# UNDERSTANDING TRAINING AND ADAPTATION IN FEATURE LEARNING: FROM TWO-LAYER NETWORKS TO FOUNDATION MODELS

by

Zhenmei Shi

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2024

Date of final oral examination: 12/19/2024

The dissertation is approved by the following members of the Final Oral Committee:
Yingyu Liang, Associate Professor, Computer Sciences
Xiaojin Zhu, Professor, Computer Sciences
Frederic Sala, Assistant Professor, Computer Sciences
Yin Li, Assistant Professor, Biostatistics & Medical Informatics and Computer Sciences

*For my wife, parents, and brother.*

## ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my love, Yingxin Jia. We met in September 2021 on the UW-Madison campus when I was a third-year graduate student, and she was a first-year graduate student. In November 2024, we married in Seattle. Throughout my PhD journey, Yingxin has been my unwavering supporter, always encouraging me to pursue my dreams and comforting me during moments of frustration. Her boundless love, steadfast encouragement, and profound understanding have been the foundation of my strength during the most challenging times. Without her, I would not have achieved this milestone.

I am profoundly grateful to my parents and brother for their unconditional support over the past 28 years. They have always provided me with the best opportunities and a happy childhood, encouraging me to strive for excellence. I am truly fortunate to have such exceptional individuals in my life, whose presence has made all the difference.

I extend my heartfelt appreciation to my PhD advisor, Prof. Yingyu Liang, for his invaluable guidance, expertise, and constant encouragement. I still vividly remember the excitement I felt when I received his email offering me a position at UW-Madison. Over the past five years, Prof. Liang has patiently mentored me, especially during the early stages when I was new to research. Together, we spent over two years working on my first publication, which laid the foundation for many successful projects thereafter. As I progressed, I learned not only to conduct research but also to mentor junior researchers, a skill I will carry forward thanks to his example. His dedication, profound knowledge, and passion for research have inspired me to push my boundaries and strive for excellence. His mentorship has been instrumental in shaping both my research trajectory and personal growth.

I am deeply thankful to my thesis committee members, Prof. Xiaojin (Jerry) Zhu, Prof. Frederic (Fred) Sala, and Prof. Yin Li, for their insightful feedback, constructive criticism, and valuable suggestions. They have also served as my preliminary exam committee members, and Jerry and Fred were on my qualification exam committee as well. Collaborations with Fred and Yin have been transforma-

tive, and the theoretical machine learning courses (CS760 and CS861) taught by Jerry were pivotal in providing tools that significantly shaped my later research. Their expertise and dedication have been critical to my academic development, and I am sincerely grateful for their time and effort.

I am indebted to all the collaborators who have enriched my PhD journey. I would like to thank the members of the Liang group—Jiefeng Chen, Junyi Wei, Zhuoyan Xu, and Yang Guo—for their hard work and camaraderie. I also appreciate my collaborators on the UW-Madison campus, including Yifei Ming, Ying Fan, Yiyou Sun, Jiayu Wang, Prof. Yixuan Li, and Prof. Somesh Jha, for their inspiring ideas and shared efforts. My sincere gratitude extends to my internship mentors, Dr. Fuhao Shi, Dr. Myra Nam, and Dr. Sercan Arik at Google, as well as Dr. Shafiq Joty at Salesforce, for their invaluable guidance. I would like to give special thanks to Prof. Tengyu Ma for inviting me to join his start-up company after completing my PhD journey.

I also wish to acknowledge Dr. Zhao Song, my mentor at Adobe, for a year of productive collaboration exploring fascinating research directions. I am grateful to the junior researchers I have had the pleasure of working with, including Yufa Zhou, Zhizhou Sha, Xiaoyu Li, Jiangxuan Long, Bo Chen, Junze Yin, Chiwun Yang, Yekun Ke, and others, who contributed to my growth as a researcher.

Finally, I would like to express my gratitude to the many individuals who supported me in various ways. I am especially thankful to Hang Yin, my roommate during the first two and a half years, for sharing the challenges of the COVID-19 period. I am deeply appreciative of Zhihan Guo for introducing me to my wife, and my close friends in Madison, including Zifan Liu, Kan Wu, and many others, for their unwavering camaraderie and encouragement. Their presence has made this journey more meaningful and enjoyable.

To everyone who contributed to my PhD journey, directly or indirectly, I extend my heartfelt thanks. This achievement would not have been possible without your support!

## CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

## ABSTRACT

Deep neural networks have achieved remarkable success across various domains of artificial intelligence. A key factor in their success is their ability to learn effective feature representations from data, distinguishing them from traditional machine learning methods. This thesis explores how feature learning emerges during neural network training and demonstrates its crucial role in foundation models' adaptation to downstream applications.

First, we provide theoretical insights into the emergence of feature learning in neural networks. We demonstrate that the networks can efficiently learn class-relevant patterns in early training stages using minimal parameters, avoiding the curse of dimensionality that affects traditional methods. Our analysis reveals that this capability stems from the networks' ability to leverage inherent input data structures. We develop a unified analysis framework for two-layer networks trained by gradient descent, characterizing how feature learning occurs beyond kernel approaches. We extend our investigation to Transformer architectures, analyzing Fourier features in one-layer Transformers and uncovering the relationship between model scale and in-context learning behavior. Our findings reveal that larger models cover more hidden features while smaller ones emphasize important features, leading to different in-context learning behaviors.

Building on these theoretical insights, we develop practical applications for foundation models. We introduce nuclear norm regularization for improved domain generalization, demonstrating consistent performance improvements across various tasks. We address the trade-off between universality and label efficiency in contrastive learning through a novel regularization method. Furthermore, we propose looped Transformers for implementing multi-step gradient descent in in-context learning and develop GemFilter, an algorithm that leverages early-layer attention features to accelerate Large Language Model inference.

This thesis advances our understanding of feature learning in neural networks and provides practical methods for improving foundation models' performance, developing more efficient and effective machine learning systems.

# 1 INTRODUCTION

Nowadays, deep neural networks dominate artificial intelligence and machine learning. Deep Learning has achieved remarkable empirical success and has been a main driving force for the recent progress in machine learning and artificial intelligence. It has been widely used in many applications, such as computer vision (He et al., 2016; Ren et al., 2015; Chen et al., 2018; Goodfellow et al., 2014; Krizhevsky et al., 2012), natural language processing (Devlin et al., 2019; Vaswani et al., 2017; Williams et al., 2018; Dolan and Brockett, 2005; Howard and Ruder, 2018; Rajpurkar et al., 2016; Peters et al., 2018), speech recognition (Hannun et al., 2014; Abdel-Hamid et al., 2014; Amodei et al., 2016; Chorowski et al., 2015) and game playing (Silver et al., 2016, 2017), etc. In particular, foundation models (Bommasani et al., 2021), e.g., Llama (AI, 2024) and ChatGPT (OpenAI, 2022), have been widely involved in people's living and work and demonstrated immense potential to enhance various aspects.

Various empirical studies have shown that an important characteristic of deep neural networks is their feature learning ability, i.e., to learn a feature mapping for the inputs which allows better performance. This is widely believed to be a key factor to their remarkable success in many applications, in particular, an advantage over traditional machine learning methods. To understand their success, it is crucial to understand the source and benefit of feature learning in neural networks. Only when we have a deeper understanding of feature learning, we can improve deep learning comprehensively in a wide range of real applications. This raises critical research questions:

*How does feature learning emerges during the neural network training?*
*If we know that, could we utilize feature learning better, particularly in foundation models?*

In this thesis, we assert the following statement:

*Feature learning emerges from input data structures during the neural network training and is crucial for foundation models adaptation to downstream applications.*

In real-life vision tasks, the raw input data has a extremely large dimension. For example, one color image from a 120 fps HD 1080p YouTube video has dimensions $1920 \times 1080 \times 3 = 6.2 \times 10^6$. A 10-min video clip will have dimensions $600 \times 120 \times 6.2 \times 10^6 = 4.5 \times 10^{11}$. We know that there is a huge information redundancy. Empirical observations show that deep networks can discover an effective feature representation for the task and learn neurons that correspond to different important semantic patterns in the inputs (e.g., human eyes, bird shapes, tires, etc. in images). Thus, it can non-linearly project the image to a feature space with much fewer dimensions, e.g., 512. However, most traditional machine learning methods, like kernel methods, suffer from the curse of dimensionality which is not tolerable in a real-world application. In contrast, due to feature learning, neural networks can routinely tackle high-dimensional datasets and adapt to the latent low-dimensional structure without suffering from the curse of dimensionality.

In this thesis, we first study how feature learning emerge in the neural network training dynamic, under settings including two-layer neural networks and one-layer Transformers (Vaswani et al., 2017). We find that the task-useful feature is emergent from the input data distribution, which explains how feature learning make neural networks successful and beyond traditional machine learning methods. Secondly, based on our understanding, we propose many algorithm utilization feature learning to make the foundation models have better downstream application adaptation, including domain generalization, few-shot adaptation, in-context learning, and Large Language Models inference acceleration. In the subsequent sections, we will delve into the specific details of these two aspects and discuss their respective contributions.

## 1.1 How does Feature Learning Emerge in the Training of the Neural Network Dynamic?

Given the importance of feature learning, it is necessary to understand how feature learning emerges to improve neural networks' feature learning ability. The analysis of feature learning is extremely challenging because the optimization of deep neural networks is non-convex. Even training a three-node network can be NP-complete in the worst case (Blum and Rivest, 1989). In contrast, the practical networks can be of hundreds of layers with millions of nodes but can be trained to small training losses with a relatively simple algorithm, in particular, Stochastic Gradient Descent (SGD) with back-propagation. Overcoming the challenge, our studies show two critical properties resulting in feature learning: (1) feature learning emergence from input data structures in the first few learning steps; (2) implicit regularization/simplicity bias of SGD towards effective features structure after the first few learning steps.

We refer readers to Chapter 2 to Chapter 4 for more details. Here, we made a summary with high-level intuition.

**Feature learning emergence from input data structures.** In Chapter 2, to thoroughly understand the importance of feature learning for success, our work (Shi et al., 2022c) proposes to analyze learning problems motivated by practical data. The labels are determined by a set of class-relevant patterns and the inputs are generated from these along with some background patterns. Our work demonstrates that a neural network with a small number of parameters is enough to learn some class-relevant patterns in a few training steps, and these patterns are sufficient to make the model a good performance on the task. On the other hand, traditional machine learning methods with fixed features do not have such power. They need exponential fixed features to cover the whole feature space so that they can contain enough class-relevant features with a high probability to gain non-trivial performance. Covering the whole feature space will lead to the curse of dimensionality. Thus, we answer that, by feature learning, neural networks can figure out the low-dimensional structure of useful features and prevent the curse of dimensionality. Furthermore, we show that feature learning in the first few steps of neural networks comes from the input data structure. Our work demonstrates that if there is no input data structure, both neural networks and traditional methods cannot be better than random guessing. It suggests that the input data structures (useful patterns) are crucial for efficient machine learning in real-world applications.

**Provable guarantees for neural networks via gradient feature learning.** The current theoretical analysis of deep neural network is not adequate for understanding their success, e.g., the Neural Tangent Kernel approach fails to capture their key feature learning ability, while recent analyses on feature learning are typically problem-specific. In Chapter 3, our work (Shi et al., 2023d) proposes a unified analysis framework for two-layer networks trained by gradient descent, which extends the first work to general data distributions. The framework is centered around the principle of feature learning from gradients, and its effectiveness is demonstrated by applications in several prototypical problems such as mixtures of Gaussians and parity functions. The framework also sheds light on interesting network learning phenomena such as feature learning beyond kernels and the lottery ticket hypothesis.

**Fourier feature in one-layer Transformers.** In the evolving landscape of machine learning, a pivotal challenge lies in deciphering the feature representations harnessed by neural networks and Transformers. In Chapter 4, our work (Li et al., 2024b) directs our focus to the complex algebraic learning task of modular addition involving $k$ inputs. Our research presents a thorough analytical characterization of the features learned by stylized one-hidden layer neural networks and one-layer Transformers in addressing this task. A cornerstone of our theoretical framework is the elucidation of how the principle of margin maximization shapes the features adopted by one-hidden layer neural networks. We demonstrate that a neuron count of these networks attain a maximum margin solution. Furthermore, we establish that each hidden-layer neuron aligns with a specific Fourier feature, integral to solving modular addition problems. By correlating our findings with the empirical observations of similar studies, we contribute to a deeper comprehension of the intrinsic computational mechanisms of neural networks. Furthermore, we observe similar computational mechanisms in attention matrices of one-layer Transformers. Our work stands as a significant stride in unraveling their operation complexities, particularly in the realm of complex algebraic tasks.

**Feature learning in in-context learning.** Large language models (LLM) have emerged as a powerful tool for AI, with the key ability of incontext learning (ICL), where they can perform well on unseen tasks based on a brief series of task examples without necessitating any adjustments to the model parameters. One recent interesting mysterious observation is that models of different scales may have different ICL behaviors: larger models tend to be more sensitive to noise in the test context. In Chapter 5, our work (Shi et al., 2024b) studies this observation theoretically aiming to improve the understanding of LLM and ICL. We analyze two stylized settings: (1) linear regression with one-layer singlehead linear Transformers and (2) parity classification with two-layer multiple attention heads Transformers (non-linear data and non-linear model). In both settings, we give closed-form optimal solutions and find that smaller models emphasize important hidden features while larger ones cover more hidden features; thus, smaller models are more robust to noise while larger ones are more easily distracted, leading to different ICL behaviors. This sheds light on where Transformers pay attention to and how that affects ICL.

## 1.2 What Forward Path will be led by Feature Learning?

Pre-trained representations (a.k.a. foundation models) have recently become a prevalent learning paradigm, where one first self-supervised pre-trains a representation function using large-scale unlabeled data and then learns simple predictors on top of the representation using small labeled data from the downstream tasks. The self-supervised pre-training can compete with or outperform supervised pre-training on the downstream prediction performance. Practical examples like GPT-3 (a language generative model introduced by OpenAI, has 175B parameters trained on 300B tokens and costs millions of dollars), CLIP, DALL·E, PaLM, and Flamingo have obtained effective representations universally useful for a wide range of downstream tasks. There are two key desiderata for the representation function: *label efficiency* (the ability to learn an accurate classifier on top of the representation with a small amount of labeled data) and *universality* (usefulness across a wide range of downstream tasks). As we have a better understanding of feature learning emerging from input data structure, it can advise us on the proper way to improve foundation models.

In Chapter 6 and Chapter 7, we provide several cases of using feature learning analysis in real-world applications to gain two key desiderata. Furthermore, in Chapter 8 and Chapter 9, we illustrate how to utilize feature learning to improve Transformers. Here, we made a summary with high-level intuition

**Domain generalization.** The ability to generalize to unseen domains is crucial for machine learning systems deployed in the real world, especially when we only have data from limited training domains. In Chapter 6, we propose a simple and effective regularization method based on the nuclear norm of the learned features for domain generalization. Intuitively, the proposed regularizer mitigates the impacts of environmental features and encourages learning domain-invariant features. Theoretically, we provide insights into why nuclear norm regularization is more effective compared to ERM and alternative regularization methods. Empirically, we conduct extensive experiments on both synthetic and real datasets. We show that nuclear norm regularization achieves strong performance compared to baselines in a wide range of domain generalization tasks. Moreover, our regularizer is broadly applicable with various methods such as ERM and SWAD with consistently improved performance, e.g., 1.7% and 0.9% test accuracy improvements respectively on the DomainBed benchmark.

**Contrastive regularization.** In Chapter 7, we focus on one of the most popular instantiations of this paradigm: contrastive learning with linear probing, i.e., learning a linear predictor on the representation pre-trained by contrastive learning. We show that there exists a trade-off between the two desiderata so that one may not be able to achieve both simultaneously. Specifically, we provide analysis using a theoretical data model and show that, while more diverse pre-training data result in more diverse features for different tasks (improving universality), it puts less emphasis on task-specific features, giving rise to larger sample complexity for downstream supervised tasks, and thus worse prediction performance. Guided by this analysis, we propose a contrastive regularization method to improve the trade-off. We validate our analysis and method empirically with systematic experiments using real-world datasets and foundation models.

**Looped Transformers in in-context learning.** In-context learning has been recognized as a key factor in the success of Large Language Models (LLMs). It refers to the model's ability to learn patterns on the fly from provided in-context examples in the prompt during inference. Previous studies have demonstrated that the Transformer architecture used in LLMs can learn feature by implementing a single-step gradient descent update by processing in-context examples in a single forward pass. In Chapter 8, our work show that, during in-context learning, a looped Transformer can learn feature by implementing multi-step gradient descent updates in forward passes. We study linear looped Transformers in-context learning on linear vector generation tasks. We show that linear looped Transformers can learn feature by implementing multi-step gradient descent efficiently for in-context learning. Our results demonstrate that as long as the input data has a constant condition number, the linear looped Transformers can achieve a small error by multi-step gradient descent during in-context learning. Our findings offer new insights into the mechanisms behind LLMs and potentially guiding the better design of efficient inference algorithms for LLMs.

**Utilizing attention feature to compress input.** Large Language Models (LLMs) have demonstrated remarkable capabilities in handling long context inputs, but this comes at the cost of increased computational resources and latency. In Chapter 9, our research (Shi et al., 2024a) introduces a novel approach for the long context bottleneck to accelerate LLM inference and reduce GPU memory consumption. Our research demonstrates that LLMs can identify relevant tokens by attention feature in the early layers before generating answers to a query. Leveraging this insight, we propose an algorithm that uses early layers of an LLM as filters to select and compress input tokens, significantly reducing the context length for subsequent processing. Our method, GemFilter, demonstrates substantial improvements in both speed and memory efficiency compared to existing techniques. Notably, it achieves a 2.4× speedup and 30% reduction in GPU memory usage compared to SOTA methods. GemFilter is simple, training-free, and broadly applicable across different LLMs. Crucially, it provides interpretability by allowing humans to inspect the selected input sequence. These findings not only offer practical benefits for LLM deployment, but also enhance our understanding of LLM internal mechanisms, paving the way for further optimizations in LLM design and inference.

## 1.3 Summary of Contributions

This thesis makes several significant contributions to understanding feature learning in neural networks and its applications in foundation models.

The first major contribution is a theoretical understanding of feature learning emergence. We demonstrated that feature learning emerges from input data structures during early training steps. Our analysis proved that neural networks can learn class-relevant patterns with minimal parameters, avoiding the curse of dimensionality that plagues traditional methods. We developed a unified analysis framework for two-layer networks trained by gradient descent, revealing how feature learning occurs beyond kernel approaches.

The second key contribution centers on novel insights into Transformer architecture. We characterized feature representations in one-layer Transformers through the lens of Fourier features and revealed the relationship between model scale and in-context learning behavior. Our work demonstrated how larger models cover more hidden features while smaller models emphasize important ones, providing crucial insights into model behavior at different scales.

The third significant contribution focuses on practical applications in foundation models. We introduced nuclear norm regularization for improved domain generalization and developed contrastive regularization methods to balance universality and label efficiency. Additionally, we created the GemFilter algorithm for efficient long-context processing in LLMs and proposed looped Transformers for implementing multi-step gradient descent in in-context learning.

## 1.4 Thesis Outline

The remainder of this thesis is organized into two main parts, focusing on theoretical understanding of feature learning emergence and practical applications in foundation models.

Chapter 2-Chapter 5 establish the theoretical foundations of feature learning emergence. Chapter 2 analyzes how feature learning emerges from input data structures in neural networks. We present our theoretical framework for understanding feature learning in the first few training steps and demonstrate why neural networks can avoid the curse of dimensionality while traditional methods cannot. Chapter 3 introduces our unified analysis framework for two-layer networks trained by gradient descent, showing how gradient-based feature learning occurs beyond kernel approaches. Chapter 4 and Chapter 5 extends our analysis to Transformers, characterizing Fourier features in one-layer Transformers and examining the relationship between model scale and in-context learning behavior.

Chapter 6-Chapter 9 focus on practical applications of feature learning in foundation models. Chapter 6 presents our nuclear norm regularization method for domain generalization, including theoretical analysis and empirical validation across various tasks. Chapter 7 explores the trade-off between universality and label efficiency in contrastive learning, introducing our contrastive regularization method to improve this balance. Chapter 8 describes our work on looped Transformers for in-context learning, demonstrating how they can implement multi-step gradient descent updates. Chapter 9 presents GemFilter, our algorithm for efficient long-context processing in Large Language Models, showing how feature learning can be leveraged to improve computational efficiency.

Chapter 10 concludes the thesis by summarizing our key findings and contributions to feature learning theory and practice. We discuss the broader implications of our work for the field of machine learning and outline promising directions for future research.

Each chapter begins with an overview of the specific problem being addressed and concludes with a discussion of results and their implications. Technical proofs and additional experimental details are provided in appendices to maintain readability while ensuring completeness.

# 2 A THEORETICAL ANALYSIS ON FEATURE LEARNING IN NEURAL NETWORKS: EMERGENCE FROM INPUTS AND ADVANTAGE OVER FIXED FEATURES

**Contribution statement.** This chapter is joint work with Junyi Wei and Yingyu Liang. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis, and completed all the experiments. The results of this chapter have been published as a conference paper in ICLR 2022 (Shi et al., 2022c).

## 2.1 Introduction

Various empirical studies have shown that an important characteristic of neural networks is their feature learning ability, i.e., to learn a feature mapping for the inputs which allow accurate prediction (e.g., Zeiler and Fergus (2014); Girshick et al. (2014); Zhang et al. (2019); Manning et al. (2020)). This is widely believed to be a key factor to their remarkable success in many applications, in particular, an advantage over traditional machine learning methods. To understand their success, it is then crucial to understand the source and benefit of feature learning in neural networks. Empirical observations show that networks can learn neurons that correspond to different semantic patterns in the inputs (e.g., eyes, bird shapes, tires, etc. in images (Zeiler and Fergus, 2014; Girshick et al., 2014)). Moreover, recent progress (e.g., Caron et al. (2018); Chen et al. (2020c); He et al. (2020a); Jing and Tian (2020)) shows that one can even learn a feature mapping using only unlabeled inputs and then learn an accurate predictor (usually a linear function) on it using labeled data. This further demonstrates the feature learning ability of neural networks and that these input distributions contain important information for learning useful features. These empirical observations strongly suggest that the structure of the input distribution is crucial for feature learning and feature learning is crucial for the strong performance. However, it is largely unclear how practical training methods (gradient descent or its variants) learn important patterns from the inputs and whether this is necessary for obtaining the superior performance, since the empirical studies do not exclude the possibility that some other training methods can achieve similar performance without feature learning or with feature learning that does not exploit the input structure. Rigorous theoretical investigations are thus needed for answering these fundamental questions: *How can effective features emerge from inputs in the training dynamics of gradient descent? Is learning features from inputs necessary for the superior performance?*

Compared to the abundant empirical evidence, the theoretical understanding still remains largely open. One line of work (e.g. Jacot et al. (2018); Li and Liang (2018); Du et al. (2019); Allen-Zhu et al. (2019b); Zou et al. (2020); Chizat et al. (2019) and many others) shows in certain regime, sufficiently overparameterized networks are approximately linear models, i.e., a linear function on the Neural Tangent Kernel (NTK). This falls into the traditional approach of linear models on fixed features, which also includes random features (Rahimi and Recht, 2008) and other kernel methods (Kamath et al., 2020). The kernel viewpoint thus does not explain feature learning in networks nor the advantage over fixed features. A recent line of work (e.g. Daniely and Malach (2020); Bai and Lee (2019); Ghorbani et al. (2020); Yehudai and Shamir (2019); Allen-Zhu and Li (2019, 2020a); Li et al. (2020); Malach et al. (2021) and others) shows examples where networks provably enjoy advantages over fixed features, under different settings and assumptions. While providing insightful results separating the two approaches, most studies have not investigated if the input structure is crucial for feature learning and thus the advantage. Also, most studies have not analyzed how gradient descent can learn important input patterns as effective features, or rely on strong assumptions like models or data atypical in practice (e.g., special networks, Gaussian data, etc).

Towards a more thorough understanding, we propose to analyze learning problems motivated by practical data, where the labels are determined by a set of class relevant patterns and the inputs are generated from these along with some background patterns. We use comparison for our study: (1) by comparing network learning approaches with fixed feature approaches on these problems, we analyze the emergence of effective features and demonstrate feature learning leads to the advantage over fixed features; (2) by comparing these problems to those with the input structure removed, we demonstrate that the input structure is crucial for feature learning and prediction performance.

More precisely, we obtain the following results. We first prove that two-layer networks trained by gradient descent can efficiently learn to small errors on these problems, and then prove that no linear models on fixed features of polynomial sizes can learn to as good errors. These two results thus establish the provable advantage of networks and implies that feature learning leads to this advantage. More importantly, our analysis reveals the dynamics of feature learning: the network first learns a rough approximation of the effective features, then improves them to get a set of good features, and finally learns an accurate classifier on these features. Notably, the improvement of the effective features in the second stage is needed for obtaining the provable advantage. The analysis also reveals the emergence and improvement of the effective features are by exploiting the data, and in particular, they rely on the input structure. To formalize this, we further prove the third result: if the specific input structure is removed and replaced by a uniform distribution, then no polynomial algorithm can even weakly learn in the Statistical Query (SQ) learning model, not to mention the advantage over fixed features. Since SQ learning includes essentially all known algorithms (in particular, mini-batch stochastic gradient descent used in practice), this implies that feature learning depends strongly on the input structure. Finally, we perform simulations on synthetic data to verify our results. We also perform experiments on real data and observe similar phenomena, which show that our analysis provides useful insights for the practical network learning.

Our analysis then provides theoretical support for the following principle: *feature learning in neural networks depends strongly on the input structure and leads to the superior performance.* In particular, our results make it explicit that learning features from the input structure is crucial for the superior performance. This suggests that input-distribution-free analysis (e.g., traditional PAC learning) may not be able to explain the practical success, and advocates an emphasis of the input structure in the analysis. While these results are for our proposed problem setting and network learning in practice can be more complicated, the insights obtained match existing empirical observations and are supported by our experiments. The compelling evidence hopefully can attract more attention to further studies on modeling the input structure and analyzing feature learning.

## 2.2 Related Work

This section provides an overview while more technical discussions can be found in Appendix A.3.

**Neural Tangent Kernel (NTK) and Linearization of Neural Networks.** One line of work (e.g. Jacot et al. (2018); Li and Liang (2018); Matthews et al. (2018); Lee et al. (2019a); Novak et al. (2019); Yang (2019); Du et al. (2019); Allen-Zhu et al. (2019b); Zou et al. (2020); Ji and Telgarsky (2019b); Cao et al. (2020); Geiger et al. (2020); Chizat et al. (2019) and more) explains the success of sufficiently over-parameterized neural network by connecting them to linear methods like NTK. Though their approaches are different, they all base on the observation that when the network is sufficiently large, the weights stay close to the initialization during the training, and training is similar to solving a kernel method problem. This is typically referred to as the NTK regime, or lazy training, or linearization. However, networks used in practice are usually not large enough to enter this regime, and the weights are frequently observed to traverse away from the initialization. Furthermore, in this regime, network learning is essentially the traditional approach of linear methods over fixed features, which cannot establish or explain feature learning and the advantage of network learning.

**Advantage of Neural Networks over Linear Models on Fixed Features.** Since the superior network learning results via gradient descent are not well explained by the NTK view, a recent line of work has turned to learning settings where neural networks provably have advantage over linear models on fixed features (e.g. Daniely and Malach (2020); Refinetti et al. (2021); Malach et al. (2021); Dou and Liang (2020); Bai and Lee (2019); Ghorbani et al. (2020); Allen-Zhu and Li

(2019); see the great summary in Malach et al. (2021)). While formally establishing the advantage, they have not thoroughly answered the two fundamental questions this work focuses on; in particular, most existing work has not studied whether the input structure is a crucial factor for feature learning and thus the advantage, and/or has not considered how the features are learned in more practical training scenarios. For example, Ghorbani et al. (2020) show the advantage of networks in approximation power and Dou and Liang (2020) show their statistical advantage, but they do not consider the learning dynamics (i.e., how the training method obtains the good network). Allen-Zhu and Li (2019) prove the advantage of the networks for PAC learning with labels given by a depth-2 ResNet and Allen-Zhu and Li (2020a) prove for Gaussian inputs with labels given by a multiple-layer network, while neither considers the influence of the input structure on feature learning or the advantage. Daniely and Malach (2020) prove the advantage of the networks for learning sparse parities on specific input distributions that help gradient descent learn effective features for prediction, and Malach et al. (2021) consider similar learning problems but with specifically designed differentiable models, while our work analyzes data distributions and models closer to those in practice and also explicitly focuses on whether the input structure is needed for the learning. There are also other theoretical studies on feature learning in networks (e.g. Yehudai and Ohad (2020); Zhou et al. (2021b); Diakonikolas et al. (2020); Frei et al. (2020)), which however do not directly relate feature learning to the input structure or the advantage of network learning.

## 2.3 Problem Setup

To motivate our setup, consider images with various kinds of patterns like lines and rectangles. Some patterns are relevant for the labels (e.g., rectangles for distinguishing indoor or outdoor images), while the others are not. If the image contains a sufficient number of the former, then we are confident that the image belongs to a certain class. Dictionary learning or sparse coding is a classic model of such data (e.g., Olshausen and Field (1997); Vinje and Gallant (2000); Blei et al. (2003)). We thus model the patterns as a dictionary, generate a hidden vector indicating the presence of the patterns, and generate the input and label from this vector.

Let $\mathbf{X} = \mathbb{R}^d$ be the input space, and $\mathcal{Y} = \{\pm 1\}$ be the label space. Suppose $M \in \mathbb{R}^{d \times D}$ is an unknown dictionary with $D$ columns that can be regarded as patterns. For simplicity, assume $M$ is orthonormal. Let $\tilde{\phi} \in \{0, 1\}^D$ be a hidden vector that indicates the presence of each pattern. Let $\mathbf{A} \subseteq [D]$ be a subset of size $k$ corresponding to the class relevant patterns. Then the input is generated by $M\tilde{\phi}$, and the label can be any binary function on the number of class relevant patterns. More precisely, let $P \subseteq [k]$. Given $\mathbf{A}$ and $P$, we first sample $\tilde{\phi}$ from a distribution $\mathcal{D}_{\tilde{\phi}}$, and then generate the input $\tilde{x}$ and the class label $y$ from $\tilde{\phi}$:

$$\tilde{\phi} \sim \mathcal{D}_{\tilde{\phi}}, \quad \tilde{x} = M\tilde{\phi}, \quad y = \begin{cases} +1, & \text{if } \sum_{i \in \mathbf{A}} \tilde{\phi}_i \in P, \\ -1, & \text{otherwise.} \end{cases} \tag{2.1}$$

**Learning with Input Structure.** We allow quite general $\mathcal{D}_{\tilde{\phi}}$ with the following assumptions:

- **(A0)** The class probabilities are balanced: $\Pr[\sum_{i \in \mathbf{A}} \tilde{\phi}_i \in P] = 1/2$.
- **(A1)** The patterns in $\mathbf{A}$ are correlated with the labels with the same correlation: for any $i \in A$, $\gamma = \mathbb{E}[y\tilde{\phi}_i] - \mathbb{E}[y]\mathbb{E}[\tilde{\phi}_i] > 0$.
- **(A2)** Each pattern outside $\mathbf{A}$ is identically distributed and independent of all other patterns. Let $p_o := \Pr[\tilde{\phi}_i = 1]$ and without loss of generality assume $p_o \leqslant 1/2$.

Let $\mathcal{D}(\mathbf{A}, P, \mathcal{D}_{\tilde{\phi}})$ denote the distribution on $(\tilde{x}, y)$ for some $\mathbf{A}, P$, and $\mathcal{D}_{\tilde{\phi}}$. Given parameters $\Xi = (d, D, k, \gamma, p_o)$, the family $\mathcal{F}_\Xi$ of distributions include all $\mathcal{D}(\mathbf{A}, P, \mathcal{D}_{\tilde{\phi}})$ with $\mathbf{A} \subseteq [D]$, $P \subseteq [k]$, and $\mathcal{D}_{\tilde{\phi}}$ satisfying the above assumptions. The labeling function includes some interesting special cases:

*Example 1.* Suppose $P = \{i \in [k] : i > k/2\}$ for some threshold, i.e., we will set the label $y = +1$ when more than a half of the relevant patterns are presented in the input.

*Example 2.* Suppose $k$ is odd, and let $P = \{i \in [k] : i \text{ is odd}\}$, i.e., the labels are given by the parity function on $\tilde{\phi}_j (j \in \mathbf{A})$. This is useful to prove our lower bounds via the properties of parities.

Appendix A.8 presents results for more general settings (e.g., incoherent dictionary, unbalanced classes, etc.). On the other hand, our problem setup does not include some important data models. In particular, one would like to model hierarchical representations often observed in practical data and believed to be important for deep learning. We leave such more general cases for future work.

**Learning Without Input Structure.** For comparison, we also consider learning problems without input structure. The data are generated as above but with different distributions $\mathcal{D}_{\tilde{\phi}}$:

**(A1′)** The patterns are uniform over $\{0, 1\}^D$: for any $i \in [D], \Pr[\tilde{\phi}_i = 1] = 1/2$ independently.

Given parameters $\Xi_0 = (d, D, k)$, the family $\mathcal{F}_{\Xi_0}$ of distributions without input structure is the set of all the distributions with $\mathbf{A} \subseteq [D]$, $P \subseteq [k]$ and $\mathcal{D}_{\tilde{\phi}}$ satisfying the above assumptions.

### 2.3.1 Neural Network Learning

**Networks.** We consider training a two-layer network via gradient descent on the data distribution:

$$g(\mathbf{x}) = \sum_{i=1}^{2m} a_i \sigma(\langle \mathbf{w}_i, \mathbf{x} \rangle + b_i) \tag{2.2}$$

where $\mathbf{w}_i \in \mathbb{R}^d, b_i, a_i \in \mathbb{R}$, and $\sigma(z) = \min(1, \max(z, 0))$ is the truncated rectified linear unit (ReLU) activation function. Let $\theta = \{\mathbf{w}_i, b_i, a_i\}_{i=1}^{2m}$ denote all the parameters, and let superscript $(t)$ denote the time step, e.g., $g^{(t)}$ denote the network at time step $t$ with $\theta^{(t)} = \{\mathbf{w}_i^{(t)}, b_i^{(t)}, a_i^{(t)}\}$.

**Loss Function.** Similar to typical practice, we will normalize the data for learning: first compute $\mathbf{x} = (\tilde{x} - \mathbb{E}[\tilde{x}])/\tilde{\sigma}$ where $\tilde{\sigma}^2 = \mathbb{E} \sum_{i=1}^{d} (\tilde{x}_i - \mathbb{E}[\tilde{x}_i])^2$ is the variance of the data, and then train on $(\mathbf{x}, y)$. This is equivalent to setting $\phi = (\tilde{\phi} - \mathbb{E}[\tilde{\phi}])/\tilde{\sigma}$ and generating $\mathbf{x} = M\phi$. For $(\tilde{x}, y)$ from $\mathcal{D}$ and the normalized $(\mathbf{x}, y)$, we will simply say $(\mathbf{x}, y) \sim \mathcal{D}$.

For the training, we consider the hinge-loss $\ell(y, \hat{y}) = \max\{1 - y\hat{y}, 0\}$. We will inject some noise $\xi$ to the neurons for the convenience of the analysis. (This can be viewed as using a smoothed version of the activation $\tilde{\sigma}(z) = \mathbb{E}_\xi \sigma(z + \xi)$ similar to those in existing studies like Allen-Zhu and Li (2022); Malach et al. (2021). See Section 2.5 for more explanations.) Formally, the loss is:

$$L_{\mathcal{D}}(g; \sigma_\xi) = \mathbb{E}_{(\mathbf{x},y)}[\ell(y, g(\mathbf{x}; \xi))], \text{ where } g(\mathbf{x}; \xi) = \sum_{i=1}^{2m} a_i \mathbb{E}_\xi[\sigma(\langle \mathbf{w}_i, \mathbf{x} \rangle + b_i + \xi_i)] \tag{2.3}$$

where $\xi \sim \mathcal{N}(0, \sigma_\xi^2 I_{m\times m})$ are independent Gaussian noise. Let $L_{\mathcal{D}}(g)$ denote the typical hinge-loss without noise. We also consider $\ell_2$ regularization: $R(g; \lambda_a, \lambda_{\mathbf{w}}) = \sum_{i=1}^{2m} \lambda_a |a_i|^2 + \lambda_{\mathbf{w}} \|\mathbf{w}_i\|_2^2$ with regularization coefficients $\lambda_a, \lambda_{\mathbf{w}}$.

**Training Process.** We first perform an unbiased initialization: for every $i \in [m]$, initialize $\mathbf{w}_i^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 I_{d\times d})$ with $\sigma_{\mathbf{w}} = 1/k$, $b_i^{(0)} \sim \mathcal{N}\left(0, \sigma_b^2\right)$ with $\sigma_b = 1/k^2$, $a_i^{(0)} \sim \mathcal{N}(0, \sigma_a^2)$ with $\sigma_a = \tilde{\sigma}^2/(\gamma k^2)$, and then set $\mathbf{w}_{m+i}^{(0)} = \mathbf{w}_i^{(0)}$, $b_{m+i}^{(0)} = b_i^{(0)}$, $a_{m+i}^{(0)} = -a_i^{(0)}$. We then do gradient updates:

$$\theta^{(t)} = \theta^{(t-1)} - \eta^{(t)} \nabla_\theta \left( L_{\mathcal{D}}(g^{(t-1)}; \sigma_\xi^{(t)}) + R(g^{(t-1)}; \lambda_a^{(t)}, \lambda_{\mathbf{w}}^{(t)}) \right), \text{ for } t = 1, 2, \ldots, T, \tag{2.4}$$

for some choice of the hyperparameters $\eta^{(t)}, \lambda_a^{(t)}, \lambda_{\mathbf{w}}^{(t)}$, $\sigma_\xi^{(t)}$, and $T$.

## 2.4 Main Results

**Provable Guarantee for Neural Networks.** The network learning has the following guarantee:

**Theorem 2.1.** *For any* $\delta, \epsilon \in (0,1)$*, if* $k = \Omega\left(\log^2(D/(\delta\gamma))\right)$*,* $p_o = \Omega(k^2/D)$*, and* $\max\{\Omega(k^{12}/\epsilon^{3/2}), D\} \leqslant m \leqslant$ *poly*$(D)$*, then with properly set hyperparameters, for any* $\mathcal{D} \in \mathcal{F}_\Xi$*, with probability at least* $1-\delta$*, there exists* $t \in [T]$ *such that* $\Pr[\text{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) \leqslant \epsilon$.

The theorem shows that for a wide range of the background pattern probability $p_o$ and the number of class relevant patterns $k$, the network trained by gradient descent can obtain a small classification error. More importantly, the analysis shows the success comes from feature learning. In the early stages, the network learns and improves the neuron weights such that on the features (i.e., the neurons' outputs) there is an accurate classifier; afterwards it learns such a classifier. The next section will provide a detailed discussion on the feature learning.

**Lower Bound for Fixed Features.** Empirical observations and Theorem 2.1 do not exclude the possibility that some methods without feature learning can achieve similar performance. We thus prove a lower bound for the fixed feature approach, i.e., linear models on data-independent features.

**Theorem 2.2.** *Suppose* $\Psi$ *is a data-independent feature mapping of dimension* $N$ *with bounded features, i.e.,* $\Psi : \mathbf{X} \to [-1,1]^N$*. For* $B > 0$*, the family of linear models on* $\Psi$ *with bounded norm* $B$ *is* $\mathcal{H}_B = \{h(\tilde{x}) : h(\tilde{x}) = \langle \Psi(\tilde{x}), w\rangle, \|w\|_2 \leqslant B\}$*. If* $3 < k \leqslant D/16$ *and* $k$ *is odd, then there exists* $\mathcal{D} \in \mathcal{F}_\Xi$ *such that all* $h \in \mathcal{H}_B$ *have hinge-loss at least* $p_o\left(1 - \frac{\sqrt{2N}B}{2^k}\right)$.

So using *fixed* features independent of the data cannot get loss nontrivially smaller than $p_o$ unless with exponentially large models. In contrast, viewing the neurons $\sigma(\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i)$ as *learned* features, network learning can achieve any loss $\epsilon \in (0,1)$ with models of polynomial sizes. We emphasize the lower bound is because the feature map $\Psi$ is independent of the data. Indeed, there exists a small linear model on a small dimensional feature map allowing 0 loss for each data distribution in our problem set $\mathcal{F}_\Xi$ (Lemma 2.5). However, this feature map $\Psi^*$ is different for different data distribution in $\mathcal{F}_\Xi$, i.e., depends on the data. On the other hand, the feature map $\Psi$ in the lower bound is data-independent, i.e., fixed before seeing the data. For $\Psi$ to work simultaneously for all distributions in $\mathcal{F}_{\Xi}$, it needs to have exponential dimensions. Intuitively, it needs a large number of features, so that there are some features to approximate each $\Psi^*_i$. There are exponentially many data distributions in $\mathcal{F}_{\Xi}$, and thus exponentially many data-dependent features $\Psi^*_i$, which requires $\Psi$ to have an exponentially large dimension. Network learning updates the hidden neurons using the data and can learn to move the features to the right positions to approximate the ground-truth data-dependent features $\Psi^*$, so it does not need an exponentially large dimension feature map.

The theorem directly applies to linear models on fixed finite-dimensional feature maps, e.g., linear models on the input or random feature approaches (Rahimi and Recht, 2008). It also implies lower bounds to infinite dimensional feature maps (e.g., some kernels) that can be approximated by feature maps of polynomial dimensions. For example, Claim 1 in Rahimi and Recht (2008) implies that a function $f$ using shift-invariant kernels (e.g., RBF) can be approximated by a model $\langle \Psi(\tilde{x}), w \rangle$ with the dimension $N$ and weight norm $B$ bounded by polynomials of the related parameters of $f$ like its RKHS norm and the input dimension. Then our theorem implies some related parameter of $f$ needs to be exponential in $k$ for $f$ to get nontrivial loss, formalized in Corollary 2.3. Kamath et al. (2020) has more discussions on approximating kernels with finite dimensional maps.

**Corollary 2.3.** *For any function* $f$ *using a shift-invariant kernel* $K$ *with RKHS norm bounded by* $L$*, or* $f(x) = \sum_i \alpha_i K(z_i, x)$ *for some data points* $z_i$ *and* $\|\alpha\|_2 \leqslant L$*. If* $3 < k \leqslant D/16$ *and* $k$ *is odd, then there exists* $\mathcal{D} \in \mathcal{F}_{\Xi}$ *such that* $f$ *has hinge-loss at least* $p_o \left(1 - \frac{\mathrm{poly}(d,L)}{2^k}\right) - \frac{1}{\mathrm{poly}(d,L)}$.

**Lower Bound for Without Input Structure.** Existing results do not exclude the possibility that some learning methods without exploiting the input structure can achieve strong performance. To show the necessity of the input structure, we consider learning $\mathcal{F}_{\Xi_0}$ with input structure removed. We obtain a lower bound for such learning problems in the classic Statistical Query (SQ) model (Kearns, 1998). In this model, the algorithm can only receive information about the data through statistical queries. A statistical query is specified by some polynomially-computable property predicate $Q$ of labeled instances and a tolerance parameter $\tau \in [0,1]$. For a query $(Q,\tau)$, the algorithm receives a response $\hat{P}_Q \in [P_Q - \tau, P_Q + \tau]$, where $P_Q = \Pr[Q(x,y) \text{ is true}]$. Notice that a query can be simulated using the average of roughly $O(1/\tau^2)$ random data samples with high probability. The SQ model captures almost all common learning algorithms (except Gaussian elimination) including the commonly used mini-batch SGD, and thus is suitable for our purpose.

**Theorem 2.4.** *For any algorithm in the Statistical Query model that can learn over $\mathcal{F}_{\Xi_0}$ to classification error less than $\frac{1}{2} - \frac{1}{\binom{D}{k}^3}$, either the number of queries or $1/\tau$ must be at least $\frac{1}{2}\binom{D}{k}^{1/3}$.*

The theorem shows that without the input structure, polynomial algorithms in the SQ model cannot get a classification error nontrivially smaller than random guessing. The comparison to the result for with input structure then shows that the input structure is crucial for network learning, in particular, for achieving the advantage over fixed feature models.

## 2.5 Proof Sketches

Here we provide the sketch of our analysis, focusing on the key intuition and discussing some interesting implications. The complete proofs are included in Appendix A.4-A.6.

### 2.5.1 Provable Guarantees of Neural Networks

**Overall Intuition.** We first show that there is a two-layer network that can represent the target labeling function, whose neurons can be viewed as the “ground-truth” features to be learned. We then show that after the first gradient step, the hidden neurons of the trained network become close to the ground-truth: their weights contain large components along the class relevant patterns but small along the background patterns. We further show that in the second gradient step, these features get improved: the “signal-noise” ratio between the components for class relevant patterns and those for the background ones becomes larger, giving a set of good features. Finally, we show that the remaining steps learn an accurate classifier on these features.

**Existence of A Good Network.** We show that there is a two-layer network that can fit the labels.

**Lemma 2.5.** *For any $\mathcal{D} \in \mathcal{F}_{\Xi}$, there exists a network $g^*(\mathbf{x}) = \sum_{i=1}^{n} a_i^* \sigma(\langle \mathbf{w}_i^*, \mathbf{x}\rangle + b_i^*)$ with $y = g^*(x)$ for any $(\mathbf{x}, y) \sim \mathcal{D}$. Furthermore, the number of neurons $n = 3(k+1)$, $|a_i^*| \leqslant 32k, 1/(32k) \leqslant |b_i^*| \leqslant 1/2$, $\mathbf{w}_i^* = \tilde{\sigma} \sum_{j \in \mathbf{A}} M_j/(4k)$, and $|\langle \mathbf{w}_i^*, \mathbf{x}\rangle + b_i^*| \leqslant 1$ for any $i \in [n]$ and $(\mathbf{x}, y) \sim \mathcal{D}$.*

In particular, the weights of the neurons are proportional to $\sum_{j \in \mathbf{A}} M_j$, the sum of the class relevant patterns. We thus focus on analyzing how the network learns such neuron weights.

**Feature Emergence in the First Gradient Step.** The gradient for $\mathbf{w}_i$ (ignoring the noise) is:

$$\begin{aligned}\frac{\partial L_{\mathcal{D}}(g)}{\partial \mathbf{w}_i} &= -a_i \mathbb{E}_{(\mathbf{x}, y)\sim\mathcal{D}} \{y \mathbb{I}[y g(\mathbf{x}) \leqslant 1] \sigma'[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i]\mathbf{x}\} \\ &= -a_i \mathbb{E}_{(\mathbf{x}, y)\sim\mathcal{D}} \{y \mathbf{x} \sigma'[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i]\}\end{aligned}$$

where the last step is due to $g(\mathbf{x}) = 0$ by the unbiased initialization. Let $q_j = \langle M_j, \mathbf{w}_i\rangle$ denote the component along the direction of the pattern $M_j$. Then the component of the gradient on $M_j$ is:

$$\begin{aligned}\left\langle M_j, \frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g)\right\rangle =& - a_i \mathbb{E}\{y \phi_j \sigma'[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i]\} \\ =& - a_i \mathbb{E}\left\{ y \phi_j \sigma' \left[ \sum_{\ell \in [D]} \phi_\ell q_\ell + b_i \right] \right\}.\end{aligned}$$

The key intuition is that with the randomness of $\phi_\ell$ (and potentially that of the injected noise $\xi$), the random variable under $\sigma'$ is not significantly affected by a small subset of $\phi_\ell q_\ell$. For example, for class relevant patterns $j \in \mathbf{A}$, let $\mathbb{I}_{[D]} :=$

$\sigma'\left[\sum_{\ell\in[D]}\phi_\ell q_\ell + b_i\right]$ and $\mathbb{I}_{-\mathbf{A}} := \sigma'\left[\sum_{\ell\notin\mathbf{A}}\phi_\ell q_\ell + b_i\right]$. We have $\mathbb{I}_{[D]} \approx \mathbb{I}_{-\mathbf{A}}$ and thus:

$$\left\langle M_j, \frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g)\right\rangle \propto \mathbb{E}\left\{y\phi_j\mathbb{I}_{[D]}\right\} \approx \mathbb{E}\{y\phi_j\mathbb{I}_{-\mathbf{A}}\} = \mathbb{E}\{y\phi_j\}\mathbb{E}[\mathbb{I}_{-\mathbf{A}}] = \frac{\gamma}{\tilde{\sigma}}\mathbb{E}[\mathbb{I}_{-\mathbf{A}}]$$

since $y$ only depends on $\phi_j(j \in \mathbf{A})$. Then the gradient has a nontrivial component along the pattern. Similarly, for background patterns $j \notin \mathbf{A}$, the component of the gradient along $M_j$ is close to 0.

**Lemma 2.6** (Informal). *Assume* $p_o, k$ *as in Theorem 2.1 and* $\sigma_\xi^{(1)} < 1/k$*, then with high probability* $\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g^{(0)}; \sigma_\xi^{(1)}) = -a_i^{(0)} \sum_{j=1}^{D} M_j T_j$ *where for a small* $\epsilon_e$*:*

- *if* $j \in A$*, then* $|T_j - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_e/\tilde{\sigma})$ *with* $\beta \in [\Omega(1), 1]$*;*
- *if* $j \notin \mathbf{A}$*, then* $|T_j| \leqslant O(\sigma_\phi^2 \epsilon_e \tilde{\sigma})$*.*

By setting $\lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)})$, we have

$$\mathbf{w}_i^{(1)} = \eta^{(1)} a_i^{(0)} \sum_{j=1}^{D} M_j T_j \approx \eta^{(1)} a_i^{(0)} \frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} M_j.$$

For small $p_o$, e.g., $p_o = \tilde{O}(k^2/D)$, these neurons can already allow accurate prediction. However, for such small $p_o$, we cannot show a provable advantage of networks over fixed features. On the other hand, for larger $p_o$ meaning a significant number of background patterns in the input, the approximation error terms $T_j(j \notin \mathbf{A})$ together can overwhelm the signals $T_j(j \in \mathbf{A})$ and lead to bad prediction, even though each term is small. Fortunately, we will show that the second gradient step can improve the weights by decreasing the ratio between $T_j(j \notin \mathbf{A})$ and $T_j(j \in \mathbf{A})$.

**Feature Improvement in the Second Gradient Step.** We note that by setting a small $\eta^{(1)}$, after the update we still have $yg(\mathbf{x}; \xi) < 1$ for most $(\mathbf{x}, y) \sim \mathcal{D}$ and thus the gradient in the second step is:

$$\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g; \sigma_\xi) \approx -a_i \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\{y\mathbf{x}\mathbb{E}_\xi \sigma'[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i + \xi_i]\}.$$

We can then follow the intuition for the first step again. For $j \in \mathbf{A}$, the component $\langle M_j, \frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g)\rangle$ is roughly proportional to $\frac{\gamma}{\tilde{\sigma}}\mathbb{E}[\mathbb{I}_{-A,\xi}]$ where

$$\mathbb{I}_{-A,\xi} := \sigma' \left[\sum_{\ell \notin \mathbf{A}} \phi_\ell q_\ell + b_i + \xi_i\right].$$

While $\phi_\ell q_\ell$ may not have large enough variance, the injected noise $\xi_i$ makes sure that a nontrivial amount of data activate the neuron.[1] Then $\mathbb{I}_{-A,\xi} \neq 0$, leading to a nontrivial component along $M_j$, similar to the first step. On the other hand, for $j \notin \mathbf{A}$, the approximation error term $T_j$ depends on how well $\sigma' \left[\sum_{\ell \notin \mathbf{A}, \ell \neq j} \phi_\ell q_\ell + b_i + \xi_i\right]$ approximates $\sigma' \left[\sum_{\ell \in [D]} \phi_\ell q_\ell + b_i + \xi_i\right]$. Since the $q_\ell$'s (the weight's component along $M_\ell$) in the second step are small compared to those in the first step, we can then get a small error term $T_j$. So the ratio between $T_j (j \notin \mathbf{A})$ over $T_j (j \in \mathbf{A})$ improves after the second step, giving better features allowing accurate prediction.

**Classifier Learning Stage.** Given the learned features, we are then ready to show the remaining gradient steps can learn accurate classifiers. Intuitively, with small hyperparameter values ($\eta^{(t)} = \frac{k^2}{Tm^{1/3}}, \lambda_a^{(t)} = \lambda_{\mathbf{w}}^{(t)} \leqslant \frac{k^3}{\tilde{\sigma} m^{1/3}}, \sigma_\xi^{(t)} = 0$ for $2 < t \leqslant T = m^{4/3}$), the first layer's weights do not change too much and thus the learning is similar to convex learning using the learned features. Formally, our proof uses the online convex optimization technique in Daniely and Malach (2020).

### 2.5.2 Lower Bounds

The lower bounds are based on the following observation: our problem setup is general enough to include learning sparse parity functions. Consider an odd $k$, and let $P = \{i \in [k] : i \text{ is odd}\}$. Then $y$ is given by $\Pi_{\mathbf{A}}(z) := \prod_{j \in \mathbf{A}} z_j$ for $z_j = 2\tilde{\phi}_j - 1$, i.e., the parity function on $z_j (j \in \mathbf{A})$. Then known results for learning parity functions can be applied to prove our lower bounds.

[1] Equivalently, the network uses $\tilde{\sigma}(z) = \mathbb{E}_\xi \sigma(z + \xi)$, a Gaussian smoothed version of $\sigma$, and the smoothing allows $z$ slightly outside the activated region of $\sigma$ to generate gradient for the learning. Empirically it is not needed since typically sufficient data can activate the neurons. One potential reason is that the data have their own noise to achieve a similar effect (a remote analog being noisy gradients can help the optimization). Further analysis on such an effect is left for future work.

**Lower Bound for Fixed Features.** We show that $\mathcal{F}_{\Xi}$ contains learning problems that consist of a mixture of two distributions with weights $p_o$ and $1 - p_o$ respectively, where in the first distribution $\mathcal{D}_{\mathbf{A}}^{(1)}$, $\tilde{x}$ is given by the uniform distribution over $\tilde{\phi}$ and the label $y$ is given by the parity function on $\mathbf{A}$. On such $\mathcal{D}_{\mathbf{A}}^{(1)}$, Daniely and Malach (2020) shows that exponentially large models over fixed features is needed to get nontrivial loss. Intuitively, there are exponentially many labeling functions $\Pi_{\mathbf{A}}$ that are uncorrelated (i.e., "orthogonal" to each other): $\mathbb{E}[\Pi_{\mathbf{A}_1}\Pi_{\mathbf{A}_2}] = 0$ for any $\mathbf{A}_1$ and $\mathbf{A}_2$. Note that the best approximation of $\Pi_{\mathbf{A}}$ by a fixed set of features $\Psi_i$'s is its projection on the linear span of the features. Then with polynomial-size models, there always exists some $\Pi_{\mathbf{A}}$ far from the linear span.

*Remark.* It is instructive to compare to network learning, which finds the effective weights $\sum_{j\in\mathbf{A}} M_j$ among the exponentially many candidates corresponding to different $\mathbf{A}$'s. This can be done efficiently by exploiting the data since the gradient is roughly proportional to $\mathbb{E}\{y\mathbf{x}\} = \sum_{j\in\mathbf{A}} M_j$. The network then learns *data-dependent* features on which polynomial size linear models can achieve small loss.

**Lower Bound for Learning without Input Structure.** Clearly, $\mathcal{F}_{\Xi_0}$ contains the distributions $\mathcal{D}_{\mathbf{A}}^{(1)}$ described above. The lower bound then follows from classic SQ learning results (Blum et al., 1994).

*Remark.* The SQ lower bound analysis does not apply to $\mathcal{F}_{\Xi}$, because in $\mathcal{F}_{\Xi}$ the input distribution is related the labeling function. This allows networks to learn with polynomial time/sample. While both the labeling function and the input distribution affect the learning, few existing studies explicitly point out the importance of the input structure. We thus emphasize the input structure is crucial for networks to learn effective features and achieve superior performance.

## 2.6 Experiments

Our experiments mainly focus on feature learning and the effect of the input structure. We first perform simulations on our learning problems to (1) verify our main theorems on the benefit of feature learning and the effect of input structure; (2) verify our analysis of feature learning in networks. We then check if our insights carry over to real data: (3) whether similar feature learning is presented in real network/data; (4) whether damaging the input structure lowers the performance. The results are consistent with our analysis and provide positive support for the theory. Below we present part of the results and include the complete experimental details and results in Appendix A.7.

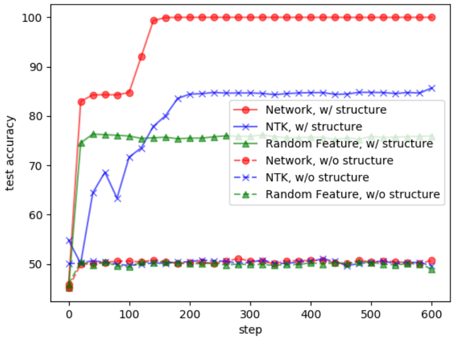


Figure 2.1: Test accuracy on simulated data with or without input structure.

**Simulation: Verification of the Main Results.** We generate data according to our problem setup, with $d = 500, D = 100, k = 5, p_o = 1/2$, a randomly sampled $\mathbf{A}$, and labels given by the parity function. We then train a two-layer network with $m = 300$ following our learning process, and for comparison, we also use two fixed feature methods (the NTK and random feature methods based on the same network). Finally, we also use these three methods on the data distribution with the input structure removed (i.e., $\mathcal{F}_{\Xi_0}$ in Theorem 2.4).

Figure 2.1 shows that the results are consistent with our results. Network learning gets high test accuracy while the two fixed feature methods get significantly lower accuracy. Furthermore, when the input structure is removed, all three methods get test accuracy similar to random guessing.

**Simulation: Feature Learning in Networks.** We compute the cosine similarities between the weights $\mathbf{w}_i$'s and visualize them by Multidimensional Scaling. (Recall that our analysis is on the *directions* of the weights without considering their *scaling*, and thus it is important to choose cosine similarity rather than say the typical

Euclidean distance.) Figure 2.2 shows that the results are as predicted by our analysis. After the first gradient step, some weights begin to cluster around the ground-truth $\sum_{j\in\mathbf{A}} M_j$ (or $-\sum_{j\in\mathbf{A}} M_j$ due to the $a_i$ in the gradient update which can be positive or negative). After the second step, the weights get improved and well-aligned with the ground-truth (with cosine similarities $> 0.99$). Furthermore, if a classifier is trained on the features after the first step, the test accuracy is about 52%; if the same is done after the second step, the test accuracy is about 100%. This demonstrates while some effective features emerge in the first step, they need to be improved in the second step to get accurate prediction.

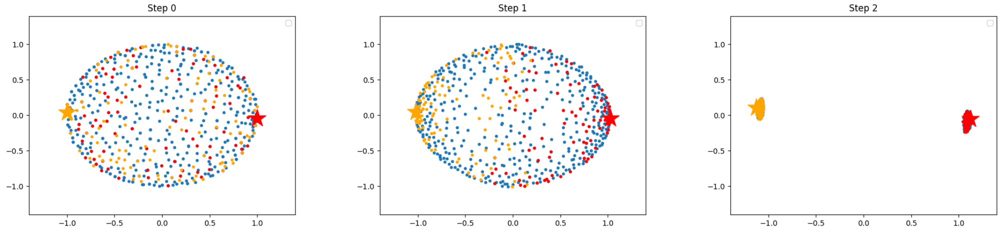


Figure 2.2: Visualization of the weights $\mathbf{w}_i$'s after initialization/one gradient step/two steps in network learning on the synthetic data. The red star denotes the ground-truth $\sum_{j\in\mathbf{A}} M_j$; the orange star is $-\sum_{j\in\mathbf{A}} M_j$. The red/orange dots are the weights closest to the red/orange star, respectively.

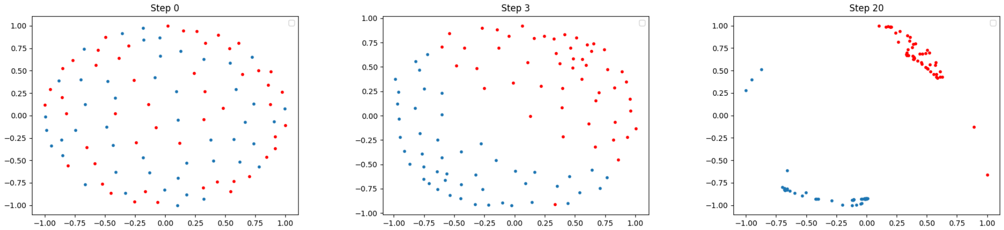
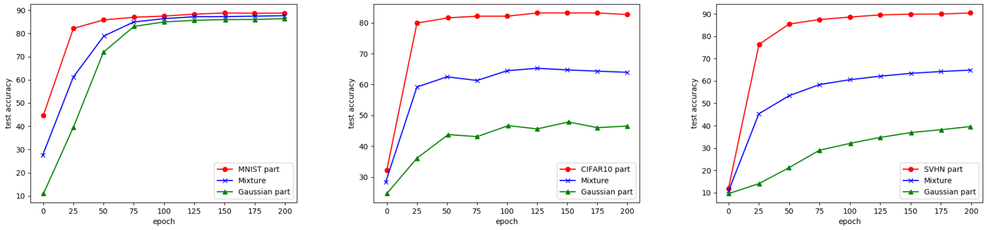


Figure 2.3: Visualization of the neurons' weights in a two-layer network trained on the subset of MNIST data with label 0/1. The weights gradually form two clusters.

**Real Data: Feature Learning in Networks.** We perform experiments on MNIST (LeCun et al., 1998; Deng, 2012), CIFAR10 (Krizhevsky, 2012), and SVHN (Netzer et al., 2011). On MNIST, we train a two-layer network with $m = 50$ on the subset



(a) (b) (c)

Figure 2.4: Test accuracy at different steps for an equal mixture of Gaussian inputs with data: (a) MNIST, (b) CIFAR10, (c) SVHN.

with labels 0/1 and visualize the neurons' weights as in the simulation. Figure 2.3 shows a similar feature learning phenomenon: effective features emerge after a few steps and then get improved to form two clusters. Similar results are observed on other datasets. These suggest the insights obtained in our analysis are also applicable to the real data.

**Real Data: The Effect of Input Structure.** Since we cannot directly manipulate the input distribution of real data, we perform controlled experiments by injecting different inputs. For labeled dataset $\mathcal{L}$ and injected input $\mathcal{U}$, we first train a teacher network fitting $\mathcal{L}$, then use the teacher network to give labels on a mixture of inputs from $\mathcal{L}$ and $\mathcal{U}$, and finally train a student network on this new dataset $\mathcal{M}$ consisting of the mixed inputs and the teacher network's labels. Checking the student' performance on different parts of $\mathcal{M}$ and comparing to those by directly training the student on the original data $\mathcal{L}$ can reveal the impact of changing the input structure. We use MNIST, CIFAR10, or SVHN as $\mathcal{L}$, and use Gaussian or images in Tiny ImageNet (Le and Yang, 2015) as $\mathcal{U}$. The networks for MNIST are two-layer with $m = 9$, and those for CIFAR10/SVHN are ResNet-18 convolutional neural networks (He et al., 2016).

Figure 2.4 shows the results on an equal mixture of data and Gaussian. It presents the test accuracy of the student on the original data part, the Gaussian part, and the whole mixture. For example, on CIFAR10, the network learns well over the CIFAR10 part (with accuracy similar to directly training on the original data) but learns slower with worse accuracy on the Gaussian part. Furthermore, the accuracy on the whole mixture is lower than that of training on the original CIFAR10. This shows that the input structure indeed has a significant impact on the learning. While MNIST+Gaussian shows a less significant trend (possibly because the tasks are simpler), the other datasets show similar significant trends as CIFAR10+Gaussian (the results using Tiny ImageNet are in the appendix).

# 3 PROVABLE GUARANTEES FOR NEURAL NETWORKS VIA GRADIENT FEATURE LEARNING

**Contribution statement.** This chapter is joint work with Junyi Wei and Yingyu Liang. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis, and completed all the experiments. The results of this chapter have been published as a conference paper in NeurIPS 2023 (Shi et al., 2023d).

## 3.1 Introduction

Neural network learning has achieved remarkable empirical success and has been a main driving force for the recent progress in machine learning and artificial intelligence. On the other hand, theoretical understandings significantly lag behind. Traditional analysis approaches are not adequate due to the overparameterization of practical networks and the non-convex optimization in the training via gradient descent. One line of work (e.g. Jacot et al. (2018); Li and Liang (2018); Chizat et al. (2019); Du et al. (2019); Allen-Zhu et al. (2019b); Zou et al. (2020) and many others) shows under proper conditions, heavily overparameterized networks are approximately linear models over data-independent features, i.e., a linear function on the Neural Tangent Kernel (NTK). While making weak assumptions about the data and thus applicable to various settings, this approach requires the network learning to be approximately using fixed data-independent features (i.e., the kernel regime, or fixed feature methods). It thus fails to capture the feature learning ability of networks (i.e., to learn a feature mapping for the inputs which allow accurate prediction), which is widely believed to be the key factor to their empirical success in many applications (e.g., Zeiler and Fergus (2014); Girshick et al. (2014); Zhang et al. (2019); Manning et al. (2020)). To study feature learning in networks, a recent line of work (e.g. Allen-Zhu and Li (2019); Bai and Lee (2019); Yehudai and Shamir (2019); Allen-Zhu and Li (2020a); Ghorbani et al. (2020); Daniely and Malach (2020); Li et al. (2020); Malach et al. (2021) and others) shows examples where networks provably enjoy advantages over fixed feature methods (including NTK), under different settings and assumptions. While providing more insights, these studies typically focus on specific problems, and their analyses exploit the specific properties of the problems and appear to be unrelated to each other. *Is there a common principle for feature learning in networks via gradient descent? Is there a unified analysis framework that can clarify the principle and also lead to provable error guarantees for prototypical problem settings?*

In this work, we take a step toward this goal by proposing a gradient feature learning framework for analyzing two-layer network learning by gradient descent. (1) The framework makes essentially no assumption about the data distribution and can be applied to various problems. Furthermore, it is centered around features from gradients, clearly illustrating how gradient descent leads to feature learning in networks and subsequently accurate predictions. (2) It leads to error guarantees competitive with the optimal in a family of networks that use the features induced by gradients on the data distribution. Then for a specific problem with structured data distributions, if the optimal in the induced family is small, the framework gives a small error guarantee.

We then apply the framework to several prototypical problems: mixtures of Gaussians, parity functions, linear data, and multiple-index models. These have been used for studying network learning (in particular, for the feature learning ability), but with different and seemingly unrelated analyses. In contrast, straightforward applications of our framework give small error guarantees, where the main effort is to compute the optimal in the induced family. Furthermore, in some cases like parities, we can handle more general data distributions than existing work.

Finally, we also demonstrate that the framework sheds light on several interesting network learning phenomena such as feature learning beyond the kernel regime, simplicity bias, and lottery ticket hypothesis (LTH). Due to space limitations, we present implications about features beyond the kernel regime in the main body but defer other implications in Section B.3 with a brief here. (1) For simplicity bias, it is generally believed that the optimization has some *implicit regularization* effect that restricts learning dynamics to a low capacity subset of the whole hypothesis class, so can lead to good generalization Neyshabur (2017); Gidel et al. (2019). Our framework provides an explanation that the learning first learns simpler functions and then more sophisticated ones where the simplicity bias is measured by the size of the family of gradient feature-induced networks. (2) For LTH, our framework allows us to formally prove LTH for two-layer networks, showing (a) the winning lottery subnetwork exists and (b) gradient descent on the subnetwork can learn to similar loss in similar runtime as on the whole network. (b) is novel and not analyzed in existing work.

## 3.2 Related Work

**Neural Networks Learning Analysis.** Recently there has been an increasing interest in the analysis of network learning. One line of work connects the sufficiently over-parameterized neural network to linear methods around its initialization like NTK (e.g. Jacot et al. (2018); Li and Liang (2018); Matthews et al. (2018); Zou et al. (2018); Oymak and Soltanolkotabi (2019); Lee et al. (2019a); Novak et al. (2019); Yang (2019); Du et al. (2019); Allen-Zhu et al. (2019b); Chizat et al. (2019); Oymak et al. (2019); Arora et al. (2019a); Cao and Gu (2019); Ji and Telgarsky (2019b); Cao et al. (2020); Geiger et al. (2020); Liang et al. (2024h); Gu et al. (2024) and more), so that the neural network training is a convex problem. The key idea is that it suffices to consider the first-order Tyler expansion of the neural network around the origin when the initialization is large enough. However, NTK lies in the lazy training (kernel) regime that excludes feature learning Chizat and Bach (2018a); Lee et al. (2018); Woodworth et al. (2020); Geiger et al. (2021). Many studies (e.g. Arora et al. (2019b); Allen-Zhu and Li (2019); Wei et al. (2019); Ghorbani et al. (2019); Yehudai and Shamir (2019); Hanin and Nica (2019); Allen-Zhu et al. (2019a); Bai and Lee (2019); Allen-Zhu and Li (2020a); Daniely and Malach (2020); Dou and Liang (2020); Lee et al. (2020); Chen et al. (2020a); Yang and Hu (2020); Huang and Yau (2020); Li et al. (2020); Ghorbani et al. (2020); Refinetti et al. (2021); Malach et al. (2021); Luo et al. (2021); Cao et al. (2022); Abbe et al. (2022b); Shi et al. (2023d) and more) show that neural networks take advantage over NTK empirically and theoretically. Another line of work is the mean-field (MF) analysis of neural networks (e.g. Mei et al. (2018); Chizat and Bach (2018b); Mei et al. (2019); Sirignano and Spiliopoulos (2020); Chen et al. (2022); Ren et al. (2023) and more). The insight is to see the training dynamics of a sufficiently large-width neural network as a PDE. It uses a smaller initialization than the NTK so that the parameters may move away from the initialization. However, the MF does not provide explicit convergence rates and requires an unrealistically large width of the neural network. One more line of work is neural networks max-margin analysis (e.g. Soudry et al. (2018); Gunasekar et al. (2018a); Nacson et al. (2019b); Ji and Telgarsky (2019a); Lyu and Li (2019); Nacson et al. (2019a); Chizat and Bach (2020); Moroshko et al. (2020); Ji and Telgarsky (2020); Telgarsky (2022); Frei et al. (2023b,a); Lyu et al. (2021) and more). They need a strong assumption that the convergence starts from weights having perfect training accuracy, while feature learning happens in the early stage of training. To explain the success of neural networks beyond the limitation mentioned above, some work introduces the low intrinsic dimension of data distributions (Chen et al., 2019a,b; Bartlett et al., 2020; Frei et al., 2021; Chatterji et al., 2021; Stöger and Soltanolkotabi, 2021). Another recent line of work is that a trained network can exactly recover the ground truth or optimal solution or teacher network (Du et al., 2018; Arora et al., 2018; Nacson et al., 2019c; Papyan et al., 2020; Oymak and Soltanolkotabi, 2020; Zhou et al., 2021b; Akiyama and Suzuki, 2021, 2023; Mousavi-Hosseini et al., 2022), but they have strong assumptions on data distribution or model structure, e.g., Gaussian marginals. Goldt et al. (2019); Wang et al. (2020a); Feng and Tu (2021); Abbe et al. (2022a); Veiga et al. (2022) show that training dynamics of neural networks have multiple phases, e.g., feature learning at the beginning, and then dynamics in convex optimization which requires proxy convexity (Frei and Gu, 2021) or PL condition (Karimi et al., 2016) or special data structure.

**Feature Learning Based on Gradient Analysis.** A recent line of work is studying how features emerge from the gradient. Allen-Zhu and Li (2022); Frei et al. (2022c) consider linear separable data and show that the first few gradient steps can learn good features, and the later steps learn a good network on neurons with these features. Daniely and Malach (2020); Shi et al. (2022c); Frei et al. (2022b) have similar conclusions on non-linear data (e.g., parity functions), while in their problems one feature is sufficient for accurate prediction (i.e., single-index data model). Damian et al. (2022) considers multiple-index with low-degree polynomials as labeling functions and shows that a one-step gradient update can learn multiple features that lead to accurate prediction. Ba et al. (2022) studies one gradient step feature improvements under different learning rates. Radhakrishnan et al. (2023) proposes Recursive Feature Machines trying to show the mechanism of recursively feature learning but without giving a final loss guarantee. These studies consider specific problems and exploit the properties of the data to analyze the gradient in a delicate way, while our work provides a general framework applicable to different problems.

## 3.3 Gradient Feature Learning Framework

**Problem Setup.** Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the input space, $\mathcal{Y} \subseteq \mathbb{R}$ the label space. Let $\mathcal{D}$ be an arbitrary data distribution over $\mathcal{X} \times \mathcal{Y}$. Denote the class of two-layer networks with $m$ neurons as:

$$\mathcal{F}_{d,m} := \left\{ g_{(\mathbf{a},\mathbf{W},\mathbf{b})} \;\middle|\; g_{(\mathbf{a},\mathbf{W},\mathbf{b})}(\mathbf{x}) := \mathbf{a}^\top \left[\sigma(\mathbf{W}^\top \mathbf{x} - \mathbf{b})\right] = \sum_{i \in [m]} \mathbf{a}_i \left[\sigma(\langle \mathbf{w}_i, \mathbf{x} \rangle - \mathbf{b}_i)\right] \right\}, \tag{3.1}$$

where $\sigma(z) = \max(z, 0)$ is the ReLU activation function, $\mathbf{a} \in \mathbb{R}^m$ is the second layer weight, $\mathbf{W} \in \mathbb{R}^{d \times m}$ is the first layer weight, $\mathbf{w}_i$ is the $i$-th column of $\mathbf{W}$ (i.e., the weight for the $i$-th neuron), and $\mathbf{b} \in \mathbb{R}^m$ is the bias for the neurons. For technical simplicity, we only train $\mathbf{a}, \mathbf{W}$ but not $\mathbf{b}$. Let superscript $(t)$ denote the time step, e.g., $g_{(\mathbf{a}^{(t)},\mathbf{W}^{(t)},\mathbf{b})}$ denote the network at time step $t$. Denote $\Xi := (\mathbf{a}, \mathbf{W}, \mathbf{b})$, $\Xi^{(t)} := (\mathbf{a}^{(t)}, \mathbf{W}^{(t)}, \mathbf{b})$.

The goal of neural network learning is to minimize the expected risk, i.e., $\mathcal{L}_\mathcal{D}(g) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \mathcal{L}_{(\mathbf{x},y)}(g)$, where $\mathcal{L}_{(\mathbf{x},y)}(g) = \ell(y g(\mathbf{x}))$ is the loss on an example $(\mathbf{x}, y)$ for some loss function $\ell(\cdot)$, e.g., the hinge loss $\ell(z) = \max\{0, 1-z\}$, and the logistic loss $\ell(z) = \log[1 + \exp(-z)]$. We also consider $\ell_2$ regularization. The regularized loss with regularization coefficient $\lambda$ is $\mathcal{L}^{\lambda}_{\mathcal{D}}(g) := \mathcal{L}_{\mathcal{D}}(g) + \frac{\lambda}{2}(\|\mathbf{W}\|_F^2 + \|a\|_2^2)$. Given a training set with $n$ i.i.d. samples $\mathcal{Z} = \{(\mathbf{x}^{(l)}, y^{(l)})\}_{l\in[n]}$ from $\mathcal{D}$, the empirical risk and its regularized version are:

$$\widetilde{\mathcal{L}}_{\mathcal{Z}}(g) : = \frac{1}{n}\sum_{l\in[n]} \mathcal{L}_{(\mathbf{x}^{(l)},y^{(l)})}(g), \qquad \widetilde{\mathcal{L}}^{\lambda}_{\mathcal{Z}}(g) := \widetilde{\mathcal{L}}_{\mathcal{Z}}(g) + \frac{\lambda}{2}(\|\mathbf{W}\|_F^2 + \|a\|_2^2). \tag{3.2}$$

Then the training process is summarized in Algorithm 1.

**Algorithm 1** Network Training via Gradient Descent

```
Initialize $(a^{(0)}, \mathbf{W}^{(0)}, b)$
for $t = 1$ to $T$ do
    Sample $\mathcal{Z}^{(t-1)} \sim \mathcal{D}^n$
    $a^{(t)} = a^{(t-1)} - \eta^{(t)} \nabla_a \widetilde{\mathcal{L}}^{\lambda^{(t)}}_{\mathcal{Z}^{(t-1)}}(g_{\Xi^{(t-1)}})$,    $\mathbf{W}^{(t)} = \mathbf{W}^{(t-1)} - \eta^{(t)} \nabla_{\mathbf{W}} \widetilde{\mathcal{L}}^{\lambda^{(t)}}_{\mathcal{Z}^{(t-1)}}(g_{\Xi^{(t-1)}})$
end for
```

In the whole paper, we need some natural assumptions about the data and the loss.

**Assumption 3.1.** *We assume* $\mathbb{E}[\|\mathbf{x}\|_2] \leqslant B_{x1}$, $\mathbb{E}[\|\mathbf{x}\|_2^2] \leqslant B_{x2}$, $\|\mathbf{x}\|_2 \leqslant B_x$ *and for any label* $y$*, we have* $|y| \leqslant 1$*. We assume the loss function* $\ell(\cdot)$ *is a 1-Lipschitz convex decreasing function, normalized* $\ell(0) = 1, |\ell'(0)| = \Theta(1)$*, and* $\ell(\infty) = 0$.

**Remark 3.2.** *The above are natural assumptions. Most input distributions have the bounded norms required, and the typical binary classification* $\mathcal{Y} = \{\pm 1\}$ *satisfies the requirement. Also, the most popular loss functions satisfy the assumption, e.g., the hinge loss and logistic loss.*

### 3.3.1 Warm Up: A Simple Setting with Frozen First Layer

To illustrate some high-level intuition, we first consider a simple setting where the first layer is frozen after one gradient update, i.e., no updates to $\mathbf{W}$ for $t \geqslant 2$ in Algorithm 1.

The first idea of our framework is to provide guarantees compared to the optimal in a family of networks. Here let us consider networks with specific weights for the first layer:

**Definition 3.3.** *For some fixed* $\mathbf{W} \in \mathbb{R}^{d \times m}, b \in \mathbb{R}^{d}$*, and a parameter* $B_{a2}$*, consider the following family of networks* $\mathcal{F}_{\mathbf{W},b,B_{a2}}$*, and the optimal approximation network loss in this family:*

$$\mathcal{F}_{\mathbf{W},b,B_{a2}} := \left\{ g_{(a,\mathbf{W},b)} \in \mathcal{F}_{d,m} \mid \|a\|_2 \leqslant B_{a2} \right\}, \qquad \mathrm{OPT}_{\mathbf{W},b,B_{a2}} := \min_{g \in \mathcal{F}_{\mathbf{W},b,B_{a2}}} \mathcal{L}_{\mathcal{D}}(f). \tag{3.3}$$

The second idea is to compare to networks using features from gradient descent. As an illustrative example, we now provide guarantees compared to networks with first layer weights $\mathbf{W}^{(1)}$ (i.e., the weights after the first gradient step):

**Theorem 3.4** (Simple Setting)**.** *Assume* $\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(f_{(a,\mathbf{W}^{(1)},b)}\right)$ *is* $L$*-smooth to* $a$*. Let* $\eta^{(t)} = \frac{1}{L}, \lambda^{(t)} = 0$*, for all* $t \in \{2, 3, \ldots, T\}$*. Training by Algorithm 1 with no updates for the first layer after the first gradient step, w.h.p., there exists* $t \in [T]$ *such that*

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) \\
\leqslant &\mathrm{OPT}_{\mathbf{W}^{(1)},b,B_{a2}} + O\left( \frac{L(\|a^{(1)}\|_2^2 + B_{a2}^2)}{T} + \sqrt{\frac{B_{a2}^2(\|\mathbf{W}^{(1)}\|_F^2 B_x^2 + \|b\|_2^2)}{n}} \right).
\end{aligned}$$

Intuitively, the theorem shows that if the weight $\mathbf{W}^{(1)}$ after one step gradient gives a good set of neurons in the sense that there exists a classifier on top of these neurons with low loss, then the network will learn to approximate this good classifier and achieve low loss. The proof is based on standard convex optimization and the Rademacher complexity (details in Section B.4.1).

Such an approach, while simple, has been used to obtain interesting results about network learning in existing work, which shows that $\mathbf{W}^{(1)}$ can indeed give good neurons due to the structure of the special problems considered (e.g., parities on uniform inputs Barak et al. (2022), or polynomials on a subspace Damian et al. (2022)). However, it is unclear whether such intuition can still yield useful guarantees for other problems. So for our purpose of building a general framework covering more prototypical problems, the challenge is what features from gradient descent should be considered so that the family of networks for comparison can achieve a low loss on other problems. The other challenge is that we would like to consider the typical case where the first layer weights are not frozen. In the following, we will introduce the core concept of Gradient Features to address the first challenge, and stipulate proper geometric properties of Gradient Features for the second challenge.

### 3.3.2 Core Concepts in the Gradient Feature Learning Framework

Now, we will introduce the core concept in our framework, Gradient Features, and use it to build the family of networks to derive guarantees. As mentioned, we consider the setting where the first layer is not frozen. After the network learns good features, to ensure the updates in later gradient steps of the first layer are still benign for feature learning, we need some geometric conditions about the gradient features, which are measured by parameters in the definition of Gradient Features. The conditions are general enough so that, as shown in Section 3.4, many prototypical problems satisfy them and the induced family of networks enjoy low loss leading to useful guarantees.

We begin by considering what features can be learned via gradients. Note that the gradient w.r.t. $\mathbf{w}_i$ is $\frac{\partial \mathcal{L}_{\mathcal{D}}(g)}{\partial \mathbf{w}_i} = a_i \mathbb{E}_{(\mathbf{x},y)} \left[\ell'(yg(\mathbf{x}))y\left[\sigma'\left(\langle \mathbf{w}_i, \mathbf{x}\rangle - b_i\right)\right]\mathbf{x}\right] = a_i \mathbb{E}_{(\mathbf{x},y)} \left[\ell'(yg(\mathbf{x}))y\mathbf{x}\mathbb{I}[\langle \mathbf{w}_i, \mathbf{x}\rangle > b_i]\right]$. Inspired by this, we define the following notion:

**Definition 3.5** (Simplified Gradient Vector)**.** *For any* $\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, *a Simplified Gradient Vector is*

$$G(\mathbf{w}, b) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[y\mathbf{x}\mathbb{I}[\mathbf{w}^\top \mathbf{x} > b]]. \tag{3.4}$$

**Remark 3.6.** *Note that the definition of* $G(\mathbf{w}, b)$ *ignores the term* $\ell'(yg(\mathbf{x}))$ *in the gradient,* *where* $f$ *is the model function. In the early stage of training (or the first gradient step),* $\ell'(\cdot)$ *is approximately a constant, i.e.,* $\ell'(yg(\mathbf{x})) \approx \ell'(0)$ *due to the symmetric initialization (see Equation* (3.9)*).*

**Definition 3.7** (Gradient Feature)**.** *For a unit vector* $D \in \mathbb{R}^d$ *with* $\|D\|_2 = 1$*, and a* $\gamma \in (0, 1)$*, a direction neighborhood (cone)* $\mathcal{C}_{D,\gamma}$ *is defined as:*

$$\mathcal{C}_{D,\gamma} := \{\mathbf{w} \mid |\langle \mathbf{w}, D\rangle|/\|\mathbf{w}\|_2 > (1-\gamma)\}. \tag{3.5}$$

*Let* $\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$ *be random variables drawn from some distribution* $\mathcal{W}, \mathcal{B}$*. A Gradient Feature set with parameters* $p, \gamma, B_G$ *is defined as:*

$$S_{p,\gamma,B_G}(\mathcal{W}, \mathcal{B}) := \left\{(D, s) \;\Big|\; \Pr_{\mathbf{w},b}\left[G(\mathbf{w}, b) \in \mathcal{C}_{D,\gamma}\,, \|G(\mathbf{w}, b)\|_2 \geqslant B_G\,, s = \frac{b}{|b|}\right] \geqslant p\right\}. \tag{3.6}$$

*When clear from context, write it as* $S_{p,\gamma,B_G}$.

**Remark 3.8.** *The gradient features are simply the normalized vectors* $D$ *that are given (approximately) by the simplified gradient vectors. (Similarly, the normalized scalar* $s$ *is given by the bias* $b$*.) To be a useful gradient feature, we require the direction to be "hit" by sufficiently large simplified gradient vectors with sufficient large probability, so as to be distinguished from noise and remain useful throughout the gradient steps. Later we will use the gradient features when* $\mathcal{W}, \mathcal{B}$ *are the initialization distributions.*

To make use of the gradient features, we consider the following family of networks using these features and with bounded norms, and will provide guarantees compared to the best in this family:

**Definition 3.9** (Gradient Feature Induced Networks)**.** *The Gradient Feature Induced Networks are:*

$$\begin{aligned}&\mathcal{F}_{d,m,B_F,S} && (3.7)\\ :=&\left\{g_{(\mathbf{a},\mathbf{W},\mathbf{b})} \in \mathcal{F}_{d,m} \;\Big|\; \forall i \in [m],\ |a_i| \leqslant B_{a1}, \|a\|_2 \leqslant B_{a2}, (\mathbf{w}_i, b_i/|b_i|) \in S,\ |b_i| \leqslant B_b\right\},\end{aligned}$$

*where* $S$ *is some Gradient Feature set and* $B_F := (B_{a1}, B_{a2}, B_b)$ *are some parameters.*

**Remark 3.10.** *In the Gradient Feature Induced Networks, the weight and the bias of a neuron are simply the scalings of some item in the feature set* $S$ *(for simplicity the scaling of* $\mathbf{w}_i$ *is absorbed into the scaling of* $a_i$ *and* $b_i$*).*

**Definition 3.11** (Optimal Approximation via Gradient Features)**.** *The optimal approximation network and loss using Gradient Feature Induced Networks* $\mathcal{F}_{d,r,B_F,S}$ *are defined as:*

$$g^* := \operatorname*{arg\,min}_{g \in \mathcal{F}_{d,r,B_F,S}} \mathcal{L}_{\mathcal{D}}(f), \qquad \mathrm{OPT}_{d,r,B_F,S} := \min_{g \in \mathcal{F}_{d,r,B_F,S}} \mathcal{L}_{\mathcal{D}}(f). \tag{3.8}$$

### 3.3.3 Provable Guarantee via Gradient Feature Learning

To obtain the guarantees, we first specify the symmetric initialization. It is convenient for the analysis and is typical in existing analysis (e.g., Daniely and Malach (2020); Damian et al. (2022); Allen-Zhu and Li (2022); Shi et al. (2022c)), though some other initialization can also work. Formally, we train a two-layer network with $4m$ neurons, $g_{(a,\mathbf{W},b)} \in \mathcal{F}_{d,4m}$. We initialize $a_i^{(0)}, \mathbf{w}_i^{(0)}$ from Gaussians and $b_i$ from a constant for $i \in \{1, \dots, m\}$, and initialize the parameters for $i \in \{m+1, \dots, 4m\}$ accordingly to get a zero-output initial network. Specifically:

$$\begin{aligned}
&\text{for } i \in \{1, \dots, m\}: \quad a_i^{(0)} \sim \mathcal{N}(0, \sigma_a^2), \mathbf{w}_i^{(0)} \sim \mathcal{N}(0, \sigma_\mathbf{w}^2 I), b_i = \tilde{b},\\
&\text{for } i \in \{m+1, \dots, 2m\}: \quad a_i^{(0)} = -a_{i-m}^{(0)}, \mathbf{w}_i^{(0)} = -\mathbf{w}_{i-m}^{(0)}, b_i = -b_{i-m},\\
&\text{for } i \in \{2m+1, \dots, 4m\}: \quad a_i^{(0)} = -a_{i-2m}^{(0)}, \mathbf{w}_i^{(0)} = \mathbf{w}_{i-2m}^{(0)}, b_i = b_{i-2m}
\end{aligned} \tag{3.9}$$

where $\sigma_a^2, \sigma_\mathbf{w}^2, \tilde{b} > 0$ are hyper-parameters. After initialization, $a, \mathbf{W}$ are updated as in Algorithm 1.

We are now ready to present our main result in the framework.

**Theorem 3.12** (Main Result). *Assume Assumption 3.1. For any* $\epsilon, \delta \in (0,1)$*, if* $m \leqslant e^d$ *and*

$$m = \Omega\left(\frac{1}{p\epsilon^4}\left(rB_{a1}B_{x1}\sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{3.10}$$

$$T = \Omega\left(\frac{1}{\epsilon}\left(\frac{\sqrt{r}B_{a2}B_bB_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m}}{\sqrt{B_bB_G}} + \frac{1}{B_{x1}(mp)^{\frac{1}{4}}}\right)\right), \tag{3.11}$$

$$n = \Omega\left(\left(\frac{mB_xB_{a2}^2\sqrt{B_b}(mp)^{\frac{1}{2}}\log m}{\epsilon rB_{a1}\sqrt{B_G}}\right)^3 + \left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G})\sqrt{B_{x2}}\right)^3\right), \tag{3.12}$$

*then with initialization* (3.9) *and proper hyper-parameter values, we have with probability* $\geqslant 1 - \delta$ *over the initialization and training samples, there exists* $t \in [T]$ *in Algorithm 1 with:*

$$\begin{aligned}\Pr[\text{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant & \mathcal{L}_{\mathcal{D}}(g_{\Xi^{(t)}}) \\ \leqslant & \text{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G n^{\frac{1}{3}}}\right)} + \epsilon.\end{aligned} \tag{3.13}$$

Intuitively, the theorem shows when a data distribution admits a small approximation error by some "ground-truth" network with $r$ neurons using gradient features from $S_{p,\gamma,B_G}$ (i.e., a small optimal approximate loss $\text{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}$), the gradient descent training can successfully learn good neural networks with sufficiently many $m$ neurons.

Now we discuss the requirements and error guarantee. Viewing boundedness parameters $B_{a1}, B_{x1}$ etc. as constants, then the number $m$ of neurons learned is roughly $\tilde{\Theta}\left(\frac{r^4}{p\epsilon^4}\right)$, a polynomial overparameterization compared to the "ground-truth" network. The proof shows that such an overparameterization is needed such that some neurons can capture the gradient features given by gradient descent. This is consistent with existing analysis about overparameterization network learning, and also consistent with existing empirical observations.

The error bound consists of three terms. The last term $\epsilon$ can be made arbitrarily small, while the other two depend on the concrete data distribution. Specifically, with larger $r$ and $\gamma$, the second term increases. While the first term (the optimal approximation loss) decreases, since larger $r$ means a larger "ground-truth" network family, and larger $\gamma$ means a larger Gradient Feature set $S_{p,\gamma,B_G}$. So there is a trade-off between these two terms. When we later apply the framework to concrete problems (e.g., mixtures of Gaussians, parity functions), we will show that depending on the specific data distribution, we can choose proper values for $r, \gamma$ to make the error small. This then leads to error guarantees for the concrete problems and demonstrates the unifying power of the framework.

**Proof Sketch.** The intuition in the proof of Theorem 3.12 is closely related to the notion of Gradient Features. First, the gradient descent will produce gradients that approximate features in $S_{p,\gamma,B_G}$. Then, the gradient descent update gives a good set of neurons, such that there exists an accurate classifier using these neurons with loss comparable to the optimal approximation loss. Finally, the training will learn to approximate the accurate classifier, resulting in the desired error guarantee. The complete proof is in Section B.4, including the proper values for hyper-parameters like $\eta^{(t)}$ in Theorem B.17. Below we briefly sketch the key ideas and omit the technical details.

We first show that a large subset of neurons has gradients at the first step as good features. (The claim can be extended to multiple steps. For simplicity, we follow existing work (e.g., Daniely and Malach (2020); Shi et al. (2022c)) and present only the first step.) Let $\nabla_i$ denote the gradient of the i-th neuron $\nabla_{\mathbf{w}_i}\mathcal{L}_{\mathcal{D}}(g_{\Xi^{(0)}})$. Denote the subset of neurons with nice gradients approximating feature $(D, s)$ as:

$$G_{(D,s),Nice} := \left\{ i \in [2m] : s = b_i/|b_i|, \langle \nabla_i, D \rangle > (1-\gamma) \|\nabla_i\|_2, \|\nabla_i\|_2 \geqslant \left| a_i^{(0)} \right| B_G \right\}. \tag{3.14}$$

**Lemma 3.13** (Feature Emergence). *For any* $r$ *size subset* $\{(D_1, s_1), \ldots, (D_r, s_r)\} \subseteq S_{p,\gamma,B_G}$*, with probability at least* $1 - re^{-\Theta(mp)}$*, for all* $j \in [r]$*, we have* $|G_{(D_j,s_j),Nice}| \geqslant \frac{mp}{4}$.

This is because $\nabla_i = \ell'(0)a_i^{(0)}\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right]\mathbf{x}\right] = \ell'(0)a_i^{(0)}G(\mathbf{w}_i^{(0)}, b_i)$. Now consider $s_j = +1$ (the case $-1$ is similar). Since $\mathbf{w}_i$ is initialized by Gaussians, by $\nabla_i$'s connection to Gradient Features, we can see that for all $i \in [m]$, $\Pr\left[i \in G_{(D_j,+1),Nice}\right] \geqslant \frac{p}{2}$. The lemma follows from concentration via a large enough $m$, i.e., sufficient overparameterization. The gradients then allow obtaining a set of neurons approximating the "ground-truth" network with comparable loss:

**Lemma 3.14** (Existence of Good Networks)**.** *For any* $\delta \in (0,1)$*, with proper hyper-parameter values, with probability at least* $1 - \delta$*, there is* $\tilde{a}$ *such that* $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$ *and* $g_{(\tilde{a},\mathbf{W}^{(1)},\mathbf{b})}(\mathbf{x}) = \sum_{i=1}^{4m} \tilde{a}_i\sigma\left(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle - b_i\right)$ *satisfies*

$$\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},\mathbf{b})}) \leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + \sqrt{2}rB_{a1}B_{x1}\left(\sqrt{\gamma} + \sqrt{\frac{2B_b}{\sqrt{mp}B_G}}\right).$$

Given the good set of neurons, we finally show that the remaining gradient steps can learn an accurate classifier. Intuitively, with small step sizes $\eta^{(t)}$, the first layer's weights $\mathbf{w}_i$ do not change too much and thus the learning is similar to convex learning using the good set of neurons. Technically, we adopt the online convex optimization analysis in Daniely and Malach (2020) to get the final loss guarantee in Theorem 3.12.

## 3.4 Applications in Special Cases

In this section we will apply the gradient feature learning framework to some specific problems, corresponding to concrete data distributions $\mathcal{D}$. We primarily focus on prototypical problems for analyzing feature learning in networks. We will present here the results for mixtures of Gaussians and parity functions, and include the complete proofs and some other results in Section B.5.

### 3.4.1 Mixtures of Gaussians

Mixtures of Gaussians are among the most fundamental and widely used statistical models. Recently, it has been used to study neural network learning, in particular, the effect of gradient descent for feature learning of two-layer neural networks and the advantage over fixed feature methods Refinetti et al. (2021); Frei et al. (2022c).
**Data Distributions.** We follow notations from Refinetti et al. (2021). The data are from a mixture of $r$ high-dimensional Gaussians, and each Gaussian is assigned to one of two possible labels in $\mathcal{Y} = \{\pm 1\}$. Let $\mathcal{S}(y) \subseteq [r]$ denote the set of indices of Gaussians associated with the label $y$. The data distribution is then: $q(\mathbf{x}, y) = q(y)q(\mathbf{x}|y), q(\mathbf{x}|y) = \sum_{j\in\mathcal{S}(y)} p_j\mathcal{N}_j(\mathbf{x})$, where $\mathcal{N}_j(\mathbf{x})$ is a multivariate normal distribution with mean $\mu_j$, covariance $\Sigma_j$, and $p_j$ are chosen such that $q(\mathbf{x}, y)$ is correctly normalized. We will make some assumptions about the Gaussians, for which we first introduce some notations.

$$D_j := \frac{\mu_j}{\|\mu_j\|_2}, \quad \tilde{\mu}_j := \mu_j/\sqrt{d}, \quad B_{\mu 1} := \min_{j\in[r]} \|\tilde{\mu}_j\|_2, \quad B_{\mu 2} := \max_{j\in[r]} \|\tilde{\mu}_j\|_2, \quad p_B := \min_{j\in[r]} p_j.$$

**Assumption 3.15.** *Let* $8 \leqslant \tau \leqslant d$ *be a parameter that will control our final error guarantee. Assume*

- *Equiprobable labels:* $q(-1) = q(+1) = 1/2$.
- *For all* $j \in [r]$, $\Sigma_j = \sigma_j I_{d\times d}$. *Let* $\sigma_B := \max_{j\in[r]} \sigma_j$ *and* $\sigma_{B+} := \max\{\sigma_B, B_{\mu 2}\}$.
- $r \leqslant 2d, \quad p_B \geqslant \frac{1}{2d}, \quad \Omega\left(1/d + \sqrt{\tau\sigma_{B+}{}^2 \log d/d}\right) \leqslant B_{\mu 1} \leqslant B_{\mu 2} \leqslant d.$
- *The Gaussians are well-separated: for all* $i \neq j \in [r]$, *we have* $-1 \leqslant \langle D_i, D_j\rangle \leqslant \theta$, *where* $0 \leqslant \theta \leqslant \min\left\{\frac{1}{2r}, \frac{\sigma_{B+}}{B_{\mu 2}}\sqrt{\frac{\tau \log d}{d}}\right\}$.

**Remark 3.16.** *The first two assumptions are for simplicity; they can be relaxed. We can generalize our analysis to the mixture of Gaussians with unbalanced label probabilities and general covariances. The third assumption is to make sure that each Gaussian has a good* *amount of probability mass to be learned. The remaining assumptions are to make sure that the Gaussians are well-separated and can be distinguished by the learning algorithm.*

We are now ready to apply the framework to these data distributions, for which we only need to compute the Gradient Feature set and the corresponding optimal approximation loss.

**Lemma 3.17** (Mixtures of Gaussians: Gradient Features)**.** $(D_j, +1) \in S_{p,\gamma,B_G}$ *for all* $j \in [r]$, *where*

$$p = \frac{B_{\mu 1}}{\sqrt{\tau \log d}\sigma_{B+} \cdot d^{\Theta\left(\tau {\sigma_{B+}}^2 / B_{\mu 1}^2\right)}}, \quad \gamma = \frac{1}{d^{0.9\tau - 1.5}}, \quad B_G = p_B B_{\mu 1}\sqrt{d} - O\left(\frac{\sigma_{B+}}{d^{0.9\tau}}\right).$$

*Let* $g^*(\mathbf{x}) = \sum_{j=1}^{r} \frac{y_{(j)}}{\sqrt{\tau \log d}\sigma_{B+}} \left[\sigma\left(\langle D_j, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+}\right)\right]$ *whose hinge loss is at most* $\frac{3}{d^{\tau}} + \frac{4}{d^{0.9\tau - 1}\sqrt{\tau \log d}}$.

Given the values on gradient feature parameters $p, \gamma, B_G$ and the optimal approximation loss $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}$, the framework immediately leads to the following guarantee:

**Theorem 3.18** (Mixtures of Gaussians: Main Result)**.** *Assume Assumption 3.15. For any* $\epsilon, \delta \in (0, 1)$, *when Algorithm 1 uses hinge loss with*

$$m = \mathrm{poly}\left(\frac{1}{\delta}, \frac{1}{\epsilon}, d^{\Theta\left(\tau {\sigma_{B+}}^2 / B_{\mu 1}^2\right)}, r, \frac{1}{p_B}\right) \leqslant e^d, \quad T = \mathrm{poly}(m), \quad n = \mathrm{poly}(m)$$

*and proper hyper-parameters, then with probability at least* $1 - \delta$, *there exists* $t \in [T]$ *such that* $\Pr[\mathrm{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \frac{\sqrt{2}r}{d^{0.4\tau - 0.8}} + \epsilon$.

The theorem shows that gradient descent can learn to a small error via learning the gradient features, given proper hyper-parameters. In particular, we need sufficient overparameterization (a sufficiently large number $m$ of neurons). When ${\sigma_{B+}}^2 / B_{\mu 1}^2$ is a constant which is the prototypical interesting case, and we choose a constant $\tau$, then $m$ is polynomial in the key parameters $\frac{1}{\delta}, \frac{1}{\epsilon}, d, r, \frac{1}{p_B}$, and the error bound is inverse polynomial in $d$. The complete proof is given in Section B.5.2, with the concrete values of the working hyper-parameters in Theorem B.35.

Frei et al. (2022c) studies (almost) linear separable cases while our setting includes non-linear separable cases, e.g., XOR. Refinetti et al. (2021) mainly studies neural network classification on 4 Gaussian clusters with XOR structured labels, while our setting is much more general, e.g., our cluster number can extend up to $2d$.

### Mixtures of Gaussians: Beyond the Kernel Regime

As discussed in the introduction, it is important for the analysis to go beyond fixed feature methods such as NTK (i.e., the kernel regime), so as to capture the feature learning ability which is believed to be the key factor for the empirical success. We first review the fixed feature methods. Following Daniely and Malach (2020), suppose $\Psi$ is a data-independent feature mapping of dimension $N$ with bounded features, i.e., $\Psi : \mathbf{X} \to [-1, 1]^N$. For $B > 0$, the family of linear models on $\Psi$ with bounded norm $B$ is $\mathcal{H}_B = \{h(\tilde{x}) : h(\tilde{x}) = \langle \Psi(\tilde{x}), w \rangle, \|w\|_2 \leqslant B\}$. This can capture linear models on fixed finite-dimensional feature maps, e.g., NTK, and also infinite dimensional feature maps, e.g., kernels like RBF, that can be approximated by feature maps of polynomial dimensions Rahimi and Recht (2008); Kamath et al. (2020); Shi et al. (2022c).

Our framework indeed goes beyond fixed features. Our framework can show the advantage of network learning over kernel methods under the setting of Refinetti et al. (2021) (4 Gaussian clusters with XOR structured labels). For large enough $d$, our framework only needs roughly $\Omega$ (log $d$) neurons and $\tilde{\Omega}(d^{1.5})$ samples to achieve arbitrary small constant error (see Theorem B.42), while fixed feature methods need $\Omega(d^2)$ features and $\Omega(d^2)$ samples to achieve nontrivial errors (as proved in Refinetti et al. (2021)). For the proof (detailed in Section B.5.3), we only need to calculate the $p, \gamma, B_G$ of the data distribution and then inject these numbers into Theorem 3.12.

### 3.4.2 Parity Functions

Parity functions are a canonical family of learning problems in computational learning theory, usually for showing theoretical computational barriers Shalev-Shwartz et al. (2017). The typical sparse parties over d-dim binary inputs $\phi \in \{\pm 1\}^d$ are $\prod_{i \in \mathbf{A}} \phi_i$ where $\mathbf{A} \subseteq [d]$ is a subset of dimensions. Recent studies have shown that when the distribution of inputs $\phi$ have structures rather than uniform, neural networks can perform feature learning and finally learn parity functions to a small error, while methods without feature learning, e.g., NTK, cannot achieve as good results Daniely and Malach (2020); Malach et al. (2021); Shi et al. (2022c). This thus has been a prototypical setting for studying the feature learning phenomena in networks. Here we consider a generalization of this problem and show that our framework can show successful learning via gradient descent.

**Data Distributions.** Suppose $M \in \mathbb{R}^{d \times D}$ is an unknown dictionary with D columns that can be regarded as patterns. For simplicity, assume $d = D$ and M is orthonormal. Let $\phi \in \mathbb{R}^d$ be a hidden representation vector. Let $\mathbf{A} \subseteq [D]$ be a subset of size rk corresponding to the class relevant patterns and r is an odd number. Then the input is generated by $M\phi$, and some function on $\phi_{\mathbf{A}}$ generates the label. WLOG, let $\mathbf{A} = \{1, \dots, rk\}$, $\mathbf{A}^\perp = \{rk+1, \dots, d\}$. Also, we split $\mathbf{A}$ such that for all $j \in [r]$, $\mathbf{A}_j = \{(j-1)k+1, \dots, jk\}$. Then the input $\mathbf{x}$ and the class label $y$ are given by: $\mathbf{x} = M\phi, y = g^*(\phi_{\mathbf{A}}) = \text{sign}\Big(\sum_{j \in [r]} \text{XOR}(\phi_{\mathbf{A}_j})\Big)$, where $g^*$ is the ground-truth labeling function mapping from $\mathbb{R}^{rk}$ to $\mathcal{Y} = \{\pm 1\}$, $\phi_{\mathbf{A}}$ is the sub-vector of $\phi$ with indices in $\mathbf{A}$, and $\text{XOR}(\phi_{\mathbf{A}_j}) = \prod_{l \in \mathbf{A}_j} \phi_l$ is the parity function. We still need to specify the distribution $\mathbf{X}$ of $\phi$, which determines the structure of the input distribution: $\mathbf{X} := (1 - 2rp_A)\mathbf{X}_U + \sum_{j \in [r]} p_A(\mathbf{X}_{j,+} + \mathbf{X}_{j,-})$. For all corresponding $\phi_{\mathbf{A}^\perp}$ in $\mathbf{X}$, we have $\forall l \in \mathbf{A}^\perp$, independently: $\phi_l = \begin{cases} +1, & \text{w.p. } p_o \\ -1, & \text{w.p. } p_o \\ 0, & \text{w.p. } 1 - 2p_o \end{cases}$, where $p_o$ controls the signal noise ratio: if $p_o$ is large, then there are many nonzero entries in $\mathbf{A}^\perp$ which are noise interfering with the learning of the ground-truth labeling function on $\mathbf{A}$. For corresponding $\phi_{\mathbf{A}}$, any $j \in [r]$, we have

- In $\mathbf{X}_{j,+}$, $\phi_{\mathbf{A}_j} = [+1, +1, \dots, +1]^\top$ and $\phi_{\mathbf{A}\setminus\mathbf{A}_j}$ only have zero elements.
- In $\mathbf{X}_{j,-}$, $\phi_{\mathbf{A}_j} = [-1, -1, \dots, -1]^\top$ and $\phi_{\mathbf{A}\setminus\mathbf{A}_j}$ only have zero elements.
- In $\mathbf{X}_U$, we have $\phi_{\mathbf{A}}$ draw from $\{+1, -1\}^{rk}$ uniformly.

**Assumption 3.19.** *Let $8 \leqslant \tau \leqslant d$ be a parameter that will control our final error guarantee. Assume $k$ is an odd number and: $k \geqslant \Omega(\tau \log d)$, $\quad d \geqslant rk + \Omega(\tau r \log d)$, $\quad p_o = O\left(\frac{rk}{d-rk}\right)$, $\quad p_A \geqslant \frac{1}{d}$.*

**Remark 3.20.** *We set up the problem to be more general than the parity function learning in existing work. If $r = 1$, the labeling function reduces to the traditional $k$-sparse parties of $d$ bits. The assumptions require $k$, $d$, and $p_A$ to be sufficiently large so as to provide enough large signals for learning. Note that when $k = \frac{d}{16}, r = 1, p_o = \frac{1}{2}$, our analysis also holds, which shows our framework is beyond the kernel regime (discuss in detail in Section 3.4.2).*

To apply our framework, again we only need to compute the Gradient Feature set and the corresponding optimal loss. We first define the Gradient Features: For all $j \in [r]$, let $D_j = \frac{\sum_{l \in \mathbf{A}_j} M_l}{\|\sum_{l \in \mathbf{A}_j} M_l\|_2}$.

**Lemma 3.21** (Parity Functions: Gradient Features)**.** *We have $(D_j, +1), (D_j, -1) \in S_{p,\gamma,B_G}$ for all $j \in [r]$, where*

$$p = \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\Theta(\tau r)}}\right), \qquad \gamma = \frac{1}{d^{\tau-2}}, \qquad B_G = \sqrt{k} p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right). \tag{3.15}$$

*With gradient features from $S_{p,\gamma,B_G}$, let*

$$g^*(\mathbf{x}) = \sum_{j=1}^{r} \sum_{i=0}^{k} (-1)^{i+1} \sqrt{k} \Big[ \sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k-1}{\sqrt{k}}\right) - 2\sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k}{\sqrt{k}}\right) + \sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k+1}{\sqrt{k}}\right) \Big]$$

*whose hinge loss is 0.*

Given the values on gradient feature parameters and the optimal approximation loss, the framework immediately leads to the following guarantee:

**Theorem 3.22** (Parity Functions: Main Result)**.** *Assume Assumption 3.19. For any* $\epsilon, \delta \in (0,1)$*, when Algorithm 1 uses hinge loss with*

$$m = \text{poly}\left(\frac{1}{\delta}, \frac{1}{\epsilon}, d^{\Theta(\tau r)}, k, \frac{1}{p_A}\right) \leqslant e^d, \quad T = \text{poly}\left(m\right), \quad n = \text{poly}\left(m\right)$$

*and proper hyper-parameters, then with probability at least* $1-\delta$*, there exists* $t \in [T]$ *such that* $\Pr[\text{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \frac{3r\sqrt{k}}{d^{(\tau-3)/2}} + \epsilon$.

The theorem shows that gradient descent can learn to a small error in this problem. We also need sufficient overparameterization: When $r$ is a constant (e.g., $r = 1$ in existing work), and we choose a constant $\tau$, $m$ is polynomial in $\frac{1}{\delta}, \frac{1}{\epsilon}, d, k, \frac{1}{p_A}$, and the error bound is inverse polynomial in $d$. The proof is in Section B.5.4, with the concrete values of the working hyper-parameters in Theorem B.52.

Our setting is more general than that in Daniely and Malach (2020); Malach et al. (2021) which corresponds to $M = I, r = 1, p_A = \frac{1}{4}, p_o = \frac{1}{2}$. Shi et al. (2022c) study single index learning, where one feature direction is enough for a two-layer network to recover the label, while our setting considers $r$ directions $D_1, \ldots, D_r$, so the network needs to learn multiple directions to get a small error.

### Parity Functions: Beyond the Kernel Regime

Again, we show that our framework indeed goes beyond fixed features under parity functions. Our problem setting in Section 3.4.2 is general enough to include the problem setting in Daniely and Malach (2020). Their lower bound for fixed feature methods directly applies to our case and leads to the following:

**Proposition 3.23.** *There exists a data distribution in the parity learning setting in Section 3.4.2 with* $M = I, r = 1, p_A = \frac{1}{4}, k = \frac{d}{16}, p_o = \frac{1}{2}$*, such that all* $h \in \mathcal{H}_B$ *have hinge-loss at least* $\frac{1}{2} - \frac{\sqrt{N}B}{2^k\sqrt{2}}$.

This means to get an inverse-polynomially small loss, fixed feature models need to have an exponentially large size, i.e., either the number of features $N$ or the norm $B$ needs to be exponential in $k$. In contrast, Theorem 3.22 shows our gradient feature learning framework guarantees a small loss with a polynomially large model, runtime, and sample complexity. Clearly, our framework is beyond the fixed feature methods.

**Parities on Uniform Inputs.** When $r = 1, p_A = 0$, our problem setting will degenerate to the classic sparse parity function on a uniform input distribution. This has also been used for analyzing network learning Barak et al. (2022). For this case, our framework can get a $k2^{O(k)} \log(k)$ network width bound and a $O(d^k)$ sample complexity bound, matching those in Barak et al. (2022). This then again confirms the advantage of network learning over kernel methods that requires $d^{\Omega(k)}$ dimensions as shown in Barak et al. (2022). See the full statement in Theorem B.56 and details in Section B.5.5.

## 3.5 Conclusion

This work proposed a general framework for analyzing two-layer neural network learning by gradient descent and showed that it can lead to provable guarantees for several prototypical problem settings for analyzing network learning. In particular, our framework goes beyond fixed feature methods, e.g., NTK. It also sheds some light on several interesting phenomena in network learning, e.g., simplicity bias and the lottery ticket hypothesis. Future directions include: (1) How to extend the framework to deeper networks? (2) While the current framework focuses on the gradient features in the early gradient steps, whether feature learning also happens in later steps and if so how to formalize that?

# 4 FOURIER CIRCUITS IN NEURAL NETWORKS AND TRANSFORMERS: A CASE STUDY OF MODULAR ARITHMETIC WITH MULTIPLE INPUTS

**Contribution statement.** This chapter is joint work with Chenyang Li, Yingyu Liang, Zhao Song, and Tianyi Zhou. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis, and completed all the experiments. The results in this chapter have appeared in ICLR 2024 Workshop, and then is submitted to AIStats 2025 (Li et al., 2024b).

## 4.1 Introduction

The field of artificial intelligence has experienced a significant transformation with the development of large language models (LLMs), particularly through the introduction of the Transformer architecture (Vaswani et al., 2017). This advancement has revolutionized approaches to challenging tasks in natural language processing, notably in machine translation (Prato et al., 2020; Gao et al., 2020) and text generation (Luo et al., 2022). Consequently, models e.g., Mistral (Jiang et al., 2023a), Llama (AI, 2024), Gemini (Team et al., 2023), Gemma (Team et al., 2024), Claude3 (Anthropic, 2024), GPT4 (Achiam et al., 2023) and so on, have become predominant in NLP.

Central to this study is the question of how these advanced models transcend mere pattern recognition to engage in what appears to be logical reasoning and problem-solving. This inquiry is not purely academic; it probes the core of "understanding" in artificial intelligence. While LLMs, such as Claude3 and GPT4, demonstrate remarkable proficiency in human-like text generation, their capability to comprehend and process mathematical logic is a topic of considerable debate. This line of investigation is crucial, given AI's potential to extend beyond text generation into deeper comprehension of complex subjects. Mathematics, often seen as the universal language, presents a uniquely challenging domain for these models (Yousefzadeh and Cao, 2023). Our research aims to determine whether Transformers with attention, noted for their NLP efficiency, can also demonstrate an intrinsic understanding of mathematical operations and reasoning.

In a recent surprising study of mathematical operations learning, Power et al. (2022) train Transformers on small algorithmic datasets, e.g., $a_1 + a_2 \mod p$ and we let $p$ be a prime number, and show the "grokking" phenomenon, where models abruptly transition from bad generalization to perfect generalization after a large number of training steps. Nascent studies, such as those by Nanda et al. (2023a), empirically reveal that Transformers can solve modular addition using Fourier-based circuits. They found that the Transformers trained by Stochastic Gradient Descent (SGD) not only reliably compute $a_1 + a_2 \mod p$, but also that the networks consistently employ a specific geometric algorithm. This algorithm, which involves composing integer rotations around a circle, indicates an inherent comprehension of modular arithmetic within the network's architecture. The algorithm relies on this identity: for any $a_1, a_2$ and $\zeta \in \mathbb{Z}_p \backslash \{0\}$, the following two quantities are equivalent

$$(a_1 + a_2) \text{ mod } p = \arg\max_{c \in \mathbb{Z}_p} \{\cos(2\pi\zeta(a_1 + a_2 - c)/p)\}.$$

Nanda et al. (2023a) further show that the attention and MLP module in the Transformer imbues the neurons with Fourier circuit-like properties. To study why networks arrive at Fourier-based circuits computational strategies, Morwani et al. (2024) theoretically study one-hidden layer neural network learning on two inputs modular addition task and certify that the trained networks will execute modular addition by employing Fourier features aligning closely with the previous empirical observations. However, the question remains whether neural networks can solve more complicated mathematical problems.

Inspired by recent developments in mechanistic interpretability (Olah et al., 2020; Elhage et al., 2021, 2022) and the study of inductive biases (Soudry et al., 2018; Vardi, 2023) in neural networks, we extend our research to modular addition with more $(k)$ inputs.

$$(a_1 + a_2 + \cdots + a_k) \bmod p. \tag{4.1}$$

This approach offers insights into why certain representations and solutions emerge from neural network training. By integrating these insights with our empirical findings, we aim to provide a comprehensive understanding of neural networks' learning mechanisms, especially in solving the modular addition problem. We also determine the necessary number of neurons for the network to learn this Fourier method for modular addition. Our paper's contributions are summarized as follows:

- **Expansion of Input for Modular Addition Problem:** We extend the input parameter range for the modular addition problem from a binary set to $k$-element sets.

- **Network's Maximum Margin:** For $p$-modular addition of $k$ inputs, we give the closed form of the maximum margin of a network (Lemma 4.10):

$$\gamma^* = \frac{2(k!)}{(2k+2)^{(k+1)/2}(p-1)p^{(k-1)/2}}.$$

- **Neuron Count in One-Hidden-Layer Networks:** We propose that in a general case, a one-hidden-layer network having $m \geqslant 2^{2k-2} \cdot (p-1)$ neurons can achieve the maximum $L_{2,k+1}$-margin solution, each hidden neuron aligning with a specific Fourier spectrum. This ensures the network's capability to effectively solve the modular addition in a Fourier-based method (Theorem 4.9).

- **Empirical Validation of Theoretical Findings:** We validate our theoretical finding that: when $m \geqslant 2^{2k-2} \cdot (p-1)$, for each spectrum $\zeta \in \{1, \ldots, \frac{p-1}{2}\}$, there exists a hidden-neuron utilizes this spectrum. It strongly supports our analysis. (Figure 4.1 and Figure 4.2).

- **Similar Findings in Transformer:** We have a similar observation in one-layer Transformer learning modular addition involving $k$ inputs. For the 2-dimensional matrix $W_K W_Q$, where $W_K, W_Q$ denotes the key and query matrix, it shows the superposition of two cosine waveforms in each dimension, each characterized by distinct frequencies (Figure 4.3).
- **Grokking under Different** $k$: We observe that as $k$ increases, the grokking phenomenon becomes weaker, as predicted by our analysis (Figure 4.4).

## 4.2 Related Work

**Max Margin Solutions in Neural Networks.** Bronstein et al. (2022) demonstrated that neurons in a one-hidden-layer ReLU network align with clauses in max margin solutions for read-once DNFs, employing a unique proof technique involving the construction of perturbed networks. Morwani et al. (2024) utilize max-min duality to certify maximum-margin solutions. Further, extensive research in the domain of margin maximization in neural networks, including works by Gunasekar et al. (2018b); Soudry et al. (2018); Gunasekar et al. (2018a); Wei et al. (2019); Lyu and Li (2019); Ji and Telgarsky (2019a); Moroshko et al. (2020); Chizat and Bach (2020); Ji and Telgarsky (2020); Lyu et al. (2021); Frei et al. (2022c, 2023a); Shi et al. (2023c); Li et al. (2024a) and more, has highlighted the implicit bias towards margin maximization inherent in neural network optimization. They provide a foundational understanding of the dynamics of neural networks and their inclination towards maximizing margins under various conditions and architectures.

**Algebraic Tasks Learning Mechanism Interpretability.** The study of neural networks trained on algebraic tasks has been pivotal in shedding light on their training dynamics and inductive biases. Notable contributions include the work of Power et al. (2022); Gromov (2023); Quirke and Barez (2023) on modular addition and subsequent follow-up studies, investigations into learning parities (Daniely and Malach, 2020; Barak et al., 2022; Shi et al., 2022c, 2024b, 2023d,b; Zhang et al., 2023d; Xu et al., 2024e), and research into algorithmic reasoning capabilities (Saxton et al., 2018; Hendrycks et al., 2021; Lewkowycz et al., 2022; Meng et al., 2022a; Damian et al., 2022; Chughtai et al., 2023; Stander et al., 2023; Nanda et al., 2023b; Zhong et al., 2023; Tigges et al., 2023; Hanna et al., 2023). The field of mechanistic interpretability, focusing on the analysis of internal representations in neural networks, has also seen significant advancements through the works of Cammarata et al. (2020); Olsson et al. (2022); Merrill et al. (2023); Rubin et al. (2023); Varma et al. (2023); Doshi et al. (2024); Shi et al. (2024a); Ke et al. (2024a); Chen et al. (2024b); Liang et al. (2024c); Saxena et al. (2024) and others.

**Grokking and Emergent Ability.** The phenomenon known as "grokking" was initially identified by Power et al. (2022) and is believed to be a way of studying the emerging abilities of LLM (Wei et al., 2022c). This research observed a unique trend in two-layer transformer models engaged in algorithmic tasks, where there was a significant increase in test accuracy, surprisingly occurring well after these models had reached perfect accuracy in their training phase. In Millidge (2022), it was hypothesized that this might be the result of the SGD process that resembles a random path along what is termed the optimal manifold. Adding to this, Nanda et al. (2023a) aligns with the findings of Belinkov (2022), indicating a steady advancement of networks towards algorithms that are better at generalization. Liu et al. (2022); Xu et al. (2024d); Lyu et al. (2024) developed smaller-scale examples of grokking and utilized these to map out phase diagrams, delineating multiple distinct learning stages. Furthermore, Thilak et al. (2022); Murty et al. (2023) suggested the possibility of grokking occurring naturally, even in the absence of explicit regularization. They attributed this to an optimization quirk they termed the slingshot mechanism, which might inadvertently act as a regularizing factor.

## 4.3 Problem Setup

### 4.3.1 Data and Network Setup

**Data.** Following Morwani et al. (2024), let $\mathbb{Z}_p = [p]$ denote the modular group on $p$ integers, where $p > 2$ is a given prime number. The input space is $\mathcal{X} := \mathbb{Z}_p^k$

for some integer $k$, and the output space is $\mathcal{Y} := \mathbb{Z}_p$. Then an input data point is $a = (a_1, \dots, a_k)$ with $a_i \in \mathbb{Z}_p$. When clear from context, we also let $x_i \in \{0,1\}^p$ be the one-hot encoding of $a_i$, and let $x = (x_1, \dots, x_k)$ denote the input point.

**Network.** We consider single-hidden layer neural networks with polynomial activation functions:

$$\begin{aligned} f(\theta, x) &:= \sum_{i=1}^{m} \phi(\theta_i, x), \\ \phi(\theta_i, x) &:= (u_{i,1}^\top x_1 + \cdots + u_{i,k}^\top x_k)^k w_i, \end{aligned} \tag{4.2}$$

where $\theta := \{\theta_1, \dots, \theta_m\} \in \mathbb{R}^{(k+1)\times p}$, $\phi(\theta_i, x)$ is one neuron, and $\theta_i := \{u_{i,1}, \dots, u_{i,k}, w_i\}$ are the parameters of the neuron with $u_{i,1}, \dots, u_{i,k}, w_i \in \mathbb{R}^p$. We use polynomial activation functions due to the homogeneous requirement in Lemma 4.7 and easy sum-to-product identities calculation in Fourier analysis. Using the notation $a$ instead of the one-hot encodings $x$, we can also write:

$$\begin{aligned} f(\theta, a) &:= \sum_{i=1}^{m} \phi(\theta_i, a), \\ \phi(\theta_i, a) &:= (u_{i,1}(a_1) + \cdots + u_{i,k}(a_k))^k w_i, \end{aligned}$$

where with $u_{i,j}(a_j)$ being the $a_j$-th component of $u_{i,j}$. We consider the parameter set: $\Theta := \{\|\theta\|_{2,k+1} \leqslant 1\}$, where $\|\theta\|_{2,k+1} := (\sum_{i=1}^{m} \|\theta_i\|_2^{k+1})^{\frac{1}{k+1}}$, and $\|\theta_i\|_2 := (\sum_{j=1}^{k} \|u_{i,j}\|_2^2 + \|w_i\|_2^2)^{\frac{1}{2}}$.

Here $\|\theta\|_{2,k+1}$ is the $L_{2,k+1}$ matrix norm of $\theta$ (Definition C.2), and $\|\theta_i\|_2$ is the $L_2$ vector norm of the concatenated vector of the parameters in $\theta_i$. The training objective over $\Theta$ is then as follows.

**Definition 4.1.** *Given a dataset* $D_p$ *and the cross-entropy loss* $l$*, the regularized training objective is:*

$$\mathcal{L}_\lambda(\theta) := \frac{1}{|D_p|} \sum_{(x,y)\in D_p} l(f(\theta, x), y) + \lambda \|\theta\|_{2,k+1}.$$

### 4.3.2 Margins of the Neural Networks

Now, we define the margin for a data point and the margin for a whole dataset.

**Definition 4.2.** *We denote* $g : \mathbb{R}^u \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ *as the margin function, where for given* $(x, y) \in D_p$, $g(\theta, x, y) := f(\theta, x)[y] - \max_{y' \in \mathcal{Y} \setminus \{y\}} f(\theta, x)[y']$.

**Definition 4.3.** *The margin for a given dataset* $D_p$ *is denoted as* $h : \mathbb{R}^u \to \mathbb{R}$ *where* $h(\theta) := \min_{(x,y) \in D_p} g(\theta, x, y)$.

For parameter $\theta$, its normalized margin is denoted as $h(\theta / \|\theta\|_{2,k+1})$. For simplicity, we define $\gamma^*$ to be the maximum normalized margin as the following:

**Definition 4.4.** *The minimum of the regularized objective is denoted as* $\theta_\lambda \in \arg\min_{\theta \in \mathbb{R}^u} \mathcal{L}_\lambda(\theta)$. *We define the normalized margin of* $\theta_\lambda$ *as* $\gamma_\lambda := h(\theta_\lambda / \|\theta_\lambda\|_{2,k+1})$ *and the maximum normalized margin as* $\gamma^* := \max_{\theta \in \Theta} h(\theta)$, *where* $\Theta = \{\|\theta\|_{2,k+1} \leqslant 1\}$.

Let $\mathcal{P}(D_p)$ denote a set containing all distributions over $D_p$. Then $\gamma^*$ can be rewritten as

$$\begin{aligned}\gamma^* = \max_{\theta \in \Theta} h(\theta) &= \max_{\theta \in \Theta} \min_{(x,y) \in D_p} g(\theta, x, y) \\ &= \max_{\theta \in \Theta} \min_{q \in \mathcal{P}(D_p)} \mathop{\mathbb{E}}_{(x,y) \sim q} [g(\theta, x, y)], \qquad (4.3)\end{aligned}$$

where the first step is from Definition 4.4, the second step is from Definition 4.3, and the last step is from the linearity of the expectation. Now, we introduce an important concept of a duality stationary pair $(\theta^*, q^*)$.

**Definition 4.5.** *We define a stationary pair* $(\theta^*, q^*)$ *when satisfying*

$$\begin{aligned} q^* &\in \arg\min_{q \in \mathcal{P}(D_p)} \mathop{\mathbb{E}}_{(x,y) \sim q} [g(\theta^*, x, y)], \qquad (4.4) \\ \theta^* &\in \arg\min_{\theta \in \Theta} \mathop{\mathbb{E}}_{(x,y) \sim q^*} [g(\theta, x, y)]. \end{aligned}$$

This means that $q^*$ is a distribution that minimizes the expected margin based on $\theta^*$, and simultaneously, $\theta^*$ is a solution that maximizes the expected margin relative to $q^*$. The max-min inequality (Boyd and Vandenberghe, 2004) indicates that presenting such a duality adequately proves $\theta^*$ to be a maximum margin solution. Recall that there is a "max" operation in Definition 4.2, which makes the swapping of expectation and summation infeasible, meaning that the expected network margin cannot be broken down into the expected margins of individual neurons. To tackle this problem, the class-weighted margin is proposed, whose intuition is similar to label smoothing. Let $\tau : D_p \to \Delta(\mathcal{Y})$ allocate weights to incorrect labels for every data point. Given $(x, y)$ in $D_p$ and for any $y' \in \mathcal{Y}$, we have $\tau(x, y)[y'] \geqslant 0$ and $\sum_{y' \in \mathcal{Y} \setminus \{y\}} \tau(x, y)[y'] = 1$. We denote a proxy $g'$ as the following to solve the issue.

**Definition 4.6.** *Draw* $(x, y) \in D_p$. *The class-weighted margin* $g'$ *is defined as*

$$g'(\theta, x, y) := \ f(\theta, x)[y] - \sum_{y' \in \mathcal{Y} \setminus \{y\}} \tau(x, y)[y'] f(\theta, x)[y'].$$

We have $g'$ uses a weighted sum rather than max, so $g(\theta, x, y) \leqslant g'(\theta, x, y)$. Following the linearity of expectation, we get the expected class-weighted margin as

$$\begin{aligned} \mathop{\mathbb{E}}_{(x,y)}[g'(\theta, x, y)] = \sum_{i=1}^{m} \mathop{\mathbb{E}}_{(x,y)} \Big[ & \phi(\theta_i, x)[y] \\ & - \sum_{y' \in \mathcal{Y} \setminus \{y\}} \tau(x, y)[y'] \phi(\theta_i, x)[y'] \Big], \end{aligned}$$

where we can move the summation $\sum_{i=1}^{m}$ out of the expectation $\mathbb{E}[]$.

### 4.3.3 Connection between Training and the Maximum Margin Solutions

We denote $\nu$ as the network's homogeneity constant, where the equation $f(\alpha\theta, x) = \alpha^\nu f(\theta, x)$ holds for any $x$ and any scalar $\alpha > 0$. Specifically, we focus on networks with homogeneous neurons that satisfy $\phi(\alpha\theta_i, x) = \alpha^\nu \phi(\theta_i, x)$ for any $\alpha > 0$. Note

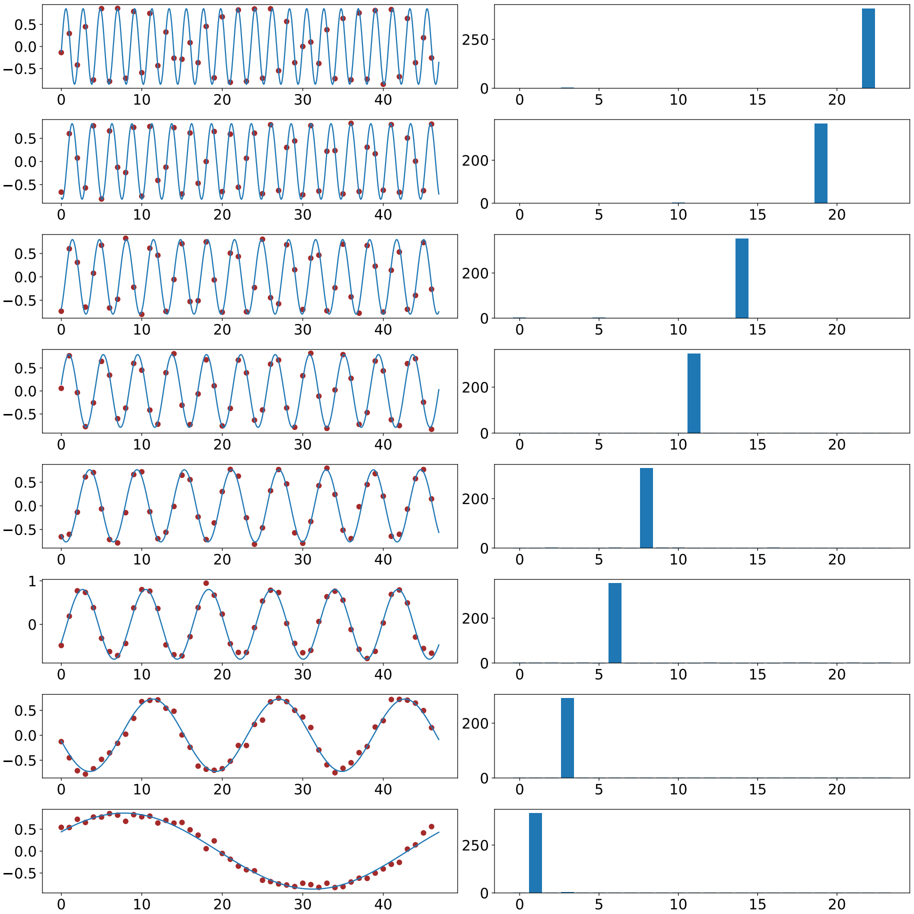


Figure 4.1: Cosine shape of the trained embeddings (hidden layer weights) and corresponding power of Fourier spectrum. The two-layer network with $m = 2944$ neurons is trained on $k = 4$-sum mod-$p = 47$ addition dataset. We even split the whole datasets ($p^k = 47^4$ data points) into the training and test datasets. Every row represents a random neuron from the network. The left figure shows the final trained embeddings, with red dots indicating the true weight values, and the pale blue interpolation is achieved by identifying the function that shares the same Fourier spectrum. The right figure shows their Fourier power spectrum. The results in these figures are consistent with our analysis statements in Lemma 4.10. See Figure C.1, C.3 in Appendix C.10.2 for similar results when $k$ is 3 or 5.

that our one-hidden layer networks (Eq. equation 4.2) are $k + 1$ homogeneous. As the following Lemma states, when $\lambda$ is small enough during training homogeneous functions, we have the $\mathcal{L}_\lambda$ global optimizers' normalized margin converges to $\gamma^*$.

**Lemma 4.7** (Wei et al. (2019), Theorem 4.1)**.** *Let* $f$ *be a homogeneous function. For any norm* $\|\cdot\|$*, if* $\gamma^* > 0$*, we have* $\lim_{\lambda\to 0}\gamma_\lambda = \gamma^*$*.*

Therefore, to comprehend the global minimize, we can explore the maximum-margin solution as a surrogate, enabling us to bypass complex analyses in non-convex optimization. Furthermore, Morwani et al. (2024) states that under the following condition, the maximum-margin solutions and class-weighted maximum-margin ($g'$) solutions are equivalent to each other.

**Condition 4.8** (Condition C.1 in page 8 in Morwani et al. (2024))**.** *We have* $g'(\theta^*, x, y) = g(\theta^*, x, y)$ *for all* $(x, y) \in \operatorname{spt}(q^*)$*, where* spt *is the support. It means:* $\{y' \in \mathcal{Y}\backslash\{y\} : \tau(x,y)[y'] > 0\} \subseteq \arg\max_{y'\in\mathcal{Y}\backslash\{y\}} f(\theta^*, x)[y']$.

Thus, under these conditions, we only need to focus on the class-weighted maximum-margin solutions in our following analysis.

## 4.4 Main Result

We characterize the Fourier features to perform modular addition with $k$ input in the one-hidden-layer neuron network. We show that every neuron only focuses on a distinct Fourier frequency. Additionally, within the network, there is at least one neuron for each frequency. When we consider the uniform class weighting, where $\mathcal{L}_\lambda(\theta)$ is based on $\tau(a_1, \ldots, a_k)[c'] := 1/(p-1) \ \ \forall c' \neq a_1 + \cdots + a_k$, we have the following main result:

**Theorem 4.9** (Main result, informal version of Theorem C.22)**.** *Let* $f(\theta, x)$ *be the one-hidden layer networks defined in Eq equation 4.2. If* $m \geqslant 2^{2k-1} \cdot \frac{p-1}{2}$*, then the max* $L_{2,k+1}$*-margin network satisfies:*

- *The maximum* $L_{2,k+1}$*-margin for a dataset* $D_p$ *is:*

$$\gamma^* = \frac{2(k!)}{(2k+2)^{(k+1)/2}(p-1)p^{(k-1)/2}}.$$

- *For each neuron* $\phi(\{u_1, \dots, u_k, w\}; a_1, \dots, a_k)$*, there is a constant scalar* $\beta \in \mathbb{R}$ *and a frequency* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$ *satisfying*

$$\begin{aligned} u_i(a_i) &= \beta \cdot \cos(\theta^*_{u_i} + 2\pi\zeta a_i/p), \;\; \forall i \in [k] \\ w(c) &= \beta \cdot \cos(\theta^*_w + 2\pi\zeta c/p), \end{aligned}$$

  *where* $\theta^*_{u_1}, \dots, \theta^*_{u_k}, \theta^*_w \in \mathbb{R}$ *are some phase offsets satisfying* $\theta^*_{u_1} + \dots + \theta^*_{u_k} = \theta^*_w$*.*

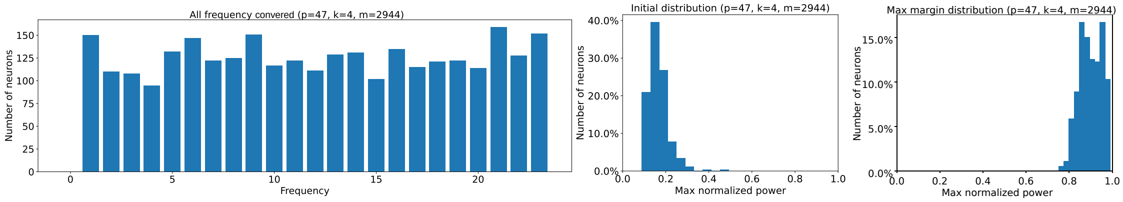
- *For each frequency* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$*, there exists one neuron using this frequency only.*

*Proof sketch of Theorem 4.9.* See formal proof in Appendix C.9.2. By Lemma 4.10, we get $\gamma^*$ and the single-neuron class-weighted maximum-margin solution set $\Omega'^*_q$. By satisfying Condition 4.8, we know it is used in the maximum-margin solution. By Lemma 4.11, we can construct the network $\theta^*$ that uses neurons in $\Omega'^*_q$. By Lemma C.6, we know that it is the maximum-margin solution. Finally, by Lemma C.20, we know that all frequencies are covered. □

Theorem 4.9 tells us when the number of neurons is large enough, e.g., $m \geqslant 2^{2k-1} \cdot \frac{p-1}{2}$ (the lower bound of $m$ may not be the tightest in our analysis), the one hidden neural network will exactly learn all Fourier spectrum/basis to recover the modular addition operation. More specifically, each neuron will only focus on one Fourier frequency. Our analysis provides a comprehensive understanding of why neural networks trained by SGD prefer to learn Fourier-based circuits.

### 4.4.1 Technique Overview

In this section, we propose techniques overview of the proof for our main result. We use $\mathbf{i}$ to denote $\sqrt{-1}$. Let $f : \mathbb{Z}_p \to \mathbb{C}$. Then, for each frequency $j \in \mathbb{Z}_p$, we define f discrete Fourier transform (DFT) as $\hat{f}(j) := \sum_{\zeta \in \mathbb{Z}_p} f(\zeta) \exp(-2\pi\mathbf{i} \cdot j\zeta/p)$. Let $\Omega'^*_q$ be the single neuron class-weighted maximum-margin solution set (formally defined in Definition C.13). We first show how we get $\Omega'^*_q$.



(a) (b) (c)

Figure 4.2: All Fourier spectrum frequencies being covered and the maximum normalized power of the embeddings (hidden layer weights). The one-hidden layer network with $m = 2944$ neurons is trained on $k = 4$-sum mod-$p = 47$ addition dataset. We denote $\hat{u}[i]$ as the Fourier transform of $u[i]$. Let $\max_i |\hat{u}[i]|^2/(\sum |\hat{u}[j]|^2)$ be the maximum normalized power. Mapping each neuron to its maximum normalized power frequency, (a) shows the final frequency distribution of the embeddings. Similar to our construction analysis in Lemma 4.11, we have an almost uniform distribution over all frequencies. (b) shows the maximum normalized power of the neural network with random initialization. (c) shows, in frequency space, the embeddings of the final trained network are one-sparse, i.e., maximum normalized power being almost 1 for all neurons. This is consistent with our max-margin analysis results in Lemma 4.11. See Figure C.2 and C.4 in Appendix C.10.2 for results when $k$ is 3 or 5.

**Lemma 4.10** (Informal version of Lemma C.15)**.** *If for any* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$*, there exists a scaling constant* $\beta \in \mathbb{R}$*, such that* $u_i(a_i) = \beta \cdot \cos(\theta^*_{u_i} + 2\pi\zeta a_i/p)$ *for any* $i \in [k]$ *and* $w(c) = \beta \cdot \cos(\theta^*_w + 2\pi\zeta c/p)$*, where* $\theta^*_{u_1}, \dots, \theta^*_{u_k}, \theta^*_w \in \mathbb{R}$ *are some phase offsets satisfying* $\theta^*_{u_1} + \dots + \theta^*_{u_k} = \theta^*_w$*. Then, we have* $\Omega'^*_q = \{(u_1, \dots, u_k, w)\}$*, and* $\gamma^* = \frac{2(k!)}{(2k+2)^{(k+1)/2}(p-1)p^{(k-1)/2}}$.

*Proof sketch of Lemma 4.10.* See formal proof in Appendix C.6.5. The proof establishes the maximum-margin solution's sparsity in the Fourier domain through several key steps. Initially, by Lemma C.14, focus is directed to maximizing Eq. equation C.10. For odd $p$, Eq. equation C.10 can be reformulated with magnitudes and phases of $\hat{u}_i$ and $\hat{w}$ (discrete Fourier transform of $u_i$ and $w$), leading to an equation involving cosine of their phase differences. Plancherel's theorem is then employed to translate the norm constraint to the Fourier domain. This allows for the optimization of the cosine term in the sum, effectively reducing the problem to maximizing the product of magnitudes of $\hat{u}_i$ and $\hat{w}$ (Eq. equation C.14). By applying the inequality of arithmetic and geometric means, we have an upper bound for the optimization problem. To achieve the upper bound, equal magnitudes are required for all $\hat{u}_i$ and $\hat{w}$ at a single frequency, leading to Eq. equation C.16. The neurons are finally expressed in the time domain, demonstrating that they assume a specific cosine form with phase offsets satisfying certain conditions. □

Next, we show the number of neurons required to solve the problem and the properties of these neurons. We demonstrate how to use these neurons to construct the network $\theta^*$.

**Lemma 4.11** (Informal version of Lemma C.18)**.** *Let* $\cos_\zeta(x)$ *denote* $\cos(2\pi\zeta x/p)$*. Then, we have the maximum* $L_{2,k+1}$*-margin solution* $\theta^*$ *will consist of* $2^{2k-1} \cdot \frac{p-1}{2}$ *neurons* $\theta_i^* \in \Omega_q^{'*}$ *to simulate* $\frac{p-1}{2}$ *type of cosine computation, where each cosine computation is uniquely determined a* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$*. In particular, for each* $\zeta$ *the cosine computation is* $\cos_\zeta(a_1 + \cdots + a_k - c), \forall a_1, \dots, a_k, c \in \mathbb{Z}_p$.

*Proof sketch of Lemma 4.11.* See formal proof in Appendix C.7.3. Our goal is to show that $2^{2k-1} \cdot \frac{p-1}{2}$ neurons $\theta_i^* \in \Omega_q^{'*}$ are able to simulate $\frac{p-1}{2}$ type of cos computation. We have the following expansion function of $\cos_\zeta(x)$, which denotes $\cos(2\pi\zeta x/p)$.

$$\begin{aligned}\cos_\zeta(\sum_{i=1}^k a_i) = & \sum_{b\in\{0,1\}^k} \prod_{i=1}^k \cos^{1-b_i}(a_i) \cdot \sin^{b_i}(a_i) \\ & \cdot \mathbf{1}[\sum_{i=1}^k b_i\%2 = 0] \cdot (-1)^{\mathbf{1}[\sum_{i=1}^k b_i\%4=2]}.\end{aligned}$$

The above equation can decompose a $\cos(\sum)$ to some basic elements. We have $2^k$ terms in the above equation. By using the following fact in Lemma C.16,

$$2^k \cdot k! \cdot \prod_{i=1}^k a_i = \sum_{c\in\{-1,+1\}^k} (-1)^{(k-\sum_{i=1}^k c_i)/2} (\sum_{j=1}^k c_j a_j)^k,$$

where each term can be constructed by $2^{k-1}$ neurons. Therefore, we need $2^{k-1}2^k$ total neurons. To simulate $\frac{p-1}{2}$ type of simulation, we need $2^{2k-1}\frac{p-1}{2}$ neurons. Then, using the Lemma C.5, we construct the network $\theta^*$. By using the Lemma C.6 from Morwani et al. (2024), we get it is the maximum-margin solution. □

## 4.5 Experiments

First, we conduct simulation experiments to verify our analysis for $k = 3, 4, 5$. Then, we show that the one-layer transformer learns 2-dimensional cosine functions in their attention weights. Finally, we show the grokking phenomenon under different $k$. Please refer to Appendix C.10.1 for details about implementation.

**One-hidden Layer Neural Network.** We conduct simulation experiments to verify our analysis. In Figure 4.1 and Figure 4.2, we use SGD to train a two-layer network with $m = 2944 = 2^{2k-2} \cdot (p-1)$ neurons, i.e., Eq. equation 4.2, on $k = 4$-sum mod-$p = 47$ addition dataset, i.e., Eq. equation 4.1. Figure 4.1 shows that the networks trained with SGD have single-frequency hidden neurons, which support our analysis in Lemma 4.10. Furthermore, Figure 4.2 demonstrates that the network will learn all frequencies in the Fourier spectrum which is consistent with our analysis in Lemma 4.11. Together, they verify our main results in Theorem 4.9 and show that the network trained by SGD prefers to learn Fourier-based circuits. There are more similar results when $k$ is 3 or 5 in Appendix C.10.2.

**One-layer Transformer.** We find similar results in one-layer transformers. Let $E$ be input embedding and $W^P, W^V, W^K, W^Q$ be projection, value, key and query matrix. The $m$-heads attention layer can be written as

$$W^P \begin{pmatrix} W_1^{V\top} E \cdot \text{softmax}\left(E^\top W_1^K W_1^{Q\top} E\right) \\ \cdots \\ W_m^{V\top} E \cdot \text{softmax}\left(E^\top W_m^K W_m^{Q\top} E\right) \end{pmatrix}.$$

We denote $W^K W^{Q\top}$ as $W^{KQ}$ and call it attention matrix. In Figure 4.3, we train a one-layer transformer with $m = 160$ heads attention and hidden dimension 128, i.e.,

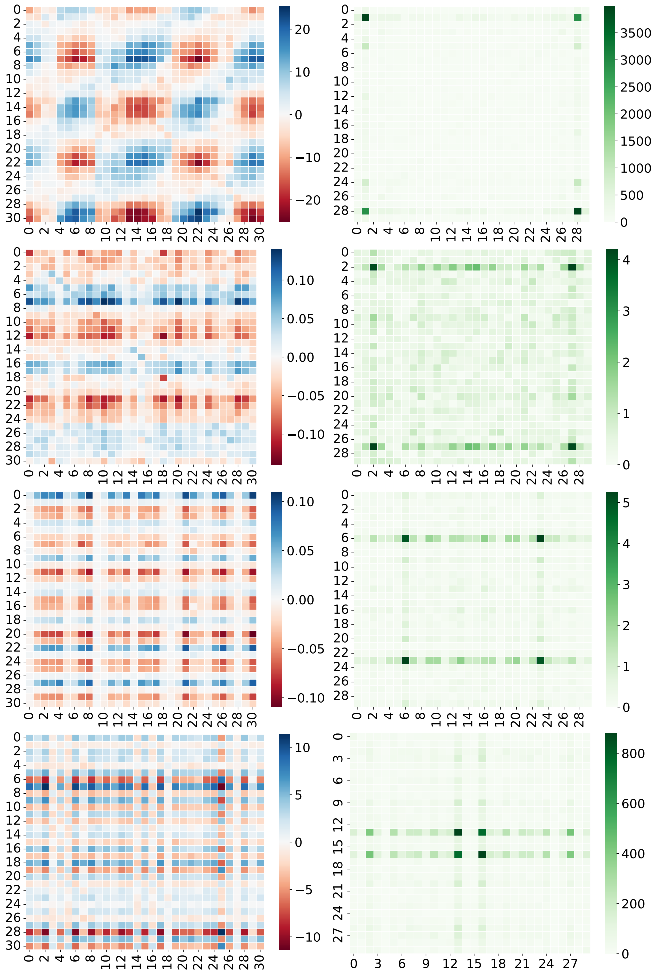


Figure 4.3: 2-dimension cosine shape of the trained $W^{KQ}$ (attention weights) and their Fourier power spectrum. The one-layer transformer with attention heads $m = 160$ is trained on $k = 4$-sum mod-$p = 31$ addition dataset. We even split the whole datasets ($p^k = 31^4$ data points) into training and test datasets. Every row represents a random attention head from the transformer. The left figure shows the final trained attention weights being an apparent 2-dim cosine shape. The right figure shows their 2-dim Fourier power spectrum. The results in the figures are consistent with Figure 4.1. See Figure C.5 and Figure C.6 in Appendix C.10.3 for similar results when $k$ is 3 or 5.

above equation, on $k = 4$-sum mod-$p = 31$ addition dataset, i.e., Eq. equation 4.1. Figure 4.3 shows that the SGD-trained one-layer transformer learns 2-dim cosine

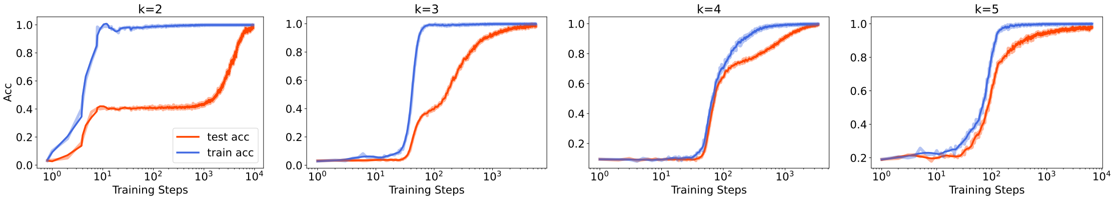


Figure 4.4: Grokking (models abruptly transition from bad generalization to perfect generalization after a large number of training steps) under learning modular addition involving $k = 2, 3, 4, 5$ inputs. We train two-layer transformers with $m = 160$ attention heads on $k = 2, 3, 4, 5$-sum mod-$p = 97, 31, 11, 5$ addition dataset with 50% of the data in the training set under AdamW Loshchilov and Hutter (2018) optimizer 1e-3 learning rate and 1e-3 weight decay. We use different $p$ to guarantee the dataset sizes are roughly equal to each other. The blue curves show training accuracy, and the red ones show validation accuracy. There is a grokking phenomenon in all figures. However, as $k$ increases, the grokking phenomenon becomes weak. See explanation in Section 4.5.

shape attention matrices, which is similar to the one-hidden layer neural networks in Figure 4.1. This means that the attention layer has a learning mechanism similar to neural networks in the modular arithmetic task. It prefers to learn (2-dim) Fourier-based circuits when trained by SGD. There are more similar results when $k$ is 3 or 5 in Appendix C.10.3.

**Grokking under Different $k$.** To support the importance of our data setting, we study the grokking phenomenon in our data distribution. Following the experiments' protocol in Power et al. (2022), we show there is the grokking phenomenon under different $k$. We train two-layer transformers with $m = 160$ attention heads and hidden dimension as 128 on $k = 2, 3, 4, 5$-sum mod-$p = 97, 31, 11, 5$ addition dataset with 50% of the data in training. We use different $p$ to guarantee the dataset sizes are roughly equal to each other. Figure 4.4 shows that the grokking weakens as the number of $k$ increases, which is consistent with our analysis. When $k$ increases, the function class will become more complicated, as we may need more neurons to achieve the max-margin solution. Thus, we use our Theorem 4.9 as a metric to measure the data complexity. It implies that when the ground-truth function class becomes "complicated", the transformers need to train more steps to fit the training datasets, and the generalization tends to be better. Brilliant recent works by Lyu et al. (2024); Kumar et al. (2023) argue that, during learning, the network will be first in the lazy training/NTK regime and then transfer to the rich/feature learning regime sharply, leading to a grokking phenomenon. We use learning steps required for regime switch as a metric of grokking strength.

NTK is a notorious overparameterized regime, which probably needs a much larger number of neurons than our max-margin convergence case, i.e., much larger than $\Omega(2^{2k})$ in Theorem 4.9. Thus, under the fixed $m$ and increasing $k$, the model may easily escape the NTK regime, or there is no longer an NTK regime. Thus, we will see a weaker grokking phenomenon as the learning steps needed to transfer from the NTK regime to the feature learning regime become fewer. With increasing $k$, the model will have an "underfitting" issue in the NTK regime, meaning the model must need feature learning to fit the task but cannot only fit the task by NTK. However, the model still has an "overfitting" in the feature learning regime.

## 4.6 Discussion

**Grokking in Transformers.** The interpretability of grokking in Transformers is explored in Nanda et al. (2023a). By examining various intermediate states within the residual stream of the Transformer model, it is validated that the model employs Fourier features to tackle the modular addition task. However, fully comprehending how the Transformer model and LLMs perform modular addition remains challenging based on the current work, particularly from a theoretical standpoint. We contend that beginning with a simplistic model setup and achieving a thorough and theoretical understanding of how the network utilizes Fourier features to address the problem serves as a valuable starting point and it provides a theoretical understanding of the grokking phenomenon. We believe that further study on Transformers will be an interesting and important future direction.

**Grokking, Benign Overfitting, and Implicit Bias.** Recently, Xu et al. (2024d) connects the grokking phenomenon to benign overfitting (Bartlett et al., 2020; Cao et al., 2022; Tsigler and Bartlett, 2023; Frei et al., 2022a, 2023a). It shows how the network undergoes a grokking period from catastrophic to benign overfitting. Lyu et al. (2024); Kumar et al. (2023) uses implicit bias (Soudry et al., 2018; Gunasekar et al., 2018a; Ji and Telgarsky, 2019a; Shah et al., 2020; Moroshko et al., 2020; Chizat and Bach, 2020; Lyu et al., 2021; Jacot, 2023; Xu et al., 2023, 2024f) to explain grokking, where grokking happens if the early phase bias implies an overfitting solution while late phase bias implies a generalizable solution. The intuition from the benign overfitting and the implicit bias well align with our observation in Section 4.5. It is interesting and valuable to rigorously analyze the grokking or emergent ability under different function class complexities, e.g., Eq equation 4.1. We leave this challenge problem as a future work.

**Connection to Parity and SQ Hardness.** If we let $p = 2$, then $(a_1 + \cdots + a_k) \bmod p$ will degenerate to parity function, i.e., $b_1, \dots, b_k \in \{\pm 1\}$ and determining $\prod_{i=1}^{k} b_i$. Parity functions serve as a fundamental set of learning challenges in computational learning theory, often used to demonstrate computational obstacles (Shalev-Shwartz et al., 2017). In particular, $(n, k)$-sparse parity problem is notorious hard to learn, i.e., Statistical Query (SQ) hardness (Blum et al., 1994). Daniely and Malach (2020) showed that one-hidden layer networks need an $\Omega(\exp(k))$ number of neurons or an $\Omega(\exp(k))$ number of training steps to successfully learn it by SGD. In our work, we are studying Eq. equation 4.2, which is a more general function than parity and indeed is a learning hardness. Our Theorem 4.9 states that we need $\Omega(\exp(k))$ number of neurons to represent the maximum-margin solution, which well aligns with existing works. Our experiential results in Section 4.5 are also consistent. Hence, our modular addition involving $k$ inputs function class is a good data model to analyze and test the model learning ability, i.e., approximation, optimization, and generalization.

**High Order Correlation Attention.** Sanford et al. (2023); Alman and Song (2023, 2024) state that, when $k = 3$, $a_1 + a_2 + a_3 \bmod p$ is hard to be captured by traditional attention. Thus, they introduce high-order attention to capture high-order correlation from the input sequence. However, in Section 4.5, we show that one-layer transformers have a strong learning ability and can successfully learn modular arithmetic tasks even when $k = 5$. This implies that the traditional attention may be more powerful than we expect.

## 4.7 Conclusion

We study neural networks and transformers learning on $(a_1 + \cdots + a_k)$ mod $p$. We theoretically show that networks prefer to learn Fourier circuits. Our experiments on neural networks and transformers support our analysis. Finally, we study the grokking phenomenon under this new data setting.

# 5 WHY LARGER LANGUAGE MODELS DO IN-CONTEXT LEARNING DIFFERENTLY?

**Contribution statement.** This chapter is joint work with Junyi Wei, Zhuoyan Xu, and Yingyu Liang. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis, and completed part of the experiments. The results of this chapter have been published as a conference paper in ICML 2024 (Shi et al., 2024b).

## 5.1 Introduction

As large language models (LLM), e.g., ChatGPT (OpenAI, 2022) and GPT4 (Achiam et al., 2023), are transforming AI development with potentially profound impact on our societies, it is critical to understand their mechanism for safe and efficient deployment. An important emergent ability Wei et al. (2022c); An et al. (2023), which makes LLM successful, is *in-context learning* (ICL), where models are given a few exemplars of input–label pairs as part of the prompt before evaluating some new input. More specifically, ICL is a few-shot Brown et al. (2020) evaluation method without updating parameters in LLM. Surprisingly, people find that, through ICL, LLM can perform well on tasks that have never been seen before, even without any finetuning. It means LLM can adapt to wide-ranging downstream tasks under efficient sample and computation complexity. The mechanism of ICL is different from traditional machine learning, such as supervised learning and unsupervised learning. For example, in neural networks, learning usually occurs in gradient updates, whereas there is only a forward inference in ICL and no gradient updates. Several recent works, trying to answer why LLM can learn in-context, argue that LLM secretly performs or simulates gradient descent as meta-optimizers with just a forward pass during ICL empirically Dai et al. (2022); Von Oswald et al. (2023); Malladi et al. (2023) and theoretically Zhang et al. (2023c); Ahn et al. (2024a); Mahankali et al. (2023); Cheng et al. (2023); Bai et al. (2024a); Hu et al. (2024a); Li et al. (2023f); Guo et al. (2024); Wu et al. (2024c). Although some insights have been obtained, the mechanism of ICL deserves further research to gain a better understanding.

Recently, there have been some important and surprising observations Min et al. (2022); Pan et al. (2023); Wei et al. (2023b); Shi et al. (2023a) that cannot be fully explained by existing studies. In particular, Shi et al. (2023a) finds that LLM is not robust during ICL and can be easily distracted by an irrelevant context. Furthermore, Wei et al. (2023b) shows that when we inject noise into the prompts, the larger language models may have a worse ICL ability than the small language models, and conjectures that the larger language models may overfit into the prompts and forget the prior knowledge from pretraining, while small models tend to follow the prior knowledge. On the other hand, Min et al. (2022); Pan et al. (2023) demonstrate that injecting noise does not affect the in-context learning that much for smaller models, which have a more strong pretraining knowledge bias. To improve the understanding of the ICL mechanism, to shed light on the properties and inner workings of LLMs, and to inspire efficient and safe use of ICL, we are interested in the following question:

*Why do larger language models do in-context learning differently?*

To answer this question, we study two settings: (1) one-layer single-head linear self-attention network Schlag et al. (2021); Von Oswald et al. (2023); Akyurek et al. (2023); Ahn et al. (2024a); Zhang et al. (2023c); Mahankali et al. (2023); Wu et al. (2024c) pretrained on linear regression in-context tasks Garg et al. (2022); Raventos et al. (2023); Von Oswald et al. (2023); Akyürek et al. (2022); Bai et al. (2024a); Mahankali et al. (2023); Zhang et al. (2023c); Ahn et al. (2024a); Li et al. (2023d); Hu et al. (2024a); Wu et al. (2024c), with rank constraint on the attention weight matrices for studying the effect of the model scale; (2) two-layer multiple-head transformers Li et al. (2023f) pretrained on sparse parity classification in-context tasks, comparing small or large head numbers for studying the effect of the model scale. In both settings, we give the closed-form optimal solutions. We show that smaller models emphasize important hidden features while larger models cover more features, e.g., less important features or noisy features. Then, we show that smaller models are more robust to label noise and input noise during evaluation, while larger models may easily be distracted by such noises, so larger models may have a worse ICL ability than smaller ones.

We also conduct in-context learning experiments on five prevalent NLP tasks utilizing various sizes of the Llama model families Touvron et al. (2023a,b), whose results are consistent with previous work Min et al. (2022); Pan et al. (2023); Wei et al. (2023b) and our analysis.

**Our contributions and novelty over existing work:**

- We formalize new stylized theoretical settings for studying ICL and the scaling effect of LLM. See Section 5.4 for linear regression and Section 5.5 for parity.
- We characterize the optimal solutions for both settings (Theorem 5.4.1 and Theorem 5.5.1).
- The characterizations of the optimal elucidate different attention paid to different hidden features, which then leads to the different ICL behavior (Theorem 5.4.2, Theorem 5.4.3, Theorem 5.5.2).
- We further provide empirical evidence on large base and chat models corroborating our theoretical analysis (Figure 5.1, Figure 5.2).

Note that previous ICL analysis paper may only focus on (1) the approximation power of transformers Garg et al. (2022); Panigrahi et al. (2023); Guo et al. (2024); Bai et al. (2024a); Cheng et al. (2023), e.g., constructing a transformer by hands which can do ICL, or (2) considering one-layer single-head linear self-attention network learning ICL on linear regression Von Oswald et al. (2023); Akyürek et al. (2022); Mahankali et al. (2023); Zhang et al. (2023c); Ahn et al. (2024a); Wu et al. (2024c), and may not focus on the robustness analysis or explain the different behaviors. In this work, (1) we extend the linear model linear data analysis to the non-linear model and non-linear data setting, i.e., two-layer multiple-head transformers leaning ICL on sparse parity classification and (2) we have a rigorous behavior difference analysis under two settings, which explains the empirical observations and provides more insights into the effect of attention mechanism in ICL.

## 5.2 Related Work

**Large language model.** Transformer-based Vaswani et al. (2017) neural networks have rapidly emerged as the primary machine learning architecture for tasks in natural language processing. Pretrained transformers with billions of parameters on broad and varied datasets are called large language models (LLM) or foundation models Bommasani et al. (2021), e.g., BERT Devlin et al. (2019), PaLM Chowdhery et al. (2022), LlamaTouvron et al. (2023a), ChatGPT (OpenAI, 2022), GPT4 (Achiam et al., 2023) and so on. LLM has shown powerful general intelligence Bubeck et al. (2023) in various downstream tasks. To better use the LLM for a specific downstream task, there are many adaptation methods, such as adaptor Hu et al. (2022); Zhang et al. (2023b); Gao et al. (2023); Shi et al. (2023b), calibration Zhao et al. (2021); Zhou et al. (2023a), multitask finetuning Gao et al. (2021a); Xu et al. (2023); Von Oswald et al. (2023); Xu et al. (2024f), prompt tuning Gao et al. (2021a); Lester et al. (2021), instruction tuning Li and Liang (2021); Chung et al. (2022); Mishra et al. (2022), symbol tuning Wei et al. (2023a), black-box tuning Sun et al. (2022), chain-of-thoughts Wei et al. (2022d); Yao et al. (2023); Zheng et al. (2024), scratchpad Nye et al. (2021), reinforcement learning from human feedback (RLHF) Ouyang et al. (2022) and many so on.

**In-context learning.** One important emergent ability Wei et al. (2022c) from LLM is in-context learning (ICL) Brown et al. (2020). Specifically, when presented with a brief series of input-output pairings (known as a prompt) related to a certain task, they can generate predictions for test scenarios without necessitating any adjustments to the model's parameters. ICL is widely used in broad scenarios, e.g., reasoning Zhou et al. (2022), negotiation Fu et al. (2023), self-correction Pourreza and Rafiei (2023), machine translation Agrawal et al. (2022) and so on. Many works trying to improve the ICL and zero-shot ability of LLM Min et al. (2021); Wang et al. (2022); Wei et al. (2022a); Iyer et al. (2022). There is a line of insightful works to study the mechanism of transformer learning Geva et al. (2021); Xie et al. (2022); Garg et al. (2022); Jelassi et al. (2022); Arora and Goyal (2023); Li et al. (2023b,f); Allen-Zhu and Li (2023); Luo et al. (2023); Tian et al. (2023a,b); Zhou et al. (2023b); Bietti et al. (2023); Xu et al. (2024e); Li et al. (2024a); Liang et al. (2024a,g,h) and in-context learning Dai et al. (2022); Mahankali et al. (2023); Raventos et al. (2023); Bai et al. (2024a); Ahn et al. (2024a); Von Oswald et al. (2023); Pan et al. (2023); Li et al. (2023f,d,e); Akyürek et al. (2022); Zhang et al. (2023a,c); Hu et al. (2024a); Cheng et al. (2023); Wibisono and Wang (2023); Wu et al. (2024c); Guo et al. (2024); Reddy (2024) empirically and theoretically. On the basis of these works, our analysis takes a step forward to show the ICL behavior difference under different scales of language models.

## 5.3 Preliminary

**Notations.** We denote $[n] := \{1, 2, \dots, n\}$. For a positive semidefinite matrix $\mathbf{A}$, we denote $\|\mathbf{x}\|_{\mathbf{A}}^2 := \mathbf{x}^\top \mathbf{A} \mathbf{x}$ as the norm induced by a positive definite matrix $\mathbf{A}$. We denote $\| \cdot \|_F$ as the Frobenius norm. $\mathrm{diag}()$ function will map a vector to a diagonal matrix or map a matrix to a vector with its diagonal terms.

**In-context learning.** We follow the setup and notation of the problem in Zhang et al. (2023c); Mahankali et al. (2023); Ahn et al. (2024a); Hu et al. (2024a); Wu et al. (2024c). In the pretraining stage of ICL, the model is pretrained on prompts. A prompt from a task $\tau$ is formed by $N$ examples $(\mathbf{x}_{\tau,1}, y_{\tau,1}), \dots, (\mathbf{x}_{\tau,N}, y_{\tau,N})$ and a query token $\mathbf{x}_{\tau,q}$ for prediction, where for any $i \in [N]$ we have $y_{\tau,i} \in \mathbb{R}$ and $\mathbf{x}_{\tau,i}, \mathbf{x}_{\tau,q} \in \mathbb{R}^d$. The embedding matrix $\mathbf{E}_\tau$, the label vector $\mathbf{y}_\tau$, and the input matrix $\mathbf{X}_\tau$ are defined as:

$$\begin{aligned}
\mathbf{E}_\tau :=& \begin{pmatrix} \mathbf{x}_{\tau,1} & \mathbf{x}_{\tau,2} & \dots & \mathbf{x}_{\tau,N} & \mathbf{x}_{\tau,q} \\ y_{\tau,1} & y_{\tau,2} & \dots & y_{\tau,N} & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(N+1)}, \\
\mathbf{y}_\tau :=& [y_{\tau,1}, \dots, y_{\tau,N}]^\top \in \mathbb{R}^N, \qquad y_{\tau,q} \in \mathbb{R}, \\
\mathbf{X}_\tau :=& [\mathbf{x}_{\tau,1}, \dots, \mathbf{x}_{\tau,N}]^\top \in \mathbb{R}^{N\times d}, \qquad \mathbf{x}_{\tau,q} \in \mathbb{R}^d.
\end{aligned}$$

Given prompts represented as $\mathbf{E}_\tau$'s and the corresponding true labels $y_{\tau,q}$'s, the pretraining aims to find a model whose output on $\mathbf{E}_\tau$ matches $y_{\tau,q}$. After pretraining, the evaluation stage applies the model to a new test prompt (potentially from a different task) and compares the model output to the true label on the query token.

Note that our pretraining stage is also called learning to learn in-context Min et al. (2021) or in-context training warmup Dong et al. (2022) in existing work. Learning to learn in-context is the first step to understanding the mechanism of ICL in LLM following previous works Raventos et al. (2023); Zhou et al. (2023b); Zhang et al. (2023c); Mahankali et al. (2023); Ahn et al. (2024a); Hu et al. (2024a); Li et al. (2023f); Wu et al. (2024c).

**Linear self-attention networks.** The linear self-attention network has been widely studied Schlag et al. (2021); Von Oswald et al. (2023); Akyürek et al. (2022); Ahn et al. (2024a); Zhang et al. (2023c); Mahankali et al. (2023); Wu et al. (2024c); Ahn et al. (2024b), and will be used as the learning model or a component of the model in our two theoretical settings. It is defined as:

$$f_{\mathrm{LSA},\theta}(\mathbf{E}) = [\mathbf{E} + \mathbf{W}^{PV}\mathbf{E} \cdot \frac{\mathbf{E}^\top \mathbf{W}^{KQ}\mathbf{E}}{\rho}], \tag{5.1}$$

where $\theta = (\mathbf{W}^{PV}, \mathbf{W}^{KQ})$, $\mathbf{E} \in \mathbb{R}^{(d+1)\times(N+1)}$ is the embedding matrix of the input prompt, and $\rho$ is a normalization factor set to be the length of examples, i.e., $\rho = N$ during pretraining. Similar to existing work, for simplicity, we have merged the projection and value matrices into $\mathbf{W}^{PV}$, and merged the key and query matrices into $\mathbf{W}^{KQ}$, and have a residual connection in our LSA network. The prediction of the network for the query token $\mathbf{x}_{\tau,q}$ will be the bottom right entry of the matrix output, i.e., the entry at location $(d+1),(N+1)$, while other entries are not relevant to our study and thus are ignored. So only part of the model parameters are relevant. To see this, let us denote

$$\mathbf{W}^{PV} = \begin{pmatrix} \mathbf{W}_{11}^{PV} & \mathbf{w}_{12}^{PV} \\ (\mathbf{w}_{21}^{PV})^\top & w_{22}^{PV} \end{pmatrix} \in \mathbb{R}^{(d+1)\times(d+1)},$$

$$\mathbf{W}^{KQ} = \begin{pmatrix}\mathbf{W}_{11}^{KQ} & \mathbf{w}_{12}^{KQ}\\ (\mathbf{w}_{21}^{KQ})^\top & w_{22}^{KQ}\end{pmatrix} \in \mathbb{R}^{(d+1)\times(d+1)},$$

where $\mathbf{W}_{11}^{PV}, \mathbf{W}_{11}^{KQ} \in \mathbb{R}^{d\times d}$; $\mathbf{w}_{12}^{PV}, \mathbf{w}_{21}^{PV}, \mathbf{w}_{12}^{KQ}, \mathbf{w}_{21}^{KQ} \in \mathbb{R}^d$; and $w_{22}^{PV}, w_{22}^{KQ} \in \mathbb{R}$. Then the prediction is:

$$\begin{aligned}\widehat{y}_{\tau,q} =& f_{\text{LSA},\theta}(\mathbf{E})_{(d+1),(N+1)} \\ =& \frac{\begin{pmatrix}(\mathbf{w}_{21}^{PV})^\top & w_{22}^{PV}\end{pmatrix}(\mathbf{E}\mathbf{E}^\top}{\rho)\begin{pmatrix}\mathbf{W}_{11}^{KQ}\\ (w_{21}^{KQ})^\top\end{pmatrix}\mathbf{x}_{\tau,q}.}\end{aligned} \tag{5.2}$$

## 5.4 Linear Regression

In this section, we consider the linear regression task for in-context learning which is widely studied empirically Garg et al. (2022); Raventos et al. (2023); Von Oswald et al. (2023); Akyürek et al. (2022); Bai et al. (2024a) and theoretically Mahankali et al. (2023); Zhang et al. (2023c); Ahn et al. (2024a); Li et al. (2023d); Hu et al. (2024a); Wu et al. (2024c).

**Data and task.** For each task $\tau$, we assume for any $i \in [N]$ tokens $\mathbf{x}_{\tau,i}, \mathbf{x}_{\tau,q} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \Lambda)$, where $\Lambda$ is the covariance matrix. We also assume a $d$-dimension task weight $\mathbf{w}_\tau \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{d\times d})$ and the labels are given by $y_{\tau,i} = \langle \mathbf{w}_\tau, \mathbf{x}_{\tau,i}\rangle$ and $y_{\tau,q} = \langle \mathbf{w}_\tau, \mathbf{x}_{\tau,q}\rangle$.

**Model and loss.** We study a one-layer single-head linear self-attention transformer (LSA) defined in Equation (5.1) and we use $\widehat{y}_{\tau,q} := f_{\text{LSA},\theta}(\mathbf{E})_{(d+1),(N+1)}$ as the prediction. We consider the mean square error (MSE) loss so that the empirical risk over $B$ independent prompts is defined as

$$\widehat{\mathcal{L}}(f_\theta) := \frac{1}{2B}\sum_{\tau=1}^{B}(\widehat{y}_{\tau,q} - \langle \mathbf{w}_\tau, \mathbf{x}_{\tau,q}\rangle)^2.$$

**Measure model scale by rank.** We first introduce a lemma from previous work that simplifies the MSE and justifies our measurement of the model scale. For notation simplicity, we denote $\mathbf{U} = \mathbf{W}_{11}^{KQ}, u = w_{22}^{PV}$.

**Lemma 5.1** (Lemma A.1 in Zhang et al. (2023c)). *Let* $\Gamma := (1+\frac{1}{N})\Lambda + \frac{1}{N}\operatorname{tr}(\Lambda) I_{d\times d} \in \mathbb{R}^{d\times d}$. *Let*

$$\begin{aligned}
\mathcal{L}(f_{\mathrm{LSA},\theta}) =& \lim_{B\, arrow \infty} \widehat{\mathcal{L}}(f_{\mathrm{LSA},\theta}) \\
=& \frac{1}{2}\mathbb{E}_{\mathbf{w}_\tau, \mathbf{x}_{\tau,1},\ldots,\mathbf{x}_{\tau,N},\mathbf{x}_{\tau,q}}[(\widehat{y}_{\tau,q} - \langle \mathbf{w}_\tau, \mathbf{x}_{\tau,q}\rangle)^2], \\
\tilde{\ell}(\mathbf{U}, u) =& \operatorname{tr}[\frac{1}{2}u^2\Gamma\Lambda\mathbf{U}\Lambda\mathbf{U}^\top - u\Lambda^2\mathbf{U}^\top],
\end{aligned}$$

*we have* $\mathcal{L}(f_{\mathrm{LSA},\theta}) = \tilde{\ell}(\mathbf{U}, u) + C$, *where* $C$ *is a constant independent with* $\theta$.

Theorem 5.1 tells us that the loss only depends on $u\mathbf{U}$. If we consider non-zero $u$, w.l.o.g, letting $u = 1$, then we can see that the loss only depends on $\mathbf{U} \in \mathbb{R}^{d\times d}$,

$$\mathcal{L}(f_{\mathrm{LSA},\theta}) = \operatorname{tr}[\frac{1}{2}\Gamma\Lambda\mathbf{U}\Lambda\mathbf{U}^\top - \Lambda^2\mathbf{U}^\top].$$

Note that $\mathbf{U} = \mathbf{W}_{11}^{KQ}$, then it is natural to measure the size of the model by rank of $\mathbf{U}$. Recall that we merge the key matrix and the query matrix in attention together, i.e., $\mathbf{W}^{KQ} = (\mathbf{W}^K)^\top\mathbf{W}^Q$. Thus, a low-rank $\mathbf{U}$ is equivalent to the constraint $\mathbf{W}^K, \mathbf{W}^Q \in \mathbb{R}^{r\times d}$ where $r \ll d$. The low-rank key and query matrix are practical and have been widely studied Hu et al. (2022); Chen et al. (2021); Bhojanapalli et al. (2020); Fan et al. (2021); Dass et al. (2023); Shi et al. (2023c). Therefore, we use $r = \operatorname{rank}(\mathbf{U})$ to measure the scale of the model, i.e., larger $r$ representing larger models. To study the behavior difference under different model scale, we will analyze $\mathbf{U}$ under different rank constraints.

### 5.4.1 Low Rank Optimal Solution

Since the token covariance matrix $\Lambda$ is positive semidefinite symmetric, we have eigendecomposition $\Lambda = \mathbf{Q}\mathbf{D}\mathbf{Q}^\top$, where $\mathbf{Q}$ is an orthonormal matrix containing eigenvectors of $\Lambda$ and $\mathbf{D}$ is a sorted diagonal matrix with non-negative entries containing eigenvalues of $\Lambda$, denoting as $\mathbf{D} = \operatorname{diag}([\lambda_1, \dots, \lambda_d])$, where $\lambda_1 \geqslant \cdots \geqslant \lambda_d \geqslant 0$. Then, we have the following theorem.

**Theorem 5.4.1** (Optimal rank-r solution for regression)**.** *Recall the loss function $\tilde{\ell}$ in Theorem 5.1. Let*

$$\mathbf{U}^*, u^* = \underset{\mathbf{U} \in \mathbb{R}^{d \times d}, \operatorname{rank}(\mathbf{U}) \leqslant r, u \in \mathbb{R}}{\arg\min} \tilde{\ell}(\mathbf{U}, u).$$

*Then $\mathbf{U}^* = c\mathbf{Q}\mathbf{V}^*\mathbf{Q}^\top, u = \frac{1}{c}$, where $c$ is any nonzero constant, and $\mathbf{V}^* = \operatorname{diag}([v_1^*, \dots, v_d^*])$ satisfies for any $i \leqslant r, v_i^* = \frac{N}{(N+1)\lambda_i + \operatorname{tr}(\mathbf{D})}$ and for any $i > r, v_i^* = 0$.*

*Proof sketch of Theorem 5.4.1.* We defer the full proof to Section D.2.1. The proof idea is that we can decompose the loss function into different ranks, so we can keep the direction by their sorted "variance", i.e.,

$$\underset{\mathbf{U} \in \mathbb{R}^{d \times d}, \operatorname{rank}(\mathbf{U}) \leqslant r, u \in \mathbb{R}}{\arg\min} \tilde{\ell}(\mathbf{U}, u) = \sum_{i=1}^{d} T_i \lambda_i^2 (v_i^* - \frac{1}{T_i})^2,$$

where $T_i = (1 + \frac{1}{N})\lambda_i + \frac{\operatorname{tr}(\mathbf{D})}{N}$. We have that $v_i^* \geqslant 0$ for any $i \in [d]$ and if $v_i^* > 0$, we have $v_i^* = \frac{1}{T_i}$. Denote $g(x) = x^2(\frac{1}{(1+\frac{1}{N})x + \frac{\operatorname{tr}(\mathbf{D})}{N}})$. We get the conclusion by $g(x)$ is an increasing function on $[0, \infty)$. □

Theorem 5.4.1 gives the closed-form optimal rank-r solution of one-layer single-head linear self-attention transformer learning linear regression ICL tasks. Let $f_{\text{LSA},\theta}$ denote the optimal rank-r solution corresponding to the $\mathbf{U}^*, u^*$ above. In detail, the optimal rank-r solution $f_{\text{LSA},\theta}$ satisfies

$$\mathbf{W}^{*PV} = \begin{pmatrix} 0_{d \times d} & 0_d \\ 0_d^\top & u \end{pmatrix}, \mathbf{W}^{*KQ} = \begin{pmatrix} \mathbf{U}^* & 0_d \\ 0_d^\top & 0 \end{pmatrix}. \tag{5.3}$$

**What hidden features does the model pay attention to?** Theorem 5.4.1 shows that the optimal rank-r solution indeed is the truncated version of the optimal full-rank solution, keeping only the most important feature directions (i.e., the first r eigenvectors of the token covariance matrix). In detail, (1) for the optimal full-rank solution, we have for any $i \in [d], v_i^* = \frac{N}{(N+1)\lambda_i + \text{tr}(\mathbf{D})}$; (2) for the optimal rank-r solution, we have for any $i \leqslant r, v_i^* = \frac{N}{(N+1)\lambda_i + \text{tr}(\mathbf{D})}$ and for any $i > r, v_i^* = 0$. That is, the small rank-r model keeps only the first r eigenvectors (viewed as hidden feature directions) and does not cover the others, while larger ranks cover more hidden features, and the large full rank model covers all features.

Recall that the prediction depends on $\mathbf{U}^* \mathbf{x}_{\tau,q} = c\mathbf{Q}\mathbf{V}^*\mathbf{Q}^\top \mathbf{x}_{\tau,q}$; see Equation (5.2) and (5.3). So the optimal rank-r model only uses the components on the first r eigenvector directions to do the prediction in evaluations. When there is noise distributed in all directions, a smaller model can ignore noise and signals along less important directions but still keep the most important directions. Then it can be less sensitive to the noise, as empirically observed. This insight is formalized in the next subsection.

### 5.4.2 Behavior Difference

We now formalize our insight into the behavior difference based on our analysis on the optimal solutions. We consider the evaluation prompt to have M examples (may not be equal to the number of examples N during pretraining for a general evaluation setting), and assume noise in labels to facilitate the study of the behavior difference (our results can be applied to the noiseless case by considering noise level $\sigma = 0$). Formally, the evaluation prompt is:

$$\begin{aligned}\widehat{\mathbf{E}} &:= \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_M & \mathbf{x}_q \\ y_1 & y_2 & \dots & y_M & 0 \end{pmatrix} \in \mathbb{R}^{(d+1)\times(M+1)} \\ &= \begin{pmatrix} \mathbf{x}_1 & \dots & \mathbf{x}_M & \mathbf{x}_q \\ \langle \mathbf{w}, \mathbf{x}_1 \rangle + \epsilon_1 & \dots & \langle \mathbf{w}, \mathbf{x}_M \rangle + \epsilon_M & 0 \end{pmatrix},\end{aligned}$$

where $\mathbf{w}$ is the weight for the evaluation task, and for any $i \in [M]$, the label noise $\epsilon_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2)$.

Recall $\mathbf{Q}$ are eigenvectors of $\Lambda$, i.e., $\Lambda = \mathbf{Q}\mathbf{D}\mathbf{Q}^\top$ and $\mathbf{D} = \text{diag}([\lambda_1, \dots, \lambda_d])$. In practice, we can view the large variance part of $\mathbf{x}$ (top $r$ directions in $\mathbf{Q}$) as a useful signal (like words "positive", "negative"), and the small variance part (bottom $d-r$ directions in $\mathbf{Q}$) as the less important or useless information (like words "even", "just").

Based on such intuition, we can decompose the evaluation task weight $\mathbf{w}$ accordingly: $\mathbf{w} = \mathbf{Q}(\mathbf{s} + \xi)$, where the $r$-dim truncated vector $\mathbf{s} \in \mathbb{R}^d$ has $\mathbf{s}_i = 0$ for any $r < i \leqslant d$, and the residual vector $\xi \in \mathbb{R}^d$ has $\xi_i = 0$ for any $1 \leqslant i \leqslant r$. The following theorem (proved in Section D.2.2) quantifies the evaluation loss at different model scales $r$ which can explain the scale's effect.

**Theorem 5.4.2** (Behavior difference for regression)**.** *Let* $\mathbf{w} = \mathbf{Q}(\mathbf{s} + \xi) \in \mathbb{R}^d$ *where* $\mathbf{s}, \xi \in \mathbb{R}^d$ *are truncated and residual vectors defined above. The optimal rank-*$r$ *solution* $f_{\mathrm{LSA},\theta}$ *in Theorem 5.4.1 satisfies:*

$$\begin{aligned}
&\mathcal{L}(f_{\mathrm{LSA},\theta}; \widehat{\mathbf{E}}) \\
:=&\mathbb{E}_{\mathbf{x}_1,\epsilon_1,\ldots,\mathbf{x}_M,\epsilon_M,\mathbf{x}_q}\left(f_{\mathrm{LSA},\theta}(\widehat{\mathbf{E}}) - \langle \mathbf{w}, \mathbf{x}_q \rangle\right)^2 \\
=&\frac{1}{M}\|\mathbf{s}\|^2_{(\mathbf{V}^*)^2\mathbf{D}^3} + \frac{1}{M}(\|\mathbf{s}+\xi\|^2_{\mathbf{D}} + \sigma^2)\,\mathrm{tr}((\mathbf{V}^*)^2\mathbf{D}^2) \\
&+ \|\xi\|^2_{\mathbf{D}} + \sum_{i\in[r]} \mathbf{s}_i^2 \lambda_i (\lambda_i v_i^* - 1)^2.
\end{aligned}$$

**Implications.** If $N$ is large enough with $N\lambda_r \gg \mathrm{tr}(\mathbf{D})$ (which is practical as we usually pretrain networks on long text), then

$$\mathcal{L}(f_{\mathrm{LSA},\theta}; \widehat{\mathbf{E}}) \approx \|\xi\|^2_{\mathbf{D}} + \frac{1}{M}((r+1)\|\mathbf{s}\|^2_{\mathbf{D}} + r\|\xi\|^2_{\mathbf{D}} + r\sigma^2).$$

The first term $\|\xi\|^2_{\mathbf{D}}$ is due to the residual features not covered by the network, so it decreases for larger $r$ and becomes 0 for full-rank $r = d$. The second term $\frac{1}{M}(\cdot)$ is significant since we typically have limited examples in evaluation, e.g., $M = 16 \ll N$. Within it, $(r+1)\|\mathbf{s}\|^2_{\mathbf{D}}$ corresponds to the first $r$ directions, and $r\sigma^2$ corresponds to the label noise. These increase for larger $r$. So there is a trade-off between the two error terms when scaling up the model: for larger $r$ the first term decreases while the second term increases. This depends on whether more signals are covered or more noise is kept when increasing the rank r.

To further illustrate the insights, we consider the special case when the model already covers all useful signals in the evaluation task: $\mathbf{w} = \mathbf{Q}\mathbf{s}$, i.e., the label only depends on the top r features (like "positive", "negative" tokens). Our above analysis implies that a larger model will cover more useless features and keep more noise, and thus will have worse performance. This is formalized in the following theorem (proved in Section D.2.2).

**Theorem 5.4.3** (Behavior difference for regression, special case)**.** *Let* $0 \leqslant r \leqslant r' \leqslant d$ *and* $\mathbf{w} = \mathbf{Q}\mathbf{s}$ *where* $\mathbf{s}$ *is* $r$*-dim truncated vector. Denote the optimal rank-*$r$ *solution as* $f_1$ *and the optimal rank-*$r'$ *solution as* $f_2$*. Then,*

$$\begin{aligned}&\mathcal{L}(f_2; \widehat{\mathbf{E}}) - \mathcal{L}(f_1; \widehat{\mathbf{E}})\\ =&\frac{1}{M}(\|\mathbf{s}\|_{\mathbf{D}}^2 + \sigma^2)(\sum_{i=r+1}^{r'}(\frac{N\lambda_i}{(N+1)\lambda_i + \mathrm{tr}(\mathbf{D})})^2).\end{aligned}$$

**Implications.** By Theorem 5.4.3, in this case,

$$\mathcal{L}(f_2; \widehat{\mathbf{E}}) - \mathcal{L}(f_1; \widehat{\mathbf{E}}) \approx \underbrace{\frac{r'-r}{M}\|\mathbf{s}\|_{\mathbf{D}}^2}_{\text{input noise}} + \underbrace{\frac{r'-r}{M}\sigma^2}_{\text{label noise}}.$$

We can decompose the above equation to input noise and label noise, and we know that $\|\mathbf{s}\|_{\mathbf{D}}^2 + \sigma^2$ only depends on the intrinsic property of evaluation data and is independent of the model size. When we have a larger model (larger $r'$), we will have a larger evaluation loss gap between the large and small models. It means larger language models may be easily affected by the label noise and input noise and may have worse in-context learning ability, while smaller models may be more robust to these noises as they only emphasize important signals. Moreover, if we increase the label noise scale $\sigma^2$ on purpose, the larger models will be more sensitive to the injected label noise. This is consistent with the observation in Wei et al. (2023b); Shi et al. (2023a) and our experimental results in Section 5.6.

## 5.5 Sparse Parity Classification

We further consider a more sophisticated setting with nonlinear data which necessitates nonlinear models. Viewing sentences as generated from various kinds of thoughts and knowledge that can be represented as vectors in some hidden feature space, we consider the classic data model of dictionary learning or sparse coding, which has been widely used for text and images Olshausen and Field (1997); Vinje and Gallant (2000); Blei et al. (2003). Furthermore, beyond linear separability, we assume the labels are given by the $(d, 2)$-sparse parity on the hidden feature vector, which is the high-dimensional generalization of the classic XOR problem. Parities are a canonical family of highly non-linear learning problems and recently have been used in many recent studies on neural network learning Daniely and Malach (2020); Barak et al. (2022); Shi et al. (2022c, 2023d).

**Data and task.** Let $\mathcal{X} = \mathbb{R}^d$ be the input space, and $\mathcal{Y} = \{\pm 1\}$ be the label space. Suppose $\mathbf{G} \in \mathbb{R}^{d \times d}$ is an unknown dictionary with $d$ columns that can be regarded as features; for simplicity, assume $\mathbf{G}$ is orthonormal. Let $\phi \in \{\pm 1\}^d$ be a hidden vector that indicates the presence of each feature. The data are generated as follows: for each task $\tau$, generate two task indices $\mathbf{t}_\tau = (i_\tau, j_\tau)$ which determines a distribution $\mathcal{T}_\tau$; then for this task, draw examples by $\phi \sim \mathcal{T}_\tau$, and setting $\mathbf{x} = \mathbf{G}\phi$ (i.e., dictionary learning data), $y = \phi_{i_\tau}\phi_{j_\tau}$ (i.e., XOR labels).

We now specify how to generate $\mathbf{t}_\tau$ and $\phi$. As some of the hidden features are more important than others, we let $A = [k]$ denote a subset of size $k$ corresponding to the important features. We denote the important task set as $S_1 := A \times A \setminus \{(l, l) : l \in A\}$ and less important task set as $S_2 := [d] \times [d] \setminus (\{(l, l) : l \in [d]\} \cup S_1)$. Then $\mathbf{t}_\tau$ is drawn uniformly from $S_1$ with probability $1 - p_{\mathcal{T}}$, and uniformly from $S_2$ with probability $p_{\mathcal{T}}$, where $p_{\mathcal{T}} \in [0, \frac{1}{2})$ is the less-important task rate. For the distribution of $\phi$, we assume $\phi_{[d]\setminus\{i_\tau, j_\tau\}}$ is drawn uniformly from $\{\pm 1\}^{d-2}$, and assume $\phi_{\{i_\tau, j_\tau\}}$ has good correlation (measured by a parameter $\gamma \in (0, \frac{1}{4})$) with the label to facilitate learning. Independently, we have

$$\Pr[(\phi_{i_\tau}, \phi_{j_\tau}) = (1, 1)] = 1/4 + \gamma,$$

$$\Pr[(\phi_{i_\tau}, \phi_{j_\tau}) = (1, -1)] = 1/4,$$
$$\Pr[(\phi_{i_\tau}, \phi_{j_\tau}) = (-1, 1)] = 1/4,$$
$$\Pr[(\phi_{i_\tau}, \phi_{j_\tau}) = (-1, -1)] = 1/4 - \gamma.$$

Note that without correlation ($\gamma = 0$), it is well-known sparse parities will be hard to learn, so we consider $\gamma > 0$.

**Model.** Following Wu et al. (2024c), we consider the reduced linear self-attention $f_{\mathrm{LSA},\theta}(\mathbf{X}, \mathbf{y}, \mathbf{x}_q) = \frac{\mathbf{y}^\top \mathbf{X}}{N} \mathbf{W}^{KQ} \mathbf{x}_q$ (which is a reduced version of Equation (5.1)), and also denote $\mathbf{W}^{KQ}$ as $\mathbf{W}$ for simplicity. It is used as the neuron in our two-layer multiple-head transformers:

$$g(\mathbf{X}, \mathbf{y}, \mathbf{x}_q) = \sum_{i \in [m]} \mathbf{a}_i \sigma[\frac{\mathbf{y}^\top \mathbf{X}}{N} \mathbf{W}^{(i)} \mathbf{x}_q],$$

where $\sigma$ is ReLU activation, $\mathbf{a} = [\mathbf{a}_1, \dots, \mathbf{a}_m]^\top \in [-1, 1]^m$, $\mathbf{W}^{(i)} \in \mathbb{R}^{d \times d}$ and $m$ is the number of attention heads. Denote its parameters as $\theta = (\mathbf{a}, \mathbf{W}^{(1)}, \dots, \mathbf{W}^{(m)})$.

This model is more complicated as it uses non-linear activation, and also has two layers with multiple heads.

**Measure model scale by head number.** We use the attention head number $m$ to measure the model scale, as a larger $m$ means the transformer can learn more attention patterns. We consider hinge loss $\ell(z) = \max(0, 1 - z)$, and the population loss with weight-decay regularization:

$$\mathcal{L}^\lambda(g) = \mathbb{E}[\ell(y_q \cdot g(\mathbf{X}, \mathbf{y}, \mathbf{x}_q))] + \lambda(\sum_{i \in [m]} \|\mathbf{W}^{(i)}\|_F^2).$$

Suppose $N arrow \infty$ and let the optimal solution of $\mathcal{L}^\lambda(g)$ be

$$g^* = \arg\min_g \lim_{\lambda arrow 0^+} \mathcal{L}^\lambda(g).$$

### 5.5.1 Optimal Solution

We first introduce some notations to describe the optimal. Let $\mathrm{bin}(\cdot)$ be the integer to binary function, e.g., $\mathrm{bin}(6) = 110$. Let $\mathrm{digit}(z, i)$ denote the digit at the $i$-th position (from right to left) of $z$, e.g., $\mathrm{digit}(01000, 4) = 1$. We are now ready to characterize the optimal solution (proved in Section D.3.1).

**Theorem 5.5.1** (Optimal solution for parity)**.** *Consider* $k = 2^{\nu_1}, d = 2^{\nu_2}$*, and let* $g_1^*$ *and* $g_2^*$ *denote the optimal solutions for* $m = 2(\nu_1 + 1)$ *and* $m = 2(\nu_2 + 1)$*, respectively.*

*When* $0 < p_{\mathcal{T}} < \frac{\frac{1}{4}-\gamma}{\frac{d(d-1)}{2}(\frac{1}{4}+\gamma)+\frac{1}{4}-\gamma}$*,* $g_1^*$ *neurons are a subset of* $g_2^*$ *neurons. Specifically, for any* $i \in [2(\nu_2 + 1)]$*, let* $\mathbf{V}^{*,(i)}$ *be diagonal matrix and*

- *For any* $i \in [\nu_2]$ *and* $i_\tau \in [d]$*, let* $\mathbf{a}_i^* = -1$ *and* $\mathbf{V}_{i_\tau,i_\tau}^{*,(i)} = (2\,\mathrm{digit}(\mathrm{bin}(i_\tau - 1), i) - 1)/(4\gamma)$*.*
- *For* $i = \nu_2 + 1$ *and any* $i_\tau \in [d]$*, let* $\mathbf{a}_i^* = +1$ *and* $\mathbf{V}_{i_\tau,i_\tau}^{*,(i)} = -\nu_j/(4\gamma)$ *for* $g_j^*$*.*
- *For* $i \in [2(\nu_2 + 1)] \setminus [\nu_2 + 1]$*, let* $\mathbf{a}_i^* = \mathbf{a}_{i-\nu_2-1}^*$ *and* $\mathbf{V}^{*,(i)} = -\mathbf{V}^{*,(i-\nu_2-1)}$*.*

*Let* $\mathbf{W}^{*,(i)} = \mathbf{G}\mathbf{V}^{*,(i)}\mathbf{G}^\top$*. Up to permutations,* $g_2^*$ *has neurons* $(\mathbf{a}^*, \mathbf{W}^{*,(1)}, \ldots, \mathbf{W}^{*,(m)})$ *and* $g_1^*$ *has the* $\{1, \ldots, \nu_1, \nu_2 + 1, \nu_2 + 2 \ldots, \nu_2 + \nu_1 + 1, 2\nu_2 + 2\}$*-th neurons of* $g_2^*$*.*

*Proof sketch of Theorem 5.5.1.* The proof is challenging as the non-linear model and non-linear data. We defer the full proof to Section D.3.1. The high-level intuition is transferring the optimal solution to patterns covering problems. For small $p_{\mathcal{T}}$, the model will "prefer" to cover all patterns in $S_1$ first. When the model becomes larger, by checking the sufficient and necessary conditions, it will continually learn to cover non-important features. Thus, the smaller model will mainly focus on important features, while the larger model will focus on all features. □

**Example for Theorem 5.5.1.** When $\nu_2 = 3$, the optimal has $\mathbf{a}_1 = \mathbf{a}_2 = \mathbf{a}_3 = -1$, $\mathbf{a}_4 = +1$ and,

$$\mathbf{V}^{(1)} = 1/4\gamma \cdot \mathrm{diag}([-1, +1, -1, +1, -1, +1, -1, +1])$$

$$\mathbf{V}^{(2)} = 1/4\gamma \cdot \mathrm{diag}([-1, -1, +1, +1, -1, -1, +1, +1])$$

$$\mathbf{V}^{(3)} = 1/4\gamma \cdot \mathrm{diag}([-1, -1, -1, -1, +1, +1, +1, +1])$$

$$\mathbf{V}^{(4)} = 3/4\gamma \cdot \mathrm{diag}([-1, -1, -1, -1, -1, -1, -1, -1])$$

and $\mathbf{V}^{(i+4)} = -\mathbf{V}^{(i)}, \mathbf{a}_{i+4} = \mathbf{a}_i$ for $i \in [4]$.

On the other hand, the optimal $g_1^*$ for $\nu_1 = 1$ has the $\{1, 4, 5, 8\}$-th neurons of $g_2^*$.

By carefully checking, we can see that the neurons in $g_1^*$ (i.e., the $\{1, 4, 5, 8\}$-th neurons of $g_2^*$) are used for parity classification task from $S_1$, i.e, label determined by the first $k = 2^{\nu_1} = 2$ dimensions. With the other neurons (i.e., the $\{2, 3, 6, 7\}$-th neurons of $g_2^*$), $g_2^*$ can further do parity classification on the task from $S_2$, label determined by any two dimensions other than the first two dimensions.

**What hidden features does the model pay attention to?** Theorem 5.5.1 gives the closed-form optimal solution of two-layer multiple-head transformers learning sparse-parity ICL tasks. It shows the optimal solution of the smaller model indeed is a sub-model of the larger optimal model. In detail, the smaller model will mainly learn all important features, while the larger model will learn more features. This again shows a trade-off when increasing the model scale: larger models can learn more hidden features which can be beneficial if these features are relevant to the label, but also potentially keep more noise which is harmful.

### 5.5.2 Behavior Difference

Similar to Theorem 5.4.3, to illustrate our insights, we will consider a setting where the smaller model learns useful features for the evaluation task while the larger model covers extra features. That is, for evaluation, we uniformly draw a task $\mathbf{t}_\tau = (i_\tau, j_\tau)$ from $S_1$, and then draw $M$ samples to form the evaluation prompt in the same way as during pretraining. To present our theorem (proved in Section D.3.2 using Theorem 5.5.1), we introduce some notations. Let

$$\mathbf{D}_1 = \big[\mathrm{diag}(\mathbf{V}^{*,(1)}), \dots, \mathrm{diag}(\mathbf{V}^{*,(\nu_1)}), \mathrm{diag}(\mathbf{V}^{*,(\nu_2+1)}), \\ \dots, \mathrm{diag}(\mathbf{V}^{*,(\nu_2+\nu_1+1)}), \mathrm{diag}(\mathbf{V}^{*,(2\nu_2+2)})\big] \in \mathbb{R}^{d\times 2(\nu_1+1)}$$

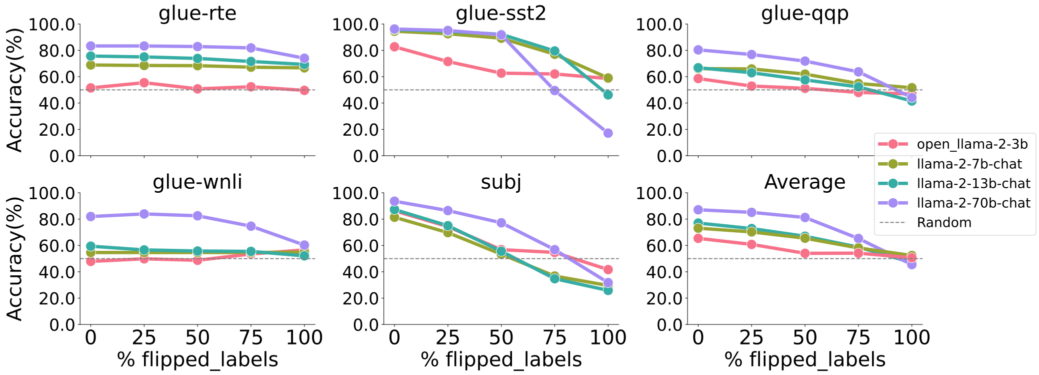


Figure 5.1: Larger models are easier to be affected by noise (flipped labels) and override pretrained biases than smaller models for different datasets and model families (chat/with instruct turning). Accuracy is calculated over 1000 evaluation prompts per dataset and over 5 runs with different random seeds for each evaluation, using $M = 16$ in-context exemplars.

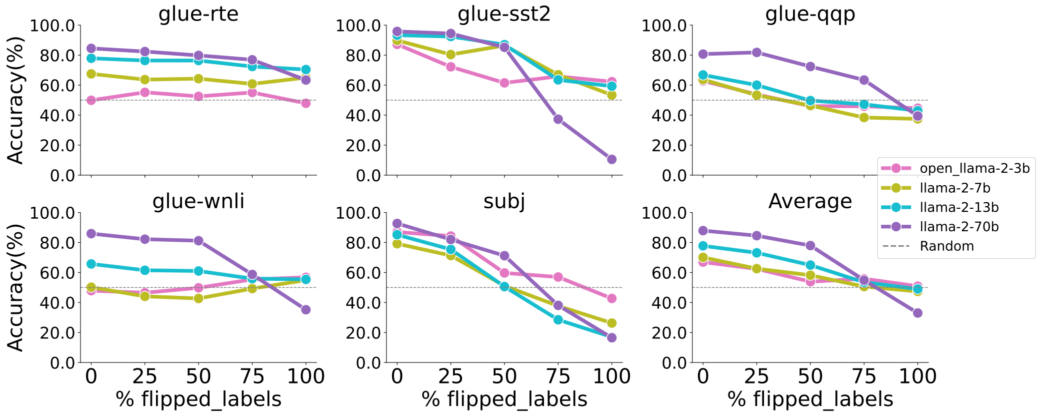


Figure 5.2: Larger models are easier to be affected by noise (flipped labels) and override pretrained biases than smaller models for different datasets and model families (original/without instruct turning). Accuracy is calculated over 1000 evaluation prompts per dataset and over 5 runs with different random seeds for each evaluation, using $M = 16$ in-context exemplars.

$$\mathbf{D}_2 = [\mathrm{diag}(\mathbf{V}^{*,(1)}), \ldots, \mathrm{diag}(\mathbf{V}^{*,(2\nu_2+2)})] \in \mathbb{R}^{d \times 2(\nu_2+1)},$$

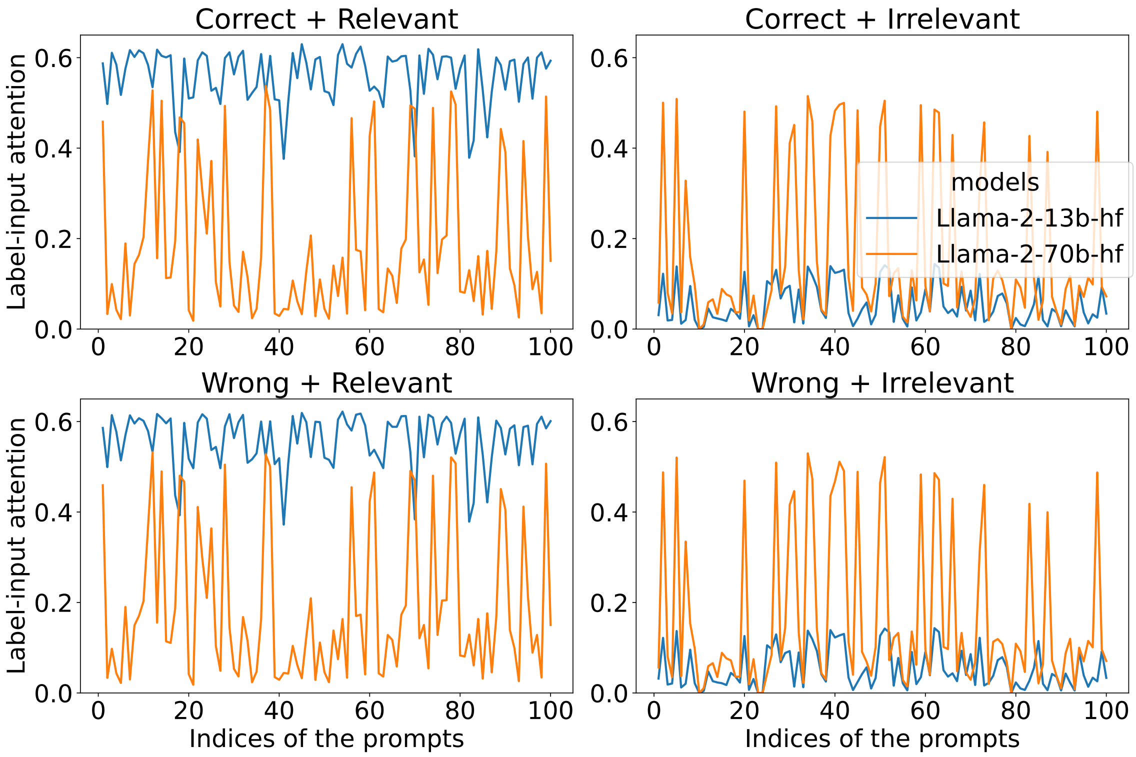


Figure 5.3: The magnitude of attention between the labels and input sentences in Llama 2-13b and 70b on 100 evaluation prompts; see the main text for the details. $x$-axis: indices of the prompts. $y$-axis: the norm of the last row of attention maps in the final layer. Correct: original label; wrong: flipped label; relevant: original input sentence; irrelevant: irrelevant sentence from other datasets. The results show that larger models focus on both sentences, while smaller models only focus on relevant sentences.

where for any $i \in [2(\nu_2 + 1)]$, $\mathbf{V}^{*,(i)}$ is defined in Theorem 5.5.1. Let $\hat{\phi}_{\tau,q} \in \mathbb{R}^d$ satisfy $\hat{\phi}_{\tau,q,i_\tau} = \phi_{\tau,q,i_\tau}, \hat{\phi}_{\tau,q,j_\tau} = \phi_{\tau,q,j_\tau}$ and all other entries being zero. For a matrix $\mathbf{Z}$ and a vector $\mathbf{v}$, let $P_{\mathbf{Z}}$ denote the projection of $\mathbf{v}$ to the space of $\mathbf{Z}$, i.e., $P_{\mathbf{Z}}(\mathbf{v}) = \mathbf{Z}(\mathbf{Z}^\top \mathbf{Z})^{-1}\mathbf{Z}^\top \mathbf{v}$.

**Theorem 5.5.2** (Behavior difference for parity)**.** *Assume the same condition as Theorem 5.5.1. For $j \in \{1,2\}$, Let $\theta_j$ denote the parameters of $g_j^*$. For $l \in [M]$, let $\xi_l$ be uniformly drawn from $\{\pm 1\}^d$, and $\Xi = \frac{\sum_{l\in[M]} \xi_l}{M}$. Then, for any $\delta \in (0,1)$, with probability at least*

$1 - \delta$ *over the randomness of test data, we have*

$$g_j^*(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}) = h(\theta_j, 2\gamma\hat{\phi}_{\tau,q} + P_{\mathbf{D}_j}(\Xi)) + \epsilon_j$$
$$:= \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\mathrm{diag}(\mathbf{V}^{*,(i)})^\top (2\gamma\hat{\phi}_{\tau,q} + P_{\mathbf{D}_j}(\Xi))] + \epsilon_j$$

*where* $\epsilon_j = O(\sqrt{\frac{\nu_j}{M} \log \frac{1}{\delta}})$ *and we have*

- $2\gamma\hat{\phi}_{\tau,q}$ *is the signal useful for prediction:* $0 = \ell(y_q \cdot h(\theta_1, 2\gamma\hat{\phi}_{\tau,q})) = \ell(y_q \cdot h(\theta_2, 2\gamma\hat{\phi}_{\tau,q}))$.
- $P_{\mathbf{D}_1}(\Xi))$ *and* $P_{\mathbf{D}_2}(\Xi))$ *is noise not related to labels, and* $\frac{\mathbb{E}[\|P_{\mathbf{D}_1}(\Xi))\|_2^2]}{\mathbb{E}[\|P_{\mathbf{D}_2}(\Xi))\|_2^2]} = \frac{\nu_1+1}{\nu_2+1}$.

**Implications.** Theorem 5.5.2 shows that during evaluation, we can decompose the input into two parts: signal and noise. Both the larger model and smaller model can capture the signal part well. However, the smaller model has a much smaller influence from noise than the larger model, i.e., the ratio is $\frac{\nu_1+1}{\nu_2+1}$. The reason is that smaller models emphasize important hidden features while larger ones cover more hidden features, and thus, smaller models are more robust to noise while larger ones are easily distracted, leading to different ICL behaviors. This again sheds light on where transformers pay attention to and how that affects ICL.

**Remark 5.2.** *Here, we provide a detailed intuition about Theorem 5.5.2.* $\Xi$ *is the input noise. When we only care about the noise part, we can rewrite the smaller model as* $g_1 = h(\theta_1, P_{D_1}(\Xi))$, *and the larger model as* $g_2 = h(\theta_2, P_{D_2}(\Xi))$, *where they share the same* $h$ *function. Our conclusion says that* $E[\|P_{D_1}(\Xi)\|_2^2]/E[\|P_{D_2}(\Xi)\|_2^2] = (\nu_1 + 1)/(\nu_2 + 1)$, *which means the smaller model's "effect" input noise is smaller than the larger model's "effect" input noise. Although their original input noise is the same, as the smaller model only focuses on limited features, the smaller model will ignore part of the noise, and the "effect" input noise is small. However, the larger model is the opposite.*

## 5.6 Experiments

Brilliant recent work Wei et al. (2023b) runs intensive and thorough experiments to show that larger language models do in-context learning differently. Following their idea, we conduct similar experiments on binary classification datasets, which is consistent with our problem setting in the parity case, to support our theory statements.

**Experimental setup.** Following the experimental protocols in Wei et al. (2023b); Min et al. (2022), we conduct experiments on five prevalent NLP tasks, leveraging datasets from GLUE Wang et al. (2018) tasks and Subj Conneau and Kiela (2018). Our experiments utilize various sizes of the Llama model families Touvron et al. (2023a,b): 3B, 7B, 13B, 70B. We follow the prior work on in-context learning Wei et al. (2023b) and use $M = 16$ in-context exemplars. We aim to assess the models' ability to use inherent semantic biases from pretraining when facing in-context examples. As part of this experiment, we introduce noise by inverting an escalating percentage of in-context example labels. To illustrate, a 100% label inversion for the SST-2 dataset implies that every "positive" exemplar is now labeled "negative". Note that while we manipulate the in-context example labels, the evaluation sample labels remain consistent. We use the same templates as Min et al. (2021), a sample evaluation for SST-2 when $M = 2$:

```
sentence: show us a good time
The answer is positive.

sentence: as dumb and cheesy
The answer is negative.

sentence: it 's a charming and often
affecting journey
The answer is
```

### 5.6.1 Behavior Difference

Figure 5.1 shows the result of model performance (chat/with instruct turning) across all datasets with respect to the proportion of labels that are flipped. When 0% label flips, we observe that larger language models have better in-context abilities. On the other hand, the performance decrease facing noise is more significant for larger models. As the percentage of label alterations increases, which can be viewed as increasing label noise $\sigma^2$, the performance of small models remains flat and seldom is worse than random guessing while large models are easily affected by the noise, as predicted by our analysis. These results indicate that large models can override their pretraining biases in-context input-label correlations, while small models may not and are more robust to noise. This observation aligns with the findings in Wei et al. (2023b) and our analysis.

We can see a similar or even stronger phenomenon in Figure 5.2: larger models are more easily affected by noise (flipped labels) and override pretrained biases than smaller models for the original/without instruct turning version (see the “Average” sub-figure). On the one hand, we conclude that both large base models and large chat models suffer from ICL robustness issues. On the other hand, this is also consistent with recent work suggesting that instruction tuning will impair LLM’s in-context learning capability.

### 5.6.2 Ablation Study

To further verify our analysis, we provide an ablation study. We concatenate an irrelevant sentence from GSM-IC (Shi et al., 2023a) to an input-label pair sentence from SST-2 in GLUE dataset. We use “correct” to denote the original label and “wrong” to denote the flipped label. Then, we measure the magnitude of correlation between label-input, by computing the norm of the last row of attention maps across all heads in the final layer. We do this between “correct”/“wrong” labels and the original/irrelevant inserted sentences. Figure 5.3 shows the results on 100 evaluation prompts; for example, the subfigure Correct+Relevant shows the correlation magnitude between the “correct” label and the original input sentence in each prompt. The results show that the small model Llama 2-13b mainly focuses on the relevant part (original input) and may ignore the irrelevant sentence, while the large model Llama 2-70b focuses on both sentences. This well aligns with our analysis.

## 5.7 More Discussions about Noise

There are three kinds of noise covered in our analysis:

**Pretraining noise.** We can see it as toxic or harmful pretraining data on the website (noisy training data). The model will learn these features and patterns. It is covered by $\xi$ in the linear regression case and $S_2$ in the parity case.

**Input noise during inference.** We can see it as natural noise as the user's wrong spelling or biased sampling. It is a finite sampling error as $x$ drawn from the Gaussian distribution for the linear regression case and a finite sampling error as $x$ drawn from a uniform distribution for the parity case.

**Label noise during inference.** We can see it as adversarial examples, or misleading instructions, e.g., deliberately letting a model generate a wrong fact conclusion or harmful solution, e.g., poison making. It is $\sigma$ in the linear regression case and $S_2$ in the parity case.

For pretraining noise, it will induce the model to learn noisy or harmful features. During inference, for input noise and label noise, the larger model will pay additional attention to these noisy or harmful features in the input and label pair, i.e., $y \cdot x$, so that the input and label noise may cause a large perturbation in the final results. If there is no pretraining noise, then the larger model will have as good robustness as the smaller model. Also, if there is no input and label noise, the larger model will have as good robustness as the smaller model. The robustness gap only happens when both pretraining noise and inference noise exist simultaneously.

## 5.8 Conclusion

In this work, we answered our research question: why do larger language models do in-context learning differently? Our theoretical study showed that smaller models emphasize important hidden features while larger ones cover more hidden features, and thus the former are more robust to noise while the latter are more easily distracted, leading to different behaviors during in-context learning. Our empirical results provided positive support for the theoretical analysis. Our findings can help improve understanding of LLMs and ICL, and better training and application of these models.

# 6 DOMAIN GENERALIZATION VIA NUCLEAR NORM REGULARIZATION

**Contribution statement.** This chapter is joint work with Yifei Ming, Ying Fan, Frederic Sala, and Yingyu Liang. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis, and completed all the experiments. The results of this chapter have been published as a conference paper in CPAL 2024 (Shi et al., 2023c).

## 6.1 Introduction

Making machine learning models reliable under distributional shifts is crucial for real-world applications such as autonomous driving, health risk prediction, and medical imaging. This motivates the area of domain generalization, which aims to obtain models that generalize to unseen domains, e.g., different image backgrounds or different image styles, by learning from a limited set of training domains. To improve model robustness under domain shifts, a plethora of algorithms have been recently proposed Zhang et al. (2018); Bahng et al. (2020); Wang et al. (2020b); Luo et al. (2020); Yao et al. (2022a). In particular, methods that learn invariant feature representations (class-relevant patterns) or invariant predictors Arjovsky et al. (2019) across domains demonstrate promising performance both empirically and theoretically Ganin et al. (2016); Li et al. (2018c); Sun and Saenko (2016); Li et al. (2018b).

Despite this, it remains challenging to improve on empirical risk minimization (ERM) when evaluating a broad range of real-world datasets Gulrajani and Lopez-Paz (2021); Koh et al. (2021). Notice that ERM is a reasonable baseline method since it must use invariant features to achieve optimal in-distribution performance. It has been empirically shown Rosenfeld et al. (2022) that ERM already learns “invariant" features sufficient for domain generalization, which means these features are only correlated with the class label, not domains or environments.

Although competitive in domain generalization tasks, the main issue ERM faces is that the invariant features it learns can be arbitrarily mixed: environmental features are hard to disentangle from invariant features. Various regularization techniques that control empirical risks across domains have been proposed Arjovsky et al. (2019); Krueger et al. (2021); Rosenfeld et al. (2021), but few directly regularize ERM, motivating this work.

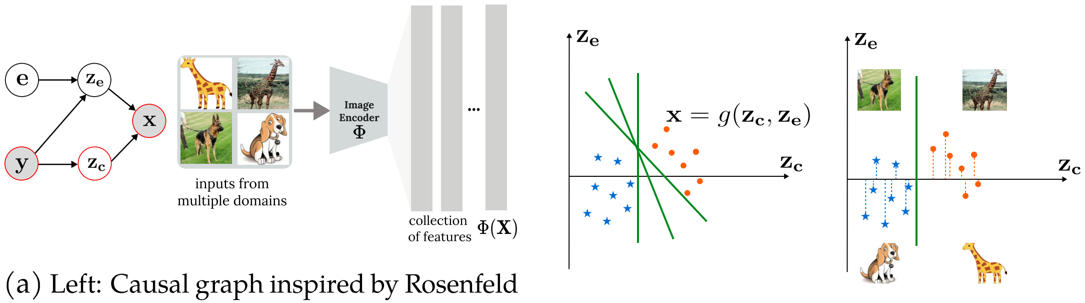
One desired property to improve ERM is disentangling the invariant features from the mixtures. A natural way to achieve this is to identify the subset of solutions from ERM with minimal information retrieved from training domains by controlling the rank. This avoids domain overfitting. We are interested in the following question: *can ERM benefit from rank regularization of the extracted feature for better domain generalization performance?*

To answer this question, we propose a simple yet effective algorithm, ERM-NU (Empirical Risk Minimization with Nuclear Norm Regularization), for improving domain generalization without acquiring domain annotations. Our method is inspired by works in low-rank matrix completion and recovery with nuclear norm minimization Candes and Romberg (2006); Candès et al. (2011); Chandrasekaran et al. (2011); Candes and Recht (2012); Korlakai Vinayak et al. (2014); Gu et al. (2014). Given latent feature representations from pre-trained models via ERM, ERM-NU aims to extract class-related (domain-invariant) features by fine-tuning the network with nuclear norm regularization. Specifically, we propose to minimize the nuclear norm of the backbone features, which is a convex envelope to the rank of the feature matrix Recht et al. (2010).

Our main contributions and findings are as follows:

- ERM-NU offers competitive empirical performance: We evaluate the performance of ERM-NU on synthetic datasets and five benchmark real-world datasets. Despite its simplicity, NU demonstrates strong performance and improves on existing methods on some large-scale datasets such as TerraInc and DomainNet.

- We provide theoretical guarantees when applying ERM-NU to domain gener-



(a) Left: Causal graph inspired by Rosenfeld et al. (2021). Shaded variables are observed. Right: Training pipeline. We collect input features from multiple domains.

(b) Left: ERM Solutions (green lines). Right: ERM solution with the smallest nuclear norm of extracted feature ($\mathbf{z_c}$ only).

Figure 6.1: Causal graph of our data assumption (6.1a), and the effect of nuclear norm regularization in ERM (6.1b) where we use a linear $g$ for a simple illustration. From Figure 6.1b, nuclear norm regularization can select a subset of ERM solutions that extract the smallest possible information (in the sense of rank) from $\mathbf{x}$ for classification, which can reduce the effect of environmental features for better generalization performance while still preserving high classification accuracy.

alization tasks: We show that even training with infinite data from in-domain (ID) tasks, ERM with weight decay may perform worse than random guessing on out-of-domain (OOD) tasks, while ERM with bounded rank (corresponding to ERM-NU) can guarantee 100% test accuracy on the out-of-domain task.

- Nuclear norm regularization (NU) is simple, efficient and broadly applicable: NU is computationally efficient as it does not require annotations from training domains. As a regularization, NU is also potentially orthogonal to other methods that are based on ERM: we get a consistent improvement of NU on ERM, Mixup Yan et al. (2020) and SWAD Cha et al. (2021) as baselines.

## 6.2 Method

We first provide the problem setup and a brief background of the nuclear norm and then introduce our method.

### 6.2.1 Preliminaries

We use $\mathcal{X}$ and $\mathcal{Y}$ to denote the input and label space, respectively. Following Koh et al. (2021); Yao et al. (2022a); Rosenfeld et al. (2021), we consider data distributions consisting of environments (domains) $\mathcal{E} = \{1, \dots, E\}$. For a given environment $e \in \mathcal{E}$ and label $y \in \mathcal{Y}$, the data generation process is the following: latent *environmental* features (e.g., image style or background information) $\mathbf{z}_e$ and *invariant* features (e.g., windows pattern for house images) $\mathbf{z}_c$ are sampled where invariant features only depend on $y$, while environmental features depend on $e$ and $y$ (i.e., environmental features and the label may have correlations), $\mathbf{z}_c \perp \mathbf{z}_e$. The input data is generated from the latent features $\mathbf{x} = g(\mathbf{z}_c, \mathbf{z}_e)$ by some injective function $g$. See illustration in Figure 6.1a. We assume that the training data is drawn from a mixture of $E^{tr} \subset \mathcal{E}$ domains and test data is drawn from some unseen domain in $E^{ts} \subset \mathcal{E}$. In the domain shift setup, training domains are disjoint from test domains: $\mathcal{E}^{tr} \cap \mathcal{E}^{ts} = \emptyset$. In this work, as we do not require domain annotations for training data, we remove notation involving $\mathcal{E}$ for simplicity and denote the training data distribution as $\mathcal{D}_{\text{id}}$ and the unseen domain test data distribution as $\mathcal{D}_{\text{ood}}$.

We consider population risk. Our objective is to learn a feature extactor $\Phi : \mathcal{X} \to \mathbb{R}^d$ that maps input data to a d-dimensional feature embedding (usually fine-tuned from a pre-trained backbone, e.g. ResNet He et al. (2016) pre-trained on ImageNet) and a classifier $\hat{f}$ to minimize the risk on *unseen* environments,

$$\mathcal{L}(\hat{f}, \Phi) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}\left[\ell(\hat{f}(\Phi(\mathbf{x})), y)\right], \tag{6.1}$$

where the function $\ell$ can be any loss appropriate to classification, e.g., cross-entropy.

The *nuclear norm* Fan (1951) (trace norm) of a matrix is the sum of the singular values of the matrix. Suppose a matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$, we have the nuclear norm

$$\|\mathbf{M}\|_* := \sum_{i}^{\min\{m,n\}} \sigma_i(\mathbf{M}),$$

where $\sigma_i(\mathbf{M})$ is the $i$-th largest singular value. From Recht et al. (2010), we know that the nuclear norm is the tightest convex envelope of the rank function of a matrix within the unit ball, i.e, the nuclear norm is smaller than the rank when the operator norm (spectral norm) $\|\mathbf{M}\|_2 = \sigma_1(\mathbf{M}) \leqslant 1$. As the matrix rank function is highly non-convex, nuclear norm regularization is often used in optimization to achieve a low-rank solution, as it has good convergence guarantees, while the rank function does not.

### 6.2.2 Method description

**Intuition.** Intuitively, to guarantee low risk on $\mathcal{D}_{\text{ood}}$, $\Phi$ needs to rely only on invariant features for prediction. It must not use environmental features in order to avoid spurious correlations to ensure domain generalization. As environmental features depend on the label $y$ and the environment $e$ in Figure 6.1a, our main hypothesis is that *environmental features have a lower correlation with the label than the invariant features*. If our hypothesis is true, we can eliminate environmental features by constraining the rank of the learned representations from the training data while minimizing the empirical risk, i.e., the invariant features will be preserved (due to empirical risk minimization) and the environmental features will be removed (due to rank minimization).

**Objectives.** We consider fine-tuning the backbone (feature extractor) $\Phi$ with a linear prediction head. Denote the linear head as $\in \mathbb{R}^{d\times m}$, where $m$ is the class number. The goal of ERM is to minimize the expected risk

$$\mathcal{L}(,\Phi) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{in}}}\left[\ell(^\top\Phi(\mathbf{x}),y)\right].$$

Consider the latent vector $\Phi(\mathbf{x}) \in \mathbb{R}^d$. This vector may contain both environment-related and class-related features. In order to obtain just the class-related features, we would like for $\Phi$ to extract as little information as possible while simultaneously optimizing the ERM loss. See illustration in Figure 6.1b. Note that, we assume that the correlation between environmental features and labels is lower than the correlation between invariant features and labels. Let $\mathbf{X}$ be a batch of training data points (batch size $>$ d). To minimize information and so rule out environmental features, we minimize the rank of $\Phi(\mathbf{X})$. Our objective is

$$\min_{,\Phi} \mathcal{L}(,\Phi) + \lambda \mathrm{rank}(\Phi(\mathbf{X})). \tag{6.2}$$

As the nuclear norm is a convex envelope to the rank of a matrix, our convex relaxation objective is

$$\min_{,\Phi} \mathcal{L}(,\Phi) + \lambda \|\Phi(\mathbf{X})\|_*, \tag{6.3}$$

where $\lambda$ is the regularization weight.

**Takeaways.** We summarize the advantages of nuclear norm minimization as follows:

- Simple and efficient: our method can be easily implemented. For example, ERM-NU only needs two more lines of code than ERM, as shown below.
- Broadly applicable: without requiring domain labels, our method can be used in conjunction with a broad range of existing domain generalization algorithms.
- Empirically effective and theoretically sound: our method demonstrates promising performance on synthetic and real-world tasks (Section 6.3) with theoretical insights presented in Section 6.4.

```
def forward(self, x, y):
    # calculate classification loss
    feature = self.featurizer(x)
    preds = self.classifier(feature)
    loss = F.cross_entropy(preds, y)
    # add nuclear norm regularization
    _,s,_ = torch.svd(feature)
    loss += self.lambda * torch.sum(s)
    return loss
```

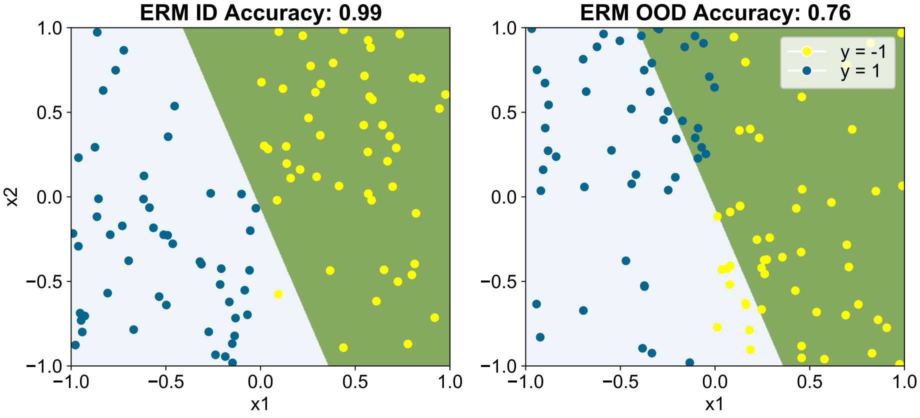


Figure 6.2: ID and OOD classification results with ERM on the synthetic dataset with two classes (shown in yellow and navy blue). We visualize the decision boundary. While the model achieves nearly perfect accuracy on ID training set, the performance drastically degrades on the OOD test set.

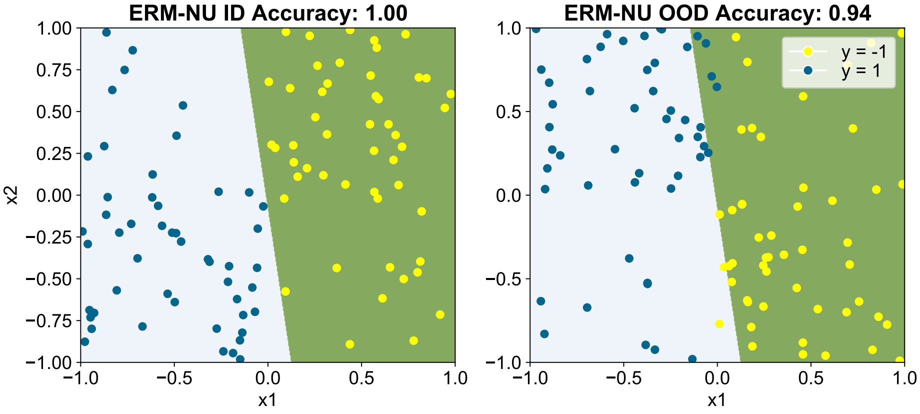


Figure 6.3: ID and OOD classification results with ERM-NU on the synthetic dataset. Nuclear norm regularization significantly reduces the OOD error rate.

## 6.3 Experiments

In this section, we start by presenting a synthetic task in Section 6.3.1 to help visualize the effects of nuclear norm regularization. Next, in Section 6.3.2, we demonstrate the effectiveness of our approach with real-world datasets. We pro-

| Algorithm | VLCS | PACS | OfficeHome | TerraInc | DomainNet | Average |
|---|---|---|---|---|---|---|
| MMD$^\dagger$ (CVPR 18) Li et al. (2018b) | 77.5 ± 0.9 | 84.6 ± 0.5 | 66.3 ± 0.1 | 42.2 ± 1.6 | 23.4 ± 9.5 | 58.8 |
| Mixstyle$^\ddagger$ (ICLR 21) Zhou et al. (2021a) | 77.9 ± 0.5 | 85.2 ± 0.3 | 60.4 ± 0.3 | 44.0 ± 0.7 | 34.0 ± 0.1 | 60.3 |
| GroupDRO$^\dagger$ (ICLR 19) Sagawa et al. (2019) | 76.7 ± 0.6 | 84.4 ± 0.8 | 66.0 ± 0.7 | 43.2 ± 1.1 | 33.3 ± 0.2 | 60.7 |
| IRM$^\dagger$ (ArXiv 20) Arjovsky et al. (2019) | 78.5 ± 0.5 | 83.5 ± 0.8 | 64.3 ± 2.2 | 47.6 ± 0.8 | 33.9 ± 2.8 | 61.6 |
| ARM$^\dagger$ (ArXiv 20) Zhang et al. (2020) | 77.6 ± 0.3 | 85.1 ± 0.4 | 64.8 ± 0.3 | 45.5 ± 0.3 | 35.5 ± 0.2 | 61.7 |
| VREx$^\dagger$ (ICML 21) Krueger et al. (2021) | 78.3 ± 0.2 | 84.9 ± 0.6 | 66.4 ± 0.6 | 46.4 ± 0.6 | 33.6 ± 2.9 | 61.9 |
| CDANN$^\dagger$ (ECCV 18) Li et al. (2018c) | 77.5 ± 0.1 | 82.6 ± 0.9 | 65.8 ± 1.3 | 45.8 ± 1.6 | 38.3 ± 0.3 | 62.0 |
| AND-mask$^*$ (ICLR 20)Parascandolo et al. (2020) | 78.1 ± 0.9 | 84.4 ± 0.9 | 65.6 ± 0.4 | 44.6 ± 0.3 | 37.2 ± 0.6 | 62.0 |
| DANN$^\dagger$ (JMLR 16) Ganin et al. (2016) | 78.6 ± 0.4 | 83.6 ± 0.4 | 65.9 ± 0.6 | 46.7 ± 0.5 | 38.3 ± 0.1 | 62.6 |
| RSC$^\dagger$ (ECCV 20) Huang et al. (2020) | 77.1 ± 0.5 | 85.2 ± 0.9 | 65.5 ± 0.9 | 46.6 ± 1.0 | 38.9 ± 0.5 | 62.7 |
| MTL$^\dagger$ (JMLR 21) Blanchard et al. (2021) | 77.2 ± 0.4 | 84.6 ± 0.5 | 66.4 ± 0.5 | 45.6 ± 1.2 | 40.6 ± 0.1 | 62.9 |
| Mixup$^\dagger$ (ICLR 18) Zhang et al. (2018) | 77.4 ± 0.6 | 84.6 ± 0.6 | 68.1 ± 0.3 | 47.9 ± 0.8 | 39.2 ± 0.1 | 63.4 |
| MLDG$^\dagger$ (AAAI 18) Li et al. (2018a) | 77.2 ± 0.4 | 84.9 ± 1.0 | 66.8 ± 0.6 | 47.7 ± 0.9 | 41.2 ± 0.1 | 63.6 |
| Fish (ICLR 22) Shi et al. (2022a) | 77.8 ± 0.3 | 85.5 ± 0.3 | 68.6 ± 0.4 | 45.1 ± 1.3 | 42.7 ± 0.2 | 63.9 |
| Fishr$^*$ (ICML 22) Rame et al. (2022) | 77.8 ± 0.1 | 85.5 ± 0.4 | 67.8 ± 0.1 | 47.4 ± 1.6 | 41.7 ± 0.0 | 64.0 |
| SagNet$^\dagger$ (CVPR 21) Nam et al. (2021) | 77.8 ± 0.5 | 86.3 ± 0.2 | 68.1 ± 0.1 | 48.6 ± 1.0 | 40.3 ± 0.1 | 64.2 |
| SelfReg (ICCV 21) Kim et al. (2021) | 77.8 ± 0.9 | 85.6 ± 0.4 | 67.9 ± 0.7 | 47.0 ± 0.3 | 41.5 ± 0.2 | 64.2 |
| CORAL$^\dagger$ (ECCV 16) Sun and Saenko (2016) | 78.8 ± 0.6 | 86.2 ± 0.3 | 68.7 ± 0.3 | 47.6 ± 1.0 | 41.5 ± 0.1 | 64.5 |
| SAM$^\ddagger$ (ICLR 21) Foret et al. (2021) | 79.4 ± 0.1 | 85.8 ± 0.2 | 69.6 ± 0.1 | 43.3 ± 0.7 | 44.3 ± 0.0 | 64.5 |
| mDSDI (NeurIPS 21) Bui et al. (2021) | 79.0 ± 0.3 | 86.2 ± 0.2 | 69.2 ± 0.4 | 48.1 ± 1.4 | 42.8 ± 0.1 | 65.1 |
| MIRO (ECCV 22) Cha et al. (2022) | 79.0 ± 0.0 | 85.4 ± 0.4 | 70.5 ± 0.4 | 50.4 ± 1.1 | 44.3 ± 0.2 | 65.9 |
| ERM$^\dagger$ Vapnik (1999) | 77.5 ± 0.4 | 85.5 ± 0.2 | 66.5 ± 0.3 | 46.1 ± 1.8 | 40.9 ± 0.1 | 63.3 |
| **ERM-NU (ours)** | **78.3** ± 0.3 | **85.6** ± 0.1 | **68.1** ± 0.1 | **49.6** ± 0.6 | **43.4** ± 0.1 | **65.0** |
| SWAD$^\ddagger$ (NeurIPS 21) Cha et al. (2021) | 79.1 ± 0.1 | 88.1 ± 0.1 | 70.6 ± 0.2 | 50.0 ± 0.3 | 46.5 ± 0.1 | 66.9 |
| **SWAD-NU (ours)** | **79.8** ± 0.2 | **88.5** ± 0.2 | **71.3** ± 0.3 | **52.2** ± 0.3 | **47.1** ± 0.1 | **67.8** |

Table 6.1: OOD accuracy for five realistic domain generalization datasets. The results marked by $^\dagger$, $^\ddagger$, $^*$ are the reported numbers from Gulrajani and Lopez-Paz (2021), Cha et al. (2021), Rame et al. (2022) respectively. We highlight **our methods** in bold. The results of Fish, SelfReg, mDSDI and MIRO are the reported ones from each paper. Average accuracy and standard errors are reported from three trials. Nuclear norm regularization is simple, effective, and broadly applicable. It significantly improves the performance over ERM and a competitive baseline SWAD across all datasets considered.

vide further discussions and ablation studies in Section 6.3.3. Code is available at https://github.com/zhmeishi/DG-NU.

### 6.3.1 Synthetic tasks

To visualize the effects of nuclear norm regularization, we start with a synthetic dataset with binary labels and two-dimensional inputs. Our expectation is that unregularized ERM will perform well for ID but struggle for OOD data, whereas nuclear norm-regularized ERM will excel in both settings. Assume inputs $\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2]$, where $\mathbf{x}_1$ is the invariant feature and $\mathbf{x}_2$ is the environmental feature. Specifically, $\mathbf{x}_1$ is drawn from a uniform distribution conditioned on $y \in \{-1, 1\}$ for both ID (training) and OOD (test) datasets:

$$\mathbf{x}_1 \mid y = 1 \sim \mathcal{U}[0, 1], \quad \mathbf{x}_1 \mid y = -1 \sim \mathcal{U}[-1, 0]$$

The environmental feature $\mathbf{x}_2$ also follows a uniform distribution but is conditioned on both $y$ and a Bernoulli random variable $b \sim \mathrm{Ber}(0.7)$. For ID data, $\mathbf{x}_2 \mid y = 1 \sim \mathcal{U}[0, 1]$ with probability (w.p.) 0.7, while $\mathbf{x}_2 \mid y = 1 \sim \mathcal{U}[-1, 0]$ w.p. 0.3. In contrast, for OOD data, $\mathbf{x}_2 \mid y = 1 \sim \mathcal{U}[-1, 0]$ w.p. 0.7, and $\mathbf{x}_2 \mid y = 1 \sim \mathcal{U}[0, 1]$ w.p. 0.3. We provide a more general setting in Section 6.4.

We visualize the ID and OOD datasets in Figure 6.2, where samples from $y = -1$ and $y = 1$ are shown in yellow and navy blue dots, respectively. We consider a simple linear feature extractor $\Phi(\mathbf{x}) = A\mathbf{x}$ with $A \in \mathbb{R}^{2\times 2}$. The models are trained with ERM and ERM-NU objectives using gradient descent until convergence. The results are shown in Figure 6.2 and Figure 6.3, respectively. To better illustrate the effects of nuclear norm minimization, we show the decision boundary along with the accuracy on ID and OOD datasets. For ID dataset, training with both objective yield nearly perfect accuracy. For OOD dataset, the model trained with ERM only achieves an accuracy of 0.76, as a result of utilizing the environmental feature. In contrast, training with ERM-NU successfully mitigates the reliance on environmental features and significantly improves the OOD accuracy to 0.94. In Section 6.4, we further provide theoretical analysis to better understand the effects of nuclear norm regularization.

### 6.3.2 Real-world tasks

In this section, we demonstrate the effects of nuclear norm regularization across real-world datasets and compare with a broad range of algorithms.

**Experimental setup.** Nuclear norm regularization is simple, flexible, and can be plugged into ERM-like algorithms. To verify its effectiveness, we consider adding the regularizer over ERM and SWAD (dubbed as ERM-NU and SWAD-NU, respectively). For a fair comparison with baseline methods, we evaluate our algorithm on the DomainBed testbed Gulrajani and Lopez-Paz (2021), an open-source benchmark that aims to rigorously compare different algorithms for domain generalization. The testbed consists of a wide range of datasets for multi-domain image classification tasks, including PACS Li et al. (2017) (4 domains, 7 classes, and 9,991 images), VLCS Fang et al. (2013) (4 domains, 5 classes, and 10,729 images), Office-Home Venkateswara et al. (2017), Terra Incognita Beery et al. (2018) (4 domains, 10 classes, and 24,788 images), and DomainNet Peng et al. (2019) (6 domains, 345 classes, and 586,575 images). Following the evaluation protocol in DomainBed, we report all performance scores by "leave-one-out cross-validation", where averaging over cases that use one domain as the test (OOD) domain and all others as the training (ID) domains. For the model selection criterion, we use the "training-domain validation set" strategy, which refers to choosing the model maximizing the accuracy on the overall validation set, 20% of training domain data. For each dataset and model, we report the test domain accuracy of the best-selected model (average over three independent runs with different random seeds). Following common practice, we use ResNet-50 He et al. (2016) as the feature backbone. We use the output features of the penultimate layer of ResNet-50 (2048-dim) for nuclear norm regularization and fine-tune the whole model. The default value of weight scale $\lambda$ is set as 0.01 and distributions for random search as $10^{\text{Uniform }(-2.5,-1.5)}$. The default batch size is 32 and the distribution for random search is $2^{\text{Uniform }(5,6)}$. During training, we perform batch-wise nuclear norm regularization, similar to Arjovsky et al. (2019) which uses batch-wise statistics for invariant risk minimization.

**Nuclear norm regularization achieves strong performance across a wide range of datasets.** We present an overview of the OOD accuracy for DomainBed datasets across various algorithms in Table 6.1. We observe that: (1) incorporating nuclear norm regularization consistently improves the performance of ERM and SWAD across all datasets considered. In particular, compared to ERM, ERM-NU yields an average accuracy improvement of 1.7%. (2) SWAD-NU demonstrates highly competitive performance relative to other baselines, including prior invariance-learning approaches such as IRM, VREx, and DANN. Notably, the approach does not require domain labels, which further underscores the versatility of nuclear norm regularization for real-world datasets.

**Nuclear norm regularization significantly improves baselines.** Across a range of realistic datasets, nuclear norm regularization enhances competitive baselines. To examine whether NU is effective with baselines other than SWAD, in Figure 6.4, we plot the average difference in accuracy with and without nuclear norm regularization for ERM, Mixup, and SWAD. Detailed results for individual datasets can be seen in Table 6.2. Encouragingly, adding nuclear norm regularization improves the performance over all three baselines across the five datasets. In particular, the average accuracy is improved by 3.5 with ERM-NU over ERM on Terra Incognita, and 3.1 with Mixup-NU over Mixup on DomainNet. This further suggests the effectiveness of nuclear norm regularization in learning invariant features. See full results in Appendix E.2.

### 6.3.3 Ablations and discussions

**Analyzing the regularization strength with stable rank.** We aim to better understand the strength of nuclear norm regularization on OOD accuracy. Due to the precision of floating point numbers and numerical perturbation, it is common to use stable rank (numerical rank) to approximate the matrix rank in numerical analysis. Suppose a matrix $\mathbf{M} \in \mathbb{R}^{m \times n}$, the stable rank is defined as:

$$\text{StableRank}(\mathbf{M}) := \frac{\|\mathbf{M}\|_F^2}{\|\mathbf{M}\|_2^2},$$

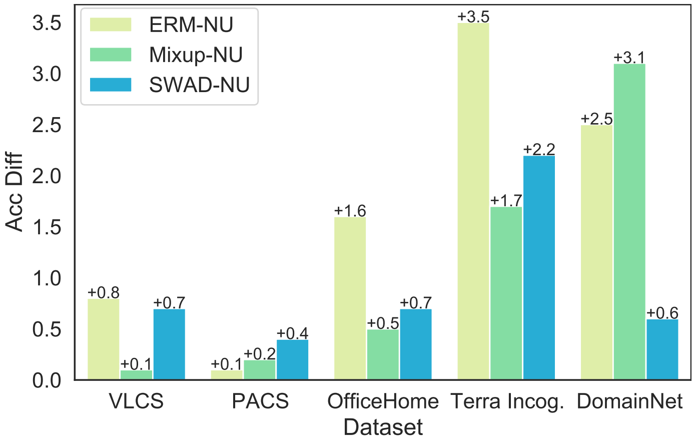
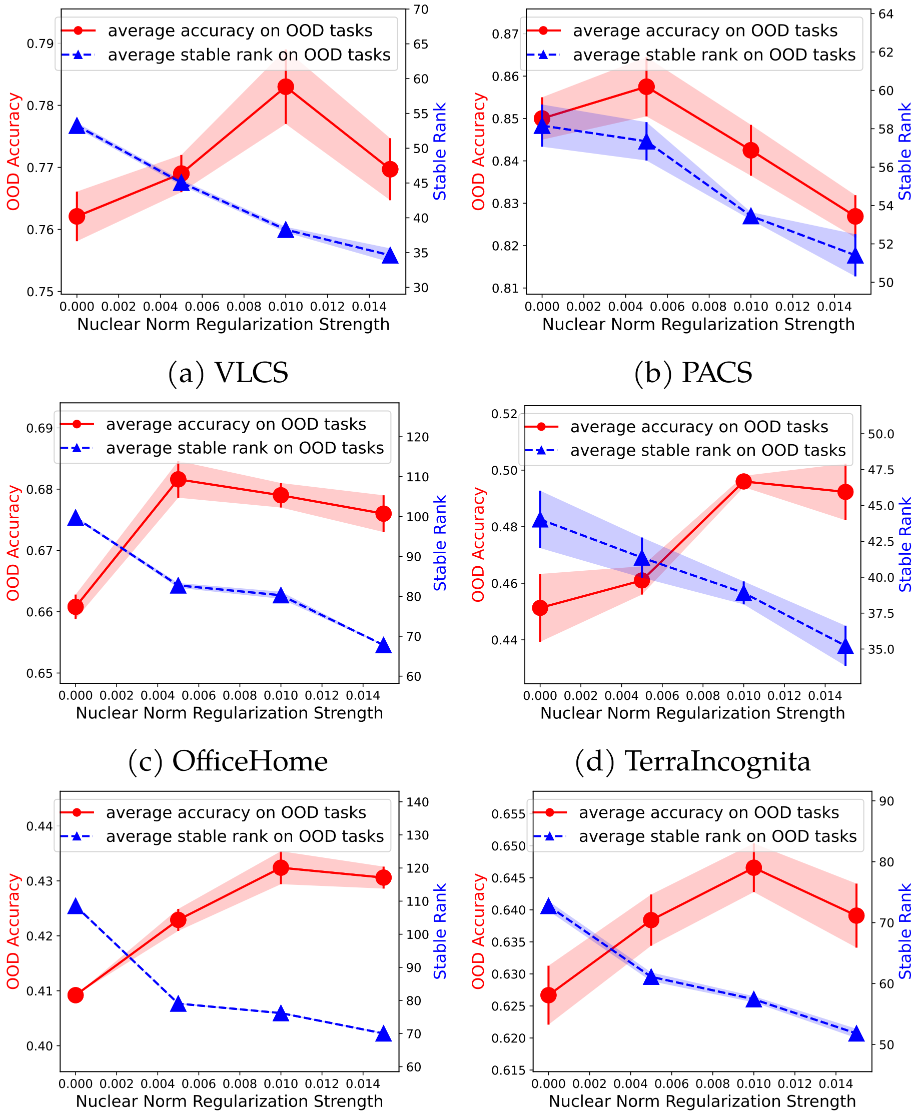


Figure 6.4: Nuclear norm regularization enhances competitive baselines across a range of realistic datasets, as demonstrated by the average difference in accuracy with nuclear norm regularization for ERM, Mixup, and SWAD. Detailed results for individual datasets can be seen in Table 6.2.

where $\|\|_2$ is the operator norm (spectral norm) and $\|\|_F$ is the Frobenius norm. The stable rank is analogous to the classical rank of a matrix but considerably more well-behaved. For example, the stable rank is a continuous and Lipschitz function while the rank function is discrete.

In Figure 6.5, we calculate the stable rank of the OOD data feature representation of the ERM-NU model trained with different nuclear norm regularization weight $\lambda$ and we plot the OOD accuracy simultaneously. We have three observations. (1) The stable rank will decrease when we have a stronger nuclear norm regularizer, which is consistent with our method intuition. (2) As nuclear norm regularization weight increases, the OOD accuracy will increase first and then decrease. In the first stage, as we increase nuclear norm regularization weight, the environmental features start to be ruled out and the OOD accuracy improves. In the second stage, when nuclear norm regularization strength is large enough, some invariant features will be ruled out, which will hurt the generalization. (3) Although the ResNet-50 is a 2048-dim feature extractor, the stable rank of the OOD data feature representation is pretty low, e.g, on average the stable rank is smaller than 100. On the other hand,



(a) VLCS

(b) PACS

(c) OfficeHome

(d) TerraIncognita

(e) DomainNet

(f) Average

Figure 6.5: Stable rank and OOD accuracy of ERM-NU with varying nuclear norm regularization weight λ (x-axis) on different datasets.

when the dataset becomes more "complicated", the stable rank will increase. For example, when $\lambda = 0.0$, the stable rank of DomainNet features (6 domains, 345 classes) is over 100, while the stable rank of VLCS (4 domains, 5 classes) features is only around 50.

**Exploring alternative regularizers.** We use SWAD Cha et al. (2021) as a baseline, which aims to find flat minima that suffers less from overfitting by a dense and overfit-aware stochastic weight sampling strategy. We consider different regularizers with SWAD: CORAL Sun and Saenko (2016) tries to minimize domain shift by aligning the second-order statistics of input data from training and test domains. MIRO Cha et al. (2022), one of the SOTA regularization methods in domain generalization, uses mutual information to reduce the distance between the pre-training model and the fine-tuned model. The performance comparison is shown in Table 6.3, where we observe that nuclear norm regularization consistently achieves competitive performance compared to alternative regularizers.

## 6.4 Theoretical Analysis

Next, we present a simple but insightful theoretical result showing that, for a more general setting defined in Section 6.3.1, the ERM-rank solution to the Equation equation 6.2 is much more robust than the ERM solution on OOD tasks.

**Data distributions.** Consider the binary classification setting. Let $\mathcal{X}$ be the input space, and $\mathcal{Y} = \{\pm 1\}$ be the label space. Let $\tilde{\mathbf{z}} : \mathcal{X} \to \mathbb{R}^d$ be a feature pattern encoder of the input data $\mathbf{x}$, i.e., $\tilde{\mathbf{z}}(\mathbf{x}) \in \mathbb{R}^d$. For any $j \in [d]$, we assume the feature embedder (defined in Section 6.2) $\Phi(\mathbf{x})_j = \mathbf{w}_j \tilde{\mathbf{z}}_j(\mathbf{x})$ where $\mathbf{w}_j$ is a scalar and $\tilde{\mathbf{z}}_j$ is a specific feature pattern encoder, i.e, $\tilde{\mathbf{z}}_j(\mathbf{x})$ being the j-th dimension of $\tilde{\mathbf{z}}(\mathbf{x})$. Suppose $\mathbf{x}$ are drawn from some distribution condition on label $y$, then we have $\tilde{\mathbf{z}}(\mathbf{x})$ drawn from some distribution condition on label $y$. We denote $\mathbf{z} = \tilde{\mathbf{z}}(\mathbf{x})y$ for simplicity. We assume, for any $j, j' \in [d]$, $\mathbf{z}_j, \mathbf{z}_{j'}$ are independent when $j \neq j'$.

Let $R \subseteq [d]$ be a subset of size $r$ corresponding to the class-relevant patterns (invariant features $\mathbf{z}_c$ in Figure 6.1a) and $U = [d] \setminus R$ be a subset of size $d - r$ corresponding to the spurious patterns (environmental features $\mathbf{z}_e$ in Figure 6.1a). For invariant features, we assume, for any $j \in R$, $\mathbf{z}_j \sim [0, 1]$ uniformly, so $\mathbb{E}[\mathbf{z}_j] = \frac{1}{2}$.

Next, we define in-domain (ID) tasks and out-of-domain (OOD) tasks. We first need to define $\mathcal{D}_\gamma$ where $\gamma \in (-\frac{1}{2}, \frac{1}{2})$. A random variable $z \sim \mathcal{D}_\gamma$ means, $z \sim [0, 1]$ uniformly with probability $\frac{1}{2} + \gamma$ and $z \sim [-1, 0]$ uniformly with probability $\frac{1}{2} - \gamma$, so $\mathbb{E}[z] = \gamma$. In ID tasks, for any environmental features $j \in \mathcal{U}$, we assume that $\mathbf{z}_j \sim \mathcal{D}_\gamma$, where $\gamma \in \left(\frac{3}{\sqrt{r}}, \frac{1}{2}\right)$. We denote this distribution as $\mathcal{D}_{\text{id}}$. In OOD tasks, for any $j \in \mathcal{U}$, we assume that $\mathbf{z}_j \sim \mathcal{D}_{-\gamma}$. We denote this distribution as $\mathcal{D}_{\text{ood}}$.

**Explanation and intuition for our data distributions.** There is an upper bound for $\gamma$ because the environmental features have a smaller correlation with the label than the invariant features, e.g., when $\gamma = \frac{1}{2}$ we cannot distinguish invariant features and environmental features. We also have a lower bound for $\gamma$ to distinguish the environmental features and noise. When $\gamma = 0$, the ID task and the OOD task will be identical (no distribution shift). We can somehow use $\gamma$ to measure the “distance" between the ID task and the OOD task. The intuition about the definition of $\mathcal{D}_{\text{id}}$ and $\mathcal{D}_{\text{ood}}$ is that the environmental features may have different correlations with labels in different tasks, while the invariant features keep the same correlations with labels through different tasks.

**Objectives.** We simplify the fine-tuning process, setting $= [1, 1, \ldots, 1]^\top$ and the trainable parameter to be $\mathbf{w}$ (varying the impact of each feature). Thus, the network output is $f_\mathbf{w}(\mathbf{x}) = \sum_{j=1}^{d} \mathbf{w}_j \tilde{\mathbf{z}}_j(\mathbf{x})$. We consider two objective functions. The first is traditional ERM with weight decay ($\ell_2$ norm regularization). The ERM-$\ell_2$ objective function is

$$\min_{\mathbf{w}} \mathcal{L}^\lambda(\mathbf{w}) := \mathcal{L}(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2, \tag{6.4}$$

where $\mathcal{L}_{(\mathbf{x},y)}(\mathbf{w}) = \ell(y f_\mathbf{w}(\mathbf{x}))$ is the loss on an example $(\mathbf{x}, y)$ and $\ell(z)$ is the logistic loss $\ell(z) = \ln(1 + \exp(-z))$.

The second objective we consider is ERM with bounded rank. Note that for a batch input data $\mathbf{X}$ with batch size $> d$, it is full rank with probability 1. Thus, we say the total feature rank is $\|\mathbf{w}\|_0 \leqslant d$ ($\|\mathbf{w}\|_0$ indicates the number of nonzero elements in $\mathbf{w}$). Thus, the ERM-rank objective function is

$$\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) \quad \text{subject to} \quad \|\mathbf{w}\|_0 \leqslant B_{\text{rank}}, \tag{6.5}$$

where $B_{\text{rank}}$ is a rank upper bound. The ERM-rank objective function is equivalent to Equation (6.2).

**Theoretical results.** First, we analyze the property of the optimal solution of ERM-$\ell_2$ on the ID task. Following the idea from Lemma B.1 of Shi et al. (2023b), we have the Lemma below.

**Lemma 6.1.** *Consider the ID setting with ERM-$\ell_2$ objective function. Then the optimal $\mathbf{w}^*$ for ERM-$\ell_2$ objective function following conditions (1) for any $j \in R$, $\mathbf{w}^*_j =: \alpha$; (2) for any $j \in U$, $\mathbf{w}^*_j := \beta$; (3) $0 < \beta < \alpha < \frac{1}{\sqrt{r}}$, $\frac{\alpha}{\beta} < \frac{3}{4\gamma}$.*

In Lemma 6.1, we show that the ERM-$\ell_2$ objective will encode all features correlated with labels, even when the correlation between spurious features and labels is weak (e.g. $\gamma = O\left(1/\sqrt{r}\right)$). However, the optimal solution of the ERM-rank objective will only encode the features which have a strong correlation with labels (intrinsic features), shown in Lemma 6.2.

**Lemma 6.2.** *Consider ID setting with ERM-rank objective function. Denote $R_{\text{rank}}$ is any subset of $R$ with size $|R_{\text{rank}}| = B_{\text{rank}}$, we have an optimal $\mathbf{w}^*$ for ERM-rank objective function following conditions (1) for any $j \in R_{\text{rank}}$, $\mathbf{w}^*_j > 0$ and (2) for any $j \notin R_{\text{rank}}$, $\mathbf{w}^*_j = 0$.*

Based on the property of two optimal solutions, we can show the performance gap between these two optimal solutions on the OOD task, considering the spurious features may change their correlation to the labels in different tasks.

**Proposition 6.3.** *Assume $1 \leqslant B_{\text{rank}} \leqslant r, \lambda > \Omega\left(\frac{\sqrt{r}}{\exp\left(\frac{\sqrt{r}}{5}\right)}\right), d > \frac{r}{\gamma^2} + r, r > C$, where $C$ is some constant $< 20$. The optimal solution for the ERM-rank objective function on the ID tasks has 100% OOD test accuracy, while the optimal solution for the ERM-$\ell_2$ objective* *function on the ID tasks has OOD test accuracy at most* $\exp\left(-\frac{r}{10}\right) \times 100\%$ (*much worse than random guessing*).

**Discussions.** The assumption of $\lambda$ and $d$ means that the regularization strength cannot be too small and the environmental features signal level should be compatible with invariant features signal level. Then, Proposition 6.3 shows that even with infinite data, the optimal solution for ERM-$\ell_2$ on the ID tasks cannot produce better performance than random guessing on the OOD task. However, the optimal solution for ERM-rank on the ID tasks can still produce 100% test accuracy. The proof idea is that, by using the gradient equal to zero and the properties of the logistic loss, the ERM-$\ell_2$ objective will encode all features correlated with labels, even when the correlation between spurious features and label is weak (e.g. $\gamma = O\left(1/\sqrt{r}\right)$). Moreover, there is a positive correlation between the feature encoding strength and the corresponding feature-label correlation (Lemma 6.1 (3)). Then, we can show that the value of $\beta$ is compatible with the value of $\alpha$ in Lemma 6.1. Thus, when the OOD tasks have a different spurious feature distribution, the optimal solution of ERM-$\ell_2$ objective may thoroughly fail, i.e., much worse than random guessing. However, the optimal solution of the ERM-rank objective will only encode the features which have a strong correlation with labels (intrinsic features). Thus, it can guarantee 100% test accuracy on OOD tasks. See the full proof in Appendix E.1.

## 6.5 Related Works

**Nuclear norm minimization.** Nuclear norm is commonly used to approximate the matrix rank Recht et al. (2010). Nuclear norm minimization has been widely used in many areas where the solution is expected to have a low-rank structure. It has been widely applied for low-rank matrix approximation, completion, and recovery Candes and Romberg (2006); Candès et al. (2011); Chandrasekaran et al. (2011); Candes and Recht (2012) with applications such as graph clustering Korlakai Vinayak et al. (2014); Demirel et al. (2022), community detection Li et al. (2021), compressed sensing Dong et al. (2014), recommendations system Koren et al. (2009) and robust Principal Component Analysis  Lu et al. (2019). Nuclear norm regularization can also be used in multi-task learning to learn shared representations across multiple tasks, which can lead to improved generalization and reduce overfitting Kumar and Daume III (2012). Nuclear norm has been used in computer vision as well to solve problems such as image denoising Gu et al. (2014) and image restoration Yair and Michaeli (2018). In this work, we focus on utilizing nuclear norm-based regularization for domain generalization. We provide extensive experiments on synthetic and realistic datasets and theoretical analysis to better understand its effectiveness.

**Contextual bias in computer vision.** There has been rich literature studying the classification performance in the presence of pre-defined contextual bias and spurious correlations Torralba (2003); Beery et al. (2018); Barbu et al. (2019); Bahng et al. (2020); Ming et al. (2022). The reliance on contextual bias such as image backgrounds, texture, and color for object detection has also been explored Zhu et al. (2017); Baker et al. (2018); Geirhos et al. (2019); Zech et al. (2018); Xiao et al. (2021); Sagawa et al. (2019); Shi et al. (2022b); Yang et al. (2019). In contrast, our study requires no prior information on the type of contextual bias and is broadly applicable to different categories of bias.

**Domain generalization and group robustness.** The task of domain generalization aims to improve the classification performance of models on new test domains. A plethora of algorithms are proposed in recent years: learning domain invariant Ganin et al. (2016); Li et al. (2018c); Sun and Saenko (2016); Li et al. (2018b); Meng et al. (2022b) and domain specific features Bui et al. (2021), minimizing the weighted combination of risks from training domains Sagawa et al. (2019), mixing risk penalty terms to facilitate invariance prediction Arjovsky et al. (2019); Krueger et al. (2021), prototype-based contrastive learning Yao et al. (2022b), meta-learning Dou et al. (2019), and data-centric approaches such as generation Zhou et al. (2020) and mixup Zhang et al. (2018); Wang et al. (2020b); Luo et al. (2020); Yao et al. (2022a). Recent works also demonstrate promising results with pre-trained models Cha et al. (2022); Kirichenko et al. (2023); Li et al. (2023a); Zhang et al. (2022b); Dubois et al. (2022). Beyond domain generalization, another closely-related task is to improve the group robustness in the presence of spurious correlations Sagawa et al. (2019); Liu et al. (2021a); Zhang et al. (2022a). However, recent works often assume access to group labels for a small dataset or require multiple stages of training. In contrast, our approach is simple and efficient, requiring no access to domain labels or multi-stage training, and can improve over ERM-like algorithms on a broad range of real-world datasets.

## 6.6 Conclusions

In this work, we propose nuclear norm minimization, a simple yet effective regularization method for improving domain generalization without acquiring domain annotations. Key to our method is minimizing the nuclear norm of the feature embeddings as a convex proxy for rank minimization. Empirically, we show that our method is broadly applicable with ERM and other competitive algorithms for domain generalization and achieves competitive performance across synthetic and a wide range of real-world datasets. Theoretically, we show that it outperforms ERM with $\ell_2$ regularization in the linear setting. We hope our work will inspire effective algorithm design and promote a better understanding of domain generalization.

| Algorithm | C | L | S | V | Average |
|---|---|---|---|---|---|
| ERM | 97.7 ± 0.4 | 64.3 ± 0.9 | 73.4 ± 0.5 | 74.6 ± 1.3 | 77.5 |
| ERM-NU | 97.9 ± 0.4 | 65.1 ± 0.3 | 73.2 ± 0.9 | 76.9 ± 0.5 | **78.3** |
| Mixup | 98.3 ± 0.6 | 64.8 ± 1.0 | 72.1 ± 0.5 | 74.3 ± 0.8 | 77.4 |
| Mixup-NU | 97.9 ± 0.2 | 64.1 ± 1.4 | 73.1 ± 0.9 | 74.8 ± 0.5 | **77.5** |
| SWAD | 98.8 ± 0.1 | 63.3 ± 0.3 | 75.3 ± 0.5 | 79.2 ± 0.6 | 79.1 |
| SWAD-NU | 99.1 ± 0.4 | 63.6 ± 0.4 | 75.9 ± 0.4 | 80.5 ± 1.0 | **79.8** |

(a) VLCS

| Algorithm | A | C | P | S | Average |
|---|---|---|---|---|---|
| ERM | 84.7 ± 0.4 | 80.8 ± 0.6 | 97.2 ± 0.3 | 79.3 ± 1.0 | 85.5 |
| ERM-NU | 87.4 ± 0.5 | 79.6 ± 0.9 | 96.3 ± 0.7 | 79.0 ± 0.5 | **85.6** |
| Mixup | 86.1 ± 0.5 | 78.9 ± 0.8 | 97.6 ± 0.1 | 75.8 ± 1.8 | 84.6 |
| Mixup-NU | 86.7 ± 0.3 | 78.0 ± 1.3 | 97.3 ± 0.3 | 77.3 ± 2.0 | **84.8** |
| SWAD | 89.3 ± 0.2 | 83.4 ± 0.6 | 97.3 ± 0.3 | 82.5 ± 0.5 | 88.1 |
| SWAD-NU | 89.8 ± 1.1 | 82.8 ± 1.0 | 97.7 ± 0.3 | 83.7 ± 1.1 | **88.5** |

(b) PACS

| Algorithm | A | C | P | R | Average |
|---|---|---|---|---|---|
| ERM | 61.3 ± 0.7 | 52.4 ± 0.3 | 75.8 ± 0.1 | 76.6 ± 0.3 | 66.5 |
| ERM-NU | 63.3 ± 0.2 | 54.2 ± 0.3 | 76.7 ± 0.2 | 78.2 ± 0.3 | **68.1** |
| Mixup | 62.4 ± 0.8 | 54.8 ± 0.6 | 76.9 ± 0.3 | 78.3 ± 0.2 | 68.1 |
| Mixup-NU | 64.3 ± 0.5 | 55.9 ± 0.6 | 76.9 ± 0.4 | 78.0 ± 0.6 | **68.8** |
| SWAD | 66.1 ± 0.4 | 57.7 ± 0.4 | 78.4 ± 0.1 | 80.2 ± 0.2 | 70.6 |
| SWAD-NU | 67.5 ± 0.3 | 58.4 ± 0.6 | 78.6 ± 0.9 | 80.7 ± 0.1 | **71.3** |

(c) OfficeHome

| Algorithm | L100 | L38 | L43 | L46 | Average |
|---|---|---|---|---|---|
| ERM | 49.8 ± 4.4 | 42.1 ± 1.4 | 56.9 ± 1.8 | 35.7 ± 3.9 | 46.1 |
| ERM-NU | 52.5 ± 1.2 | 45.0 ± 0.5 | 60.2 ± 0.2 | 40.7 ± 1.0 | **49.6** |
| Mixup | 59.6 ± 2.0 | 42.2 ± 1.4 | 55.9 ± 0.8 | 33.9 ± 1.4 | 47.9 |
| Mixup-NU | 55.1 ± 3.1 | 45.8 ± 0.7 | 56.4 ± 1.2 | 41.1 ± 0.6 | **49.6** |
| SWAD | 55.4 ± 0.0 | 44.9 ± 1.1 | 59.7 ± 0.4 | 39.9 ± 0.2 | 50.0 |
| SWAD-NU | 58.1 ± 3.3 | 47.7 ± 1.6 | 60.5 ± 0.8 | 42.3 ± 0.9 | **52.2** |

(d) Terra Incognita

| Algorithm | clip | info | paint | quick | real | sketch | Average |
|---|---|---|---|---|---|---|---|
| ERM | 58.1 ± 0.3 | 18.8 ± 0.3 | 46.7 ± 0.3 | 12.2 ± 0.4 | 59.6 ± 0.1 | 49.8 ± 0.4 | 40.9 |
| ERM-NU | 60.9 ± 0.0 | 21.1 ± 0.2 | 49.9 ± 0.3 | 13.7 ± 0.2 | 62.5 ± 0.2 | 52.5 ± 0.4 | **43.4** |
| Mixup | 55.7 ± 0.3 | 18.5 ± 0.5 | 44.3 ± 0.5 | 12.5 ± 0.4 | 55.8 ± 0.3 | 48.2 ± 0.5 | 39.2 |
| Mixup-NU | 59.5 ± 0.3 | 20.5 ± 0.1 | 49.3 ± 0.4 | 13.3 ± 0.5 | 59.6 ± 0.3 | 51.5 ± 0.2 | **42.3** |
| SWAD | 66.0 ± 0.1 | 22.4 ± 0.3 | 53.5 ± 0.1 | 16.1 ± 0.2 | 65.8 ± 0.4 | 55.5 ± 0.3 | 46.5 |
| SWAD-NU | 66.6 ± 0.2 | 23.2 ± 0.2 | 54.3 ± 0.2 | 16.2 ± 0.2 | 66.1 ± 0.6 | 56.2 ± 0.2 | **47.1** |

(e) DomainNet

Table 6.2: Nuclear norm regularization improves the domain generalization performance over various baselines such as ERM, Mixup, and SWAD.

| Algorithm | VLCS | PACS | DomainNet | Average |
|---|---|---|---|---|
| SWAD Cha et al. (2021) | 79.1 ± 0.1 | 88.1 ± 0.1 | 46.5 ± 0.1 | 71.2 |
| SWAD-CORAL Sun and Saenko (2016) | 78.9 ± 0.1 | 88.3 ± 0.1 | 46.8 ± 0.0 | 71.3 |
| SWAD-MIRO Cha et al. (2022) | 79.6 ± 0.2 | 88.4 ± 0.1 | 47.0 ± 0.0 | 71.7 |
| SWAD-NU (ours) | **79.8** ± 0.2 | **88.5** ± 0.2 | **47.1** ± 0.1 | **71.8** |

Table 6.3: Alternative regularizers with SWAD on the DomainBed benchmark. Full Table is in Appendix E.2.

# 7 THE TRADE-OFF BETWEEN UNIVERSALITY AND LABEL EFFICIENCY OF REPRESENTATIONS FROM CONTRASTIVE LEARNING

**Contribution statement.** This chapter is joint work with Jiefeng Chen, Kunyang Li, Jayaram Raghuram, Xi Wu, Yingyu Liang, and Somesh Jha. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis, and completed part of the experiments. The results of this chapter have been published as a conference paper in ICLR 2023 (Shi et al., 2023b).

## 7.1 Introduction

Representation pre-training is a recent successful approach that utilizes large-scale unlabeled data to address the challenges of scarcity of labeled data and distribution shift. Different from the traditional supervised learning approach using a large labeled dataset, representation learning first pre-trains a representation function using large-scale diverse unlabeled datasets by self-supervised learning (e.g., contrastive learning), and then learns predictors on the representation using small labeled datasets for downstream target tasks. The pre-trained model is commonly referred to as a *foundation model* (Bommasani et al., 2021), and has achieved remarkable performance in many applications, e.g., BERT (Devlin et al., 2019), GPT-3 (Brown et al., 2020), CLIP (Radford et al., 2021), and Flamingo (Alayrac et al., 2022). To this end, we note that there are two properties that are key to their success: (1) *label efficiency*: with the pre-trained representation, only a small amount of labeled data is needed to learn accurate predictors for downstream target tasks; (2) *universality*: the pre-trained representation can be used across various downstream tasks.

In this work, we focus on *contrastive learning with linear probing* that learns a linear predictor on the representation pre-trained by contrastive learning, which is an exemplary pre-training approach (e.g., (Saunshi et al., 2019; Chen et al., 2020d)). We highlight and study a fundamental trade-off between label efficiency and universality, though ideally, one would like to have these two key properties simultaneously. Since pre-training with large-scale diverse unlabeled data is widely used in practice, such a trade-off merits deeper investigation.

Theoretically, we provide an analysis of the features learned by contrastive learning, and how the learned features determine the downstream prediction performance and lead to the trade-off. We propose a *hidden representation data model*, which first generates a hidden representation containing various features, and then uses it to generate the label and the input. We first show that contrastive learning is essentially generalized nonlinear PCA that can learn *hidden features invariant to the transformations* used to generate positive pairs. We also point out that additional assumptions on the data and representations are needed to obtain non-vacuous guarantees for prediction performance. We thus consider a setting where the data are generated by linear functions of the hidden representation, and formally prove that the difference in the learned features leads to the trade-off. In particular, pre-training on more diverse data learns more diverse features and is thus useful for prediction on more tasks. But it also down-weights task-specific features, implying larger sample complexity for predictors and thus worse prediction performance on a specific task. This analysis inspires us to propose a general method – *contrastive regularization* – that adds a contrastive loss to the training of predictors to improve the accuracy on downstream tasks.

Empirically, we first perform controlled experiments to reveal the trade-off. Specifically, we first pre-train on a specific dataset similar to that of the target task, and then incrementally add more datasets into pre-training. In the end, the pre-training data includes both datasets similar to the target task and those not so similar, which mimics the practical scenario that foundation models are pre-trained on diverse data to be widely applicable for various downstream tasks. Fig. 7.1 gives an example of this experiment: As we increase task diversity for contrastive learning, it increases the average accuracy on *all tasks* from 18.3% to 20.1%, while it harms the label efficiency of an individual task, on CIFAR-10 the accuracy drops from 88.5% to 76.4%. We also perform experiments on contrastive regularization, and demonstrate that it can *consistently improve* over the typical fine-tuning method

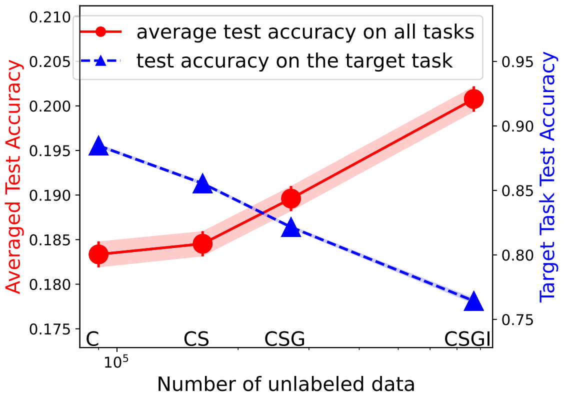


Figure 7.1: Illustration of the trade-off between universality and label efficiency. x-axis: from left to right, incrementally add CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I) for pre-training MoCo v2. For example, "CS" means CINIC-10+SVHN. The average test accuracy of prediction on all 4 datasets (red line) increases with more diverse pre-training data, while that on the target task CIFAR-10 (blue line) decreases. (The variance of the blue line is too small to be seen.) Please refer to Section 7.3.1 for details.

across multiple datasets. In several cases, the improvement is significant: 1.3% test accuracy improvement for CLIP on ImageNet, 4.8% for MoCo v3 on GTSRB (see Table 7.1 and 7.2 for details). With these results, we believe that it is of importance to bring the community's attention to this trade-off and the forward path of foundation models.

Our main contributions are summarized as follows:

- We propose a hidden representation data model and prove that contrastive learning is essentially generalized nonlinear PCA, and can encode hidden features invariant to the transformations used in positive pairs (Section 7.2.1).

- We formally prove the trade-off in a simplified setting with linear data (Section 7.2.2).

- We empirically demonstrate the trade-off across different methods and datasets for contrastive learning with linear probing (Section 7.3.1 and 7.3.2).

- We propose a contrastive regularization method for training the predictor on a target task (Section 7.2.2), which achieves consistent improvement in our experiments (Section 7.3.3).

**Related Work on Representation Pre-training.** This paradigm pre-trains a representation function on a large dataset and then uses it for prediction on various downstream tasks (Devlin et al., 2019; Kolesnikov et al., 2020; Brown et al., 2020; Newell and Deng, 2020). The representations are also called foundation models (Bommasani et al., 2021). There are mainly two kinds of approaches: (1) supervised approaches (e.g., (Kolesnikov et al., 2020)) that pre-train on large labeled datasets; (2) self-supervised approaches (e.g., (Newell and Deng, 2020)) that pre-train on large and diverse unlabeled datasets. Recent self-supervised pre-training can compete with or outperform supervised pre-training on the downstream prediction performance (Ericsson et al., 2021). Practical examples like BERT (Devlin et al., 2019), GPT-3 (Brown et al., 2020), CLIP (Radford et al., 2021), DALL·E (Ramesh et al., 2022), PaLM (Chowdhery et al., 2022) and Flamingo (Alayrac et al., 2022) have obtained effective representations universally useful for a wide range of downstream tasks.

A popular method is contrastive learning, i.e., to distinguish matching and non-matching pairs of augmented inputs (e.g., (van den Oord et al., 2018; Chen et al., 2020d; He et al., 2020b; Grill et al., 2020; Chen and He, 2021; Zbontar et al., 2021; Gao et al., 2021b)). Some others solve "pretext tasks" like predicting masked parts of the inputs (e.g.,(Doersch et al., 2015; Devlin et al., 2019)).

**Related Work on Analysis of Self-supervised Pre-training.** There exist abundant studies analyzing self-supervised pre-training (Saunshi et al., 2019; Tsai et al., 2020; Yang et al., 2020; Wang and Isola, 2020; Garg and Liang, 2020; Zimmermann et al., 2021; Tosh et al., 2021; HaoChen et al., 2021; Wen and Li, 2021; Liu et al., 2021b; Kotar et al., 2021; Van Gansbeke et al., 2021; Lee et al., 2021; Saunshi et al., 2022; Shen et al., 2022; Kalibhat et al., 2022; Sun et al., 2023b,a; Wang et al., 2024). They typically focus on pre-training or assume the same data distribution in pre-training and prediction. Since different distributions are the critical reason for the trade-off we focus on, we provide a new analysis. Some studies have connected contrastive learning to component analysis (Balestriero and LeCun, 2022; Tian, 2022; Ko et al., 2022). Our analysis focuses on the trade-off, while also showing a connection to PCA based on our notion of invariant features and is thus fundamentally different. Recently, Cole et al. have attempted to identify successful conditions for contrastive learning and pointed out that diverse pre-training data can decrease prediction performance compared to pre-training on the specific task data. However, they do not consider universality and provide no systematic study. Similarly, Bommasani et al. call for more research on specialization vs. diversity in pre-training data but provide no study. We aim to provide a better understanding of the trade-off between universality and label efficiency.

## 7.2 Theoretical Analysis

Our experiments in Section 7.3.1 demonstrate a trade-off between the universality and label efficiency of contrastively pre-trained representations when used for prediction on a distribution different from the pre-training data distribution. See Fig. 7.1 for an example. Intuitively, from the unlabeled data, pre-training can learn semantic features useful for prediction on even different data distributions. To analyze this, we need to formalize the notion of useful semantic features. So we introduce a *hidden representation data model* where a hidden representation (i.e., a set of semantic features) is sampled and then used for generating the data. Similar models have been used in some studies (HaoChen et al., 2021; Zimmermann et al., 2021), while we introduce the notion of spurious and invariant features and obtain a novel analysis for contrastive learning.

Using this theoretical model of data, Section 7.2.1 investigates what features are learned by contrastive learning. We show that *contrastive learning can be viewed as a generalization of Principal Components Analysis, and it encodes the invariant features not affected by the transformations but removes the others*. We also show that further assumptions on the data and the representations are needed necessary for any non-vacuous bounds for downstream prediction. So Section 7.2.2 considers a simplified setting with linear data. We show that when pre-trained on diverse datasets (modeled as a mixture of unlabeled data from different tasks), it encodes all invariant features from the different tasks and thus is useful for all tasks. On the other hand, it essentially emphasizes those that are shared among the tasks, but down-weights those that are specific to a single task. Compared to pre-training only on unlabeled data from the target task, this then leads to a larger sample complexity and thus worse generalization for prediction on the target task. Therefore, we show that the trade-off between universality and label efficiency occurs due to the fact that *when many useful features from diverse data are packed into the representation, those for a specific target task can be down-weighted and thus worsen the prediction performance on it*. Based on this insight, we propose a contrastive regularization method for using representations in downstream prediction tasks, which achieves consistent improvement over the typical fine-tuning method in our experiments in Section 7.3.3.

**Contrastive Learning.** Let $\mathcal{X} \subseteq \mathbb{R}^d$ denote the input space, $\mathcal{Y}$ the label space, and $\overline{\mathcal{Z}} \subseteq \mathbb{R}^k$ the output vector space of the learned representation function. Let $\Phi$ denote the hypothesis class of representations $\phi : \mathcal{X} \to \overline{\mathcal{Z}}$, and $\mathcal{F}_\phi$ the hypothesis class of predictors on $\phi$. A task is simply a data distribution over $\mathcal{X} \times \mathcal{Y}$. In pre-training, using transformations on unlabeled data from the tasks, we have some pre-train distribution $\mathcal{D}_{pre}$ over positive pairs $(x, x^+)$ and negative examples $x^-$, where $x, x^+$ are obtained by applying random transformations on the same input (e.g., cropping or color jitter for images), and $x^-$ is an independent example. The contrastive loss is $\ell\left(\phi(x)^\top(\phi(x^+) - \phi(x^-))\right)$ where $\ell(t)$ is a suitable loss function. Typically, the logistic loss $\ell(t) = \log(1 + \exp(-t))$ is used, while our analysis also holds for other loss functions. A representation $\phi$ is learned by:

$$\min_{\phi \in \Phi} \mathbb{E}_{(x,x^+,x^-)\sim\mathcal{D}_{pre}} \left[\ell\left(\phi(x)^\top(\phi(x^+) - \phi(x^-))\right)\right]. \tag{7.1}$$

(We simply consider the population loss since pre-training data are large-scale.) Then a predictor $f$ is learned on top of $\phi$ using $m$ labeled points $\{(x_i, y_i)\}_{i=1}^m$ from a specific target task $\mathcal{D}$:

$$\min_{f \in \mathcal{F}_\phi} \frac{1}{m} \sum_{i=1}^{m} \ell_c(f(\phi(x_i)), y_i) \tag{7.2}$$

where $\ell_c$ is a prediction loss (e.g. cross-entropy). Usually, $f$ is a linear classifier (Linear Probing) with a bounded norm: $\mathcal{F}_\phi = \{f(z) = u^\top z : u \in \mathbb{R}^k, \|u\| \leqslant B\}$, where $\|\cdot\|$ denotes the $\ell_2$ norm.

**Hidden Representation Data Model.** We now consider the pre-train distribution $\mathcal{D}_{\mathrm{pre}}$ over $(x, x^+, x^-)$. To capture that pre-training can learn useful features, we assume a hidden representation for generating the data: first sample a hidden representation $z \in \mathcal{Z}$ from a distribution $\mathcal{D}_z$ over some hidden representation space $\mathcal{Z} \subseteq \mathbb{R}^d$, and then generate the input $x$ and the label $y$ from $z$. (The space $\mathcal{Z}$ models semantic features, and can be different from the learned representation space $\overline{\mathcal{Z}}$.) The dimensions of $z$ are partitioned into two disjoint subsets of $[d] := \{1, \cdots, d\}$: *spurious features* $U$ that are affected by the transformations, and *invariant features* $R$ that are not. Specifically, let $\mathcal{D}_U, \mathcal{D}_R$ denote the distributions of $z_U$ and $z_R$, respectively, and let $x = g(z)$ denote the generative function for $x$. Then the positive pairs $(x, x^+)$ are generated as follows:

$$z = [z_R; z_U] \sim \mathcal{D}_z, z_U^+ \sim \mathcal{D}_U,\ z^+ = [z_R; z_U^+], \quad x = g(z), x^+ = g(z^+). \tag{7.3}$$

That is, $x, x^+$ are from the same $z_R$ but two random copies of $z_U$ that model the random transformations. Finally, $x^-$ is an i.i.d. sample from the same distribution as $x$: $z^- \sim \mathcal{D}_z, x^- = g(z^-)$.

### 7.2.1 What Features are Learned by Contrastive Learning?

To analyze prediction performance, we first need to analyze what features are learned in pre-training.

**Contrastive Learning is Generalized Nonlinear PCA.** Recall that given data $x$ from a distribution $\mathcal{D}$, Principal Components Analysis (PCA) (Pearson, 1901; Hotelling,

1933) aims to find a linear projection function $\phi$ on some subspace such that the variance of the projected data $\phi(x)$ is maximized, i.e., it is minimizing the following PCA objective:

$$-\mathbb{E}_{x\sim\mathcal{D}}[\|\phi(x) - \mathbb{E}_{x'\sim\mathcal{D}}[\phi(x')]\|^2] = -\mathbb{E}_{x\sim\mathcal{D}}[\|\phi(x) - \phi_0\|^2] \tag{7.4}$$

where $\phi_0 := \mathbb{E}[\phi(x')]$ is the mean of the projected data. Nonlinear PCA replaces linear representation functions $\phi$ with nonlinear ones. We next show that contrastive learning is a generalization of nonlinear PCA on the smoothed representation after smoothing out the transformations.

**Theorem 7.1.** *If $\ell(t) = -t$, then the contrastive loss is equivalent to the PCA objective on $\phi_{z_R}$:*

$$\mathbb{E}\left[\ell\left(\phi(x)^\top[\phi(x^+) - \phi(x^-)]\right)\right] = -\mathbb{E}\left[\|\phi_{z_R} - \phi_0\|^2\right] \tag{7.5}$$

*where $\phi_{z_R} := \mathbb{E}[\phi(x) \mid z_R] = \mathbb{E}[\phi(g(z)) \mid z_R]$. If additionally $\phi(x)$ is linear in $x$, then it is equivalent to the linear PCA objective $-\mathbb{E}\left[\|\phi(\bar{x}) - \phi_0\|^2\right]$ on data $\bar{x} := \mathbb{E}[x|z_R] = \mathbb{E}[g(z)|z_R]$.*

So contrastive learning is essentially nonlinear PCA when $\ell(t) = -t$, and further specializes to linear PCA when the representation is linear. As PCA finds directions with large variances, the analogue is that contrastive learning encodes important invariant features but not spurious ones.

**Contrastive Learning Encodes Invariant Features and Removes Spurious Features.** For a formal statement we need some weak assumptions on the data, the representations, and the loss:

**(A1)** $z_R$ can be recovered from $x$, i.e., the inputs $x = g(z)$ from different $z_R$'s are disjoint.

**(A2)** The representation functions are the *regular* functions with $\|\phi(x)\| = B_r$ $(\forall x)$ for some $B_r > 0$. Being regular means there are a finite $L$ and a partition of $\mathcal{Z}$ into a finite number of subsets, such that in each subset all $\phi \circ g$ have Lipschitz constants bounded by $L$.

**(A3)** The loss $\ell(t)$ is convex, decreasing, and lower-bounded.

The first condition means the invariant features $z_R$ can be extracted from $x$ (note that $g$ need not be invertible). The regular condition on the representation is to exclude some pathological cases like the Dirichlet function; essentially reasonable functions relevant for practice satisfy this condition, e.g., when $g$ is Lipschitz and $\phi$ are neural networks with the ReLU activation. Also, note that the logistic loss typically used in practice satisfies the last condition.

We say a function $f(z)$ is independent of a subset of input dimensions $z_S$, if there exists a function $f'$ such that $f(z) = f'(z_{-S})$ with probability 1, where $z_{-S}$ denotes the set of all $z_j$ with $j \notin S$. We say the representation $\phi$ encodes a feature $z_i$, if $\phi \circ g : \mathcal{Z} \to \overline{\mathcal{Z}}$ is not independent of $z_i$ as long as the generative function $g(z)$ is not independent of $z_i$.

**Theorem 7.2.** *Under Assumptions (**A1**)(**A2**)(**A3**), the optimal representation $\phi^*$ satisfies:*

(1) *$\phi^*$ does not encode the spurious features $z_U$: $\phi^* \circ g(z)$ is independent of $z_U$.*

(2) *For any invariant feature $i \in R$, there exists $B_i > 0$ such that as long as the representations' norm $B_r \geqslant B_i$, then $\phi^*$ encodes $z_i$. Furthermore, if $\mathcal{Z}$ is finite, then $B_i$ is monotonically decreasing in $\Pr[z_{R\setminus\{i\}} = z^-_{R\setminus\{i\}}, z_i \neq z^-_i]$, the probability that in $z_R$ and $z^-_R$, the $i$-th feature varies while the others remain the same.*

So contrastive learning aims to remove the spurious features and preserve the invariant features. Then the transformations should be chosen such that they will not affect the useful semantic features, but change those irrelevant to the label. Interestingly, the theorem further suggests that contrastive learning tends to favor the more "spread-out" invariant features $z_i$, as measured by $\Pr[z_{R\setminus\{i\}} = z^-_{R\setminus\{i\}}, z_i \neq z^-_i]$. As we increase the representation capacity $B_r$, $B_r$ passes the threshold $B_i$ for

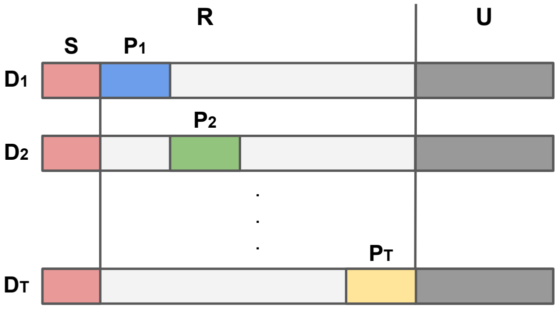


Figure 7.2: Illustration of the features in our data distributions.

more features $z_i$, so $\phi^*$ first encodes the more spread-out invariant features and then the others.

This further suggests the following intuition for the trade-off. When pre-trained on diverse data modeled as a mixture from multiple tasks with different invariant features, the representation encodes all the invariant features and thus is useful for prediction on all the tasks. When pre-trained on only a specific task, features specific to this task are favored over those that only show up in other tasks, which leads to smaller sample complexity for learning the predictor and thus better prediction. However, to formalize this, some inductive bias assumptions about the data and the representation are necessary to get any non-vacuous guarantee for the prediction (see discussion in Appendix F.1.1). Therefore, Section 7.2.2 introduces additional assumptions and formalizes the trade-off.

### 7.2.2 Analyzing the Trade-Off: Linear Data

To analyze the prediction performance, we first need to model the relation between the pre-training data and the target task. We model the diverse pre-training data as a mixture of data from T different tasks $\mathcal{D}_t$'s, while the target task is one of the tasks. All tasks share a public feature set S of size s, and each task $\mathcal{D}_t$ additionally owns a private disjoint feature set $P_t$ of size $r - s$, i.e., $P_t \cap S = \emptyset$ and $P_{t_1} \cap P_{t_2} = \emptyset$ for $t_1 \neq t_2$ (Fig. 7.2). The invariant features for $\mathcal{D}_t$ are then $R_t = S \cup P_t$. All invariant features are $R = \cup_{t=1}^{T} R_t$, and spurious features are $U = [d] \setminus R$. In task $\mathcal{D}_t$, the $(x, x^+)$ are generated as follows:

$$z_{R_t} \sim \mathcal{N}(0, I),\ z_{R\setminus R_t} = 0,\ z_U \sim \mathcal{N}(0, I), z = [z_R; z_U], \quad x = g(z), \tag{7.6}$$

$$z_U^+ \sim \mathcal{N}(0, I), z^+ = [z_R; z_U^+], \quad x^+ = g(z^+), \tag{7.7}$$

and $x^-$ is simply an i.i.d. copy from the same distribution as $x$. In practice, multiple independent negative examples are used, and thus we consider the following contrastive loss $\min_{\phi \in \Phi}\ \mathbb{E}_{(x,x^+)} \left[\ell\left(\phi(x)^\top(\phi(x^+) - \mathbb{E}_{x^-}\phi(x^-))\right)\right]$ for a convex and decreasing $\ell(t)$ to pre-train a representation $\phi$. Then, when using $\phi$ for prediction in the target task $\mathcal{D}_t$, the predictor class should contain a predictor matching the ground-truth label:

$$\mathcal{F}_{\phi,t} = \{f(z) = u^\top z : u \in \mathbb{R}^k, \|u\| \leqslant B_{\phi,t}\} \tag{7.8}$$

where $B_{\phi,t}$ is the minimum value such that there exists $u_t \in \mathcal{F}_{\phi,t}$ with $y = u_t^\top \phi(x)$ on $\mathcal{D}_t$.

Now, given the necessity of inductive biases for non-vacuous guarantees (see Appendix F.1.1), and inspired by classic dictionary learning and recent analysis on such data (e.g., Olshausen and Field (1997); Wen and Li (2021); Shi et al. (2022c)), we assume linear data and linear representations:

- $x$ is linear in $z$: $x = g(z) = Mz$ where $M \in \mathbb{R}^{d\times d}$ is an orthonormal dictionary. Since linear probing has strong performance on pre-trained representations, we thus assume that the label in each task $t$ is linear in its invariant features $y = (u_t^*)^\top z_{R_t}$ for some $u_t^* \in \mathbb{R}^r$.

- The representations are linear functions with weights of bounded spectral/Frobenius norms:

$$\Phi = \{\phi(x) = Wx : W \in \mathbb{R}^{k\times d}, \|W\| \leqslant 1, \|W\|_F \leqslant \sqrt{r}\}.$$

  Here the norm bounds are chosen to be the minimum values to allow recovering the invariant features in the target task, i.e., there exists $\phi \in \Phi$ such that

$\phi(x) = [z_{R_t}; \mathbf{0}]$.

We compare two representations: a specific one pre-trained on unlabeled data from the target task $\mathcal{D}_t$, and a universal one pre-trained on an even mixture of data from T tasks. (Appendix F.2 provides analysis for more general cases like uneven mixtures.) This captures the situation that the pre-training data contains some data similar to the target task and also other less similar data. Let $\nu_{t,1} = \sum_{j \in S} (u_t^*)_j^2$ and $\nu_{t,2} = \sum_{j \in P_t} (u_t^*)_j^2$ be the weights on the shared and task-specific invariant features, respectively. Also, assume the prediction loss $\ell_c$ is L-Lipschitz.

**Proposition 7.3.** *The representation $\phi^*$ obtained on an even mixture of data from all the tasks $\{\mathcal{D}_t : 1 \leqslant t \leqslant T\}$ satisfies $\phi^* \circ g(z) = Q\left(\sum_{j \in S} \sqrt{\alpha} z_j e_j + \sum_{j \in R \setminus S} \sqrt{\beta} z_j e_j\right)$ for some $\alpha \in [0, 1]$, $\beta = \min\left(1, \frac{r - \alpha s}{T(r-s)}\right)$, where $e_j$'s are the basis vectors and $Q$ is any orthonormal matrix.*
*The Empirical Risk Minimizer $\hat{u} \in \mathcal{F}_{\phi^*, t}$ on $\phi^*$ using $m$ labeled data points from $\mathcal{D}_t$ has risk*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}^\top \phi^*(x), y)]$$
$$\leqslant 4L\sqrt{\frac{1}{m}\left(\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\beta}\right)}\left(\sqrt{s\alpha + (r-s)\beta} + O\left(\sqrt{\frac{r}{s\alpha + (r-s)\beta}}\right)\right) + 8\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

**Proposition 7.4.** *The representation $\phi_t^*$ obtained on data from $\mathcal{D}_t$ satisfies $\phi_t^* \circ g(z) = Q\left(\sum_{j \in R_t} z_j e_j\right)$ where $e_j$'s are the basis vectors and $Q$ is any orthonormal matrix.*
*The Empirical Risk Minimizer $\hat{u} \in \mathcal{F}_{\phi_t^*, t}$ on $\phi_t^*$ using $m$ labeled data points from $\mathcal{D}_t$ has risk*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}^\top \phi_t^*(x), y)] \leqslant 4L\sqrt{\frac{r}{m}}\|u_t^*\| + 8\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

*While on task $\mathcal{D}_i (i \neq t)$, any linear predictor on $\phi_t^*$ has error at least* $\min_u \mathbb{E}_{\mathcal{D}_i}[\ell_c(u^\top z_S, y)]$.

**Difference in Learned Features Leads to the Trade-off.** The key of the analysis (in Appendix F.2) is about what features are learned in the representations. Pre-trained on all $\mathsf{T}$ tasks, $\phi^*$ is a rotation of the weighted features, where the shared features are weighted by $\sqrt{\alpha}$ and task-specific ones are weighted by $\sqrt{\beta}$. Pre-trained on one task $\mathcal{D}_t$, $\phi_t^*$ is a rotation of the task-specific features $R_t$. So compared to $\phi_t^*$, $\phi^*$ encodes all invariant features but down-weights the task-specific features $P_t$.

The difference in the learned features then determines the prediction performance and results in a trade-off between universality and label efficiency: compared to $\phi_t^*$, $\phi^*$ is useful for more tasks but has worse performance on the specific task $\mathcal{D}_t$. For illustration, suppose $r = 2s$, and the shared and task-specific features are equally important for the labels on the target task: $\nu_{t,1} = \nu_{t,2} = \|u_t^*\|^2/2$. In Appendix F.2.3 we show that $\phi^*$ has $\alpha = 1, \beta = \frac{1}{T}$ and the error is $O\left(L\sqrt{\frac{Tr}{m}}\|u_t^*\|\right)$, while the error using $\phi_t^*$ is $O\left(L\sqrt{\frac{r}{m}}\|u_t^*\|\right)$. Therefore, the error when using representations pre-trained on data from $\mathsf{T}$ tasks is $O(\sqrt{T})$ worse than that when just pre-training on data from the target task. On the other hand, the former can be used in all $\mathsf{T}$ tasks and the prediction error diminishes with the labeled data number $m$. While the latter only encodes $R_t$ and the only useful features on the other tasks are $z_S$, then even with infinite labeled data the error can be large ($\geqslant \min_u \mathbb{E}[\ell_c(u^\top z_S, y)]$, the approximation error using only the common features $z_S$ for prediction).

**Improving the Trade-off via Contrastive Regularization.** The above analysis provides some guidance on improving the trade-off, in particular, improving the target prediction accuracy when given a pre-trained representation $\phi^*$. It suggests that when $\phi^*$ is pre-trained on diverse data, one can update it by contrastive learning on some unlabeled data from the target task, which can get better features and better predictions. This is indeed the case for the illustrative example above. We can show that updating $\phi^*$ by contrastive learning on $\mathcal{D}_t$ can increase the weights $\beta$ on the task-specific features $z_{P_t}$, and thus improve the generalization error (formal analysis in Appendix F.2.4).

In practice, typically one will learn the classifier and also fine-tune the representation with a labeled dataset $\{(x_i, y_i)\}_{i=1}^m$ from the target task. We thus propose *contrastive regularization* for fine-tuning: for each data point $(x, y)$, generate contrastive pairs $\mathcal{R} = \{(\tilde{x}, \tilde{x}^+, \tilde{x}^-)\}$ by applying transformations, and add the contrastive loss on these pairs as a regularization term to the classification loss:

$$\ell_c(f(\phi(x)), y) + \frac{\lambda}{|\mathcal{R}|} \sum_{(\tilde{x}, \tilde{x}^+, \tilde{x}^-) \in \mathcal{R}} \ell \left( \phi(\tilde{x})^\top (\phi(\tilde{x}^+) - \phi(\tilde{x}^-)) \right). \tag{7.9}$$

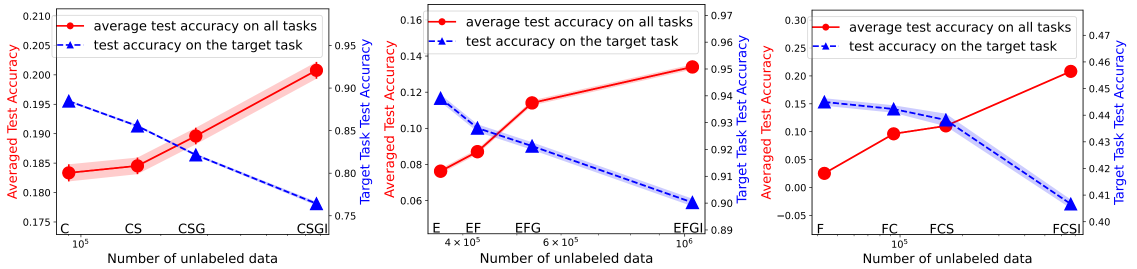
This method is simple and generally applicable to different models and algorithms. Similar ideas have been used in graph learning (Ma et al., 2021), domain generalization (Kim et al., 2021) and semi-supervised learning (Lee et al., 2022), while we use it in fine-tuning for learning predictors. Our experiments in Section 7.3.3 show that it can consistently improve the prediction performance compared to the typical fine-tuning approach.

## 7.3 Experiments

We conduct experiments to answer the following questions. (**Q1**) Does the trade-off between universality and label efficiency exist when training on real datasets? (**Q2**) What factors lead to the trade-off? (**Q3**) How can we alleviate the trade-off, particularly in large foundation models? Our experiments provide the following answers: (**A1**) The trade-off widely exists in different models and datasets when pre-training on large-scale unlabeled data and adapting with small labeled data (see Section 7.3.1). This justifies our study and aligns with our analysis. (**A2**) Different datasets own many private invariant features leading to the trade-off, e.g., FaceScrub and CIFAR-10 do not share many invariant features (see Section 7.3.2). It supports our analysis in Section 7.2.2. (**A3**) Our proposed method, Finetune with Contrastive Regularization, can improve the trade-off consistently (see Section 7.3.3).

### 7.3.1 Verifying the Existence of the Trade-off

**Evaluation & Methods.** We first pre-train a ResNet18 backbone (He et al., 2016) with different contrastive learning methods and then do Linear Probing (LP, i.e., train a linear classifier on the feature extractor) with the labeled data from the target task. We report the test accuracy on a specific target task and the average test

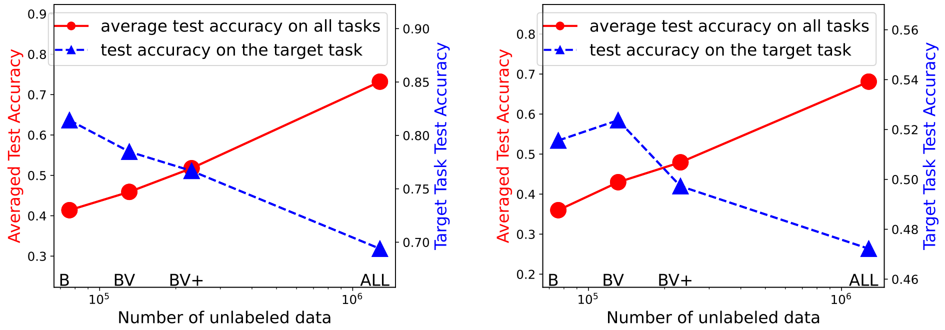


(a) CIFAR-10 (b) MNIST (c) Fer2013

Figure 7.3: Trade-off between universality and label efficiency for MoCo v2. Appendix F.3.5 shows similar results for more methods and datasets. $x$-axis: incrementally add datasets for pre-training MoCo v2. **(a)** Pre-training data: CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I). E.g., "CS" on the $x$-axis means CINIC-10+SVHN. Target task: CIFAR-10. Red line: average test accuracy of Linear Probing on all 4 datasets. Blue line: test accuracy on the target task. **(b)** EMNIST-Digits&Letters (E), Fashion-MNIST (F), GTSRB (G), ImageNet32 (I). Target: MNIST. **(c)** FaceScrub (F), CIFAR-10 (C), SVHN (S), ImageNet32 (I). Target: Fer2013. Note that training does *not* follow the online learning fashion, e.g., the model will pre-train from scratch (random initialization) on the CSG datasets, rather than using the model pre-trained on the CS datasets.

accuracy on all pre-training datasets (i.e., using them as the downstream tasks). Appendix F.3.2 presents full details and additional results, while Fig. 7.3 shows the results for the method MoCo v2. The size and diversity of pre-training data are increased on the $x$-axis by incrementally adding unlabeled training data from: (a) CINIC-10, SVHN, GTSRB, ImageNet32 (using only a 500k subset); (b) EMNIST-Digits&Letters, Fashion-MNIST, GTSRB, ImageNet32; (c) FaceScrub, CIFAR-10, SVHN, ImageNet32. We further perform larger-scale experiments: (1) on ImageNet (see Fig. 7.4); (2) on ImageNet22k and GCC-15M (see Appendix F.3.2).

**Results.** The results show that when the pre-training data becomes more diverse, the average test accuracy on all pre-training datasets increases (i.e., universality improves), while the test accuracy on the specific target task decreases (i.e., label efficiency drops). This shows a clear trade-off between universality and label efficiency. It supports our claim that diverse pre-training data allow learning diverse features for better universality, but can down-weight the features for a specific task resulting in worse prediction. Additional results in the appendix



(a) MoCo v3 (backbone ViT-S)

(b) SimSiam (backbone ResNet50)

Figure 7.4: Trade-off between universality and label efficiency on ImageNet. $x$-axis: from left to right, incrementally add ImageNet-Bird (B), ImageNet-Vehicle (V), ImageNet-Cat/Ball/Shop/Clothing/Fruit (+), and ImageNet (ALL) for pre-training (a) MoCo v3 with backbone ViT-S (b) SimSiam with backbone ResNet50. For example, "BV" means ImageNet-Bird + ImageNet-Vehicle. Target: ImageNet-Bird.

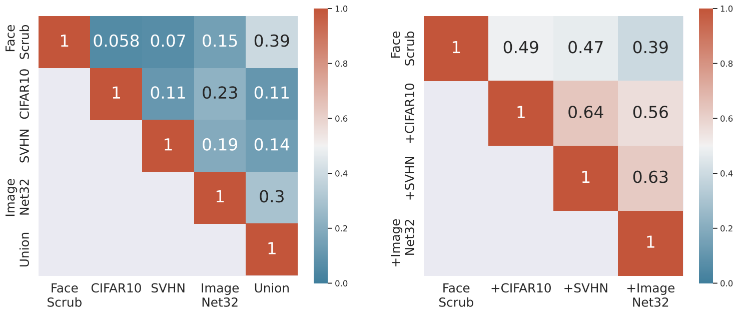


Figure 7.5: Linear CKA similarity among Fer2013 features from MoCo v2 pre-trained on different datasets. Left: each representation in the first four columns/rows is pre-trained on a single dataset. "Union" indicates the model pre-trained on the union of the four disjoint datasets. Right: from left column to right, from top row to bottom, we incrementally add datasets for pre-training.

show similar trends (e.g., for methods NNCLR and SimSiam). This validates our theoretical analysis of the trade-off.

| | **Pre-training** dataset | | | |
|---|---|---|---|---|
| Method | CINIC-10 | +SVHN | +GTSRB | +ImageNet32 |
| LP | 88.41±0.01 | 85.18±0.01 | 82.07±0.01 | 75.64±0.03 |
| FT | 93.58±0.14 | 93.35±0.10 | 93.42±0.13 | 92.92±0.06 |
| Ours | **94.51**±0.02 | **94.26**±0.01 | **94.32**±0.13 | **93.66**±0.12 |

Table 7.1: Test accuracy on CIFAR-10 with different evaluation methods on MoCo v2 by using all CIFAR-10 training data. From left to right: incrementally add datasets for pre-training.

### 7.3.2 Inspecting the Trade-off: Feature Similarity

Here we compute the similarity of the features learned from different pre-training datasets for a target task. For each pre-trained model, we extract a set of features for the target task Fer2013 using the pre-trained representation function. Then we compute the similarities between the extracted features based on different pre-training dataset pairs using linear Centered Kernel Alignment (CKA) (Kornblith et al., 2019), a widely used tool for high-dimensional feature comparison. Figure 7.5 reports the results (rows/columns are pre-training data; numbers/colors show the similarity). The left figure shows that the features from different pre-training datasets have low similarities. This is consistent with our setup in Section 7.2.2 that different tasks only share some features and each owns many private ones. The right figure shows a decreasing trend of similarity along each row. This indicates that when gradually adding more diverse pre-training data, the learned representation will encode more downstream-task-irrelevant features, and become less similar to that prior to adding more pre-training data. Additional results with similar observations, finer-grained investigation into the trade-off, and some ablation studies are provided in Appendix F.3.3.

### 7.3.3 Improving the Trade-off: Finetune with Contrastive Regularization

**Evaluation & Methods.** We pre-train ResNet18 by MoCo v2 as in Section 7.3.1 and report the test accuracy on CIFAR-10 when the predictor is learned by: Linear Probing (LP), Finetune (FT), and Finetune with Contrastive Regularization (Ours).

LP follows the training protocol in Section 7.3.1. FT and Ours learn a linear predictor and update the representation, and use the same data augmentation for a fair comparison. FT follows MAE (He et al., 2022), while Ours uses MoCo v2 contrastive loss and regularization coefficient $\lambda = 0.1$. More details and results are given in Appendix F.3.4.

**Results.** Table 7.1 shows that our method can consistently outperform the other baselines. In particular, it outperforms the typical fine-tuning method by about 0.7% – 1%, even when the latter also uses the same amount of data augmentation. This confirms the benefit of contrastive regularization. To further support our claim, Fig. F.8 in Appendix F.3.4 visualizes the features of different methods by t-SNE, showing that contrastive regularization can highlight the task-specific features and provide cleaner clustering, and thus improve the generalization, as discussed in our theoretical analysis.

| | CLIP | | | MoCo v3 | | | SimCSE | |
|---|---|---|---|---|---|---|---|---|
| Method | ImageNet | SVHN | GTSRB | CIFAR-10 | SVHN | GTSRB | IMDB | AGNews |
| LP | 77.84±0.02 | 63.44±0.01 | 86.56±0.01 | 95.82±0.01 | 61.92±0.01 | 75.37±0.01 | 86.49±0.16 | 87.76±0.66 |
| FT | 83.65±0.01 | 78.22±0.18 | 90.74±0.06 | 96.17±0.12 | 65.36±0.33 | 76.45±0.29 | 92.31±0.26 | 93.57±0.23 |
| Ours | **84.94**±0.09 | **78.72**±0.37 | **92.01**±0.28 | **96.71**±0.10 | **66.29**±0.20 | **81.28**±0.10 | **92.85**±0.03 | **93.94**±0.02 |

Table 7.2: Test accuracy for different evaluation methods on different datasets using all training data and using foundation models from CLIP, MoCo v3, and SimCSE. Data augmentation is not used for LP (Linear Probing). For FT (Finetune) and Ours (our method), 10 augmentations to each training images are used for CLIP, MoCo v3, and unique augmentation in each training step is used for SimCSE. More results are in Appendix F.3.4.

**Larger Foundation Models.** We further evaluate our method on several popular real-world large representation models (foundation models). On some of these models, the user may be able to fine-tune the representation when learning predictors. On very large foundation models, the user typically extracts feature embeddings of their data from the models and then trains a small predictor, called adapter (Hu et al., 2022; Sung et al., 2022), on these embeddings. We evaluate CLIP (ViT-L (Dosovitskiy et al., 2020) as the representation backbone), MoCo v3 (ViT-B backbone), and SimCSE (Gao et al., 2021b) (BERT backbone). They are trained on (image, text), (image, image), and (text, text) pairs, respectively, so cover a good spectrum of methods. For CLIP and MoCo v3, the backbone is fixed. LP uses a linear classifier, while FT and Ours insert a two-layer ReLU network as an adapter between the backbone and the linear classification layer. Ours uses the SimCLR contrastive loss on the output of the adapter. For SimCSE, all methods use linear classifiers. LP fixes the backbone, while FT and Ours train the classifier and fine-tune the backbone simultaneously. Ours uses the SimCSE contrastive loss on the backbone feature. We set the regularization coefficient $\lambda = 1.0$.

Table 7.2 again shows that our method can consistently improve the downstream prediction performance for all three models by about 0.4% – 4.8%, and quite significantly in some cases (e.g., 1.3% for CLIP on ImageNet, 4.8% for MoCo v3 on GTSRB). This shows that our method is also useful for large foundation models, even when the foundation models cannot be fine-tuned and only the extracted embeddings can be adapted. Full details and more results are provided in Appendix F.3.4.

## 7.4 Conclusion and Future Work

In this work, we have shown and analyzed the trade-off between universality and label efficiency of representations in contrastive learning. There are many interesting open questions for future work. (1) What features does the model learn from specific pre-training and diverse pre-training datasets beyond linear data? (2) Do the other self-supervised learning methods have a similar trade-off? (3) Can we address the trade-off better to gain both properties at the same time?

# 8 BYPASSING THE EXPONENTIAL DEPENDENCY: LOOPED TRANSFORMERS EFFICIENTLY LEARN IN-CONTEXT BY MULTI-STEP GRADIENT DESCENT

**Contribution statement.** This chapter is joint work with Bo Chen, Xiaoyu Li, Yingyu Liang, and Zhao Song. The author Zhenmei Shi proposed the method, contributed to part of the theoretical analysis. The results in this chapter is submitted to AIStats 2025 (Chen et al., 2024b).

## 8.1 Introduction

Large Language Models (LLMs) have gained immense success and have been widely used in our daily lives, e.g., GPT4 (Achiam et al., 2023), Claude 3.5 (Anthropic., 2024), and so on, based on its Transformer architecture (Vaswani et al., 2017). One core emergent ability of LLMs is In-Context Learning (ICL) (Brown et al., 2020). During ICL, the user provides an input sequence (prompts) containing some question-answer pairs as in-context examples and the goal query that the user cares about, where the examples and query are drawn from an unknown task. The LLMs can in-context learn from these examples and generate the correct answer for the goal query without any parameter update. Notably, these unknown tasks may never seen by LLMs during their pre-training and post-training, e.g., synthetic task (Wei et al., 2023a). Thus, many believe that the in-context learning mechanism is different from supervised learning or unsupervised learning, where the latter may focus on feature learning, while the ICL may perform some algorithm to learn. For instance, the Transformer can implement algorithm selection (Bai et al., 2024a) or gradient descent (Dong et al., 2022; Von Oswald et al., 2023) by in-context learning forward pass.

Many works have tried to understand how Transformers perform single-step gradient descent (Zhang et al., 2023c; Ahn et al., 2024a; Mahankali et al., 2023; Cheng et al., 2023; Bai et al., 2024a; Huang et al., 2023; Li et al., 2023c; Guo et al., 2024; Wu et al., 2024c). They study one-layer linear Transformer in-context learning on linear regression tasks and show that the one-layer linear Transformer can perform single gradient updates based on the in-context examples during a forward pass. Recent work (Gatmiry et al., 2024) has further shown that a linear looped Transformer can implement multi-step gradient descent updates with multiple forward passes, meaning that Transformers can express multi-step algorithms during in-context learning. However, their theoretical results require an exponential number of in-context examples, $\exp(\Omega(T))$, where $T$ is the number of loops or passes, to achieve a reasonably low error for linear regression tasks. This violates the intuition that more gradient descent updates lead to better performance.

Thus, it is natural to ask the following question:

*Is it necessary to use an exponential number of examples for Transformers to implement multi-step gradient descent during in-context learning?*

In this work, we study linear looped Transformers (Definition 8.7) in-context learning on linear vector generation tasks (Definition 8.5), which is as hard as linear regression. We show that the linear looped Transformer can efficiently perform multi-step gradient descent as long as in-context examples are well-conditioned. We present our main result in the following theorem.

**Theorem 8.1** (Main result. Informal version of Theorem 8.16)**.** *Let $T$ be the number of loops, $n$ be the number of in-context examples, and $d$ be the feature dimension. Let $(X, y) \in \mathbb{R}^{n \times d} \times \mathbb{R}^n$ be the in-context examples. Let $\kappa$ be the condition number of $X^\top X$ (Definition 8.12). Then, given a query $\alpha \in \mathbb{R}$, the linear looped Transformer can predict a vector output with error $\leqslant |\alpha| \cdot \exp(-\frac{T}{2\kappa})$ by explicitly performing multi-step gradient descent in its hidden states.*

In Theorem 8.1, as long as the condition number is constant, we can see that the linear looped Transformer will perform better when the loop number is increasing, i.e., the error will exponentially decay to 0. Informally, for a small constant $\epsilon$, if we draw $X$ from Gaussian distribution, the condition number of $X^\top X$ will be

$1 \leqslant \kappa \leqslant 1 + O(\epsilon)$ when $n \geqslant \Omega(d/\epsilon^2)$. Thus, informally, we only need $O(d)$ numbers of in-context examples to guarantee a good performance. Furthermore, our preliminary experiments (Section 8.6) validate the above arguments and our theoretical analysis.

The main intuition of our analysis is that we find that a linear looped Transformer can explicitly perform gradient descent in its hidden states (Lemma 8.13 and Theorem 8.14). Thus, the error analysis can be directly solved by the standard convex optimization technique (Theorem 8.25).

**Our contributions:**

- We study linear looped Transformers in-context learning on linear vector generation tasks (Definition 8.4), which is as hard as linear regression.
- We find that linear looped Transformers can explicitly perform gradient descent in their hidden states (Lemma 8.13 and Theorem 8.14).
- We demonstrate that linear looped Transformers can efficiently perform multi-step gradient descent as long as in-context examples are well-conditioned, e.g., $n = O(d)$, where the error will exponentially decay to 0 after $T$ loops (Theorem 8.16).
- Our preliminary experiments on synthetic data validate our main theoretical results (Section 8.6).

**Roadmap.** This paper is structured as follows: we begin with a review of related work in Section 8.2, followed by essential definitions and foundational concepts in Section 8.3. Section 8.4 delves into the gradient computation analysis within the Looped Transformer architecture, examining both individual layer computations and the full looped structure. In Section 8.5, we analyze the error convergence of looped transformers under strong convexity and smoothness conditions, providing an upper bound on error after $T$ gradient descent iterations for linear vector generation. Section 8.6 presents our experimental results and findings. Finally, we conclude in Section 8.7.

## 8.2 Related Work

This section briefly reviews the related research work on Large Language Models (LLM), In-Context Learning (ICL), and looped transformers. These topics have a close connection to our work.

**Large Language Models.** Neural networks based on the Transformer architecture (Vaswani et al., 2017) have swiftly become the dominant paradigm in machine learning for natural language processing applications. Expansive Transformer models, trained on diverse and extensive datasets and comprising billions of parameters, are called large language models (LLM) or foundation models (Bommasani et al., 2021). Examples include BERT (Devlin et al., 2019), PaLM (Chowdhery et al., 2022), Llama (Touvron et al., 2023a), ChatGPT (OpenAI, 2022), GPT4 (Achiam et al., 2023), among others. These LLMs have demonstrated remarkable general intelligence capabilities (Bubeck et al., 2023) across various downstream tasks.

Researchers have developed numerous adaptation techniques to optimize LLM performance for specific applications. These include methods such as adapters (Hu et al., 2022; Zhang et al., 2023b; Gao et al., 2023; Shi et al., 2023b), calibration approaches (Zhao et al., 2021; Zhou et al., 2023a), multitask fine-tuning strategies (Gao et al., 2021a; Xu et al., 2023; Von Oswald et al., 2023; Xu et al., 2024f), prompt tuning techniques (Gao et al., 2021a; Lester et al., 2021), scratchpad approaches (Nye et al., 2021), instruction tuning methodologies (Li and Liang, 2021; Chung et al., 2022; Mishra et al., 2022), symbol tuning (Wei et al., 2023a), black-box tuning (Sun et al., 2022), reinforcement learning from the human feedback (RLHF) (Ouyang et al., 2022), chain-of-thought reasoning (Wei et al., 2022d; Khattab et al., 2022; Yao et al., 2023; Zheng et al., 2024) and various other strategies. And here are more related works aiming at enhancing model efficiency without compromising performance, such as (Chen et al., 2024c; Liang et al., 2024d,b; Hu et al., 2023; Wu et al., 2024b; Hu et al., 2024c; Xu et al., 2024a; Wu et al., 2024a; Liang et al., 2024e; Hu et al., 2024a,d,b; Li et al., 2024e; Song et al., 2023c; Shi et al., 2024a; Ke et al., 2024b; Chen et al., 2024d; Li et al., 2024d; Chen et al., 2024a; Li et al., 2024f; Liang et al., 2024f).

**In-context Learning.** A significant capability that has emerged from Large Language Models (LLMs) is in-context learning (ICL) (Brown et al., 2020; Wei et al., 2022b; Shi et al., 2024b). This feature allows LLMs to generate predictions for new scenarios when provided with a concise set of input-output examples (referred to as a prompt) for a specific task, without requiring any modification to the model's parameters. ICL has found widespread application across various domains, including reasoning (Zhou et al., 2022), self-correction (Pourreza and Rafiei, 2023), machine translation (Agrawal et al., 2022), and many so on.

Numerous studies have focused on enhancing the ICL and zero-shot capabilities of LLMs (Min et al., 2021; Wang et al., 2022; Wei et al., 2022a; Iyer et al., 2022). A substantial body of research has been dedicated to investigating the underlying mechanisms of transformer learning (Geva et al., 2021; Xie et al., 2022; Garg et al., 2022; Jelassi et al., 2022; Arora and Goyal, 2023; Li et al., 2023b,f; Allen-Zhu and Li, 2023; Luo et al., 2023; Tian et al., 2023a,b; Zhou et al., 2023b; Bietti et al., 2023; Xu et al., 2024e; Li et al., 2024a,b; Liang et al., 2024a,g,h) and in-context learning (Mahankali et al., 2023; Raventos et al., 2023; Dong et al., 2022; Ahn et al., 2024a; Pan et al., 2023; Li et al., 2023c,e; Akyurek et al., 2023; Zhang et al., 2023a,c; Huang et al., 2023; Cheng et al., 2023; Wibisono and Wang, 2023; Wu et al., 2024c; Guo et al., 2024; Reddy, 2024) through both empirical and theoretical approaches. Building upon these insights, our analysis extends further to elucidate ICL implementing multi-step gradient descent.

**Looped Transformer.** The concept of recursive inductive bias was first introduced by Dehghani et al. (2018) into Transformers. Looping Transformers are also related to parameter-efficient weight-tying Transformers (Chi et al., 2021; Gatmiry et al., 2024) theoretically showed that the recursive structure of Looped Transformers enables them to function as Turing machines. Yang et al. (2023) demonstrate that increasing the number of loop iterations can improve performance on some tasks. Recent studies (Artur Back and Fountoulakis, 2024; Giannou et al., 2024) have provided theoretical insights into the emulation capabilities of specific algorithms and their convergence during training, with a particular emphasis on in-context learning. However, the expressive capacity of Looped Transformers or Looped Neural Newrok Liang et al. (2024c); Ke et al. (2024a) in function approximation and their associated approximation rates remain largely unexplored territories.

## 8.3 Preliminary

In this section, we present some preliminary concepts and definitions of our paper. In Section 8.3.1, we introduce some basic notations used in our paper. In Section 8.3.2, we defined some important variables to set up our problem.

### 8.3.1 Notations

For any integer $n$, we define $[n] = \{1, 2, \dots, n\}$. For two vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^n$, we use $\langle x, y \rangle$ to denote the inner product between $x, y$, i.e., $\langle x, y \rangle = \sum_{i=1}^n x_i y_i$. For each $a, b \in \mathbb{R}^n$, we use $a \circ b \in \mathbb{R}^n$ to denote the Hardamard product, i.e. the $i$-th entry of $(a \circ b)$ is $a_i b_i$ for all $i \in [n]$. We use $\mathbf{1}_n$ to denote a length-$n$ vector where all the entries are ones. We use $\|x\|_p$ to denote the $\ell_p$ norm of a vector $x \in \mathbb{R}^n$, i.e. $\|x\|_1 := \sum_{i=1}^n |x_i|$, $\|x\|_2 := (\sum_{i=1}^n x_i^2)^{1/2}$, and $\|x\|_\infty := \max_{i \in [n]} |x_i|$. For $n > d$, for any matrix $A \in \mathbb{R}^{n \times d}$, we use $\|A\|$ to denote the spectral norm of $A$, i.e. $\|A\| := \sup_{x \in \mathbb{R}^d} \|Ax\|_2 / \|x\|_2$. For a matrix $A$, we use $\lambda_{\min}(A)$ and $\lambda_{\max}(A)$ to denote the minimum and maximum eigenvalue of $A$, respectively.

### 8.3.2 In-context Learning

First, we introduce some definitions of in-context examples and their labels.

**Definition 8.2** (In-context prompt data)**.** *Let* $X \in \mathbb{R}^{n \times d}$ *be the input data with* $n$ *tokens and each token with feature dimension* $d$*, i.e,* $X = [x_1, x_2, \dots, x_n]^\top$*, where* $x_i \in \mathbb{R}^d$ *for any* $i \in [n]$.

**Definition 8.3** (In-context prompt label)**.** *Assume* $\theta^*$ *uniformly draw form* $d$ *dimension unit sphere* $\mathbb{S}^d$*, where* $\|\theta^*\|_2 = 1$*. Let labels be* $y := X\theta^* \in \mathbb{R}^n$.

Note that the $\theta^*$ is unseen to the model. Then, our vector generation tasks are defined as follows.

**Definition 8.4** (In-context task)**.** *Given in-context prompt examples/data* $X \in \mathbb{R}^{n \times d}$ *with their labels* $y = X\theta^*$*, where* $\theta^*$ *is unseen to model, and given a query* $\alpha \neq 0 \in \mathbb{R}$*, the task is to output/generate a vector* $q$ *such that* $\langle q, \theta^* \rangle$ *is as close to* $\alpha$ *as possible. Thus, the prediction error is*

$$|\langle q, \theta^* \rangle - \alpha|.$$

We remark that our vector generation task is as hard as the linear regression task, as they are dual problems. Solving any one of them requires to estimate the $\theta^*$.

Combining all above, we have the ICL prompt/input data for the model.

**Definition 8.5** (Input data)**.** *Let input be*

$$Z^{(0)} := \begin{bmatrix} X & y \\ q^{(0)\top} & \alpha \end{bmatrix} \in \mathbb{R}^{(n+1)\times(d+1)}.$$

In Definition 8.5, the model will iteratively update $q^{(0)} \in \mathbb{R}^d$ and make it as the generation output. In this work, we initialize $q^{(0)}$ as $\mathbf{0}_d$.

### 8.3.3 Linear Looped Transformer

In line with recent work by Gatmiry et al. (2024) and Ahn et al. (2024a), we consider a linear self-attention model, formally defined as follows:

**Definition 8.6** (Linear attention)**.** *Let* $Q, P \in \mathbb{R}^{d \times d}$ *be the query-key matrix and the value-output matrix. Let* $Z \in \mathbb{R}^{n \times d}$ *be input data. The linear attention is defined as*

$$\mathsf{Attn}(Z; Q, P) := (M \circ (ZQZ^\top))ZP,$$

*where* $M = \begin{bmatrix} \mathbf{0}_{n \times n} & \mathbf{0}_{n \times 1} \\ \mathbf{1}_{1 \times n} & 0 \end{bmatrix}$ *is a casual attention mask for text generation.*

In Definition 8.6, we combine the query matrix and key matrix as Q and combine the value matrix and output matrix as P for simplicity following previous works (Zhang et al., 2023c; Gatmiry et al., 2024).

Building upon this, we introduce the concept of a Linear Looped Transformer (Gatmiry et al., 2024):

**Definition 8.7** (Linear looped transformer)**.** *Let* $T$ *be the loop number. Let* $\eta^{(t)} > 0$ *for any* $t \in \{0, 1, \ldots, T-1\}$*. The linear looped transformer* $\mathsf{TF}(Z^{(0)}; Q, P)$ *is defined as*

$$Z^{(t)} := Z^{(t-1)} - \eta^{(t-1)} \mathsf{Attn}(Z^{(t-1)}; Q, P), \; \forall t \in [T]$$
$$\mathsf{TF}(Z^{(0)}; Q, P) := -{Z_{n+1,1:d}^{(T)}}^{\top}.$$

The Looped Transformer is to simulate a real multiple-layer Transformer with residual connections (He et al., 2016), where $\eta$ represents the weights of the residual components.

**Remark 8.8.** *Our settings are more practical than Gatmiry et al. (2024) in the following sense.*

- *We have a more practical casual attention mask used in generation. Our mask requires Hardamard product, the same as the standard attention, while Gatmiry et al. (2024) uses matrix product mask.*
- *Our model does not have prior knowledge of* $X$ *distribution, while the model in Gatmiry et al. (2024) knows the distribution of* $X$*, i.e., distribution free. In practice, the LLMs do not know any information about in-context examples.*

### 8.3.4 Linear Regression with Gradient Descent

In this section, we introduce some key concept of linear regression with gradient descent.

**Definition 8.9** (Linear regression)**.** *Given* $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, *and* $\theta \in \mathbb{R}^d$, *the linear regression loss function is defined as*

$$\ell(\theta) := 0.5\|y - X\theta\|_2^2.$$

*Then, the gradient formulation is*

$$\nabla_\theta \ell(\theta) = X^\top X\theta - X^\top y \in \mathbb{R}^d. \tag{8.1}$$

*For any* $t \in \mathbb{N}_+$, *let* $\eta^{(t)} > 0$ *and by Gradient Descent, we have*

$$\theta^{(t)} := \theta^{(t-1)} - \eta^{(t-1)}(X^\top X\theta^{(t-1)} - X^\top y). \tag{8.2}$$

We define the optimizer below.

**Definition 8.10.** *Let* $\widetilde{\theta} = (X^\top X)^{-1} X^\top y$ *be the optimizer of* $\ell(\theta)$.

In this work, we consider $n > d$, assuming $X^\top X$ is inevitable. Then, we have $\theta^* = \widetilde{\theta}$.

**Lemma 8.11** (Forklore)**.** *In the realizable setting* (*no noise term*), *if* $n > d$, *then* $\theta^* = \widetilde{\theta}$.

Finally, we define the condition number, which will be used in our final convergence bound.

**Definition 8.12.** *We define the condition number of input data as* $\kappa := \frac{\lambda_{\max}(X^\top X)}{\lambda_{\min}(X^\top X)}$.

## 8.4 Gradient Computation in Looped Transformer

In this section, we present a comprehensive analysis of the gradient computation process within the Looped Transformer architecture. Our investigation begins with an examination of computations in individual layers and subsequently extends to the full looped structure. This approach allows us to build a nuanced understanding of the Looped Transformer's behavior, starting from its fundamental components.

We commence our analysis by establishing a crucial result regarding the output of a single layer in our Looped Transformer model. This foundational lemma serves as a cornerstone for our subsequent derivations and provides valuable insights into the model's inner workings.

**Lemma 8.13** (Single layer output)**.** *Let* $Z^{(0)}$ *be defined in Definition 8.5. Let* $Q = I_{d+1,d+1}$*. Let* $P = \begin{bmatrix} I_{d\times d} & \mathbf{0}_{d\times 1} \\ \mathbf{0}_{1\times d} & 0 \end{bmatrix}$*. Let causal attention mask be* $M = \begin{bmatrix} \mathbf{0}_{n\times n} & \mathbf{0}_{n\times 1} \\ \mathbf{1}_{1\times n} & 0 \end{bmatrix}$*. Then, we have*

$$\mathsf{Attn}(Z^{(0)}; Q, P) = \begin{bmatrix} \mathbf{0}_{n\times d} & \mathbf{0}_{n\times 1} \\ q^{(0)\top} X^\top X + \alpha y^\top X & 0 \end{bmatrix}.$$

*Proof.* We can show that

$$\begin{aligned}
& \mathsf{Attn}(Z^{(0)}; Q, P) \\
= & (M \circ (Z^{(0)} Q (Z^{(0)\top}))) Z^{(0)} P \\
= & (M \circ \begin{bmatrix} X & y \\ q^{(0)\top} & \alpha \end{bmatrix} \begin{bmatrix} X^\top & q^{(0)} \\ y^\top & \alpha \end{bmatrix}) Z^{(0)} P \\
= & (M \circ \begin{bmatrix} XX^\top + yy^\top & Xq^{(0)} + \alpha y \\ q^{(0)\top} X^\top + \alpha y^\top & q^{(0)\top} q^{(0)} + \alpha^2 \end{bmatrix}) Z^{(0)} P \\
= & \begin{bmatrix} \mathbf{0}_{n\times n} & \mathbf{0}_{n\times 1} \\ q^{(0)\top} X^\top + \alpha y^\top & 0 \end{bmatrix} Z^{(0)} P \\
= & \begin{bmatrix} \mathbf{0}_{n\times n} & \mathbf{0}_{n\times 1} \\ q^{(0)\top} X^\top + \alpha y^\top & 0 \end{bmatrix} \begin{bmatrix} X & y \\ q^{(0)\top} & \alpha \end{bmatrix} P \\
= & \begin{bmatrix} \mathbf{0}_{n\times d} & \mathbf{0}_{n\times 1} \\ q^{(0)\top} X^\top X + \alpha y^\top X & q^{(0)\top} X^\top y + \alpha y^\top y \end{bmatrix} P \\
= & \begin{bmatrix} \mathbf{0}_{n\times d} & \mathbf{0}_{n\times 1} \\ q^{(0)\top} X^\top X + \alpha y^\top X & 0 \end{bmatrix}
\end{aligned}$$

where the first step follows from Definition 8.6, the second step follows from

Definition 8.5, and the rest steps directly follow from the matrix multiplication. □

This result illuminates the specific form of the attention mechanism's output in a single layer, which is essential for understanding the model's overall behavior, where the output only has non-zero terms in the position of $q^{(0)}$ and the format is close to Eq. equation 8.1.

Having characterized the behavior of a single layer, we now extend our analysis to encompass the full-looped transformer structure. Turning our attention to the core of our analysis with the groundwork laid for single-layer computations. The following theorem establishes a crucial relationship between the transformer's output and the iteratively updated parameters:

**Theorem 8.14.** *Let* $\alpha \neq 0 \in \mathbb{R}$. *Let* $\theta^{(0)} = -\frac{1}{\alpha} q^{(0)}$ . *Let* $\theta^{(i)}$ *correspondingly be defined in Definition 8.9 for any* $i \in \{2, 3, \ldots, T\}$. *Let* $\mathsf{TF}(Z^{(0)}; Q, P)$ *be defined in Definition 8.7. We have*

$$\mathsf{TF}(Z^{(0)}; Q, P) = \alpha \theta^{(T)}.$$

*Proof.* We will prove this theorem by induction on $t$, where $t \in [T]$. From Lemma 8.13, we have:

$$\mathsf{Attn}(Z^{(0)}; Q, P) = \begin{bmatrix} \mathbf{0}_{n \times d} & \mathbf{0}_{n \times 1} \\ q^{(0)\top} X^\top X + \alpha y^\top X & 0 \end{bmatrix}.$$

For $t = 1$, we can calculate:

$$\begin{aligned} Z^{(1)} &= Z^{(0)} - \eta^{(0)} \mathsf{Attn}(Z^{(0)}; Q, P) \\ &= \begin{bmatrix} X & y \\ q^{(0)\top} & \alpha \end{bmatrix} - \eta^{(0)} \begin{bmatrix} \mathbf{0}_{n \times d} & \mathbf{0}_{n \times 1} \\ q^{(0)\top} X^\top X + \alpha y^\top X & 0 \end{bmatrix} \\ &= \begin{bmatrix} X & y \\ q^{(0)\top} - \eta^{(0)} (q^{(0)\top} X^\top X + \alpha y^\top X) & \alpha \end{bmatrix} \end{aligned}$$

Where the first step follows from Definition 8.7, the second step follows from Definition 8.5 and Definition 8.6, the third step follows from basic algebra. Then we extract $q^{(0)\top} - \eta^{(0)}(q^{(0)\top} X^\top X + \alpha y^\top X)$, we have

$$\begin{aligned}
& q^{(0)\top} - \eta^{(0)}(q^{(0)\top} X^\top X + \alpha y^\top X) \\
= & -\alpha(-\frac{1}{\alpha} q^{(0)\top} - \eta^{(0)}(-\frac{1}{\alpha} q^{(0)\top} X^\top X - y^\top X)) \\
= & -\alpha(\theta^{(0)\top} - \eta^{(0)}(\theta^{(0)\top} X^\top X - y^\top X)) \\
= & -\alpha\theta^{(1)\top}.
\end{aligned}$$

where the first step follows from basic algebra, the second step follows from we defined $\theta^{(0)} = -\frac{1}{\alpha} q^{(0)}$, the third step follows from basic algebra. Thus, $q^{(1)} = -\alpha\theta^{(1)}$, so $\theta^{(1)} = -\frac{1}{\alpha} q^{(1)}$.

Similarly, by math induction, we can have $\theta^{(T)} = -\frac{1}{\alpha} q^{(T)}$. Thus, we finish the proof by $\mathsf{TF}(Z^{(0)}; Q, P) := -q^{(T)}$. □

To further refine our understanding of the Looped Transformer's performance, we introduce a bound on the final prediction error:

**Lemma 8.15.** *The final prediction error satisfies*

$$|\langle \mathsf{TF}(Z^{(0)}; Q, P), \theta^* \rangle - \alpha| \leqslant |\alpha| \cdot \|\theta^{(T)} - \theta^*\|_2.$$

*Proof.* By Definition 8.4, we have $q = \mathsf{TF}(Z^{(0)}; Q, P), \theta^* \rangle$. Then, we have

$$\begin{aligned}
|\langle \mathsf{TF}(Z^{(0)}; Q, P), \theta^* \rangle - \alpha| & = |\langle \alpha\theta^{(T)}, \theta^* \rangle - \alpha| \\
& = |\alpha| \cdot |\langle \theta^{(T)}, \theta^* \rangle - \langle \theta^*, \theta^* \rangle| \\
& = |\alpha| \cdot |\langle \theta^{(T)} - \theta^*, \theta^* \rangle| \\
& \leqslant |\alpha| \cdot \|\theta^{(T)} - \theta^*\|_2.
\end{aligned}$$

where the first step is from Definition 8.4 and Theorem 8.14, the second step follows $\|\theta^*\|_2 = 1$, the third step is from the linear properties of inner product, and the fourth step is from Cauchy-Schwarz inequality. □

This result provides a quantitative measure of the model's accuracy, linking it directly to the number of iterations and multi-step gradient descent results of linear regression in Eq. equation 8.2.

Finally, we present our main theoretical contribution, which encapsulates the core findings of our work:

**Theorem 8.16** (Main result. Formal version of Theorem 8.1)**.** *Let $\kappa$ be the condition number defined in Definition 8.12. Let $T$ be the number of loops. Let the initial point $q^{(0)} = \mathbf{0}_d$. Then, we have the final prediction error satisfies*

$$|\langle \mathsf{TF}(Z^{(0)}; Q, P), \theta^* \rangle - \alpha| \leqslant |\alpha| \cdot \exp(-\frac{T}{2\kappa}).$$

*Proof.* The proof directly follows Lemma 8.15 and Theorem 8.25. □

In Theorem 8.16, as long as the condition number is constant, we can see that the linear looped Transformer will perform better when the loop number is increasing, i.e., the error will exponentially decay to 0. Usually, we only need $O(d)$ numbers of in-context examples to guarantee a constant $\kappa$.

The above theorem offers a comprehensive characterization of the Looped Transformer's behavior, providing a tight bound on the prediction error that decays exponentially with the number of iterations. The intuition is that the Linear Looped Transformer can explicitly perform gradient descent in its hidden states. Furthermore, our theoretical finding is also supported by our experiments in Section 8.6.

**Comparison with Previous Works.** We restate the results in Gatmiry et al. (2024).

**Theorem 8.17** (Theorem 4.1 in Gatmiry et al. (2024))**.** *Under condition $\frac{8Td^2}{\sqrt{n}} \leqslant \frac{1}{2^{2T}}$, we have the optimal linear regression error is $\leqslant \frac{8Td^2 2^{2T}}{\sqrt{n}}$.*

In their work, the linear looped transformer has an error bound $8Td^2 2^{2T}/\sqrt{n}$, while our bound is $|\alpha| \cdot \exp(-\frac{T}{2\kappa})$. As the looped number $T$ increases, our error bounds will exponentially decay, while theirs is exponentially increase. Note that our linear vector generation task and their linear regression task are dual problems.

Our results align with the common intuition that more steps of gradient descent lead to better performance.

## 8.5 Error Convergence

In this section, we explore the convergence properties of looped transformers, focusing on their behavior under conditions of strong convexity and smoothness. We begin by defining these key concepts and then proceed to establish their implications.

### 8.5.1 Convexity and Smoothness Analysis

We first introduce some crucial definitions.

**Definition 8.18** (Strong convexity). *Let* $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ *and* $\mu > 0$. *We say that* $f$ *is* $\mu$*-strongly convex if, for every* $x, y \in \mathbb{R}^d$*, and every* $t \in (0, 1)$ *we have that*

$$\mu \frac{\gamma(1-\gamma)}{2} \|x - y\|_2^2 + f(\gamma x + (1-\gamma)y) \\ \leqslant \gamma f(x) + (1-\gamma) f(y).$$

*We say that* $\mu$ *is the strong convexity constant of* $f$.

**Definition 8.19** ($L$-smooth). *Let* $f : \mathbb{R}^d \to \mathbb{R}$ *and* $L > 0$. *We say that* $f$ *is* $L$*-smooth if it is differentiable and if* $\nabla f : \mathbb{R}^d \to \mathbb{R}^d$ *is* $L$*-Lipschitz for all* $x, y \in \mathbb{R}^d$

$$\|\nabla f(x) - \nabla f(y)\|_2 \leqslant L \|x - y\|_2.$$

To quantitatively analyze the parameter dynamics in our linear vector generation task, we first derive the Lipschitz and convexity constants for the model introduced in Definition 8.9.

**Lemma 8.20.** *Given* $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, *and* $\theta \in \mathbb{R}^d$ *in Definition 8.9, we have*

$$L = \|X^\top X\|$$

*where* $L$ *is the Lipschitz constant defined in Definition 8.19.*

*Proof.* From Definition 8.9, we have

$$\ell(\theta) = 0.5\|y - X\theta\|_2^2.$$

The gradient will be

$$\nabla_\theta \ell(\theta) = X^\top X\theta - X^\top y \in \mathbb{R}^d.$$

Then for $\theta_1, \theta_2 \in \mathbb{R}^d$, we have

$$\begin{aligned} \|\nabla \ell_{\theta_1}(\theta_1) - \nabla \ell_{\theta_2}(\theta_2)\|_2 &= \|X^\top X(\theta_1 - \theta_2)\|_2 \\ &\leqslant \|X^\top X\| \cdot \|\theta_1 - \theta_2\|_2 \end{aligned}$$

where the first step follows from Definition 8.9, and the second step follows from properties of norm. From Definition 8.19, we observe that

$$L = \|X^\top X\|,$$

where $\|X^\top X\|$ is the spectral norm of $X^\top X$ denoting the maximum eigenvalue. □

The following two lemmas are closely related and build upon each other to establish the strong convexity constant for a specific optimization problem.

**Lemma 8.21** (Forklore)**.** *For* $X \in \mathbb{R}^{n \times d}$ *and* $v \in \mathbb{R}^d$, *we have*

$$v^\top (X^\top X) v \geqslant \lambda_{\min}(X^\top X)\|v\|_2^2.$$

**Lemma 8.22.** *Given $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^n$, and $\theta \in \mathbb{R}^d$ in Definition 8.9, we have*

$$\mu = \lambda_{\min}(X^\top X)$$

*where $\mu$ is the strong convexity constant defined in Definition 8.18*

*Proof.* For any $\theta_1, \theta_2 \in \mathbb{R}^d$ and $t \in (0, 1)$, we have

$$\begin{aligned}
& \ell(\gamma\theta_1 + (1-\gamma)\theta_2) \\
= & 0.5\|y - X(\gamma\theta_1 + (1-\gamma)\theta_2)\|_2^2 \\
= & 0.5\|\gamma(y - X\theta_1) + (1-\gamma)(y - X\theta_2)\|_2^2 \\
\leqslant & 0.5(\gamma\|y - X\theta_1\|_2^2 + (1-\gamma)\|y - X\theta_2\|_2^2) \\
& - 0.5\gamma(1-\gamma)\|X(\theta_1 - \theta_2)\|_2^2 \\
= & 0.5(\gamma\|y - X\theta_1\|_2^2 + (1-\gamma)\|y - X\theta_2\|_2^2) \\
& - 0.5\gamma(1-\gamma)(\theta_1 - \theta_2)^T X^T X(\theta_1 - \theta_2) \\
\leqslant & 0.5(\gamma\|y - X\theta_1\|_2^2 + (1-\gamma)\|y - X\theta_2\|_2^2) \\
& - 0.5\gamma(1-\gamma)\lambda_{\min}(X^T X)\|\theta_1 - \theta_2\|_2^2
\end{aligned}$$

where the first step follows from Definition 8.9, the rest step follow from basic algebra and Lemma 8.21. From Definition 8.18, we observe that

$$\mu = \lambda_{\min}(X^\top X)$$

where $\lambda_{\min}(X^\top X)$ denotes the minimum eigenvalue of $X^\top X$. □

## 8.5.2 Main Result

We first commence with a statement of Lemma 8.23, which furnishes a convergence rate for gradient descent on strongly convex and smooth functions.

**Lemma 8.23** (Theorem 3.6 in Garrigos and Gower (2023))**.** *Let $\ell : \mathbb{R}^d \to \mathbb{R}$ be a differentiable function and assume $\ell$ is $\mu$-strongly convex and $L$-smooth. Let $\{\theta^{(t)}\}_{t \in \mathbb{N}}$ be* *the sequence generated by the gradient descent algorithm, with a stepsize* $\eta \in (0, \frac{1}{L}]$. *Then for* $\theta^* = \arg\min_\theta \ell(\theta)$ *and for all* $t \in \mathbb{N}$*, we have*

$$\|\theta^{(t)} - \theta^*\|_2^2 \leqslant (1 - \eta\mu)^t \|\theta^{(0)} - \theta^*\|_2^2.$$

**Fact 8.24** (Folklore)**.** *For any* $n, T \in N_+$*, we have*

$$(1 - \frac{1}{n})^T \leqslant e^{-T/n}.$$

We now present a rigorous upper bound on the error magnitude of the gradient descent algorithm's output after $T$ iterations, elucidating the convergence properties of this optimization method in the context of linear vector generation.

**Theorem 8.25.** *If the following holds:*

- *Let* $T$ *be the loop number.*
- *Let* $X \in \mathbb{R}^{n \times d}$*,* $y \in \mathbb{R}^n$*, and* $\theta \in \mathbb{R}^d$ *be defined in Definition 8.9.*
- *Let the condition number* $\kappa = \frac{\lambda_{\max}(X^\top X)}{\lambda_{\min}(X^\top X)}$.
- *The step size* $\eta = \frac{1}{L}$.
- *Let* $\mu$ *and* $L$ *be defined in Definition 8.18 and Definition 8.19.*
- *For* $\ell(\theta)$ *defined in Definition 8.9, we have* $\ell(\theta)$ *is* $L$*-smooth and* $\mu$ *strong convex, where* $L = \|X^\top X\|$ *and* $\mu = \lambda_{\min}(X^\top X)$.
- *The initial point* $\theta^{(0)}$ *satisfies* $\|\theta^{(0)} - \theta^*\|_2 \leqslant R$.

*Then, we have*

$$\|\theta^* - \theta^{(T)}\|_2^2 \leqslant e^{-T/\kappa} R^2.$$

*Proof.* First, we have

$$\|\theta^{(t)} - \theta^*\|_2^2 \leqslant (1 - \eta\mu)^t \|\theta^{(0)} - \theta^*\|_2^2$$

$$\leqslant (1 - \frac{\mu}{L})^t \|\theta^{(0)} - \theta^*\|_2^2$$

where the first step follows from Lemma 8.23, the second step follows from we choose $\eta^{(t)} = \frac{1}{L}$. Then consider the term $\frac{\mu}{L}$, we have

$$\frac{\mu}{L} = \frac{\lambda_{\min}(X^\top X)}{\|X^\top X\|} = \frac{1}{\kappa}$$

where the first step follows from Lemma 8.20 and Lemma 8.22, the second step follows the definition of condition number $\kappa = \frac{\lambda_{\max}(X^\top X)}{\lambda_{\min}(X^\top X)}$.

Then substituting this back, we have

$$\begin{aligned} \|\theta^{(T)} - \theta^*\|_2^2 &\leqslant (1 - \frac{1}{\kappa})^T \|\theta^{(0)} - \theta^*\|_2^2 \\ &\leqslant e^{-T/\kappa} \|\theta^{(0)} - \theta^*\|_2^2 \\ &\leqslant e^{-T/\kappa} R^2 \end{aligned}$$

where the first step follows from basic algebra, the second step follows from $(1 - 1/\kappa)^\kappa < e^{-1}$ (Fact 8.24), and the last step follows from $\|\theta^{(0)} - \theta^*\|_2 \leqslant R$. □

Theorem 8.25 tells us that GD can well solve linear regression tasks. In particular, when the input data has a good condition number, the approximation error will exponentially decay to 0. We use the above insights in the proof of our main results (Theorem 8.16).

## 8.6 Experiments

In this section, we aim to verify our theory by evaluating the convergence behavior of gradient descent for linear vector generation. We designed our experiment to examine the impact of varying sample sizes on convergence rates while keeping the feature dimension fixed. Our results demonstrate that empirical convergence rates consistently outperform theoretical upper bounds across all sample sizes, with significant improvement in convergence speed as the condition number decreases, validating our theoretical predictions.

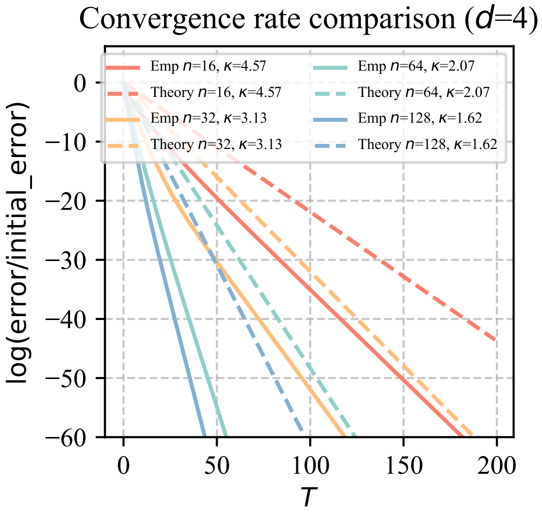


Figure 8.1: The convergence rate comparison for gradient descent in linear vector generation with a fixed dimension $d = 4$ and varying sample sizes $n \in \{16, 32, 64, 128\}$ and their corresponding condition number $\kappa$. The 'Emp' means the empirical error of our experiments. The 'Theory' means the theoretical bound in Theorem 8.16. The $y$-axis is the logarithm of normalized error and the $x$-axis is the number of loops $T$. Both empirical (solid lines) and theoretical (dashed lines) results are presented for each $n$. The plot demonstrates that as the sample size $n$ increases, the condition number will decrease, so the convergence rate improves. Thus, with larger $n$ values, there will be a steeper slope and faster convergence to the optimal solution.

**Experiment Setup.** In this experiment, we aimed to investigate the convergence behavior of multi-step gradient descent for Linear Looped Transformer (Definition 8.7) in-context learning the linear vector generation task (Definition 8.4). We randomly draw each entry of $X \in \mathbb{R}^{n \times d}$ from standard Gaussian distribution, $\mathcal{N}(0, 1)$, and response variables $y = X\theta^*$, where $\theta^* \in \mathbb{R}^d$ was randomly chosen. Our experiments focused on scenarios with $d$ fixed at 4 and $n$ varying in $\{16, 32, 64, 128\}$. For each $(n, d)$ combination, we implemented gradient descent with $T = 200$ iterations and learning rate $\eta = 1/L$, where $L = \|X^\top X\|$. To ensure statistical robustness, we conducted 10 independent trials for each configuration. Convergence was measured by tracking $\|\theta^{(t)} - \theta^*\|_2^2$, where $\theta^*$ is the optimal least squares solution. To facilitate comparison across different problem sizes, we normalized error plots by the initial error, plotting $\log(\|\theta^{(t)} - \theta^*\|_2^2 / \|\theta^{(0)} - \theta^*\|_2^2)$ against $t$. This normalization enabled a clear comparison of relative decay rates, irrespective of initial error magnitudes, thus providing insights into the impact of condition number on gradient descent convergence in the linear vector generation task.

**Result Interpretation.** Our experiment investigates the convergence behavior of gradient descent for linear vector generation with varying sample sizes $n$ and a fixed feature dimension $d = 4$. Figure 8.1 illustrates the convergence rates for different $n$ values $\{16, 32, 64, 128\}$, comparing empirical results with theoretical bounds in Theorem 8.16. A key aspect of this experiment is the condition number $\kappa$, which decreases as the number of examples increases. The average $\kappa$ values for the different sample sizes are $\{4.57, 3.13, 2.07, 1.62\}$, corresponding to $n = \{16, 32, 64, 128\}$ respectively. This inverse relationship between $n$ and $\kappa$ is noteworthy, as it significantly influences the convergence rates. The results demonstrate that as the sample size $n$ increases, the convergence rate improves substantially. This is evident from the steeper slopes of both empirical and theoretical lines for larger $n$ values. Importantly, the empirical convergence rates consistently outperform the theoretical upper bounds across all sample sizes, with the gap between empirical and theoretical performance narrowing as $n$ increases. This observation aligns with our theoretical expectations in Theorem 8.16 and highlights the crucial role of the condition number in determining convergence behavior.

## 8.7 Conclusion

In this work, we have demonstrated that linear looped Transformers can efficiently implement multi-step gradient descent for in-context learning, requiring only a reasonable number of examples when input data is well-conditioned. This finding relieves the previous assumptions of an exponential number of in-context examples and offers new insights into the capabilities of Transformer architectures. Our theoretical analysis and preliminary experiments pave the way for more efficient inference algorithms in large language models and open avenues for future research in this domain.

# 9 DISCOVERING THE GEMS IN EARLY LAYERS: ACCELERATING LONG-CONTEXT LLMS WITH 1000X INPUT TOKEN REDUCTION

**Contribution statement.** This chapter is joint work with Yifei Ming, Xuan-Phi Nguyen, Yingyu Liang, and Shafiq Joty. The author Zhenmei Shi proposed the method, contributed to all theoretical analysis, and completed all the experiments. The results in this chapter is submitted to ICLR 2025 (Shi et al., 2024a).

## 9.1 Introduction

Large Language Models (LLMs) have demonstrated impressive abilities (Wei et al., 2022c; Bubeck et al., 2023) and found widespread application in various AI systems, such as ChatGPT (Schulman et al., 2022), Gemini (Team et al., 2023), and Claude (Anthropic, 2024), and so on. They are also a fundamental component in building language-based AI agents that can orchestrate plans and execute complex tasks through interaction with external tools. A key requirement for many of these applications is the ability to process long-context inputs. This ability can also potentially eliminate the need of a retriever in retrieval augmented generation (RAG) (Xu et al., 2024b) or enhance its performance (Jiang et al., 2024c). Therefore, significant efforts have been made recently to build LLMs that support long context inputs. For instance, LLaMA 3.1 (AI, 2024), Mistral (Jiang et al., 2023a), and Phi 3.5 (Abdin et al., 2024) now support input sequences of up to 128K tokens, while Gemini can handle inputs of up to 1M tokens. However, processing such lengthy inputs comes at a substantial cost in terms of computational resources and time. Therefore, accelerating the LLM generation speed while simultaneously reducing GPU memory consumption for long-context inputs is essential to minimize response latency and increase throughput for LLM API calls.

One prominent optimization for fast text generation in decoder-only LLMs (i.e., using a causal attention mask) is the *KV cache*. Specifically, there are two phases involved in auto-regressive generation. Given a long context input, the first is the *prompt computation* phase, when the LLM computes the KV cache for all layers, storing the intermediate attention keys and values of the input tokens. Next, in the *iterative generation* phase, the LLM generates tokens iteratively using the precomputed KV cache, avoiding redundant computations. GPU memory usage and running time scale linearly with the KV cache size, meaning that the computational is high for long inputs.

To reduce GPU memory usage and running time during the iterative generation phase, H2O (Zhang et al., 2023e) and SnapKV (Li et al., 2024g) introduce static methods to compress/evict the KV cache. These techniques can shrink the KV cache size from 128K to 1024 with negligible performance loss, resulting in faster speeds and lower GPU memory consumption during the iterative generation phase. However, these methods do not improve the efficiency of the prompt computation phase, which becomes the dominant bottleneck as the input context lengthens. Thus, we ask:

*Can we accelerate the speed and reduce memory usage during the prompt computation phase?*

We observe that when serving a query, LLMs often find the necessary information in the early layers, even before generating the answer. Specifically, the relevant tokens can be identified using the attention matrix from these early layers (Figure 9.2), which we refer to as *filter layers*. Figure 9.1 provides a real example from the Needle in a Haystack task, where LLMs must find a small piece of information within a large context. For LLaMA 3.1 8B, we observe that the information needed to answer the query can be distilled from the attention matrix in any of the 13th-19th layers. Furthermore, LLMs explicitly summarize the required information in these filter layers. As a consequence, we only need to per-

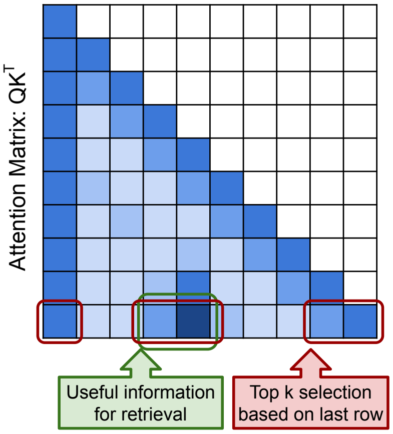
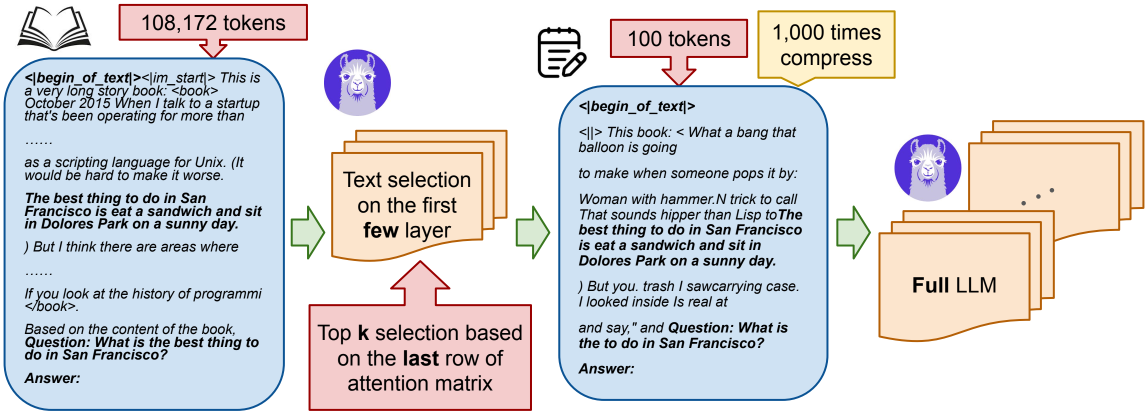

Attention Matrix: $QK^T$
Useful information for retrieval
Top k selection based on last row

Figure 9.2: The last row of attention matrices in early layers can locate answer-related tokens.


108,172 tokens
Text selection on the first few layer
Top k selection based on the last row of attention matrix
100 tokens
1,000 times compress
Full LLM

**<|begin_of_text|>**<|im_start|> This is a very long story book: <book> October 2015 When I talk to a startup that's been operating for more than

……

as a scripting language for Unix. (It would be hard to make it worse.

**The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.**

) But I think there are areas where

……

If you look at the history of programmi </book>.

Based on the content of the book, **Question: What is the best thing to do in San Francisco?**

**Answer:**

**<|begin_of_text|>**

<||> This book: < What a bang that balloon is going

to make when someone pops it by:

Woman with hammer.N trick to call That sounds hipper than Lisp to**The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.**

) But you. trash I sawcarrying case. I looked inside Is real at

and say," and **Question: What is the to do in San Francisco?**

**Answer:**

Figure 9.1: Illustration of our method GemFilter: generation with context selection based on early filter layers. We demonstrate a real Needle in a Haystack task (Section 9.4.1). The original input consists of 108,172 tokens, including the initial instruction, key message, and the query. In the first step, we use the 13th layer of the LLM (LLaMA 3.1 8B Instruct) as a filter to compress the input tokens by choosing the top k indices from the last row of the attention matrix. Notably, the selected input retains the initial instruction, key message, and query. GemFilter achieves a 1000× compression, reducing the input token length to 100. In the second step, we feed the selected tokens for full LLM inference using a standard generation function, which produces the correct output. GemFilter significantly reduces running time and GPU memory with negligible performance loss.

form the prompt computation on a long context input for the filter layers, allowing us to compress the input tokens into a smaller subset (e.g., reducing from 128K tokens to 100), saving both time and GPU memory. We then feed the selected tokens for full model inference and proceed with a standard generation function. Algorithm 2 in Section 9.3 presents our method GemFilter.

As shown in Figure 9.3, GemFilter runs faster and consumes less GPU memory than SnapKV/H2O and standard attention (full KV cache) during the prompt computation phase. During the iterative generation phase, GemFilter has the same running time and GPU memory consumption as SnapKV/H2O, both of which outperform standard attention. We discuss the complexity further in Section 9.3.2 theoretically and in Section 9.4.5 empirically. GemFilter significantly outperforms standard attention and SnapKV on the Needle in a Haystack benchmark (Sec-

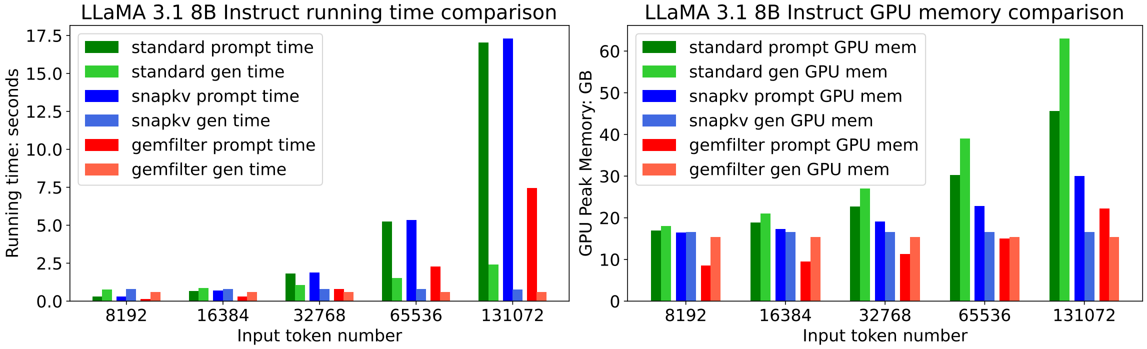


Figure 9.3: Comparison of time and GPU memory usage across different methods on LLaMA 3.1 8B Instruct. 'gemfilter' represents our method, using the 13th layer as the filter. It achieves a 2.4× speedup and reduces GPU memory usage by 30% compared to SnapKV. The iterative generation is evaluated on 50 tokens generation. Additional results can be found in Section 9.4.5.

tion 9.4.1). Additionally, on LongBench, a multi-task benchmark designed to rigorously evaluate long-context understanding across various datasets, GemFilter achieves performance comparable to SnapKV/H2O (Section 9.4.2). Furthermore, our ablation study in Section 9.4.3 shows that our method is quite robust to the filter layer selection strategy and Section 9.4.4 shows that each component in our algorithm is essential.

**Our contributions and advantages are:**

- We found that LLMs can identify relevant tokens using attention matrices in the early layers, suggesting crucial information is recognized before the answer generation. Furthermore, LLMs explicitly summarize this information within specific filter layers. This observation provides insights into LLM mechanisms and opens avenues for LLM understanding and algorithm design.

- Leveraging this insight, we develop GemFilter, formulated in Algorithm 2, an inference strategy that utilizes early LLM layers as a filter to select and compress input tokens into a small subset to be processed by the full model (Figure 9.1). GemFilter achieves a 2.4× speedup and reduces GPU memory consumption by 30% compared to state-of-the-art methods like SnapKV.

- GemFilter significantly outperforms both standard attention (all KV cache) and SnapKV on the Needle in a Haystack benchmark (Section 9.4.1), while maintaining performance comparable to SnapKV/H2O on the LongBench benchmark (Table 9.1).
- We provide a thorough ablation studies for the GemFilter in Section 9.4.3 and Section 9.4.4.
- Our approach offers several advantages: it is simple, training-free, and broadly applicable to various LLMs. Furthermore, it enhances interpretability by allowing humans to directly inspect the selected token sequence.

## 9.2 Related Works

**Generation Speed-up with Long Context Input.** One effective technique to accelerate auto-regressive generation is KV cache compression/eviction. During generation, LLMs store the previous key and value matrices to reduce computational complexity. However, when the input context is long (e.g., 128K tokens), the memory consumption and running time associated with the KV cache dominate iterative generation. Many studies have focused on KV cache eviction. For instance, Ge et al. (2023) evict long-range contexts on attention heads to prioritize local contexts, using the KV cache only for heads that broadly attend to all tokens. Streaming LLM (Xiao et al., 2023) introduces an attention sink that retains only the first few tokens and the latest $k$ tokens in the KV cache to enable fast streaming generation. LOOK-M (Wan et al., 2024) applies KV eviction in the multimodality so that the model only needs to look once for the image. LongWriter (Bai et al., 2024b) uses KV eviction to enable LLMs to generate coherent outputs exceeding 20,000 words. MInference 1.0 (Jiang et al., 2024a) introduces $\wedge$-shape, vertical-slash, and block-sparse attention head and determines the optimal KV cache pattern for each attention head offline and dynamically builds sparse indices based on the assigned query during inference. QuickLLaMA (Li et al., 2024c) classifies the KV cache to many subsets, e.g., query tokens, context tokens, global tokens, and local tokens, and only preserves some types of tokens in the KV cache. ThinK (Xu et al., 2024c) proposes a query-dependent KV cache pruning method by pruning the least significant channel dimensions of the KV cache. H2O (Zhang et al., 2023e) retains only tokens contributing to cumulative attention. SnapKV (Li et al., 2024g) evicts non-essential KV positions for each attention head based on observation windows. While the aforementioned studies focus on eviction and compression of the KV cache during the prompt computation phase to optimize the iterative generation phase, they do not reduce the running time or GPU memory usage during the prompt computation phase. In contrast, our method, GemFilter, achieves both reduced running time and GPU memory usage in the prompt computation phase, as well as during the iterative generation phase. We provide a more detailed comparison in Appendix G.2.

More related to our work, Li et al. (2023g) compress input sequences by pruning redundancy in the context, making inputs more compact. However, they need to keep 50% of input tokens to keep the LLMs' performance, whereas GemFilter achieves comparable performance by only reserving 1% of input tokens. For further details, we refer the reader to Section 9.4.1. The LLMLingua series methods (Jiang et al., 2023b; Pan et al., 2024; Jiang et al., 2024b) present a coarse-to-fine approach for prompt compression. It leverages a budget controller to ensure semantic integrity even at high compression ratios, employs a token-level iterative compression algorithm to model interdependencies within the compressed content, and utilizes an instruction-tuning strategy to achieve distribution alignment across language models.

## 9.3 Method

**Notations and Preliminary.** While the Transformer and self-attention architecture (Vaswani et al., 2017) have already become overwhelmingly popular, we first introduce preliminary definitions to provide a better methodological connection to our proposed GemFilter method in Section 9.3.1.

For any positive integer $n$, we use $[n]$ to denote the set $\{1, 2, \cdots, n\}$. We use $\circ$ to denote function composition and $\odot$ to denote the Hardamard product. Let $n$ be the input token/prompt length, $d$ the hidden feature dimension, and $\mathcal{V}$ the vocabulary set. We now introduce the key concept of attention and transformers. We first define the query, key, and value matrices. It is important to note that during text generation, the key and value matrices are also referred to as the KV cache, as they are stored in GPU memory to reduce running time during the iterative prediction of the next token.

**Definition 9.1** (Single layer self-attention). *Let* $Q \in \mathbb{R}^{n \times d}$ *be the query matrix ,* $K \in \mathbb{R}^{n \times d}$ *the key cache, and* $V \in \mathbb{R}^{n \times d}$ *the value cache. Let* $M_c \in \{0, 1\}^{n \times n}$ *be the causal attention mask, where* $(M_c)_{i,j}$ *is* 1 *if* $i \geqslant j$ *and* 0 *otherwise. The self-attention function* $\mathsf{Attn}$ *is defined as:*

$$\mathsf{Attn}(Q, K, V) = M_c \odot \mathsf{Softmax}(QK^\top/\sqrt{d}) \cdot V$$

**Definition 9.2** (Multi-layer transformer). *Let* $T \in \mathcal{V}^n$ *represent the input tokens, and let* $m$ *denote the number of transformer layers. Let* $g_i$ *represent components in the* $i$*-th transformer layer other than self-attention, such as layer normalization, residual connections, and the MLP block, where* $g_i : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d}$ *for any* $i \in \{0, 1, \ldots, m\}$*. Let* $\mathsf{Attn}_i$ *denote the self-attention module in the* $i$*-th transformer layer. We define an* $m$*-layer transformer* $F_{1:m} : \mathcal{V}^n \to \mathbb{R}^{n \times d}$ *as*

$$F_{1:m}(T) := g_m \circ \mathsf{Attn}_m \circ g_{m-1} \circ \cdots \circ g_1 \circ \mathsf{Attn}_1 \circ g_0 \circ \mathcal{E}(T) \quad \in \mathbb{R}^{n \times d},$$

*where* $\mathcal{E}$ *is the input embedding function mapping the input tokens to hidden features using the vocabulary dictionary, i.e.,* $\mathcal{E}(T) \in \mathbb{R}^{n \times d}$.

Note that the above definitions use a single attention head for simplicity, but in practice, multi-head attention is used (Vaswani et al., 2017).

**Algorithm 2** GemFilter: Generation with Token Selection Based on Early Layers

```
1: procedure SelectionGen($F_{1:m}, T \in [\mathcal{V}]^n, r \in [m], k \in [n]$)
2:         ▷ $F_{1:m}$ : An $m$-layer transformer network; $T$: input sequence of tokens
3:         ▷ $r$: filter layer index for token selection; $k$: number of selected tokens
4:     Get $Q^{(r)}, K^{(r)}$ by doing a $r$-layer forward pass: $F_{1:r}(T)$
5:             ▷ $Q^{(r)}, K^{(r)} \in \mathbb{R}^{n \times d}$: the $r$-th layer query, key
6:     $J \leftarrow \mathsf{topk\_index}(Q^{(r)}_n {K^{(r)}}^\top, k)$  ▷ $Q^{(r)}_n$: the last row of $Q^{(r)}$; $Q^{(r)}_n {K^{(r)}}^\top \in \mathbb{R}^n$ are attn scores
7:     Sort the indices in $J$            ▷ $J \subseteq [n]$ and $|J| = k$
8:     return Gen($F_{1:m}, T_J$) ▷ Gen is generation function, $T_J \in [\mathcal{V}]^k$ is a sub-sequence of $T$ on $J$
9: end procedure
```

### 9.3.1 Our Algorithm: GemFilter

We present our method, GemFilter, in Algorithm 2. We also present PyTorch code in Appendix G.4.1 for the reader's interests. The high-level idea is to run the LLM twice. In the first pass, we run only the early layers of the LLM to select the key input tokens. This corresponds to the prompt computation phase (Line 4-7 of Algorithm 2). This process selects the top $k$ tokens that receive the most attention from the last query token. In the second pass, we feed the selected tokens to the full LLM and run the generation function, corresponding to the iterative generation phase (Line 8). Below, we explain Algorithm 2 step by step.

The input of the algorithm is an $m$-layer transformer $F_1$ (Definition 9.2), an input token sequence $T \in \mathcal{V}^n$, and two hyperparameters $r \leqslant m, k \leqslant n$, where $r$ represents the index of the filter layer for context token selection and $k$ denotes the number of tokens to select. For example, in the case of LLaMA 3.1 8B Instruct (Figure 9.1), we have $m = 32$, $r = 13$, and $k = 1024$.

In the first step (Line 4), we run only the first $r$ layers forward to serve as a filter, obtaining the $r$-th layer's query and key matrices, $Q^{(r)}$ and $K^{(r)}$. Note that we do not need to run all layers of the LLM on a long context input, thereby saving both computation time and memory (see detailed analysis in Section 9.3.2). In Line 6, we select token indices based on the $r$-th layer attention matrix. The selection is made by identifying the $k$ largest values from the last row of the attention matrix, i.e., the inner product between the last query token $Q_n^{(r)}$ and all key tokens $K^{(r)}$. For multi-head attention, the top-$k$ indices are selected based on the summation of the last row across the attention matrices of all heads. For instance, suppose we have $h$ attention heads, and let $Q^{(r,j)}, K^{(r,j)} \in \mathbb{R}^{n \times d}$ represent the query and key matrices for the $r$-th layer and $j$-th attention head. Then, we compute $J \leftarrow \mathsf{topk\_index}(\sum_{j=1}^{h} Q_n^{(r,j)} {K^{(r,j)}}^\top, k)$, where $J$ is a set of top $k$ index selection. Note that our method uses a single index set $J$, whereas SnapKV (Li et al., 2024g) and H2O (Zhang et al., 2023e) use different index sets for each layer and attention head, resulting in $m \cdot h$ index sets in total. A detailed discussion is provided in Appendix G.2.

In Line 6, $J$ is sorted by inner product values. However, we need to re-sort $J$ so that the selected tokens follow their original input order, ensuring, for example, that the $\langle \mathsf{bos} \rangle$ token is placed at the beginning. Line 7 performs this reordering operation. Finally, in Line 8, we can run any language generation function using the selected tokens $T_J$, which is a sub-sequence of $T$ on the index set $J$, across all layers. This generation is efficient as the input context length is reduced from $n$ to $k$, e.g., from 128K to 1024 tokens in Figure 9.1. Below, we provide a formal time complexity analysis.

### 9.3.2 Running Time and Memory Complexity Analysis

The results of our analysis on time complexity and GPU memory consumption are presented in Theorem 9.3 below, with the proof deferred to Appendix G.3.

**Theorem 9.3** (Complexity analysis)**.** *Let $n$ be the input sequence (prompt) length and $d$ the hidden feature dimensions. In our Algorithm 2, GemFilter uses the $r$-th layer as a filter to select $k$ input tokens. Let SnapKV and H2O also use $k$ as their cache size. Assume the LLM has $m$ attention layers, each with $h$ attention heads, and each transformer layer's parameters consume $w$ GPU memory. Assuming that we generate $t$ tokens with the* Gen *function and $n \geqslant \max\{d, k, t\}$, the following table summarizes the complexity for standard attention, SnapKV and H2O, and GemFilter:*

| *Complexity* | | *Standard attention* | *SnapKV and H2O* | *GemFilter* |
|---|---|---|---|---|
| *Time* | *Prompt Comp.* | $\Theta(mhn^2d)$ | $\Theta(mhn^2d)$ | $\Theta(rhn^2d)$ |
| | *Iter. generation* | $\Theta(mh(nt+t^2)d)$ | $\Theta(mh(kt+t^2)d)$ | $\Theta(mh(k^2+t^2)d)$ |
| *GPU mem.* | *Prompt Comp.* | $mw+2mhnd$ | $mw+2hnd+2mhkd$ | $rw+2hnd$ |
| | *Iter. generation* | $mw+2mh(n+t)d$ | $mw+2mh(k+t)d$ | $mw+2mh(k+t)d$ |

Recall that there are two phases in text generation. The first phase is *prompt computation*, which involves attention computation on the long context input tokens and generating the KV cache. The second phase is *iterative generation*, where auto-regressive generation occurs based on the pre-computed KV cache. Theorem 9.3 demonstrates that GemFilter is faster and consumes less GPU memory than SnapKV/H2O and standard attention during the prompt computation phase. Additionally, during the iterative generation phase, GemFilter has the same running time and GPU memory consumption as SnapKV/H2O, which is significantly better than standard attention. This conclusion aligns with our experimental results in Section 9.4.5.

**Case Study.** Let us consider the case $n \gg k \approx t$, e.g., $n =$128K, $k = t = 1024$ and $r < m$. During the prompt computation phase, we have the running time and the GPU memory consumption:

$$\text{Standard attention} : \text{SnapKV/H2O} : \text{GemFilter} = \Theta(m : m : r),$$
$$\text{Standard attention} : \text{SnapKV/H2O} : \text{GemFilter} \approx mw + mhnd : mw + hnd : rw + hnd,$$

We see that GemFilter has a lower time complexity and less GPU memory consumption than standard attention, SnapKV, and H2O. During the iterative generation phase, we have the running time and the GPU memory consumption:

$$\text{Standard attention} : \text{SnapKV/H2O} : \text{GemFilter} = \Theta(n : k : k),$$
$$\text{Standard attention} : \text{SnapKV/H2O} : \text{GemFilter} \approx w/hd + 2n : w/hd + 4k : w/hd + 4k,$$

As such, GemFilter has the same time complexity and GPU memory consumption as SnapKV/H2O, while significantly outperforming the standard attention. The running time bottleneck for all methods occurs during prompt computation, which takes $\Theta(mhn^2d)$ for standard attention, SnapKV, and H2O. In contrast, GemFilter only requires $\Theta(rhn^2d)$ for prompt computation, as it only processes the early layers of the LLMs to select and compress the input tokens during the first run. See detailed proof in Appendix G.3. Note that the GPU memory bottleneck for standard attention occurs during iterative generation, while for other methods, the memory bottleneck arises during prompt computation due to the reduced KV cache. GemFilter consumes less GPU memory than SnapKV and H2O because it only requires loading some layer model weights when processing the long context input in its first run. Our empirical results in Section 9.4.5 support our complexity analysis findings.

## 9.4 Experiments

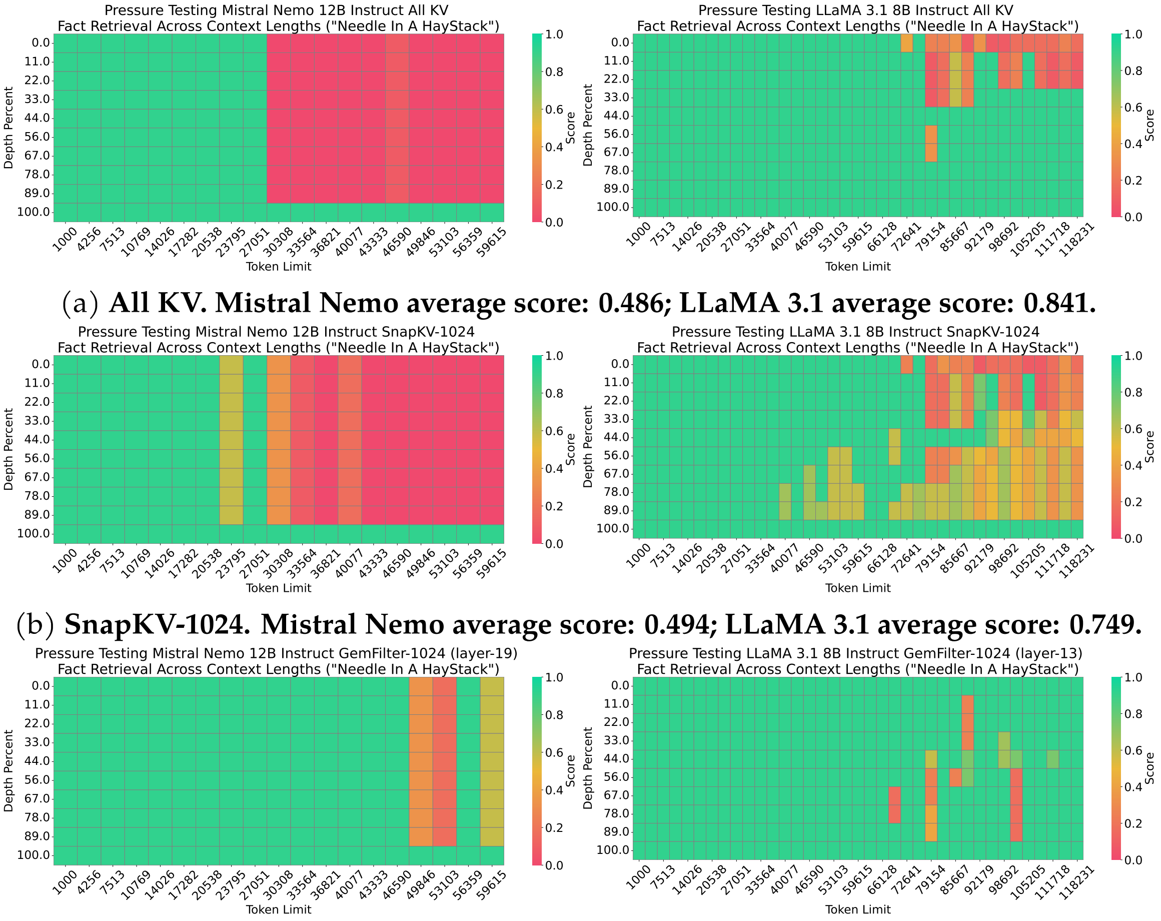
**Model and Datasets.** We evaluated our approach using three popular long-context models: LLaMA 3.1 8B Instruct[1] (AI, 2024), Mistral Nemo 12B Instruct[2] (Jiang et al., 2023a), and Phi 3.5 Mini 3.8B Instruct[3] (Abdin et al., 2024), all of which support an input token length of 128K. We compared our method, GemFilter, against standard attention and two state-of-the-art methods, SnapKV (Li et al., 2024g) and H2O (Zhang et al., 2023e)[4]. For our experiments, we used two popular datasets: Needle in a Haystack (Kamradt, 2024) (Section 9.4.1) and LongBench (Bai et al., 2023) (Section 9.4.2). More implementation details are provided in Appendix G.4.2.

[1] https://huggingface.co/meta-llama/Meta-Llama-3.1-8B-Instruct

[2] https://huggingface.co/mistralai/Mistral-Nemo-Base-2407

[3] https://huggingface.co/microsoft/Phi-3.5-mini-instruct

[4] While there are many other generation acceleration methods, they may not be directly comparable to ours as they use orthogonal techniques. We refer the reader to Section 9.2 for further details.


Pressure Testing Mistral Nemo 12B Instruct All KV
Fact Retrieval Across Context Lengths ("Needle In A HayStack")

Pressure Testing LLaMA 3.1 8B Instruct All KV
Fact Retrieval Across Context Lengths ("Needle In A HayStack")

(a) **All KV. Mistral Nemo average score: 0.486; LLaMA 3.1 average score: 0.841.**

Pressure Testing Mistral Nemo 12B Instruct SnapKV-1024
Fact Retrieval Across Context Lengths ("Needle In A HayStack")

Pressure Testing LLaMA 3.1 8B Instruct SnapKV-1024
Fact Retrieval Across Context Lengths ("Needle In A HayStack")

(b) **SnapKV-1024. Mistral Nemo average score: 0.494; LLaMA 3.1 average score: 0.749.**

Pressure Testing Mistral Nemo 12B Instruct GemFilter-1024 (layer-19)
Fact Retrieval Across Context Lengths ("Needle In A HayStack")

Pressure Testing LLaMA 3.1 8B Instruct GemFilter-1024 (layer-13)
Fact Retrieval Across Context Lengths ("Needle In A HayStack")

(c) **GemFilter-1024. Mistral Nemo average score: 0.838; LLaMA 3.1 average score: 0.887.**

Figure 9.4: Needle in a Haystack performance comparison of different methods using the Mistral Nemo 12B Instruct model (left column) and the LLaMA 3.1 8B Instruct model (right column). Results for the Phi 3.5 Mini 3.8B Instruct model are provided in Appendix G.4.3. The $x$-axis represents the length of the input tokens, while the $y$-axis shows the position depth percentage of the 'needle' information (e.g., 0% indicates the beginning, and 100% indicates the end). A higher score reflects better performance, meaning more effective retrieval of the 'needle' information. GemFilter significantly outperforms both standard attention (full KV cache) and SnapKV.

**Filter Layer.** Except for Section 9.4.3, for context selection, we always use the index of 13 out of 32, 19 out of 40, and 19 out of 32 layers as the input filter for LLaMA

3.1, Mistral Nemo and Phi 3.5, respectively. In Section 9.4.3, we provide an ablation study for the filter layer choice.

### 9.4.1 Needle in a Haystack

The Needle in a Haystack (Kamradt, 2024) benchmark serves as a pressure test, challenging LLMs to retrieve accurate information from a specific sentence (the 'needle') hidden within an extensive document (the 'haystack'), where the sentence can appear at any arbitrary location. The difficulty increases as the length of the haystack grows. We use input lengths of 60K for Mistral Nemo 12B Instruct and 120K for LLaMA 3.1 8B Instruct, as these are the maximum lengths for standard attention on two A100-40GB GPUs. The KV cache size is set to 1024 for both SnapKV and GemFilter. In Figure 9.4, we see that GemFilter significantly outperforms both All KV (standard attention) and SnapKV with Mistral Nemo and LLaMA 3.1.[5] The Needle in a Haystack results suggest that our method, GemFilter, achieves superior retrieval performance for long input contexts compared to SnapKV and standard attention. Additional results are provided in Appendix G.4.3.

### 9.4.2 LongBench

LongBench (Bai et al., 2023) is a multi-task benchmark designed to rigorously evaluate long-context understanding capabilities across various datasets, including single- and multi-document Question Answering (QA), summarization, few-shot learning, and synthetic tasks. We evaluate the English-only dataset, following Li et al. (2024g); Xu et al. (2024c). Note that we do not use a chat template in Table 9.1. See Table G.1 in Appendix G.4.7 for more results of using a chat template.

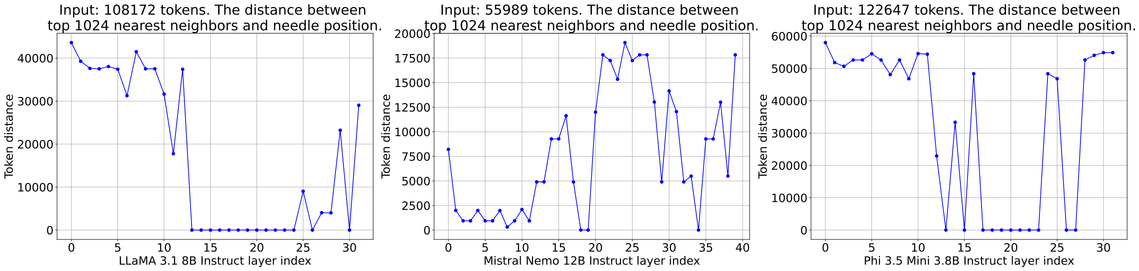
For each LLM, we evaluate GemFilter and SnapKV with selected tokens/KV caches of 1024, 2048, and 4096. We also evaluated standard attention (all KV cache) and H2O with a KV cache size of 4096 on the LongBench dataset to further demon-

[5] H2O cannot be implemented with FlashAttention due to its cumulative attention score strategy and is therefore unable to handle super long input contexts, which is why we exclude it here, following Li et al. (2024g); Xu et al. (2024c).

Table 9.1: Performance comparison on LongBench across various LLMs and methods. A larger number means better performance. The best score is **boldfaced**.

| Method | Single-Document QA | | | Multi-Document QA | | | Summarization | | | Few-shot Learning | | | Synthetic | | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | NrtvQA | Qasper | MF-en | HotpotQA | 2WikiMQA | Musique | GovReport | QMSum | MultiNews | TREC | TriviaQA | SAMSum | PCount | PRe | |
| **LLaMA 3.1 8B Instruct** | | | | | | | | | | | | | | | |
| All KV | 32.02 | 13.04 | 27.34 | 16.23 | 16.05 | 11.22 | 34.52 | 23.41 | **26.89** | **73.0** | 91.64 | 43.8 | 7.16 | 97.73 | **36.72** |
| H2O-4096 | 22.94 | 12.61 | 26.48 | 16.63 | 15.81 | 10.14 | 33.51 | 23.47 | 26.81 | 69.0 | 91.15 | **43.97** | 6.66 | 71.67 | 33.63 |
| MInference | 27.52 | **14.72** | 28.89 | 17.55 | 15.22 | 10.58 | **34.76** | 22.34 | 26.64 | 72.5 | 89.78 | 41.94 | 7.59 | 92.91 | 35.92 |
| LLMLingua-1024 | 11.73 | 6.28 | 12.43 | 13.82 | 12.92 | 8.15 | 22.82 | 20.18 | 23.32 | 24.0 | 66.75 | 24.02 | **9.09** | 4.24 | 18.55 |
| SnapKV-1024 | 31.98 | 11.17 | 25.33 | 14.81 | 15.73 | 10.69 | 26.95 | 22.89 | 25.86 | 67.5 | 91.89 | 42.85 | 7.67 | 98.16 | 35.25 |
| GemFilter-1024 | 20.71 | 11.0 | **29.28** | 19.12 | 17.01 | 13.01 | 30.37 | 21.75 | 25.17 | 63.0 | 90.7 | 42.5 | 7.15 | 92.22 | 34.50 |
| SnapKV-2048 | 31.45 | 11.94 | 26.24 | 15.73 | 16.03 | 11.66 | 29.64 | 23.24 | 26.44 | 69.5 | 91.48 | 42.68 | 7.21 | 98.03 | 35.80 |
| GemFilter-2048 | 24.36 | 12.63 | 25.39 | **19.58** | **17.03** | **14.11** | 33.15 | 22.31 | 26.49 | 69.5 | 91.59 | 42.64 | 4.61 | **98.75** | 35.87 |
| SnapKV-4096 | **32.13** | 13.12 | 27.38 | 16.11 | 16.08 | 11.6 | 32.39 | 23.47 | 26.76 | 71.5 | 91.64 | 43.46 | 7.33 | 97.24 | 36.44 |
| GemFilter-4096 | 25.66 | 12.95 | 27.38 | 17.76 | 15.6 | 12.02 | 34.17 | 23.25 | 26.87 | 70.0 | **92.36** | 43.34 | 5.96 | 98.0 | 36.09 |
| **Mistral Nemo 12B Instruct** | | | | | | | | | | | | | | | |
| All KV | 28.91 | 40.74 | 54.65 | 52.15 | 48.36 | 30.28 | **30.66** | **23.53** | 26.31 | 75.0 | 89.66 | 44.32 | 4.5 | 100.0 | 46.36 |
| H2O-4096 | **31.61** | 39.52 | 54.75 | 47.83 | 48.09 | 27.0 | 30.44 | 23.21 | 26.42 | 72.5 | 89.76 | 44.47 | 3.0 | 73.0 | 43.69 |
| LLMLingua-1024 | 19.24 | 16.92 | 21.43 | 30.94 | 25.09 | 13.24 | 21.96 | 19.8 | 23.94 | 24.5 | 68.48 | 33.33 | 4.0 | 5.0 | 23.42 |
| SnapKV-1024 | 26.42 | 38.49 | 52.96 | 51.21 | 47.86 | 27.06 | 24.32 | 22.66 | 25.52 | 73.0 | 89.82 | 43.16 | 3.5 | 100.0 | 44.71 |
| GemFilter-1024 | 27.53 | 40.68 | 53.86 | 55.51 | **55.43** | 34.11 | 27.25 | 21.16 | 25.56 | 69.0 | 87.32 | 42.49 | 4.0 | 88.06 | 45.14 |
| SnapKV-2048 | 25.85 | 40.69 | 54.48 | 51.96 | 49.06 | 26.95 | 26.29 | 23.17 | 25.9 | 74.5 | 89.66 | 43.89 | 4.0 | 99.5 | 45.42 |
| GemFilter-2048 | 29.27 | **41.53** | **54.91** | 57.62 | 54.97 | **35.09** | 29.34 | 22.58 | 26.19 | 72.0 | 89.65 | **44.93** | 4.0 | 97.5 | **47.11** |
| SnapKV-4096 | 27.92 | 40.9 | 54.75 | 51.69 | 48.16 | 29.19 | 29.17 | 23.36 | 26.35 | 75.0 | 89.66 | 43.93 | 4.5 | 100.0 | 46.04 |
| GemFilter-4096 | 30.29 | 39.9 | 56.48 | **58.78** | 51.48 | 32.81 | 30.32 | 23.21 | **26.48** | 71.5 | **90.24** | 42.13 | 2.0 | 99.5 | 46.79 |
| **Phi 3.5 Mini 3.8B Instruct** | | | | | | | | | | | | | | | |
| All KV | **27.51** | 17.23 | 35.63 | 21.7 | 25.7 | 11.68 | 34.14 | **23.17** | 24.95 | **71.5** | 87.37 | 13.08 | **7.17** | 83.85 | **34.62** |
| H2O-4096 | 19.74 | 16.23 | 34.17 | 21.02 | 23.05 | 10.49 | 33.42 | 21.95 | 24.95 | 67.5 | 86.13 | 16.71 | 1.55 | 47.46 | 30.31 |
| LLMLingua-1024 | 8.58 | 6.74 | 14.93 | 12.37 | 11.01 | 4.48 | 21.23 | 17.08 | 20.75 | 24.0 | 56.09 | 23.01 | 0.96 | 3.79 | 16.07 |
| SnapKV-1024 | 24.31 | 16.03 | 34.93 | 20.72 | 26.02 | 13.74 | 28.27 | 22.03 | 24.02 | 67.5 | **87.71** | 14.57 | 6.08 | **85.6** | 33.68 |
| GemFilter-1024 | 16.57 | 18.29 | 35.91 | 24.22 | 26.1 | 9.7 | 30.29 | 18.96 | 23.64 | 64.5 | 85.85 | **23.02** | 0.2 | 81.12 | 32.74 |
| SnapKV-2048 | 26.41 | 16.59 | **36.99** | 21.8 | 26.07 | 12.57 | 30.88 | 22.37 | 24.51 | 69.5 | 87.54 | 13.13 | 6.57 | 83.92 | 34.20 |
| GemFilter-2048 | 19.63 | 14.84 | 35.99 | 21.38 | 19.72 | 10.13 | 32.39 | 21.24 | 24.71 | 65.0 | 86.49 | 20.47 | 2.17 | 69.5 | 31.69 |
| SnapKV-4096 | 27.25 | 17.42 | 36.9 | 21.37 | 25.42 | 12.55 | 32.9 | 22.6 | 24.87 | 70.5 | 87.45 | 13.28 | 6.81 | 84.04 | 34.53 |
| GemFilter-4096 | 20.95 | **19.98** | 35.22 | **28.82** | **28.21** | **13.98** | **34.2** | 22.45 | **25.08** | 64.5 | 85.86 | 18.68 | 3.43 | 65.56 | 33.35 |

strate the performance of GemFilter, following Li et al. (2024g). Table 9.1 shows a negligible performance drop in LLMs using GemFilter compared to standard attention, even with only 1024 selected tokens. In some cases, GemFilter even outperforms standard attention, such as GemFilter-2048 for Mistral Nemo 12B Instruct. It demonstrates significantly better performance than H2O and comparable performance with SnapKV. Furthermore, GemFilter effectively filters key information in long contexts, provides interpretable summaries, and compresses the input context



(a) LLaMA 3.1 8B Instruct (b) Mistral Nemo 12B Instruct (c) Phi 3.5 Mini 3.8B Instruct

Figure 9.5: Distance between the needle position and selected token index position across three LLMs. The position depth percentage of the "needle" information is 50%. The $x$-axis means the layer index of different LLMs. The $y$-axis means $\min(\text{topk\_index} - \text{niddle\_index})$. When $y = 0$, it means the needle information is covered by the selected token. The needle information has been successfully discovered in the early layers of all three LLMs.

effectively, e.g., it reduces input tokens to an average of 8% when using 1024 tokens, and 32% when using 4096, with negligible accuracy drops.

In the section, we also evaluated on two important baselines, MInference (Jiang et al., 2024a) and LLMLingua (Jiang et al., 2023b)[6]. We can see that MInference (Jiang et al., 2024a) has compatible performance with SnapKV, while it requires offline to determine the best attention pattern, which cannot save the prompt computation phase running time. We can see that although LLMLingua (Jiang et al., 2023b) achieves a good comparison rate, the performance may not be satisfactory.

### 9.4.3 Ablation Study: Filter Layer Choice

In this section, we explore which layer should be chosen as the input filter. First, we aim to determine which layer of the LLM can best identify the position of the needle information. In Figure 9.5, we plot the distance between the needle's position and the selected token index across all layers in the LLM. The results reveal three stages in the prompt computation of LLMs. In the first stage, the initial layers preprocess

[6] We skip LongLLMLingua Jiang et al. (2024b) for a fair comparison, as it requires explicitly separating the input context into text information and questions, while other methods do not require that.

the input context and search for the 'needle'. In the second stage, some early to middle layers identify the needle information. Finally, in the third stage, the LLM prepares to generate the output based on the selected tokens.

Table 9.2: Performance of our method on LongBench using different layers as an input filter. A larger number means better performance. The best score is **boldfaced**.

| Filter layer | Single-Document QA | | | Multi-Document QA | | | Summarization | | | Few-shot Learning | | | Synthetic | | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | NrtvQA | Qasper | MF-en | HotpotQA | 2WikiMQA | Musique | GovReport | QMSum | MultiNews | TREC | TriviaQA | SAMSum | PCount | PRe | |
| | | | | | | | **LLaMA 3.1 8B Instruct (32 layers)** | | | | | | | | |
| layer-1 | 16.32 | 7.38 | 13.86 | 13.9 | 13.21 | 5.22 | 25.61 | 20.09 | 24.51 | 47.0 | 76.59 | 39.78 | 2.55 | 23.01 | 23.50 |
| layer-7 | 16.89 | 6.83 | 13.47 | 13.78 | 12.23 | 9.67 | 26.56 | 19.49 | 24.55 | 58.0 | 84.87 | 41.07 | 6.5 | 50.69 | 27.47 |
| layer-12 | 15.53 | 7.73 | 16.53 | 17.08 | 13.33 | 9.88 | 28.94 | 20.32 | 25.01 | 58.0 | 88.16 | 40.42 | 8.36 | 43.06 | 28.03 |
| layer-13 | 20.71 | 11.0 | **29.28** | 19.12 | 17.01 | 13.01 | **30.37** | 21.75 | 25.17 | **63.0** | **90.7** | **42.5** | 7.15 | 92.22 | **34.50** |
| layer-14 | 21.14 | **13.06** | 25.45 | 20.89 | **17.32** | 12.9 | 29.85 | **22.06** | 24.91 | 62.0 | 89.88 | 42.33 | 6.17 | 92.17 | 34.30 |
| layer-19 | 19.06 | 11.69 | 27.12 | **20.98** | 16.98 | **14.04** | 29.17 | 21.88 | **25.18** | 58.0 | 89.65 | 40.4 | **8.75** | **94.84** | 34.12 |
| layer-25 | **24.74** | 12.33 | 26.18 | 18.56 | 16.3 | 12.54 | 28.66 | 21.75 | 25.14 | 61.5 | 88.78 | 39.47 | 8.67 | 90.59 | 33.94 |
| layer-31 | 20.62 | 9.13 | 17.51 | 19.13 | 13.76 | 10.07 | 28.21 | 21.11 | 25.16 | 58.0 | 88.4 | 42.37 | 8.23 | 58.8 | 30.04 |

We then use the first layer that accurately identifies the needle's position as the input filter. In our experiments, we find that this layer remains consistent across different inputs. As shown in Table 9.2, performance first increases and then decreases as we select the input filter layer from the beginning to the end. The peak performance is observed at the 13th layer, which supports our layer selection strategy. Performance remains robust between layers 13 and 25, providing flexibility in layer selection. Exploring the distinct functions of different layers presents an interesting direction for future research.

### 9.4.4 More Ablation Study

To understand the intuition behind selecting tokens with the most attention specifically from the last query, we study using different rows rather than the last row in the attention matrix for indices selection, as shown in Figure 9.2 in Appendix G.4.4. In Figure G.3, we introduce two methods: (a) selecting middle rows of the attention matrix and (2) selecting rows with the largest $\ell_2$ norm. Both methods fail in the Needle in a Haystack task, verifying that selecting the last query token is essential.

Note that the performance improvement of GemFilter may stem from two factors: (1) the selection of important tokens, and (2) the re-computation of these tokens, which might mitigate issues like “lost-in-the-middle”. To understand whether both factors made contributions, we provide an ablation study to isolate the contribution of each factor in Figure G.4 of Appendix G.4.5. Furthermore, in Appendix G.4.6 Figure G.5, we show the index selection difference between Gemfilter and SnapKV.

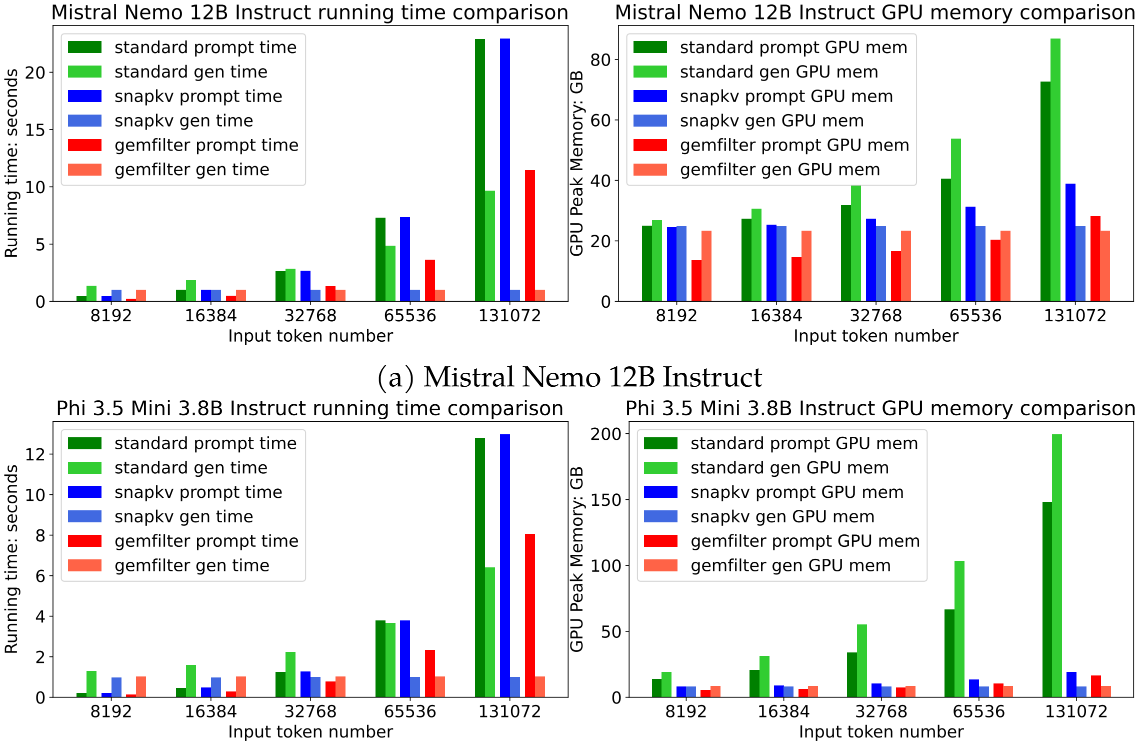
### 9.4.5 Running Time and GPU Memory Consumption

In this section, we compare the running time and GPU memory consumption of different methods with FlashAttention (Dao et al., 2022; Dao, 2023; Shah et al., 2024) support.[7] The iterative generation running time and memory consumption are evaluated on 50 tokens generation. As shown in Figure 9.3, our method, GemFilter, achieves a 2.4× speedup compared to SnapKV and standard attention, with 30% and 70% reductions in GPU memory usage, respectively. It saves both running time and GPU memory by processing the long input context only during the first stage, as described in Section 9.4.3. For the latter two stages, the LLMs only need to handle compressed inputs. In Figure 9.6, we present a comparison of running time and GPU memory consumption for Mistral Nemo 12B Instruct and Phi 3.5 Mini 3.8B Instruct using various methods. GemFilter runs faster and uses less GPU memory than the state-of-the-art methods, as discussed above. Additionally, Figure 9.3 and Figure 9.6 further support our Theorem 9.3 in Section 9.3.2.

## 9.5 Conclusion

In this work, we presented a novel approach, GemFilter, to accelerate LLM inference and reduce memory consumption for long context inputs. By leveraging the ability of early LLM layers to identify relevant information, GemFilter achieves significant improvements over existing techniques. It demonstrates a 2.4× speedup and 30%

[7] We exclude H2O as it does not support FlashAttention and thus requires more GPU memory and running time than standard attention during prompt computation.



(a) Mistral Nemo 12B Instruct

(b) Phi 3.5 Mini 3.8B Instruct

Figure 9.6: Comparison of time and GPU memory usage across different methods on Mistral Nemo 12B Instruct and Phi 3.5 Mini 3.8B Instruct. GemFilter uses the 19th layer as an input filter for both LLMs. It achieves a 2.4× speedup and reduces GPU memory usage by 30% compared to SnapKV.

reduction in GPU memory usage compared to SOTA methods, while also showing superior performance on the Needle in a Haystack benchmark. Our approach is simple, training-free, applicable to various LLMs, and offers enhanced interpretability by directly inspecting selected tokens. These results not only provide practical benefits for LLM deployment, but also provide insight into a better understanding of LLM internal mechanisms.

## 10.1 Thesis Overview

This thesis has explored the fundamental mechanisms of feature learning in neural networks and their practical applications in foundation models. Through theoretical analysis and empirical studies, we have demonstrated that feature learning emerges from input data structures during neural network training and plays a crucial role in foundation models' adaptation to downstream applications.

Our research journey began with a theoretical investigation of feature learning emergence in neural networks. We demonstrated that neural networks can efficiently learn class-relevant patterns in the early stages of training, using a surprisingly small number of parameters. This capability stands in stark contrast to traditional machine learning methods, which require exponentially many fixed features to achieve comparable performance. Our analysis revealed that this efficiency stems from the networks' ability to discover and leverage inherent input data structures, thereby avoiding the curse of dimensionality that plagues conventional approaches.

We further developed a unified analysis framework for two-layer networks trained by gradient descent, centered around the principle of feature learning from gradients. This framework successfully explained several important phenomena, including learning beyond kernel methods and the lottery ticket hypothesis. Our investigation extended to Transformers, where we characterized the Fourier features in one-layer architectures and uncovered intriguing relationships between model scale and in-context learning behavior.

In the realm of practical applications, our work has yielded several significant advances. We introduced nuclear norm regularization for domain generalization, which effectively mitigates the impact of environmental features while promoting the learning of domain-invariant features. Our contrastive regularization method addressed the fundamental trade-off between universality and label efficiency in foundation models. We also developed innovative approaches for Transformer architectures, including looped Transformers for multi-step gradient descent in-context learning and the GemFilter algorithm for efficient processing of long-context inputs in Large Language Models.

## 10.2 Future Research Directions

Looking ahead, our findings point to several promising research directions.

The first critical area involves adaptive feature selection in Retrieval-Augmented Generation (RAG) systems. Current systems often use static feature extraction methods, but our understanding of how feature learning emerges suggests the potential for dynamic approaches. Future research should explore mechanisms for adjusting feature extraction based on query complexity and context, potentially leading to more efficient and accurate retrieval systems. This could involve developing adaptive architectures that can modulate their feature extraction depth and breadth based on the input characteristics.

Cross-modal feature learning represents another frontier for investigation. As foundation models increasingly handle multiple modalities, understanding how features learned in one domain can enhance retrieval in another becomes crucial. Future work should examine the transfer of features across modalities and develop unified representations that can effectively capture cross-modal relationships while maintaining computational efficiency.

Feature compression and efficiency present significant opportunities for advancement. As models grow larger and handle more extensive contexts, developing methods for efficient feature storage and retrieval becomes paramount. Research should focus on lossy compression techniques that preserve retrieval quality while substantially reducing storage and computational requirements. This includes investigating how to identify and retain the most task-relevant features while discarding redundant or less important ones.

Temporal feature learning in continuously updated systems poses another important challenge. As knowledge bases expand and evolve, understanding how to maintain feature relevance over time becomes crucial. Future research should examine mechanisms for feature adaptation and retention in long-running systems, including strategies for balancing the preservation of important historical features with the incorporation of new knowledge.

## 10.3 Concluding Remarks

The exploration of feature learning in neural networks has revealed its fundamental role in the success of modern machine learning systems. Our theoretical insights into how feature learning emerges from data structures, combined with practical applications in foundation models, have advanced our understanding of these systems' capabilities and limitations. As we look toward future developments, the principles of feature learning will continue to guide improvements in efficiency, adaptability, and performance.

The path forward requires careful balance between theoretical understanding and practical implementation. While our work has illuminated many aspects of feature learning, numerous questions remain about optimal feature extraction, cross-modal integration, and temporal adaptation. Addressing these challenges will be crucial for developing the next generation of intelligent systems that can efficiently process, store, and utilize features across diverse applications and modalities.

As the field continues to evolve, the insights gained from studying feature learning will undoubtedly play a vital role in shaping more efficient, adaptable, and capable machine learning systems. The future of artificial intelligence lies not just in larger models, but in better understanding and utilizing the fundamental mechanisms that make learning possible.

Section A.3 presents more technical discussion on related work. Section A.4-A.6 provides the complete proofs for our results in the main text. Section A.7 provides the complete details and experimental results for our experiments.

Finally, Section A.8 provides the theoretical results and complete proofs for a setting more general than that in the main text, allowing incoherent dictionaries, unbalanced classes, and Gaussian noise in the data.

## A.1 Ethics Statement

Our paper is mostly theoretical in nature and thus we foresee no immediate negative ethical impact. We are of the opinion that our theoretical framework may lead to better understanding and inspire development of improved network learning methods, which may have a positive impact in practice. In addition to the theoretical machine learning community, we perceive that our conceptual message that the input structure is crucial for the network learning's performance can be beneficial to engineering-inclined machine learning researchers.

## A.2 Reproducibility Statement

For theoretical results in the Section 2.4, a complete proof is provided in the Appendix A.4-A.6. The theoretical results and complete proofs for a setting more general than that in the main text are provided in the Appendix A.8. For experiments in the Section 2.6, complete details and experimental results are provided in the Appendix Section A.7. The source code with explanations and comments is provided in the supplementary material.

## A.3 More Technical Discussion on Related Work

**Advantage of Neural Networks over Linear Models on Fixed Features.** A recent line of work has turned to show learning settings where network learning provably has advantage over linear models on fixed features; see the nice summary in Malach et al. (2021). Here we highlight the results and focuses of the existing related studies and discuss the differences from ours.

Yehudai and Shamir (2019) shows that the random feature method fails to learn even a single ReLU neuron on Gaussian inputs unless its size is exponentially large in dimension. This points out the limitation of the random feature method (belonging to the fixed feature approach) but does not consider feature learning in networks.

Some studies show that a single ReLU neuron can be learnt by gradient descent (Yehudai and Ohad, 2020; Diakonikolas et al., 2020; Frei et al., 2020). The analysis typically involves feature learning. However, their focus is different: they do not show the advantage over fixed feature methods and do not consider the effect of the input structures.

Zhou et al. (2021b) shows that in a special teacher-student setting, the student network will do exact local convergence in a surprising way that all student neurons will converge to one of the teacher neurons. The work does not consider the effect of the input structure nor the advantage over fixed features.

Dou and Liang (2020) explains the advantage of network learning by constructing adaptive Reproducing Kernel Hilbert Space (RKHS) indexed by the training process of the neural network, and shows that adaptive RKHS benefits from a smaller function space containing the residue comparing to RKHS. The work shows the statistical advantage of networks over data-independent kernels, but does not consider the optimization for learning the network.

Ghorbani et al. (2020) considers data generated from a hidden vector with two subsets of variables, each uniformly distributed in a high-dimensional sphere (with a different radius), while the label is determined by only the first subset of variables. It shows the existence of good neural networks that can overcome the curse of dimensionality by representing the best low-dimensional hidden structure. However, it studies the approximation power of neural networks rather than the learning, i.e., it does not show how to learn the good network.

Fang et al. (2019) argues that in the infinite width limit, a two-layer neural network will learn a nearly optimal feature representation in the distribution sense, thanks to the convexity of the limit problem. It is unclear how this result helps to understand the feature learning procedure for practical networks, which is usually a non-convex process.

Chen et al. (2020a) considers a fixed, randomly initialized neural network as a representation function fed into another trainable network which is the quadratic Taylor model of a wide two-layer network. It shows that learning over the random representation can achieve improved sample complexities compared to learning over the raw data. However, the representation considered is not learned, which is different from our focus on feature learning.

Allen-Zhu and Li (2020a) considers Gaussian inputs with labels given by a multiple-layer network with quadratic activations and skip connections (with the assumption of information gap on the weights), and studies training a deep network with quadratic activation. It shows that the trained network can learn proper representations and obtain small errors while no polynomial fixed feature methods can. On the other hand, it does not focus on the influence of input structure on feature learning: note that its input distribution contains no information about the "ground-truth" features in the target network. It also points out that the learned features get improved during training: higher-level layers will help lower-level layers to improve by backpropagating correction signals. Our analysis also shows feature improvement which however is by signals from the input distribution.

Allen-Zhu and Li (2019) considers PAC learning with labels given by a depth-2 ResNet, and studies training an overparameterized depth-2 ResNet (using uniform inputs over Boolean hypercube as an example). It shows the trained network can obtain small errors while no polynomial kernel methods can obtain as good errors. Similar to Allen-Zhu and Li (2020a), it does not focus on the influence of input structure on feature learning or the advantage of networks.

Allen-Zhu and Li (2020b) studies how ensemble of deep learning models can improve test accuracy and how the ensemble can be distilled into a single model. It develops a theory which assumes the data has multi-view structure and shows that the ensemble of independently trained networks can provably improve test accuracy and the ensemble can also be provably distilled into a single model. The analysis also relies on showing that the data structure can help the ensemble and the distillation. On the other hand, their focus is on ensembles and is quite different from ours: the analysis is on showing the multi-view input structure allows the ensembles of networks to improve over single ones and ensembles of fixed feature mappings do not have improvement. While our focus is on supervisedly learning one single network that outperforms the fixed feature approaches.

Daniely and Malach (2020) considers the task of learning sparse parities with two-layer networks, and the analysis suggests that the ability to learn the label-correlated features also seems to be critical towards the success of neural networks, although the authors did not explore much in this direction. Malach et al. (2021) also considers similar learning problems but with specifically designed models for the problems. The learning problems considered in Daniely and Malach (2020); Malach et al. (2021) have input distributions that leak information about the target labeling function, which is similar to our setting, and their analysis also shows that the first gradient descent can learn a set of good features and later steps can learn an accurate classifier on top. Our work is inspired by their studies, while there are some important differences. First, their focuses are different from ours. Daniely and Malach (2020) focuses on showing neural networks can learn targets (i.e., $k$-parity functions) that are inherently non-linear. Our analysis generalizes to more general distributions, including practically motivated ones. Malach et al. (2021) focuses on strong separations between learning with gradient descent on differentiable models (including typical neural networks) and learning using the corresponding tangent kernels. The analysis is on specific differentiable models, while our work is on two-layer neural networks similar to practical ones. Second, our analysis relies on the feature improvement in the second gradient step. This is not an artifact of the analysis but comes from our problem setup. While in Daniely and Malach (2020) the data distribution allows some neurons to be sufficiently good after the first gradient step and needs no feature improvement, our setup is more general where the data distribution may not have a similar strong benign effect and thus needs feature improvement in the second gradient step.

Most related to our work is Daniely and Malach (2020). Therefore, we provide a detailed discussion to highlight the connections and differences.

1. Our problem setting is *more general* than that in Daniely and Malach (2020). To see this, let our dictionary be the identity matrix, the set $P$ to be the odd numbers (i.e., the labeling function is a sparse parity). Furthermore, let the distribution of the hidden representation be an equal mixture of the following two:

   a) $\mathcal{D}_1$: Uniform distribution over the hypercube.

   b) $\mathcal{D}_2$: Irrelevant patterns $\tilde{\phi}_j (j \not\in A)$ have appearance probability $p_0 = 1/2$. And the distribution of relevant patterns $\tilde{\phi}_j (j \in A)$ is: all 0's with probability $1/2$, and all 1's with probability $1/2$.

   Then our problem setting reduces to their setting (up to scaling/translation of $\tilde{\phi}_j$'s). On the other hand, in general our setting allows for more choices for the labeling, the dictionary, and the distributions over $\tilde{\phi}$.

2. Upper bound: Because of the more general setting, our upper bound proof requires *technical novelty*. Recall that in their work, the input distribution is essentially a mixture of $\mathcal{D}_1$ and $\mathcal{D}_2$ above. In $\mathcal{D}_2$, the relevant patterns $\tilde{\phi}_j (j \in A)$ have the specific structure of all 0's or all 1's with probability $1/2$. This allows to show that neurons with weight $w$ satisfying $\sum_{j \in A} w_j = 0$ will have good gradients: small components from irrelevant patterns (their Lemma 7) and large components from relevant patterns (their Lemma 8). However, in our setting, the relevant patterns do not have this specific structure, and thus their proof technique is not applicable (or can be applied only when we have an exponentially large number of hidden neurons so that some hit the good positions at random initialization). What we showed is that the

gradient has some correlation with the good feature direction. So after the first gradient step, the neuron weights are not good yet but are in a better position for further improvement (in particular, their setting corresponds to $p_0 = 1/2$ which means large noise in the weights after the first step; see discussion after our Lemma 2.6 in Section 2.5). Then the latter gradient steps are able to improve the weights to better "signal-to-noise-ratio". In summary, our proof does not rely on their specific input structure or an exponentially large number of hidden neurons for hitting some good positions. The key is that the good feature will emerge with the help of the input structure, and once in a better position, the neurons' weights can be improved to the desired quality.

3. Lower bound: On the other hand, our lower bound is proved by a reduction to the lower bound results in Daniely and Malach (2020). They have shown that $\mathcal{D}_1$ above can lead to large errors for fixed feature models of polynomial size. Our proof is essentially constructing a mixture of $\mathcal{D}_1$ and $\mathcal{D}_2$ with mixture weights $p_0$ and $(1-p_0)$, and applying their lower bound for $\mathcal{D}_1$. See our proof in Appendix A.5.

4. Conceptually, our work belongs to the same line of research as Daniely and Malach (2020), to analyze how feature learning leads to the superior performance of networks. While their analysis also relies on feature learning from good gradients induced by input structure, their focus is more on separating network learning and fixed feature models and has not explicitly explored the impact of input structures (while we agree that such an explicit study will not be difficult in their setting). More importantly, their input distribution is specific and atypical in practice, which allows a specific type of feature learning (as explained in the above discussion on upper bounds). Our work thus considers a more general setting that is motivated by practical problems. Our results then bring theoretical insights closer for explaining the feature learning in practice and provide some positive evidence for the importance of analysis under proper models of the input distributions.

**Sparse Coding and Subspace Data Models.** To analyze neural networks' performance, various data models have been considered. A practical way to model the underlying structure of data is by assuming that a set of hidden variables exists and the input data is a high dimensional projection of the hidden vector (possibly with noise). Along this line, the classic sparse coding model has been used in existing works for analyzing networks. Koehler and Risteski (2018) considers such a data distribution where the label is given by a linear function on the hidden sparse vector, but studies the approximation power of networks and classic polynomial methods rather than the learning. Allen-Zhu and Li (2022) considers similar data distributions, but studies the performance of networks under adversarial perturbations. Another type of related data models assumes that the label is determined by a subset of hidden variables. Ghorbani et al. (2020) considers a hidden vector with two subsets of variables, each uniformly distributed in a high-dimensional sphere (with a different radius), while the label is determined by only the first subset of variables. However, Ghorbani et al. (2020) studies the approximation power of neural networks rather than the learning. Compared to these studies, our work assumes the input is given by a dictionary multiplied with a hidden vector (not necessarily sparse) while the label is determined by a subset of the hidden vector, as motivated by pattern recognition applications in practice. Furthermore, we focus on the learning ability of networks instead of approximation.

## A.4 Complete Proofs for Provable Guarantees of Neural Networks

We first make a few remarks about the proof.

*Remark.* The analysis can be carried out for more gradient steps following similar intuition, while we analyze two steps for simplicity.

*Remark.* Readers may notice that the network can be overparameterized. With sufficient overparameterization and proper initialization and step sizes, network learning becomes approximately NTK. However, here our learning scheme allows going beyond this kernel regime: we use aggressive gradient updates $\lambda_{\mathbf{w}}^{(t)} = 1/(2\eta^{(t)})$ in the first two steps, completely forgetting the old weights to learn effective features. Using proper initialization and aggressive updates early to escape the kernel regime has been studied in existing work (e.g., Woodworth et al. (2020); Li et al. (2019)). Our result thus adds another concrete example.

**Notations.** For a vector $v$ and an index set I, let $v_I$ denote the vector containing the entries of $v$ indexed by I, and $v_{-I}$ denote the vector containing the entries of $v$ with indices outside I.

By initialization, $\mathbf{w}_i^{(0)}$ for $i \in [m]$ are i.i.d. copies of the same random variable $\mathbf{w}^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 I_{d\times d})$; similar for $a^{(0)}$ and $b^{(0)}$. Let $q_\ell := \langle \mathbf{w}^{(0)}, M_\ell \rangle$, then $\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle = \langle \phi, q \rangle$. Similarly, define $q_{i,\ell}^{(t)} := \langle \mathbf{w}_i^{(t)}, M_\ell \rangle$. Let $\sigma_\phi^2 := p_o(1-p_o)/\tilde{\sigma}^2$ denote the variance of $\phi_\ell$ for $\ell \notin \mathbf{A}$.

We also define the following sets to denote typical initialization. For a fixed $\delta \in (0,1)$, define

$$\mathcal{G}_{\mathbf{w}}(\delta) := \left\{ \mathbf{w} \in \mathbb{R}^d : q_\ell = \langle \mathbf{w}, M_\ell \rangle, \frac{\sigma_{\mathbf{w}}^2 (D-k)}{2} \leqslant \sum_{\ell \notin \mathbf{A}} q_\ell^2 \leqslant \frac{3\sigma_{\mathbf{w}}^2(D-k)}{2}, \right.$$
$$\left. \max_\ell |q_\ell| \leqslant \sigma_{\mathbf{w}} \sqrt{2\log(Dm/\delta)} \right\}, \tag{A.1}$$

$$\mathcal{G}_a(\delta) := \{a \in \mathbb{R} : |a| \leqslant \sigma_a \sqrt{2\log(m/\delta)}\}. \tag{A.2}$$

$$\mathcal{G}_b(\delta) := \{b \in \mathbb{R} : |b| \leqslant \sigma_b \sqrt{2\log(m/\delta)}\}. \tag{A.3}$$

### A.4.1 Existence of A Good Network

we first show that there exists a network that can fit the data distribution.

**Lemma A.1.** *For some* $s, a, b \in \mathbb{R}$ *with* $a, b \geqslant 0$*, define a function* $\delta_{s,a,b} : \mathbb{R} \to \mathbb{R}$ *as*

$$\delta_{s,a,b}(z) = a\text{œ}_r(z - s + b) - 2a\text{œ}_r(z - s) + a\text{œ}_r(z - s - b). \tag{A.4}$$

*where* $\text{œ}_r(z) = \max\{z, 0\}$ *is the ReLU activation function. Then*

$$\delta_{s,a,b}(z) = \begin{cases} 0 & \textit{when } z \leqslant s - b, \\ a(z - s) + ab & \textit{when } s - b \leqslant z \leqslant s, \\ -a(z - s) + ab & \textit{when } s \leqslant z \leqslant s + b, \\ 0 & \textit{when } s + b \leqslant z. \end{cases} \tag{A.5}$$

*That is,* $\delta_{s,a,b}(z)$ *linearly interpolates between* $(s - b, 0), (s, ab), (s + b, 0)$ *when* $z \in [s - b, s + b]$*, and is 0 elsewhere.*

*Proof of Lemma A.1.* This can be simply verified for the four cases of the value of $z$. □

**Lemma A.2** (Restatement of Lemma 2.5)**.** *For any* $\mathcal{D} \in \mathcal{F}_\Xi$*, there exists a network* $g^*(\mathbf{x}) = \sum_{i=1}^n a_i^* \sigma(\langle \mathbf{w}_i^*, \mathbf{x} \rangle + b_i^*)$ *with* $y = g^*(x)$ *for any* $(\mathbf{x}, y) \sim \mathcal{D}$*. Furthermore, the number of neurons* $n = 3(k + 1)$*,* $|a_i^*| \leqslant 32k, 1/(32k) \leqslant |b_i^*| \leqslant 1/2$*,* $\mathbf{w}_i^* = \tilde{\sigma} \sum_{j \in \mathbf{A}} M_j/(4k)$*, and* $|\langle \mathbf{w}_i^*, \mathbf{x} \rangle + b_i^*| \leqslant 1$ *for any* $i \in [n]$ *and* $(\mathbf{x}, y) \sim \mathcal{D}$*.*

*Proof of Lemma 2.5.* Let $\mathbf{w} = \tilde{\sigma} \sum_{j \in \mathbf{A}} M_j$ and let $\mu = \sum_{j \in \mathbf{A}} \mathbb{E}[\tilde{\phi}_j]$. We have

$$\langle \mathbf{w}, \mathbf{x} \rangle = \tilde{\sigma} \sum_{j \in \mathbf{A}} \langle M_j, M\phi \rangle = \tilde{\sigma} \sum_{j \in \mathbf{A}} \phi_j = \sum_{j \in \mathbf{A}} \tilde{\phi}_j - \mu. \tag{A.6}$$

Then by Lemma A.1,

$$\begin{aligned} g_1^*(x) &:= \sum_{p \in P} \delta_{p-\mu,2,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p-\mu,2,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) && \text{(A.7)} \\ &= \sum_{p \in P} \delta_{p,2,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + \mu) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p,2,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + \mu) && \text{(A.8)} \end{aligned}$$

$$
\begin{aligned}
&= \sum_{p \in P} \delta_{p,2,1/2}\left(\sum_{j \in \mathbf{A}} \tilde{\phi}_j\right) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p,2,1/2}\left(\sum_{j \in \mathbf{A}} \tilde{\phi}_j\right) && \text{(A.9)}\\
&= y && \text{(A.10)}
\end{aligned}
$$

for any $(x, y) \sim \mathcal{D}$. Similarly,

$$
\begin{aligned}
g_2^*(x) &:= \sum_{p \in P} \delta_{p-\mu+1/4,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p-\mu+1/4,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) && \text{(A.11)}\\
&= \sum_{p \in P} \delta_{p+1/4,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + \mu) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p+1/4,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + \mu) && \text{(A.12)}\\
&= \sum_{p \in P} \delta_{p+1/4,4,1/2}\left(\sum_{j \in \mathbf{A}} \tilde{\phi}_j\right) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p+1/4,4,1/2}\left(\sum_{j \in \mathbf{A}} \tilde{\phi}_j\right) && \text{(A.13)}\\
&= y && \text{(A.14)}
\end{aligned}
$$

for any $(x, y) \sim \mathcal{D}$. Note that the bias terms in $g_1^*$ and $g_2^*$ have distance at least $1/4$, then at least one of them satisfies that all its bias terms have absolute value $\geqslant 1/8$. Pick that one and denote it as $g(\mathbf{x}) = \sum_{i=1}^{n} a_i \text{œ}_r(\langle \mathbf{w}_i, \mathbf{x} \rangle + b_i)$. By the positive homogeneity of $\text{œ}_r$, we have

$$
g(\mathbf{x}) = \sum_{i=1}^{n} 4k a_i \text{œ}_r(\langle \mathbf{w}_i, \mathbf{x} \rangle/(4k) + b_i/(4k)). \tag{A.15}
$$

Since for any $(\mathbf{x}, y) \sim \mathcal{D}$, $|\langle \mathbf{w}_i, \mathbf{x} \rangle/(4k) + b_i/(4k)| \leqslant 1$, then

$$
g(\mathbf{x}) = \sum_{i=1}^{n} 4k a_i \sigma(\langle \mathbf{w}_i, \mathbf{x} \rangle/(4k) + b_i/(4k)) \tag{A.16}
$$

where $\sigma$ is the truncated ReLU. Now we can set $a_i^* = 4k a_i, \mathbf{w}_i^* = \mathbf{w}_i/(4k), b_i^* = b_i/(4k)$, to get our final $g^*$. □

### A.4.2 Initialization

We first show that with high probability, the initial weights are in typical positions.

**Lemma A.3.** *For any* $\delta \in (0,1)$*, with probability at least* $1-\delta-2\exp\left(-\Theta(D-k)\right)$ *over* $\mathbf{w}^{(0)}$*,*

$$\sigma_{\mathbf{w}}^2(D-k)/2 \leqslant \sum_{\ell \notin \mathbf{A}} q_\ell^2 \leqslant 3\sigma_{\mathbf{w}}^2(D-k)/2,$$

$$\max_\ell |q_\ell| \leqslant \sigma_{\mathbf{w}}\sqrt{2\log(D/\delta)}.$$

*With probability at least* $1-\delta$ *over* $b^{(0)}$*,*

$$|b^{(0)}| \leqslant \sigma_b\sqrt{2\log(1/\delta)}.$$

*With probability at least* $1-\delta$ *over* $a^{(0)}$*,*

$$|a^{(0)}| \leqslant \sigma_a\sqrt{2\log(1/\delta)}.$$

*Proof of Lemma A.3.* From $q \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 I_{d\times d})$, we have:

- With probability $\geqslant 1-\delta/2$, $\max_\ell |q_\ell| \leqslant \sqrt{2\sigma_{\mathbf{w}}^2 \log\frac{D}{\delta}}$, and
- For any subset $S \subseteq [D]$, with probability $\geqslant 1-2\exp\left(-\Theta(|S|)\right)$, $\|q_S\|_2^2 \in \left(\frac{|S|\sigma_{\mathbf{w}}^2}{2}, \frac{3|S|\sigma_{\mathbf{w}}^2}{2}\right)$.

Similar for $b^{(0)}$ and $a^{(0)}$. The lemma then follows. □

**Lemma A.4.** *We have:*

- *With probability* $\geqslant 1-\delta-2m\exp(-\Theta(D-k))$ *over* $\mathbf{w}_i^{(0)}$*'s, for all* $i \in [2m]$*,* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta)$*.*
- *With probability* $\geqslant 1-\delta$ *over* $b_i^{(0)}$*'s, for all* $i \in [2m]$*,* $b_i^{(0)} \in \mathcal{G}_b(\delta)$*.*
- *With probability* $\geqslant 1-\delta$ *over* $a_i^{(0)}$*'s, for all* $i \in [2m]$*,* $a_i^{(0)} \in \mathcal{G}_a(\delta)$*.*

*Proof of Lemma A.4.* This follows from Lemma A.3 by union bound. □

The following lemma about the typical $\mathbf{w}_i^{(0)}$'s will be useful for later analysis.

**Lemma A.5.** *Fix* $\delta \in (0,1)$. *For any* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta)$, *we have*

$$\Pr_{\phi}\left[\sum_{\ell \notin \mathbf{A}} \phi_\ell q_{i,\ell}^{(0)} \geqslant \Theta\left(\sqrt{(D-k)\sigma_\phi^2 \sigma_{\mathbf{w}}^2}\right)\right] = \Theta(1) - \frac{O(\log^{3/2}(Dm/\delta))}{\sqrt{(D-k)\sigma_\phi^2 \tilde{\sigma}^2}}. \tag{A.17}$$

*Consequently, when* $p_o = \Omega(k^2/D)$ *and* $k = \Omega(\log^2(Dm/\delta))$,

$$\Pr_{\phi}\left[\sum_{\ell \notin \mathbf{A}} \phi_\ell q_{i,\ell}^{(0)} \geqslant \Theta\left(\sigma_{\mathbf{w}}\right)\right] = \Theta(1) - \frac{O(1)}{k^{1/4}}. \tag{A.18}$$

*Proof of Lemma A.5.* Note that for $\ell \notin \mathbf{A}$, $\mathbb{E}[\phi_\ell] = 0$, $\mathbb{E}[\phi_\ell^2] = \sigma_\phi^2$, and $\mathbb{E}[|\phi_\ell|^3] = \Theta(\sigma_\phi^2/\tilde{\sigma})$. Then the statement follows from Berry-Esseen Theorem. □

### A.4.3 Some Auxiliary Lemmas

The expression of the gradients will be used frequently.

**Lemma A.6.**

$$\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g;\sigma_\xi) = -a_i \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg(\mathbf{x};\xi) \leqslant 1]\mathbb{E}_{\xi_i}\mathbb{I}[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i + \xi_i \in (0,1)]\mathbf{x}\right\}, \tag{A.19}$$

$$\frac{\partial}{\partial b_i} L_{\mathcal{D}}(g;\sigma_\xi) = -a_i \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg(\mathbf{x};\xi) \leqslant 1]\mathbb{E}_{\xi_i}\mathbb{I}[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i \in (0,1)]\right\}, \tag{A.20}$$

$$\frac{\partial}{\partial a_i} L_{\mathcal{D}}(g;\sigma_\xi) = -\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg(\mathbf{x};\xi) \leqslant 1]\mathbb{E}_{\xi_i}\sigma(\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i + \xi_i)\right\}. \tag{A.21}$$

*Proof of Lemma A.6.* It follows from straightforward calculation. □

We now show that a small subset of the entries in $\phi, q$ does not affect the probability distribution of $\langle \phi, q\rangle$ much.

**Lemma A.7.** *Suppose* $\nu \sim \mathcal{N}(0, \sigma^2)$*. For any* $B \supseteq \mathbf{A}$ *and any* $b$*:*

$$\left| \Pr_{\phi_{-B},\nu} \{\langle \phi, q\rangle + \nu \geqslant b\} - \Pr_{\phi_{-B},\nu} \{\langle \phi_{-B}, q_{-B}\rangle + \nu \geqslant b\} \right| \tag{A.22}$$

$$\leqslant O\left( \frac{|\langle \phi_B, q_B\rangle|}{(\sigma_\phi^2 \|q_{-B}\|_2^2 + \sigma^2)^{1/2}} + \frac{\sigma^3 + \sigma_\phi^2 \|q_{-B}\|_3^3/\tilde{\sigma}}{(\sigma^2 + \sigma_\phi^2 \|q_{-B}\|_2^2)^{3/2}} \right). \tag{A.23}$$

*Similarly,*

$$\left| \Pr_{\phi_{-B}} \{\langle \phi, q\rangle \geqslant b\} - \Pr_{\phi_{-B}} \{\langle \phi_{-B}, q_{-B}\rangle \geqslant b\} \right| \tag{A.24}$$

$$\leqslant O\left( \frac{|\langle \phi_B, q_B\rangle|}{\sigma_\phi \|q_{-B}\|_2} + \frac{\|q_{-B}\|_3^3}{\tilde{\sigma}\sigma_\phi \|q_{-B}\|_2^3} \right). \tag{A.25}$$

*Proof of Lemma A.7.* Note that for $\ell \notin \mathbf{A}$, $\mathbb{E}[\phi_\ell] = 0$, $\mathbb{E}[\phi_\ell^2] = \sigma_\phi^2$, and $\mathbb{E}[|\phi_\ell|^3] = \Theta(\sigma_\phi^2/\tilde{\sigma})$. Let $t = |\langle \phi_B, q_B\rangle|$. Then by the Berry-Esseen Theorem,

$$\left| \Pr_{\phi_{-B}} \{\langle \phi, q\rangle + \nu \geqslant b\} - \Pr_{\phi_{-B}} \{\langle \phi_{-B}, q_{-B}\rangle + \nu \geqslant b\} \right| \tag{A.26}$$

$$\leqslant \Pr_{\phi_{-B}} \{\langle \phi_{-B}, q_{-B}\rangle + \nu \in [-t + b, t + b]\} \tag{A.27}$$

$$\leqslant \frac{2t}{(\sigma_\phi^2 \|q_{-B}\|_2^2 + \sigma^2)^{1/2}} + \frac{O(\sigma^3 + \sigma_\phi^2 \|q_{-B}\|_3^3/\tilde{\sigma})}{(\sigma^2 + \sigma_\phi^2 \|q_{-B}\|_2^2)^{3/2}}. \tag{A.28}$$

The second statement follows from a similar argument. □

We also have the following auxiliary lemma for later calculations.

**Lemma A.8.**

$$\mathbb{E}_{\phi_\mathbf{A}}\{y\} = 0, \tag{A.29}$$

$$\mathbb{E}_{\phi_\mathbf{A}}\{|y|\} = 1, \tag{A.30}$$

$$\mathbb{E}_{\phi_j}\{|\phi_j|\} = 2\sigma_\phi^2 \tilde{\sigma}, \textit{ for } j \notin \mathbf{A}, \tag{A.31}$$

$$\mathbb{E}_{\phi_\mathbf{A}}\{y\phi_j\} = \frac{\gamma}{\tilde{\sigma}}, \tag{A.32}$$

$$\mathbb{E}_{\phi_A}\{|y\phi_j|\} \leqslant \frac{1}{\tilde{\sigma}}, \textit{ for all } j \in [D]. \tag{A.33}$$

*Proof of Lemma A.8.*

$$\mathbb{E}_{\phi_A}\{y\} = \sum_{\nu\in\{\pm1\}} \mathbb{E}_{\phi_A}\{y|y=\nu\}\Pr[y=\nu] \tag{A.34}$$

$$= \frac{1}{2}\sum_{\nu\in\{\pm1\}} \mathbb{E}_{\phi_A}\{y|y=\nu\} \tag{A.35}$$

$$= 0. \tag{A.36}$$

$$\mathbb{E}_{\phi_A}\{|y|\} = \sum_{\nu\in\{\pm1\}} \mathbb{E}_{\phi_A}\{|y|\,|y=\nu\}\Pr[y=\nu] \tag{A.37}$$

$$= \frac{1}{2}\sum_{\nu\in\{\pm1\}} \mathbb{E}_{\phi_A}\{|y|\,|y=\nu\} \tag{A.38}$$

$$= 1. \tag{A.39}$$

$$\mathbb{E}_{\phi_j}\{|\phi_j|\} = \frac{|-p_o|(1-p_o)+|1-p_o|p_o}{\tilde{\sigma}} = 2\sigma_\phi^2\tilde{\sigma}. \tag{A.40}$$

$$\mathbb{E}_{\phi_A}\{y\phi_j\} = \mathbb{E}_{\phi_A}\left\{y\frac{\tilde{\phi}_j - \mathbb{E}[\tilde{\phi}_j]}{\tilde{\sigma}}\right\} \tag{A.41}$$

$$= \frac{1}{\tilde{\sigma}}\mathbb{E}_{\phi_A}\left\{y\tilde{\phi}_j - y\mathbb{E}[\tilde{\phi}_j]\right\} \tag{A.42}$$

$$= \frac{\gamma}{\tilde{\sigma}}. \tag{A.43}$$

$$\mathbb{E}_{\phi_A}\{|y\phi_j|\} = \mathbb{E}_{\phi_A}\{|\phi_j|\} \tag{A.44}$$

$$\leqslant \frac{1}{\tilde{\sigma}}. \tag{A.45}$$

□

### A.4.4 Feature Emergence: First Gradient Step

We will show that w.h.p. over the initialization, after the first gradient step, there are neurons that represent good features.

We begin with analyzing the gradients.

**Lemma A.9** (Full version of Lemma 2.6)**.** *Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta)$ *for all* $i \in [2m]$*. Let*

$$\epsilon_e := \frac{k \log^{1/2}(Dm/\delta) + \log^{3/2}(Dm/\delta)}{\sqrt{\sigma_\phi^2 \tilde{\sigma}^2 (D-k)}}.$$

*If* $p_o = \Omega(k^2/D)$*,* $k = \Omega(\log^2(Dm/\delta))$*, and* $\sigma_\xi^{(1)} < 1/k$*, then*

$$\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g^{(0)}; \sigma_\xi^{(1)}) = -a_i^{(0)} \sum_{j=1}^{D} M_j T_j \tag{A.46}$$

*where* $T_j$ *satisfies:*

- *if* $j \in A$*, then* $|T_j - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_e/\tilde{\sigma})$*, where* $\beta \in [\Omega(1), 1]$ *and depends only on* $\mathbf{w}_i^{(0)}, b_i^{(0)}$*;*
- *if* $j \notin \mathbf{A}$*, then* $|T_j| \leqslant O(\sigma_\phi^2 \epsilon_e \tilde{\sigma})$*.*

*Proof of Lemma A.9.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. Since the unbiased initialization leads to $g^{(0)}(\mathbf{x}; \xi^{(1)}) = 0$, we have

$$\frac{\partial}{\partial \mathbf{w}} L_{\mathcal{D}}(g^{(0)}; \sigma_\xi^{(1)}) \tag{A.47}$$

$$= -a^{(0)} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left\{ y \mathbb{I}[y g^{(0)}(\mathbf{x}; \xi^{(1)}) \leqslant 1] \mathbb{E}_{\xi^{(1)}} \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \mathbf{x} \right\} \tag{A.48}$$

$$= -a^{(0)} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}} \left\{ y \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \mathbf{x} \right\} \tag{A.49}$$

$$= -a^{(0)} \sum_{j=1}^{D} M_j \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}} \left\{ y \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \phi_j \right\}}_{:=T_j}. \tag{A.50}$$

First, consider $j \in \mathbf{A}$.

$$T_j = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}} \left\{ y \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \phi_j \right\} \tag{A.51}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}}, \xi^{(1)}} \left\{ y \phi_j \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \in (0, 1) \right] \right\}. \tag{A.52}$$

Let

$$I_a := \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \in (0, 1) \right], \tag{A.53}$$

$$I'_a := \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \in (0, 1) \right]. \tag{A.54}$$

We have

$$|\mathbb{E}_{\xi^{(1)}}(I_a - I'_a)| \tag{A.55}$$

$$\leqslant \mathbb{E}_{\xi^{(1)}} \left| \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \geqslant 0 \right] - \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \geqslant 0 \right] \right| \tag{A.56}$$

$$+ \Pr_{\phi_{-\mathbf{A}}, \xi^{(1)}} \left[ \langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \geqslant 1 \right] + \Pr_{\phi_{-\mathbf{A}}, \xi^{(1)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \geqslant 1 \right]. \tag{A.57}$$

Then by Lemma A.7,

$$\left| \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \geqslant 0 \right] - \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \geqslant 0 \right] \right| = O(\epsilon_e). \tag{A.58}$$

Note that $\sum_{\ell \notin \mathbf{A}} \mathrm{Var}(\phi_\ell q_\ell) = \Theta(\sigma_\phi^2 \sigma_{\mathbf{w}}^2 (D - k)) = \Theta(\sigma_{\mathbf{w}}^2)$, and $|\phi_\ell| \leqslant \frac{1}{\tilde{\sigma}}$, $\max_\ell |q_\ell| \leqslant \sigma_{\mathbf{w}} \sqrt{2 \log(Dm/\delta)}$. Applying Bernstein's inequality for bounded distributions, we have:

$$\Pr_{\phi_{-\mathbf{A}}} [\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle \geqslant 1/4] = \exp(-\Omega(k)) = O(\epsilon_e). \tag{A.59}$$

We also have:

$$\Pr_{\xi^{(1)}} \left[ b^{(0)} + \xi^{(1)} \geqslant 1/4 \right] = \exp(-\Omega(k)) = O(\epsilon_e). \tag{A.60}$$

Therefore,

$$\Pr_{\phi_{-\mathbf{A}}, \xi^{(1)}} \left[ \langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \geqslant 1 \right] = \exp(-\Omega(k)) = O(\epsilon_e) \tag{A.61}$$

where the last step follows from the assumption on $\sigma_{\mathbf{w}}$ and $k$. A similar argument gives:

$$\Pr_{\phi_{-\mathbf{A}}, \xi^{(1)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \geqslant 1 \right] = \exp(-\Omega(k)) = O(\epsilon_e). \tag{A.62}$$

Then we have

$$\begin{aligned} & \left| T_j - \mathbb{E}_{\phi_{\mathbf{A}}, \xi^{(1)}} \{ y \phi_j I'_a \} \right| & \text{(A.63)} \\ \leqslant\ & \mathbb{E}_{\phi_{\mathbf{A}}} \left\{ |y \phi_j| \left| \mathbb{E}_{\xi^{(1)}} (I_a - I'_a) \right| \right\} & \text{(A.64)} \\ \leqslant\ & O(\epsilon_e) \mathbb{E}_{\phi_{\mathbf{A}}} \{ |y \phi_j| \} & \text{(A.65)} \\ \leqslant\ & O(\epsilon_e / \tilde{\sigma}) & \text{(A.66)} \end{aligned}$$

where the last step is from Lemma A.8. Furthermore,

$$\begin{aligned} & \mathbb{E}_{\phi_{\mathbf{A}}, \xi^{(1)}} \{ y \phi_j I'_a \} & \text{(A.67)} \\ =\ & \mathbb{E}_{\phi_{\mathbf{A}}} \{ y \phi_j \} \mathbb{E}_{\xi^{(1)}} [I'_a] & \text{(A.68)} \\ =\ & \mathbb{E}_{\phi_{\mathbf{A}}} \{ y \phi_j \} \Pr_{\phi_{-\mathbf{A}}, \xi^{(1)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \in (0, 1] \right] & \text{(A.69)} \end{aligned}$$

By Lemma A.5, the assumption on $p_o$, and (A.59), we have

$$\begin{aligned} & \Pr_{\phi_{-\mathbf{A}}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} \in (0, 1/2) \right] \geqslant \Omega(1) - O(1/k^{1/4}), & \text{(A.70)} \\ & \Pr_{\xi^{(1)}} \left[ \xi^{(1)} \in (0, 1/2) \right] = 1/2 - \exp(-\Omega(k)), & \text{(A.71)} \end{aligned}$$

This leads to

$$\beta := \mathbb{E}_{\xi^{(1)}} [I'_a] = \Pr_{\phi_{-\mathbf{A}}, \xi^{(1)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \in (0, 1] \right] \geqslant \Omega(1). \tag{A.72}$$

By Lemma A.8, $\mathbb{E}_{\phi_{\mathbf{A}}}\{y\phi_j\} = \gamma/\tilde{\sigma}$. Therefore,

$$|T_j - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_e/\tilde{\sigma}). \tag{A.73}$$

Now, consider $j \notin \mathbf{A}$. Let B denote $\mathbf{A} \cup \{j\}$.

$$\begin{aligned} T_j &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{y\phi_j\mathbb{I}\left[\langle\phi, q\rangle + b^{(0)} + \xi^{(1)} \in (0,1)\right]\right\} && \text{(A.74)}\\ &= \mathbb{E}_{\phi_B}\mathbb{E}_{\phi_{-B},\xi^{(1)}}\left\{y\phi_j\mathbb{I}\left[\langle\phi, q\rangle + b^{(0)} + \xi^{(1)} \in (0,1)\right]\right\} && \text{(A.75)}\\ &= \mathbb{E}_{\phi_B,\xi^{(1)}}\left\{y\phi_j \Pr_{\phi_{-B}}\left[\langle\phi, q\rangle + b^{(0)} + \xi^{(1)} \in (0,1)\right]\right\}. && \text{(A.76)} \end{aligned}$$

Let

$$\begin{aligned} I_b &:= \Pr_{\phi_{-B}}\left[\langle\phi, q\rangle + b^{(0)} + \xi^{(1)} \in (0,1)\right], && \text{(A.77)}\\ I_b' &:= \Pr_{\phi_{-B}}\left[\langle\phi_{-B}, q_{-B}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)\right]. && \text{(A.78)} \end{aligned}$$

Similar as above, we have $|\mathbb{E}_{\xi^{(1)}}(I_b - I_b')| \leqslant O(\epsilon_e)$ by Lemma A.7. Then by Lemma A.8,

$$\begin{aligned} &\left|T_j - \mathbb{E}_{\phi_B,\xi^{(1)}}\{y\phi_j I_b'\}\right| && \text{(A.79)}\\ &\leqslant \mathbb{E}_{\phi_B}\left\{|y\phi_j||\mathbb{E}_{\xi^{(1)}}(I_b - I_b')|\right\} && \text{(A.80)}\\ &\leqslant O(\epsilon_e)\mathbb{E}_{\phi_{\mathbf{A}}}\{|y|\}\mathbb{E}_{\phi_j}\{|\phi_j|\} && \text{(A.81)}\\ &\leqslant O(\epsilon_e) \times O(\sigma_\phi^2\tilde{\sigma}) && \text{(A.82)}\\ &= O(\sigma_\phi^2\epsilon_e\tilde{\sigma}). && \text{(A.83)} \end{aligned}$$

Furthermore,

$$\mathbb{E}_{\phi_B,\xi^{(1)}}\{y\phi_j I_b'\} = \mathbb{E}_{\phi_{\mathbf{A}}}\{y\}\mathbb{E}_{\phi_j}\{\phi_j\}\mathbb{E}_{\xi^{(1)}}[I_b'] = 0. \tag{A.84}$$

Therefore,

$$|T_j| \leqslant O(\sigma_\phi^2\epsilon_e\tilde{\sigma}). \tag{A.85}$$

□

**Lemma A.10.** *Under the same assumptions as in Lemma A.9,*

$$\frac{\partial}{\partial b_i} L_{\mathcal{D}}(g^{(0)}; \sigma_{\xi}^{(1)}) = -a_i^{(0)} T_b \tag{A.86}$$

*where* $|T_b| \leqslant O(\epsilon_e)$.

*Proof of Lemma A.10.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. Since the unbiased initialization leads to $g^{(0)}(\mathbf{x}; \xi^{(1)}) = 0$, we have

$$\begin{aligned}
&\frac{\partial}{\partial b} L_{\mathcal{D}}(g^{(0)}; \sigma_{\xi}^{(1)}) && \text{(A.87)}\\
&= -a^{(0)} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left\{ y \mathbb{I}[y g^{(0)}(\mathbf{x}; \xi) \leqslant 1] \mathbb{E}_{\xi^{(1)}} \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \right\} && \text{(A.88)}\\
&= -a^{(0)} \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}} \left\{ y \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \right\}}_{:=T_b}. && \text{(A.89)}
\end{aligned}$$

Similar to the proof in Lemma 2.6,

$$\left| \Pr_{\phi_{-\mathbf{A}}} [\langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] - \Pr_{\phi_{-\mathbf{A}}} [\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \right| = O(\epsilon_e). \tag{A.90}$$

Then

$$\begin{aligned}
&\left| T_b - \mathbb{E}_{\phi_{\mathbf{A}},\xi^{(1)}} \left\{ y \Pr_{\phi_{-\mathbf{A}}} [\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \right\} \right| && \text{(A.91)}\\
&= \mathbb{E}_{\phi_{\mathbf{A}},\xi^{(1)}} \left\{ |y| \left| \Pr_{\phi_{-\mathbf{A}}} [\langle \phi, q \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] - \Pr_{\phi_{-\mathbf{A}}} [\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}} \rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \right| \right\} && \text{(A.92)}\\
&\leqslant O(\epsilon_e) \mathbb{E}_{\phi_{\mathbf{A}}} \{|y|\} && \text{(A.93)}\\
&\leqslant O(\epsilon_e). && \text{(A.94)}
\end{aligned}$$

Also,

$$\mathbb{E}_{\phi_{\mathbf{A}},\xi^{(1)}}\left\{y \Pr_{\phi_{-\mathbf{A}}}[\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} \tag{A.95}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}}}\{y\} \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)] \tag{A.96}$$

$$= 0. \tag{A.97}$$

Therefore, $|T_b| \leqslant O(\epsilon_e)$. □

**Lemma A.11.** *We have*

$$\frac{\partial}{\partial a_i} L_{\mathcal{D}}(g^{(0)}; \sigma_{\xi}^{(1)}) = -T_a \tag{A.98}$$

*where* $|T_a| \leqslant O(\max_\ell q_{i,\ell}^{(0)})$. *So if* $w_i^{(0)} \in \mathcal{G}(\delta)$, $|T_a| \leqslant O(\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)})$.

*Proof of Lemma A.11.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. Since the unbiased initialization leads to $g^{(0)}(\mathbf{x}; \xi^{(1)}) = 0$, we have

$$\frac{\partial}{\partial a} L_{\mathcal{D}}(g^{(0)}; \sigma_{\xi}^{(1)}) \tag{A.99}$$

$$= -\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg^{(0)}(\mathbf{x};\xi^{(1)}) \leqslant 1]\mathbb{E}_{\xi^{(1)}}\sigma(\langle\mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})\right\} \tag{A.100}$$

$$= -\underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{y\sigma(\langle\mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})\right\}}_{:=T_a}. \tag{A.101}$$

Let $\phi'_{\mathbf{A}}$ be an independent copy of $\phi_{\mathbf{A}}$, $\phi'$ be the vector obtained by replacing in $\phi$ the entries $\phi_{\mathbf{A}}$ with $\phi'_{\mathbf{A}}$, and let $x' = M\phi'$ and its label is $y'$. Then

$$|T_a| = \left|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{y\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(1)}}\sigma(\langle\mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})\right\}\right| \tag{A.102}$$

$$\leqslant \frac{1}{2}\left|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(1)}}\sigma(\langle\mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})|y = 1\right\}\right. \tag{A.103}$$

$$\left. - \mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(1)}}\sigma(\langle\mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})|y = -1\right\}\right| \tag{A.104}$$

$$\leqslant \frac{1}{2}\Bigg|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})|y = 1\right\} \tag{A.105}$$

$$- \mathbb{E}_{\phi'_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}'\rangle + b^{(0)} + \xi^{(1)})|y' = -1\right\}\Bigg|. \tag{A.106}$$

Since σ is 1-Lipschitz,

$$\begin{aligned}
|T_a| &\leqslant \frac{1}{2}\mathbb{E}_{\phi_{\mathbf{A}},\phi'_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}}}\left|\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle - \langle \mathbf{w}^{(0)}, \mathbf{x}'\rangle\right| |y = 1, y' = -1\right\} && \text{(A.107)}\\
&\leqslant \frac{1}{2}\mathbb{E}_{\phi_{-\mathbf{A}}}\left(\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\left|\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle\right| |y = 1\right\} + \mathbb{E}_{\phi'_{\mathbf{A}}}\left\{\left|\langle \mathbf{w}^{(0)}, \mathbf{x}'\rangle\right| |y' = -1\right\}\right) && \text{(A.108)}\\
&= \mathbb{E}_{\phi_{-\mathbf{A}},\phi_{\mathbf{A}}}\left|\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle\right| && \text{(A.109)}\\
&= \mathbb{E}_{\mathbf{x}}\left|\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle\right| && \text{(A.110)}\\
&\leqslant \sqrt{\mathbb{E}_{\mathbf{x}}\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle^2} && \text{(A.111)}\\
&\leqslant \max_{\ell} q_{i,\ell}^{(0)}\sqrt{\mathbb{E}_{\mathbf{x}}\left(\sum_{\ell\in[D]}\phi_\ell^2 + \sum_{j\neq\ell: j,\ell\in\mathbf{A}}|\phi_j\phi_\ell|\right)} && \text{(A.112)}\\
&\leqslant \max_{\ell} q_{i,\ell}^{(0)}\sqrt{\mathbb{E}_{\mathbf{x}}\left(1 + O(1)\right)} && \text{(A.113)}\\
&= \Theta(\max_{\ell} q_{i,\ell}^{(0)}). && \text{(A.114)}
\end{aligned}$$

□

With the bounds on the gradient, we now summarize the results for the weights after the first gradient step.

**Lemma A.12.** *Set*

$$\lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \lambda_a^{(1)} = \lambda_b^{(1)} = 0, \sigma_\xi^{(1)} = 1/k^{3/2}.$$

*Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta)$ *for all* $i \in [2m]$. *If* $p_o = \Omega(k^2/D)$, $k = \Omega(\log^2(Dm/\delta))$, *then for all* $i \in [m]$, $\mathbf{w}_i^{(1)} = \sum_{\ell=1}^D q_{i,\ell}^{(1)} M_\ell$ *satisfying*

- *if* $\ell \in A$*, then* $|q_{i,\ell}^{(1)} - \eta^{(1)} a_i^{(0)} \beta\gamma/\tilde{\sigma}| \leqslant O\left(\frac{|\eta^{(1)} a_i^{(0)}|\epsilon_e}{\tilde{\sigma}}\right)$*, where* $\beta \in [\Omega(1), 1]$ *and depends only on* $\mathbf{w}_i^{(0)}, b_i^{(0)}$*;*
- *if* $\ell \notin \mathbf{A}$*, then* $|q_{i,\ell}^{(1)}| \leqslant O\left(\sigma_\phi^2 |\eta^{(1)} a_i^{(0)}|\epsilon_e \tilde{\sigma}\right)$*;*

*and*

- $b_i^{(1)} = b_i^{(0)} + \eta^{(1)} a_i^{(0)} T_b$ *where* $|T_b| = O(\epsilon_e)$*;*
- $a_i^{(1)} = a_i^{(0)} + \eta^{(1)} T_a$ *where* $|T_a| = O(\sigma_{\mathbf{w}} \sqrt{\log(Dm/\delta)})$.

*Proof of Lemma A.12.* This follows from Lemma A.4 and Lemma A.9-A.11. □

### A.4.5 Feature Improvement: Second Gradient Step

We first show that with properly set $\eta^{(1)}$, for most $\mathbf{x}$, $|g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})| < 1$ and thus $yg^{(1)}(\mathbf{x}; \sigma_\xi^{(2)}) < 1$.

**Lemma A.13.** *Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta), a_i^{(0)} \in \mathcal{G}_a(\delta)$ *for all* $i \in [2m]$*. If* $p_o = \Omega(k^2/D)$*,* $k = \Omega(\log^2(Dm/\delta))$*,* $\sigma_a \leqslant \tilde{\sigma}^2/(\gamma k^2)$*,* $\eta^{(1)} = O\left(\frac{\gamma}{km\sigma_a}\right)$*, and* $\sigma_\xi^{(2)} \leqslant 1/k$*, then with probability* $\geqslant 1 - \exp(-\Theta(k))$ *over* $(\mathbf{x}, y)$*, we have* $yg^{(1)}(\mathbf{x}; \sigma_\xi^{(2)}) < 1$*. Furthermore, for any* $i \in [2m]$*,* $\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle\right| = \left|\langle q_i^{(1)}, \phi\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$*,* $\left|\langle (q_i^{(1)})_{-\mathbf{A}}, \phi_{-\mathbf{A}}\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$*, and* $\left|b_i^{(1)} - b_{m+i}^{(1)}\right| = O(|\eta^{(1)} a_i^{(0)}|\epsilon_e)$*.*

*Proof of Lemma A.13.* Note that $\mathbf{w}_i^{(0)} = \mathbf{w}_{m+i}^{(0)}$, $b_i^{(0)} = b_{m+i}^{(0)}$, and $a_i^{(0)} = -a_{m+i}^{(0)}$. Then the gradient for $\mathbf{w}_i$ is the negation of that for $\mathbf{w}_{m+i}$, the gradient for $b_i$ is the negation of that for $b_{m+i}$, and the gradient for $a_i$ is the same as that for $a_{m+i}$. With probability $\geqslant 1 - \exp(-\Theta(\max\{2p_o(D-k), k\}))$, among all $j \notin \mathbf{A}$, we have that at most $2p_o(D-k) + k$ of $\phi_j$ are $(1-p_o)/\tilde{\sigma}$, while the others are $-p_o/\tilde{\sigma}$. For data points with $\phi$ satisfying this, we have:

$$\left|g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})\right| \tag{A.115}$$

$$= \left| \sum_{i=1}^{2m} a_i^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} + \xi_i^{(2)}) \right| \tag{A.116}$$

$$= \left| \sum_{i=1}^{m} \left( a_i^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} + \xi_i^{(2)}) + a_{m+i}^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_{m+i}^{(1)}, \mathbf{x} \rangle + b_{m+i}^{(1)} + \xi_{m+i}^{(2)}) \right) \right| \tag{A.117}$$

$$\leqslant \left| \sum_{i=1}^{m} \left( a_i^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} + \xi_i^{(2)}) + a_{m+i}^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} + \xi_i^{(2)}) \right) \right| \tag{A.118}$$

$$+ \left| \sum_{i=1}^{m} \left( -a_{m+i}^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} + \xi_i^{(2)}) + a_{m+i}^{(1)} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_{m+i}^{(1)}, \mathbf{x} \rangle + b_{m+i}^{(1)} + \xi_i^{(2)}) \right) \right|. \tag{A.119}$$

Then we have

$$\begin{aligned}
\left| g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)}) \right| &\leqslant \sum_{i=1}^{m} \left| 2\eta^{(1)} T_a \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} + \xi_i^{(2)}) \right| \qquad &\text{(A.120)} \\
&\quad + \sum_{i=1}^{m} \left| a_{m+i}^{(1)} \right| \left( \left| \langle \mathbf{w}_i^{(1)} - \mathbf{w}_{m+i}^{(1)}, \mathbf{x} \rangle \right| + \left| b_i^{(1)} - b_{m+i}^{(1)} \right| \right) &\text{(A.121)} \\
&\leqslant \sum_{i=1}^{m} \left| 2\eta^{(1)} T_a \right| \left( \left| \langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle + b_i^{(1)} \right| + \mathbb{E}_{\xi^{(2)}} \left| \xi_i^{(2)} \right| \right) &\text{(A.122)} \\
&\quad + \sum_{i=1}^{m} \left| a_{m+i}^{(1)} \right| \left( \left| \langle \mathbf{w}_i^{(1)} - \mathbf{w}_{m+i}^{(1)}, \mathbf{x} \rangle \right| + \left| b_i^{(1)} - b_{m+i}^{(1)} \right| \right). &\text{(A.123)}
\end{aligned}$$

We have $|T_a| = O(\sigma_{\mathbf{w}} \sqrt{\log(Dm/\delta)})$, and

$$\begin{aligned}
\left| \langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle \right| &\leqslant O(|\eta^{(1)} a_i^{(0)}|) \left( \beta\gamma/\tilde{\sigma} + \epsilon_e/\tilde{\sigma} \right) \frac{k}{\tilde{\sigma}} &\text{(A.124)} \\
&\quad + O(|\eta^{(1)} a_i^{(0)}| \sigma_\phi^2 \epsilon_e \tilde{\sigma}) \left( (2p_o(D-k)+k)(1-p_o)/\tilde{\sigma} + p_o D/\tilde{\sigma} \right) \qquad &\text{(A.125)} \\
&\leqslant O(|\eta^{(1)} a_i^{(0)}|) \left( k\gamma/\tilde{\sigma}^2 + \epsilon_e k/\tilde{\sigma}^2 + (k + p_o D)\sigma_\phi^2 \epsilon_e \right) &\text{(A.126)} \\
&\leqslant O(\eta^{(1)}(1 + p_o \tilde{\sigma})/\gamma). &\text{(A.127)}
\end{aligned}$$

$$\left|b_i^{(1)}\right| \leqslant \left|b_i^{(0)}\right| + \left|\eta^{(1)} a_i^{(0)} T_b\right| \tag{A.128}$$

$$\leqslant \frac{\sqrt{\log(m/\delta)}}{k^2} + \left|\eta^{(1)} a_i^{(0)} \frac{\epsilon_e}{\tilde{\sigma}}\right|. \tag{A.129}$$

$$\mathbb{E}_{\xi^{(2)}} \left|\xi_i^{(2)}\right| \leqslant O(\sigma_\xi^{(2)}). \tag{A.130}$$

$$|a_{m+i}^{(1)}| \leqslant |a_i^{(0)}| + |\eta^{(1)} T_a| \leqslant |a_i^{(0)}| + O(\eta^{(1)} \sigma_{\mathbf{w}} \sqrt{\log(Dm/\delta)}). \tag{A.131}$$

$$\left|\langle \mathbf{w}_i^{(1)} - \mathbf{w}_{m+i}^{(1)}, \mathbf{x}\rangle\right| = 2\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle\right| = O(\eta^{(1)}(1 + p_o\tilde{\sigma})/\gamma). \tag{A.132}$$

$$\left|b_i^{(1)} - b_{m+i}^{(1)}\right| = 2|\eta^{(1)} a_i^{(0)} T_b| = O(|\eta^{(1)} a_i^{(0)}|\epsilon_e). \tag{A.133}$$

Then we have

$$\left|g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})\right| \tag{A.134}$$

$$\leqslant O\left(m\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}\right)\left(\frac{\eta^{(1)}}{\gamma} + \frac{\sqrt{\log(m/\delta)}}{k^2} + \left|\eta^{(1)} a_i^{(0)} \frac{\epsilon_e}{\tilde{\sigma}}\right| + \sigma_\xi^{(2)}\right) \tag{A.135}$$

$$+ O\left(m(|a_i^{(0)}| + \eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)})\right)\left(\frac{\eta^{(1)}}{\gamma} + \left|\eta^{(1)} a_i^{(0)} \frac{\epsilon_e}{\tilde{\sigma}}\right|\right) \tag{A.136}$$

$$= O\left(m\eta^{(1)}\sigma_{\mathbf{w}}\frac{\log(Dm/\delta)}{k} + m|a_i^{(0)}|\left(\frac{\eta^{(1)}}{\gamma} + \left|\eta^{(1)} a_i^{(0)} \frac{\epsilon_e}{\tilde{\sigma}}\right|\right)\right) \tag{A.137}$$

$$= O\left(m\eta^{(1)}\sigma_{\mathbf{w}}\frac{\log(Dm/\delta)}{k} + m|a_i^{(0)}|\frac{\eta^{(1)}}{\gamma} + m\sigma_a\eta^{(1)}\frac{k}{\gamma}\right) \tag{A.138}$$

$$< 1. \tag{A.139}$$

Then $\left|yg^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})\right| < 1$. Finally, the statement on $\left|\langle (q_i^{(1)})_{-\mathbf{A}}, \phi_{-\mathbf{A}}\rangle\right|$ follows from a similar calculation on $\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle\right| = \left|\langle q_i^{(1)}, \phi\rangle\right|$. □

We are now ready to analyze the gradients in the second gradient step.

**Lemma A.14.** *Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta), a_i^{(0)} \in \mathcal{G}_a(\delta)$ *for* *all* $i \in [2m]$*. Let* $\epsilon_{e2} := O\left(\frac{\eta^{(1)}|a_i^{(0)}|k(\gamma+\epsilon_e)}{\tilde{\sigma}^2\sigma_\xi^{(2)}}\right) + \exp(-\Theta(k))$*. If* $k = \Omega(\log^2(Dm/\delta))$ *and* $k = O(D)$*,* $\sigma_a \leqslant \tilde{\sigma}^2/(\gamma k^2)$*,* $\eta^{(1)} = O\left(\frac{\gamma}{km\sigma_a}\right)$*, and* $\sigma_\xi^{(2)} = 1/k^{3/2}$*, then*

$$\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}(g^{(1)}; \sigma_\xi^{(2)}) = -a_i^{(1)} \sum_{j=1}^{D} M_j T_j \tag{A.140}$$

*where* $T_j$ *satisfies:*

- *if* $j \in A$*, then* $|T_j - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_{e2}/\tilde{\sigma} + \eta^{(1)}/\sigma_\xi^{(2)} + \eta^{(1)}|a_i^{(0)}|\epsilon_e/(\tilde{\sigma}\sigma_\xi^{(2)}))$*, where* $\beta \in [\Omega(1), 1]$ *and depends only on* $\mathbf{w}_i^{(0)}, b_i^{(0)}$*;*
- *if* $j \notin \mathbf{A}$*, then* $|T_j| \leqslant \frac{1}{\tilde{\sigma}}\exp(-\Theta(k)) + O(\sigma_\phi^2 \epsilon_{e2}\tilde{\sigma})$*.*

*Proof of Lemma A.14.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. By Lemma A.13, $\Pr[yg^{(1)}(\mathbf{x}; \xi^{(2)}) > 1] \leqslant \exp(-\Theta(k))$. Let $\mathbb{I}_{\mathbf{x}} = \mathbb{I}[yg^{(1)}(\mathbf{x}; \xi^{(2)}) \leqslant 1]$.

$$\frac{\partial}{\partial \mathbf{w}} L_{\mathcal{D}}(g^{(1)}; \sigma_\xi^{(2)}) \tag{A.141}$$

$$= -a^{(1)} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left\{ y \mathbb{I}_{\mathbf{x}} \mathbb{E}_{\xi^{(2)}} \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\mathbf{x} \right\} \tag{A.142}$$

$$= -a^{(1)} \sum_{j=1}^{D} M_j \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y \mathbb{I}_{\mathbf{x}} \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j \right\}}_{:=T_j}. \tag{A.143}$$

Let $T_{j1} := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j \right\}$. We have

$$|T_j - T_{j1}| \tag{A.144}$$

$$= \left| \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y(1-\mathbb{I}_{\mathbf{x}}) \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j \right\} \right| \tag{A.145}$$

$$\leqslant \frac{1}{\tilde{\sigma}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} |1 - \mathbb{I}_{\mathbf{x}}| \tag{A.146}$$

$$\leqslant \frac{1}{\tilde{\sigma}} \exp(-\Theta(k)). \tag{A.147}$$

So it is sufficient to bound $T_{j1}$. For simplicity, we use $q$ as a shorthand for $q_i^{(1)}$.

First, consider $j \in \mathbf{A}$.

$$T_{j1} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j \right\} \tag{A.148}$$

$$= \mathbb{E}_{\phi_\mathbf{A}} \left\{ y\phi_j \Pr_{\phi_{-\mathbf{A}},\xi^{(2)}} \left[ \langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1) \right] \right\}. \tag{A.149}$$

Let

$$I_a := \Pr_{\xi^{(2)}} \left[ \langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1) \right], \tag{A.150}$$

$$I'_a := \Pr_{\xi^{(2)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(1)} + \xi^{(2)} \in (0,1) \right]. \tag{A.151}$$

By the property of the Gaussian $\xi^{(2)}$, that $|\langle \phi_\mathbf{A}, q_\mathbf{A}\rangle| = O(\frac{\eta^{(1)}|a_i^{(0)}|k(\gamma+\epsilon_e)}{\tilde{\sigma}^2})$, and that $|\langle \phi, q\rangle| = |\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle| = O(\eta^{(1)}/\gamma) < O(1/k)$ and $|\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle| = O(\eta^{(1)}/\gamma) < O(1/k)$, we have

$$|I_a - I'_a| \leqslant \left| \Pr_{\xi^{(2)}} \left[ \langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \geqslant 0 \right] - \Pr_{\xi^{(2)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(1)} + \xi^{(2)} \geqslant 0 \right] \right| \tag{A.152}$$

$$+ \Pr_{\xi^{(2)}} \left[ \langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \geqslant 1 \right] + \Pr_{\xi^{(2)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(1)} + \xi^{(2)} \geqslant 1 \right] \tag{A.153}$$

$$= O\left( \frac{\eta^{(1)}|a_i^{(0)}|k(\gamma+\epsilon_e)}{\tilde{\sigma}^2 \sigma_\xi^{(2)}} \right) + \exp(-\Theta(k)) = O(\epsilon_{e2}). \tag{A.154}$$

This leads to

$$\left| T_{j1} - \mathbb{E}_{\phi_\mathbf{A},\phi_{-\mathbf{A}}} \{ y\phi_j I'_a \} \right| \tag{A.155}$$

$$\leqslant \mathbb{E}_{\phi_\mathbf{A}} \left\{ |y\phi_j| \left| \mathbb{E}_{\phi_{-\mathbf{A}}} (I_a - I'_a) \right| \right\} \tag{A.156}$$

$$\leqslant O(\epsilon_{e2}) \mathbb{E}_{\phi_\mathbf{A}} \{ |y\phi_j| \} \tag{A.157}$$

$$\leqslant O(\epsilon_{e2}/\tilde{\sigma}) \tag{A.158}$$

where the last step is from Lemma A.8. Furthermore,

$$\begin{aligned}
&\mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}}\{y\phi_j I'_a\} && \text{(A.159)}\\
&= \mathbb{E}_{\phi_{\mathbf{A}}}\{y\phi_j\}\mathbb{E}_{\phi_{-\mathbf{A}}}[I'_a] && \text{(A.160)}\\
&= \mathbb{E}_{\phi_{\mathbf{A}}}\{y\phi_j\}\Pr_{\phi_{-\mathbf{A}},\xi^{(2)}}\left[\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right]. && \text{(A.161)}
\end{aligned}$$

By Lemma A.13, we have $|\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle| \leqslant O(\eta^{(1)}\tilde{\sigma}/\gamma)$. Also, $|b^{(1)} - b^{(0)}| \leqslant O(\eta^{(1)}|a_i^{(0)}|\epsilon_e)$. By the property of $\xi^{(2)}$,

$$\begin{aligned}
&\left|\Pr_{\xi^{(2)}}\left[\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right] - \Pr_{\xi^{(2)}}\left[b^{(0)} + \xi^{(2)} \in (0,1)\right]\right| && \text{(A.162)}\\
&\leqslant O(\eta^{(1)}\tilde{\sigma}/(\gamma\sigma_\xi^{(2)})) + O(\eta^{(1)}|a_i^{(0)}|\epsilon_e/\sigma_\xi^{(2)}). && \text{(A.163)}
\end{aligned}$$

On the other hand,

$$\begin{aligned}
\beta := \Pr_{\phi_{-\mathbf{A}},\xi^{(2)}}\left[b^{(0)} + \xi^{(2)} \in (0,1)\right] &= \Pr_{\xi^{(2)}}\left[\xi^{(2)} \in (-b^{(0)}, 1 - b^{(0)})\right] && \text{(A.164)}\\
&= \Omega(1) && \text{(A.165)}
\end{aligned}$$

and $\beta$ only depends on $b^{(0)}$. By Lemma A.8, $\mathbb{E}_{\phi_{\mathbf{A}}}\{y\phi_j\} = \gamma/\tilde{\sigma}$. Therefore,

$$|T_{j1} - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_{e2}/\tilde{\sigma}) + O(\eta^{(1)}/\sigma_\xi^{(2)}) + O(\eta^{(1)}|a_i^{(0)}|\epsilon_e/(\tilde{\sigma}\sigma_\xi^{(2)})). \qquad \text{(A.166)}$$

Now, consider $j \notin \mathbf{A}$. Let B denote $\mathbf{A} \cup \{j\}$.

$$\begin{aligned}
T_{j1} &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{y\phi_j\mathbb{I}\left[\langle\phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\} && \text{(A.167)}\\
&= \mathbb{E}_{\phi_B}\mathbb{E}_{\phi_{-B},\xi^{(2)}}\left\{y\phi_j\mathbb{I}\left[\langle\phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\} && \text{(A.168)}\\
&= \mathbb{E}_{\phi_B}\left\{y\phi_j\Pr_{\phi_{-B},\xi^{(2)}}\left[\langle\phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\}. && \text{(A.169)}
\end{aligned}$$

Let

$$I_b := \Pr_{\xi^{(2)}}\left[\langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right], \tag{A.170}$$

$$I'_b := \Pr_{\xi^{(2)}}\left[\langle \phi_{-B}, q_{-B}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right]. \tag{A.171}$$

Similar as above, we have $|I_b - I'_b| \leqslant \epsilon_{e2}$. Then by Lemma A.8,

$$\left|T_{j1} - \mathbb{E}_{\phi_B,\phi_{-B}}\{y\phi_j I'_b\}\right| \tag{A.172}$$

$$\leqslant \mathbb{E}_{\phi_B}\left\{|y\phi_j||\mathbb{E}_{\phi_{-B}}(I_b - I'_b)|\right\} \tag{A.173}$$

$$\leqslant O(\epsilon_{e2})\mathbb{E}_{\phi_j}\{|\phi_j|\} \tag{A.174}$$

$$\leqslant O(\epsilon_e) \times O(\sigma_\phi^2\tilde{\sigma}) \tag{A.175}$$

$$= O(\sigma_\phi^2\epsilon_{e2}\tilde{\sigma}). \tag{A.176}$$

Furthermore,

$$\mathbb{E}_{\phi_B,\phi_{-B}}\{y\phi_j I'_b\} = \mathbb{E}_{\phi_\mathbf{A}}\{y\}\mathbb{E}_{\phi_j}\{\phi_j\}\mathbb{E}_{\phi_{-B}}[I'_b] = 0. \tag{A.177}$$

Therefore,

$$|T_{j1}| \leqslant O(\sigma_\phi^2\epsilon_{e2}\tilde{\sigma}). \tag{A.178}$$

□

**Lemma A.15.** *Under the same assumptions as in Lemma A.14,*

$$\frac{\partial}{\partial b}L_{\mathcal{D}}(g^{(1)}; \sigma_\xi^{(2)}) = -a_i^{(1)}T_b \tag{A.179}$$

*where* $|T_b| \leqslant \exp(-\Omega(k)) + O(\epsilon_{e2})$.

*Proof of Lemma A.15.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. By Lemma A.13, $\Pr[yg^{(1)}(\mathbf{x};\xi^{(2)}) > 1] \leqslant \exp(-\Omega(k))$. Let $\mathbb{I}_\mathbf{x} =$

$\mathbb{I}[yg^{(1)}(\mathbf{x};\xi^{(2)}) \leqslant 1]$.

$$\frac{\partial}{\partial b} L_{\mathcal{D}}(g^{(1)};\sigma_{\xi}^{(2)}) \tag{A.180}$$

$$= -a^{(1)} \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left\{ y\mathbb{I}_{\mathbf{x}} \mathbb{E}_{\xi^{(2)}} \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)] \right\}}_{:=T_b}. \tag{A.181}$$

Let $T_{b1} := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)] \right\}$. We have

$$|T_b - T_{b1}| \tag{A.182}$$

$$= \left| \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y(1-\mathbb{I}_{\mathbf{x}})\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)] \right\} \right| \tag{A.183}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} |1 - \mathbb{I}_{\mathbf{x}}| \tag{A.184}$$

$$\leqslant \exp(-\Omega(k)). \tag{A.185}$$

So it is sufficient to bound $T_{b1}$. For simplicity, we use $q$ as a shorthand for $q_i^{(1)}$.

$$T_{b1} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ y\mathbb{I}\left[\langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right] \right\} \tag{A.186}$$

$$= \mathbb{E}_{\phi_A} \mathbb{E}_{\phi_{-A},\xi^{(2)}} \left\{ y\mathbb{I}\left[\langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right] \right\} \tag{A.187}$$

$$= \mathbb{E}_{\phi_A} \left\{ y \Pr_{\phi_{-A},\xi^{(2)}} \left[\langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right] \right\}. \tag{A.188}$$

Let

$$I_b := \Pr_{\xi^{(2)}} \left[\langle \phi, q\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right], \tag{A.189}$$

$$I_b' := \Pr_{\xi^{(2)}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)\right]. \tag{A.190}$$

Similar as in Lemma A.14, we have $|I_b - I_b'| \leqslant \epsilon_{e2}$. Then by Lemma A.8,

$$\left| T_{b1} - \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}} \{ yI_b' \} \right| \tag{A.191}$$

$$\leqslant \mathbb{E}_{\phi_{\mathbf{A}}} \left\{ |\mathbb{E}_{\phi_{-\mathbf{A}}}(I_b - I_b')| \right\} \tag{A.192}$$

$$\leqslant O(\epsilon_{e2}). \tag{A.193}$$

Furthermore,

$$\mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}}\{y I'_b\} = \mathbb{E}_{\phi_{\mathbf{A}}}\{y\}\mathbb{E}_{\phi_{-\mathbf{A}}}[I'_b] = 0. \tag{A.194}$$

Therefore, $|T_{b1}| \leqslant O(\epsilon_{e2})$ and the statement follows. □

**Lemma A.16.** *Under the same assumptions as in Lemma A.14,*

$$\frac{\partial}{\partial a_i} L_{\mathcal{D}}(g^{(1)}; \sigma_{\xi}^{(2)}) = -T_a \tag{A.195}$$

*where* $|T_a| = O(\eta^{(1)}\tilde{\sigma}/\gamma) + \exp(-\Omega(k))\mathrm{poly}(Dm)$.

*Proof of Lemma A.16.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. By Lemma A.13, $\Pr[y g^{(1)}(\mathbf{x}; \xi^{(2)}) > 1] \leqslant \exp(-\Omega(k))$. Let $\mathbb{I}_{\mathbf{x}} = \mathbb{I}[y g^{(1)}(\mathbf{x}; \xi^{(2)}) \leqslant 1]$.

$$\frac{\partial}{\partial a} L_{\mathcal{D}}(g^{(1)}; \sigma_{\xi}^{(2)}) \tag{A.196}$$

$$= -\underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y \mathbb{I}_{\mathbf{x}} \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}}_{:=T_a}. \tag{A.197}$$

Let $T_{a1} := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}$. We have

$$|T_a - T_{a1}| \tag{A.198}$$

$$= \left|\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y(1 - \mathbb{I}_{\mathbf{x}}) \mathbb{E}_{\xi^{(2)}} \sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}\right| \tag{A.199}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} |1 - \mathbb{I}_{\mathbf{x}}| \tag{A.200}$$

$$\leqslant \exp(-\Omega(k)). \tag{A.201}$$

So it is sufficient to bound $T_{a1}$. For simplicity, we use $q$ as a shorthand for $q_i^{(1)}$.

Let $\phi'_{\mathbf{A}}$ be an independent copy of $\phi_{\mathbf{A}}$, $\phi'$ be the vector obtained by replacing in $\phi$ the entries $\phi_{\mathbf{A}}$ with $\phi'_{\mathbf{A}}$, and let $x' = M\phi'$ and its label is $y'$. Then

$$|T_{a1}| := \left|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{y \mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(2)}} \sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}\right| \tag{A.202}$$

$$\leqslant \frac{1}{2}\Bigg|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)},\mathbf{x}\rangle + b^{(1)} + \xi^{(1)})|y=1\right\} \tag{A.203}$$

$$-\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)},\mathbf{x}\rangle + b^{(1)} + \xi^{(2)})|y=-1\right\}\Bigg| \tag{A.204}$$

$$\leqslant \frac{1}{2}\Bigg|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)},\mathbf{x}\rangle + b^{(1)} + \xi^{(2)})|y=1\right\} \tag{A.205}$$

$$-\mathbb{E}_{\phi'_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)},\mathbf{x}'\rangle + b^{(1)} + \xi^{(2)})|y'=-1\right\}\Bigg| \tag{A.206}$$

$$\leqslant \frac{1}{2}\mathbb{E}_{\phi_{\mathbf{A}},\phi'_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}}}\left|\langle \mathbf{w}^{(1)},\mathbf{x}\rangle - \langle \mathbf{w}^{(1)},\mathbf{x}'\rangle\right| |y=1,y'=-1\right\} \tag{A.207}$$

$$\leqslant \frac{1}{2}\mathbb{E}_{\phi_{-\mathbf{A}}}\left(\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\left|\langle \mathbf{w}^{(1)},\mathbf{x}\rangle\right| |y=1\right\} + \mathbb{E}_{\phi'_{\mathbf{A}}}\left\{\left|\langle \mathbf{w}^{(1)},\mathbf{x}'\rangle\right| |y'=-1\right\}\right) \tag{A.208}$$

$$\leqslant \mathbb{E}_{\phi_{-\mathbf{A}},\phi_{\mathbf{A}}}\left|\langle \mathbf{w}^{(1)},\mathbf{x}\rangle\right| \tag{A.209}$$

$$= \mathbb{E}_{\mathbf{x}}\left|\langle \mathbf{w}^{(1)},\mathbf{x}\rangle\right| \tag{A.210}$$

$$= O(\eta^{(1)}\tilde{\sigma}/\gamma) + \exp(-\Omega(k)) \times D \times \|q^{(1)}\|_\infty\|\phi\|_\infty \tag{A.211}$$

$$= O(\eta^{(1)}\tilde{\sigma}/\gamma) + \exp(-\Omega(k))\frac{D|\eta^{(1)}a^{(0)}|(\gamma+\epsilon_e)}{\tilde{\sigma}^2} \tag{A.212}$$

$$= O(\eta^{(1)}\tilde{\sigma}/\gamma) + \exp(-\Omega(k))\mathrm{poly}(Dm) \tag{A.213}$$

where the fourth step follows from that $\sigma$ is 1-Lipschitz, the third to the last step from Lemma A.13, and the second to the last step from Lemma A.12. □

With the above lemmas about the gradients, we are now ready to show that at the end of the second step, we get a good set of features for accurate prediction.

**Lemma A.17.** *Set*

$$\eta^{(1)} = \frac{\gamma^2\tilde{\sigma}}{km^3}, \lambda_a^{(1)} = 0, \lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \sigma_\xi^{(1)} = 1/k^{3/2}, \tag{A.214}$$

$$\eta^{(2)} = 1, \lambda_a^{(2)} = \lambda_{\mathbf{w}}^{(2)} = 1/(2\eta^{(2)}), \sigma_\xi^{(2)} = 1/k^{3/2}. \tag{A.215}$$

*Fix $\delta \in (0, O(1/k^3))$. If $p_o = \Omega(k^2/D)$, $k = \Omega\left(\log^2\left(\frac{Dm}{\delta\gamma}\right)\right)$, and $m \geqslant \max\{\Omega(k^4), D\}$, then with probability at least $1-\delta$ over the initialization, there exist $\tilde{a}_i$'s such that*

$\tilde{g}(\mathbf{x}) := \sum_{i=1}^{2m} \tilde{a}_i \sigma(\langle \mathbf{w}_i^{(2)}, \mathbf{x}\rangle + b_i^{(2)})$ *satisfies* $L_{\mathcal{D}}(\tilde{g}) = 0$. *Furthermore,* $\|\tilde{a}\|_0 = O(m/k)$, $\|\tilde{a}\|_\infty = O(k^5/m)$, *and* $\|\tilde{a}\|_2^2 = O(k^9/m)$. *Finally,* $\|a^{(2)}\|_\infty = O\left(\frac{1}{km^2}\right)$, $\|\mathbf{w}_i^{(2)}\|_2 = O(\tilde{\sigma}/k)$, *and* $|b_i^{(2)}| = O(1/k^2)$ *for all* $i \in [2m]$.

*Proof of Lemma A.17.* By Lemma 2.5, there exists a network

$$g^*(\mathbf{x}) = \sum_{\ell=1}^{3(k+1)} a_\ell^* \sigma(\langle \mathbf{w}_\ell^*, \mathbf{x}\rangle + b_\ell^*)$$

satisfying $g^*(\mathbf{x})$ for all $(\mathbf{x}, y) \sim \mathcal{D}$. Furthermore, $|a_i^*| \leqslant 32k, 1/(32k) \leqslant |b_i^*| \leqslant 1/2$, $\mathbf{w}_i^* = \tilde{\sigma} \sum_{j \in \mathbf{A}} M_j/(4k)$, and $|\langle \mathbf{w}_i^*, \mathbf{x}\rangle + b_i^*| \leqslant 1$ for any $i \in [n]$ and $(\mathbf{x}, y) \sim \mathcal{D}$. Now we fix an $\ell$, and show that with high probability there is a neuron in $g^{(2)}$ that can approximate the $\ell$-th neuron in $g^*$.

By Lemma A.4, with probability $1 - 2\delta$ over $\mathbf{w}_i^{(0)}$'s, they are all in $\mathcal{G}_{\mathbf{w}}(\delta)$; with probability $1 - \delta$ over $a_i^{(0)}$'s, they are all in $\mathcal{G}_a(\delta)$; with probability $1 - \delta$ over $b_i^{(0)}$'s, they are all in $\mathcal{G}_b(\delta)$. Under these events, by Lemma A.12, Lemma A.14 and A.15, for any neuron $i \in [2m]$, we have

$$\mathbf{w}_i^{(2)} = a_i^{(1)} \sum_{j=1}^{D} M_j T_j, \tag{A.216}$$

$$b_i^{(2)} = b_i^{(1)} + a_i^{(1)} T_b. \tag{A.217}$$

where

- if $j \in A$, then $|T_j - \beta\gamma/\tilde{\sigma}| \leqslant \epsilon_{\mathbf{w}1} := O(\epsilon_{e2}/\tilde{\sigma} + \eta^{(1)}/\sigma_\xi^{(2)} + \eta^{(1)}|a_i^{(0)}|\epsilon_e/(\tilde{\sigma}\sigma_\xi^{(2)}))$, where $\beta \in [\Omega(1), 1]$ and depends only on $\mathbf{w}_i^{(0)}, b_i^{(0)}$;
- if $j \notin \mathbf{A}$, then $|T_j| \leqslant \epsilon_{\mathbf{w}2} := \frac{1}{\tilde{\sigma}} \exp(-\Theta(k)) + O(\sigma_\phi^2 \epsilon_{e2} \tilde{\sigma})$.
- $|T_b| \leqslant \epsilon_b := \frac{1}{\tilde{\sigma}} \exp(-\Theta(k)) + O(\epsilon_{e2})$.

Given the initialization, with probability $\Omega(1)$ over $b_i^{(0)}$, we have

$$|b_i^{(0)}| \in \left[\frac{1}{2k^2}, \frac{2}{k^2}\right], \text{sign}(b_i^{(0)}) = \text{sign}(b_\ell^*). \tag{A.218}$$

Finally, since $\frac{4k|b_\ell^*|\beta\gamma}{|b_i^{(0)}|\tilde{\sigma}^2} \in [\Omega(k^2\gamma/\tilde{\sigma}^2), O(k^3\gamma/\tilde{\sigma}^2)]$ and depends only on $\mathbf{w}_i^{(0)}, b_i^{(0)}$, we have that for $\epsilon_a = \Theta(1/k^2)$, with probability $\Omega(\epsilon_a) > \delta$ over $a_i^{(0)}$,

$$\left|\frac{4k|b_\ell^*|\beta\gamma}{|b_i^{(0)}|\tilde{\sigma}^2} a_i^{(0)} - 1\right| \leqslant \epsilon_a, \quad |a_i^{(0)}| = O\left(\frac{\tilde{\sigma}^2}{k^2\gamma}\right). \tag{A.219}$$

Let $n_a = \epsilon_a m/4$. For the given value of $m$, by (A.216)-(A.219) we have with probability $\geqslant 1 - 5\delta$ over the initialization, for each $\ell$ there is a different set of neurons $I_\ell \subseteq [m]$ with $|I_\ell| = n_a$ and such that for each $i_\ell \in I_\ell$,

$$|b_{i_\ell}^{(0)}| \in \left[\frac{1}{2k^2}, \frac{2}{k^2}\right], \quad \operatorname{sign}(b_{i_\ell}^{(0)}) = \operatorname{sign}(b_\ell^*), \tag{A.220}$$

$$\left|\frac{4k|b_\ell^*|\beta\gamma}{|b_{i_\ell}^{(0)}|\tilde{\sigma}^2} a_{i_\ell}^{(0)} - 1\right| \leqslant \epsilon_a, \quad |a_{i_\ell}^{(0)}| = O\left(\frac{\tilde{\sigma}^2}{k^2\gamma}\right). \tag{A.221}$$

We also have

$$\left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \tag{A.222}$$

$$\leqslant \left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| + \left|\frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \tag{A.223}$$

$$= \left|a_{i_\ell}^{(1)} \sum_{j=1}^{D} T_j\phi_j - \frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \phi_j\right| + \left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right| \left|\frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \phi_j\right| \tag{A.224}$$

$$= \left|a_{i_\ell}^{(1)}\right| \left|\sum_{j=1}^{D} T_j\phi_j - \frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \phi_j\right| + \left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right| \left|\frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \phi_j\right|. \tag{A.225}$$

We have $\left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right| = O(\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)})$, and

$$\left|\sum_{j=1}^{D} T_j\phi_j - \frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \phi_j\right| \leqslant \left|\sum_{j\in\mathbf{A}} (T_j - \frac{\beta\gamma}{\tilde{\sigma}})\phi_j\right| + \left|\sum_{j\notin\mathbf{A}} T_j\phi_j\right| \tag{A.226}$$

$$\leqslant O(k\epsilon_{\mathbf{w}1}/\tilde{\sigma}) + O(D\epsilon_{\mathbf{w}2}/\tilde{\sigma}) =: \epsilon_{\phi}. \tag{A.227}$$

For the given values of parameters, we have

$$\epsilon_{e2} = O\left(\frac{\gamma}{m^2}\right), \tag{A.228}$$

$$\epsilon_{\mathbf{w}1} = O\left(\frac{k\gamma}{m^2\tilde{\sigma}} + \frac{\gamma\epsilon_e}{km^2}\right), \tag{A.229}$$

$$\epsilon_{\mathbf{w}2} = O\left(\frac{\gamma}{m^2\tilde{\sigma}}\right), \tag{A.230}$$

$$\epsilon_b = O\left(\frac{\gamma}{m^2}\right), \tag{A.231}$$

$$\epsilon_{\phi} = O\left(\frac{k^2\gamma}{m^2\tilde{\sigma}^2} + \frac{\gamma\epsilon_e}{m^2\tilde{\sigma}} + \frac{\gamma}{m\tilde{\sigma}^2}\right). \tag{A.232}$$

Therefore,

$$\left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}}\sum_{j\in\mathbf{A}}\langle M_j, \mathbf{x}\rangle\right| \tag{A.233}$$

$$\leqslant \left|a_{i_\ell}^{(1)}\right|\epsilon_{\phi} + \left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right|\frac{k\gamma}{\tilde{\sigma}^2} \tag{A.234}$$

$$\leqslant O\left(\frac{\tilde{\sigma}^2}{k^2\gamma} + \eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}\right)\left(\frac{k^2\gamma}{m^2\tilde{\sigma}^2} + \frac{\gamma\epsilon_e}{m^2\tilde{\sigma}} + \frac{\gamma}{m\tilde{\sigma}^2}\right) \tag{A.235}$$

$$+ O\left(\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}\right)\frac{k\gamma}{\tilde{\sigma}^2} \tag{A.236}$$

$$\leqslant O\left(\frac{1}{m}\right). \tag{A.237}$$

We also have by Lemma A.12 and A.15:

$$|b_{i_\ell}^{(2)} - b_{i_\ell}^{(0)}| \leqslant O\left(\eta^{(1)}|a_{i_\ell}^{(0)}|\epsilon_e + |a_{i_\ell}^{(1)}|\left(\frac{1}{\tilde{\sigma}}\exp(-\Theta(k)) + \epsilon_{e2}\right)\right) \leqslant O\left(\frac{1}{m}\right). \tag{A.238}$$

Now, construct $\tilde{a}$ such that $\tilde{a}_{i_\ell} = \frac{2a_\ell^*|b_\ell^*|}{|b_{i_\ell}^{(0)}|n_a}$ for each $\ell$ and each $i_\ell \in I_\ell$, and $\tilde{a}_i = 0$

elsewhere. Then

$$
\begin{aligned}
&|\tilde{g}(\mathbf{x}) - 2g^*(\mathbf{x})| && \text{(A.239)}\\
&= \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \tilde{a}_{i_\ell} \sigma\left( \langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x} \rangle + b_{i_\ell}^{(2)} \right) - \sum_{\ell=1}^{3(k+1)} 2a_\ell^* \sigma\left( \langle \mathbf{w}_\ell^*, \mathbf{x} \rangle + b_\ell^* \right) \right| && \text{(A.240)}\\
&= \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}| n_a} \sigma\left( \langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x} \rangle + b_{i_\ell}^{(2)} \right) - \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}| n_a} \sigma\left( \frac{|b_{i_\ell}^{(0)}|}{|b_\ell^*|} \langle \mathbf{w}_\ell^*, \mathbf{x} \rangle + b_{i_\ell}^{(0)} \right) \right| && \text{(A.241)}\\
&\leqslant \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{1}{n_a} \left( \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x} \rangle + b_{i_\ell}^{(2)} \right) - \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x} \rangle + b_{i_\ell}^{(0)} \right) \right) \right| && \text{(A.242)}\\
&\quad + \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{1}{n_a} \left( \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x} \rangle + b_{i_\ell}^{(0)} \right) - \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \frac{|b_{i_\ell}^{(0)}|}{|b_\ell^*|} \langle \mathbf{w}_\ell^*, \mathbf{x} \rangle + b_{i_\ell}^{(0)} \right) \right) \right| && \text{(A.243)}\\
&\leqslant 3(k+1) \max_\ell \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} O\left( \frac{1}{m} \right) + && \text{(A.244)}\\
&\quad 3(k+1) \max_\ell \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \frac{\tilde{\sigma} |b_{i_\ell}^{(0)}|}{4k |b_\ell^*|} \left| \frac{4k a_{i_\ell}^{(0)} \beta\gamma |b_\ell^*|}{\tilde{\sigma}^2 |b_{i_\ell}^{(0)}|} - 1 \right| \frac{k}{\tilde{\sigma}} && \text{(A.245)}\\
&= O\left( \frac{k^4}{m} + k^2 \epsilon_a \right) && \text{(A.246)}\\
&\leqslant 1. && \text{(A.247)}
\end{aligned}
$$

Here the second equation follows from that $\sigma$ is positive-homogeneous in $[0,1]$, $|\langle \mathbf{w}_\ell^*, \mathbf{x} \rangle + b_\ell^*| \leqslant 1$, $|b_{i_\ell}^{(0)}| / |b_\ell^*| \leqslant 1$. This guarantees $y\tilde{g}(\mathbf{x}) \geqslant 1$. Changing the scaling of $\delta$ leads to the statement.

Finally, the bounds on $\tilde{a}$ follow from the above calculation. The bound on $\|a^{(2)}\|_2$ follows from Lemma A.16, and those on $\|\mathbf{w}_i^{(2)}\|_2$ and $\|b_i^{(2)}\|_2$ follow from (A.216)(A.217) and the bounds on $a_i^{(1)}$ and $b_i^{(1)}$ in Lemma A.12. □

### A.4.6 Classifier Learning Stage

Once we have a set of good features, we are now ready to prove that the later steps will learn an accurate classifier. The intuition is that the first layer's weights do not change much and the second layer's weights get updated till achieving good accuracy. In particular, we will employ the online optimization technique from Daniely and Malach (2020).

We begin by showing that the first layer's weights do not change too much.

**Lemma A.18.** *Assume the same conditions as in Lemma A.17. Suppose for* $t > 2$, $\lambda_a^{(t)} = \lambda_{\mathbf{w}}^{(t)} = \lambda$, $\eta^{(t)} = \eta$ *for some* $\lambda, \eta \in (0, 1)$, *and* $\sigma_\xi^{(t)} = 0$. *Then for any* $t > 2$ *and* $i \in [2m]$,

$$|a_i^{(t)}| \leqslant \eta t + O\left(\frac{1}{km^2}\right), \tag{A.248}$$

$$\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(2)}\|_2 \leqslant O\left(\frac{t\eta\lambda\tilde{\sigma}}{k}\right) + \eta^2 t^2 + O\left(\frac{t}{km^2}\right), \tag{A.249}$$

$$|b_i^{(t)} - b_i^{(2)}| \leqslant O\left(\frac{t\eta\lambda}{k^2}\right) + \eta^2 t^2 + O\left(\frac{t}{km^2}\right). \tag{A.250}$$

*Proof of Lemma A.18.* First, we bound the size of $|a_i^{(t)}|$:

$$\begin{aligned}
|a_i^{(t)}| &= \left|(1 - 2\eta\lambda)a_i^{(t-1)} - \eta\frac{\partial}{\partial a_i}L_{\mathcal{D}}(g^{(t-1)})\right| && \text{(A.251)}\\
&\leqslant \left|(1 - 2\eta\lambda)a_i^{(t-1)} - \eta\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg^{(t-1)}(\mathbf{x}) \leqslant 1]\sigma(\langle\mathbf{w}_i^{(t-1)}, \mathbf{x}\rangle + b_i^{(t-1)})\right\}\right| && \text{(A.252)}\\
&\leqslant |a_i^{(t-1)}| + \eta && \text{(A.253)}
\end{aligned}$$

which leads to

$$|a_i^{(t)}| \leqslant \eta t + |a_i^{(2)}| \tag{A.254}$$

where $|a_i^{(2)}| = O\left(\frac{1}{km^2}\right)$. We are now to bound the change of $\mathbf{w}_i^{(t)}$ and $b_i^{(t)}$.

$$\begin{aligned}
&\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(2)}\|_2 && \text{(A.255)}\\
&= \left\|(1-2\eta\lambda)\mathbf{w}_i^{(t-1)} - \eta\frac{\partial}{\partial \mathbf{w}_i}L_{\mathcal{D}}(g^{(t-1)}) - \mathbf{w}_i^{(2)}\right\|_2 && \text{(A.256)}\\
&\leqslant \Bigg\|(1-2\eta\lambda)\mathbf{w}_i^{(t-1)} && \text{(A.257)}\\
&\quad + \eta a_i^{(t-1)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg^{(t-1)}(\mathbf{x}) \leqslant 1]\mathbb{I}[\langle \mathbf{w}_i^{(t-1)}, \mathbf{x}\rangle + b_i^{(t-1)} \in (0,1)]\mathbf{x}\right\} - \mathbf{w}_i^{(2)}\Bigg\|_2 && \text{(A.258)}\\
&\leqslant \left\|(1-2\eta\lambda)\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(2)}\right\|_2 && \text{(A.259)}\\
&\quad + \eta\left\|a_i^{(t-1)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg^{(t-1)}(\mathbf{x}) \leqslant 1]\mathbb{I}[\langle \mathbf{w}_i^{(t-1)}, \mathbf{x}\rangle + b_i^{(t-1)} \in (0,1)]\mathbf{x}\right\}\right\|_2 && \text{(A.260)}\\
&\leqslant (1-2\eta\lambda)\left\|\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(2)}\right\|_2 + 2\eta\lambda\left\|\mathbf{w}_i^{(2)}\right\|_2 + \eta\left|a_i^{(t-1)}\right| && \text{(A.261)}
\end{aligned}$$

leading to

$$\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(2)}\|_2 \leqslant 2t\eta\lambda\left\|\mathbf{w}_i^{(2)}\right\|_2 + \eta^2 t^2 + t|a_i^{(2)}|. \tag{A.262}$$

Note that $\|\mathbf{w}_i^{(2)}\|_2 = O(\tilde{\sigma}/k)$.

$$\begin{aligned}
&|b_i^{(t)} - b_i^{(2)}| && \text{(A.263)}\\
&= \left|b_i^{(t-1)} - \eta\frac{\partial}{\partial b_i}L_{\mathcal{D}}(g^{(t-1)}) - b_i^{(2)}\right| && \text{(A.264)}\\
&\leqslant \left|b_i^{(t-1)} - b_i^{(2)}\right| && \text{(A.265)}\\
&\quad + \eta\left|a_i^{(t-1)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{y\mathbb{I}[yg^{(t-1)}(\mathbf{x}) \leqslant 1]\mathbb{I}[\langle \mathbf{w}_i^{(t-1)}, \mathbf{x}\rangle + b_i^{(t-1)} \in (0,1)]\right\}\right| && \text{(A.266)}\\
&\leqslant \left|b_i^{(t-1)} - b_i^{(2)}\right| + \eta\left|a_i^{(t-1)}\right| && \text{(A.267)}
\end{aligned}$$

leading to

$$|b_i^{(t)} - b_i^{(2)}| \leqslant \eta^2 t^2 + t|a_i^{(2)}|. \tag{A.268}$$

Note that $\left|b_i^{(2)}\right| = O(1/k^2)$. □

**Lemma A.19.** *Assume the same conditions as in Lemma A.18. Let*

$$g_{\tilde{a}}^{(t)}(\mathbf{x}) = \sum_{i=1}^{m} \tilde{a}_i \sigma(\langle \mathbf{w}_i^{(t)}, \mathbf{x}\rangle + b_i^{(t)}).$$

*Then*

$$|\ell(g_{\tilde{a}}^{(t)}(\mathbf{x}), y) - \ell(g_{\tilde{a}}^{(2)}(\mathbf{x}), y)| \leqslant \|\tilde{a}\|_2 \sqrt{\|\tilde{a}\|_0} \left( O\left(\frac{t\eta\lambda\tilde{\sigma}}{k}\right) + \eta^2 t^2 + O\left(\frac{t}{km^2}\right)\right). \tag{A.269}$$

*Proof of Lemma A.19.* It follows from that

$$\begin{aligned}
&|\ell(g_{\tilde{a}}^{(t)}(\mathbf{x}), y) - \ell(g_{\tilde{a}}^{(2)}(\mathbf{x}))| && \text{(A.270)}\\
&\leqslant |g_{\tilde{a}}^{(t)}(\mathbf{x}) - g_{\tilde{a}}^{(2)}(\mathbf{x})| && \text{(A.271)}\\
&\leqslant \|\tilde{a}\|_2 \sqrt{\|\tilde{a}\|_0} \max_{i \in [2m]} \left|\sigma(\langle \mathbf{w}_i^{(t)}, \mathbf{x}\rangle + b_i^{(t)}) - \sigma(\langle \mathbf{w}_i^{(2)}, \mathbf{x}\rangle + b_i^{(2)})\right| && \text{(A.272)}\\
&\leqslant \|\tilde{a}\|_2 \sqrt{\|\tilde{a}\|_0} \max_{i \in [2m]} \left(\left|\langle \mathbf{w}_i^{(t)} - \mathbf{w}_i^{(2)}, \mathbf{x}\rangle\right| + \left|b_i^{(t)} - b_i^{(2)}\right|\right). && \text{(A.273)}
\end{aligned}$$

and Lemma A.18. □

### A.4.7 Proof of Theorem 2.1

Based on the above lemmas, following the same argument as in the proof of Theorem 2 in Daniely and Malach (2020), we get our main theorem.

**Theorem A.20** (Full version of Theorem 2.1)**.** *Set*

$$\eta^{(1)} = \frac{\gamma^2 \tilde{\sigma}^2}{km^3}, \lambda_a^{(1)} = 0, \lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \sigma_\xi^{(1)} = 1/k^2, \tag{A.274}$$

$$\eta^{(2)} = 1, \lambda_a^{(2)} = \lambda_{\mathbf{w}}^{(2)} = 1/(2\eta^{(2)}), \sigma_\xi^{(2)} = 1/k^2, \tag{A.275}$$

$$\eta^{(t)} = \eta = \frac{k^2}{Tm^{1/3}}, \lambda_a^{(t)} = \lambda_{\mathbf{w}}^{(t)} = \lambda \leqslant \frac{k^3}{\tilde{\sigma} m^{1/3}}, \sigma_\xi^{(t)} = 0, \text{ for } 2 < t \leqslant T. \tag{A.276}$$

*For any* $\delta \in (0,1)$*, if* $p_o = \Omega(k^2/D)$*,* $k = \Omega\left(\log^2\left(\frac{D}{\delta\gamma}\right)\right)$*,* $\max\{\Omega(k^4), D\} \leqslant m \leqslant$ *poly*$(D)$*, then we have for any* $\mathcal{D} \in \mathcal{F}_\Xi$*, with probability at least* $1 - \delta$*, there exists* $t \in [T]$ *such that*

$$\Pr[\text{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) = O\left(\frac{k^8}{m^{2/3}} + \frac{k^3 T}{m^2} + \frac{k^2 m^{2/3}}{T}\right). \tag{A.277}$$

*Consequently, for any* $\epsilon \in (0,1)$*, if* $T = m^{4/3}$*, and* $\max\{\Omega(k^{12}/\epsilon^{3/2}), D\} \leqslant m \leqslant$ *poly*$(D)$*, then*

$$\Pr[\text{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) \leqslant \epsilon. \tag{A.278}$$

*Proof of Theorem 2.1.* Consider $\tilde{L}_{\mathcal{D}}(g^{(t)}) = \mathbb{E}[\ell(g^{(t)}, y)] + \lambda_a^{(t)} \|a^{(t)}\|_2^2$. Note that the gradient update using $\tilde{L}_{\mathcal{D}}(g^{(t)})$ is the same as the update in our learning algorithm. Then by Theorem A.21, Lemma A.17, and Lemma A.19,

$$\frac{1}{T}\sum_{t=3}^{T} \tilde{L}_{\mathcal{D}}(g^{(t)}) \leqslant \frac{\|\tilde{a}\|_2^2}{2} + \|\tilde{a}\|_2 \sqrt{\|\tilde{a}\|_0}\left(O\left(\frac{T\eta\lambda\tilde{\sigma}}{k}\right) + \eta^2 T^2 + O\left(\frac{T}{km^2}\right)\right) \tag{A.279}$$

$$+ \frac{\|\tilde{a}\|_2^2}{2\eta T} + \|a^{(2)}\|_2 \sqrt{m} + \eta m \tag{A.280}$$

$$\leqslant O\left(\frac{k^9}{m} + k^4 \eta^2 T^2 + \frac{k^3 T}{m^2} + \frac{k^9}{\eta T m} + \eta m\right). \tag{A.281}$$

$$\leqslant O\left(\frac{k^8}{m^{2/3}} + \frac{k^3 T}{m^2} + \frac{k^2 m^{2/3}}{T}\right). \tag{A.282}$$

The statement follows from that 0-1 classification error is bounded by the hinge-loss. □

**Theorem A.21** (Theorem 13 in Daniely and Malach (2020))**.** *Fix some* $\eta$*, and let* $f_1, \ldots, f_T$ *be some sequence of convex functions. Fix some* $\theta_1$*, and assume we update*

$\theta_{t+1} = \theta_t - \eta \nabla f_t(\theta_t)$. *Then for every* $\theta^*$ *the following holds:*

$$\frac{1}{T}\sum_{t=1}^{T} f_t(\theta_t) \leqslant \frac{1}{T}\sum_{t=1}^{T} f_t(\theta^*) + \frac{1}{2\eta T}\|\theta^*\|_2^2 + \|\theta_1\|_2 \frac{1}{T}\sum_{t=1}^{T} \|\nabla f_t(\theta_t)\|_2 + \eta \frac{1}{T}\sum_{t=1}^{T} \|\nabla f_t(\theta_t)\|_2^2. \tag{A.283}$$

## A.5 Lower Bound for Linear Models on Fixed Feature Mappings

**Theorem A.22** (Restatement of Theorem 2.2)**.** *Suppose* $\Psi$ *is a data-independent feature mapping of dimension* $N$ *with bounded features, i.e.,* $\Psi : \mathbf{X} \to [-1, 1]^N$. *Define for* $B > 0$*:*

$$\mathcal{H}_B = \{h(\tilde{x}) : h(\tilde{x}) = \langle \Psi(\tilde{x}), w \rangle, \|w\|_2 \leqslant B\}. \tag{A.284}$$

*Then, if* $3 < k \leqslant D/16$ *and* $k$ *is odd, then there exists* $\mathcal{D} \in \mathcal{F}_\Xi$ *such that all* $h \in \mathcal{H}_B$ *have hinge-loss at least* $p_o\left(1 - \frac{\sqrt{2N}B}{2^k}\right)$.

*Proof of Theorem 2.2.* We first show that $\mathcal{F}_\Xi$ contains some distributions that are essentially sparse parity learning problems, and then we invoke the lower bound result from existing work for such problems.

Consider $\mathcal{D}$ defined as follows.

- Let $P = \{i \in [k] : i \text{ is odd}\}$. That is, if there are odd numbers of 1's in $\tilde{\phi}_\mathbf{A}$, then $y = +1$.
- Let $\mathcal{D}_{\tilde{\phi}}^{(0)}$ be a distribution where all entries $\tilde{\phi}_j$ are i.i.d. with $\Pr[\tilde{\phi}_j = 0] = \Pr[\tilde{\phi}_j = 1] = 1/2$. Let $\mathcal{D}^{(0)}$ be the distribution over $(\tilde{x}, y)$ induced by $\mathcal{D}_{\tilde{\phi}}^{(0)}$ and the above $P$.
- Let $\mathcal{D}_{\tilde{\phi}}^{(1)}$ be a distribution where all entries $\tilde{\phi}_j$ for $j \notin \mathbf{A}$ are i.i.d. with $\Pr[\tilde{\phi}_j = 1] = p_o/(2 - 2p_o)$, while $\Pr[\tilde{\phi}_\mathbf{A} = (0, 0, \ldots, 0)] = \Pr[\tilde{\phi}_\mathbf{A} = (1, 1, \ldots, 1)] = 1/2$. Let $\mathcal{D}^{(1)}$ be the distribution over $(\tilde{x}, y)$ induced by $\mathcal{D}_{\tilde{\phi}}^{(1)}$ and the above $P$.
- Let $\mathcal{D}_\mathbf{A}^{\text{mix}} = p_o\mathcal{D}^{(0)} + (1 - p_o)\mathcal{D}^{(1)}$.

It can be verified that such distributions are included in $\mathcal{F}_\Xi$ for $\gamma = \Theta(1)$.

Assume for contradiction that for all $\mathcal{D} \in \mathcal{F}_\Xi$, there exists $h^* \in \mathcal{H}_B$ such that $h = \langle \Psi, w^* \rangle$ loss smaller than $p_o \left(1 - \frac{\sqrt{2N}B}{2^k}\right)$. Then for all the distributions $\mathcal{D}_{\mathbf{A}}^{\text{mix}}$ defined above, we have

$$\mathbb{E}_{\mathcal{D}^{(0)}}[\ell(h^*(\tilde{x}), y)] < 1 - \frac{\sqrt{2N}B}{2^k}. \tag{A.285}$$

Now let $\mathcal{D}_z$ be a distribution over $z \in \{-1, +1\}^D$ with i.i.d. entries $z_j$ and $\Pr[z_j = -1] = \Pr[z_j = +1] = 1/2$. Let $f_{\mathbf{A}}(z) = \prod_{j \in \mathbf{A}} z_j$ be the k-sparse parity functions. Let $\Psi'(z) = \Psi(M(z+1)/2)$. Then we have $h'(z) = \langle \Psi'(z), w^* \rangle$ such that for all $\mathbf{A}$,

$$\mathbb{E}_{\mathcal{D}_z}[\ell(h'(z), f_{\mathbf{A}}(z))] < 1 - \frac{\sqrt{2N}B}{2^k}. \tag{A.286}$$

This is contradictory to Theorem A.23. □

The following theorem is implicit in the proof in Theorem 1 in Daniely and Malach (2020).

**Theorem A.23.** *For a subset $\mathbf{A} \subseteq [D]$ of size $k$, let the distribution $\mathcal{D}_{\mathbf{A}}$ over $(z, y)$ defined as follows: $z$ is uniform over $\{\pm 1\}^D$ and $y = \prod_{i \in \mathbf{A}} z_i$. Fix some $\Psi : \{\pm 1\}^D \to [-1, +1]^N$, and define:*

$$\mathcal{H}_\Psi^B = \{z \to \langle \Psi(z), w \rangle : \|w\|_2 \leqslant B\}.$$

*If $k$ is odd and $k \leqslant D/16$, then there exists some $\mathbf{A}$ such that*

$$\min_{h \in \mathcal{H}_\Psi^B} \mathbb{E}_{\mathcal{D}_{\mathbf{A}}}[\ell(h(z), y)] \geqslant 1 - \frac{\sqrt{2N}B}{2^k}.$$

We now prove the corollary.

**Corollary A.24** (Restatement of Corollary 2.3)**.** *For any function $f$ using a shift-invariant kernel $K$ with RKHS norm bounded by $L$, or $f(x) = \sum_i \alpha_i K(z_i, x)$ for some data points $z_i$ and $\|\alpha\|_2 \leqslant L$. If $3 < k \leqslant D/16$ and $k$ is odd, then there exists $\mathcal{D} \in \mathcal{F}_\Xi$ such that $f$ have hinge-loss at least $p_o(1 - \frac{\text{poly}(d,L)}{2^k}) - \frac{1}{\text{poly}(d,L)}$.*

*Proof.* By Claim 1 in Rahimi and Recht (2008), for any $\nu > 0$, there exists $N = \text{poly}(d, 1/\nu)$ Fourier features $\Psi_j$ that can approximate the shift-invariant kernel up to error $\nu$. For any $\epsilon > 0$, consider $\sum_i \alpha_i \langle \Psi(z_i), \Phi(x) \rangle = \langle \sum_i \alpha_i \Psi(z_i), \Psi(x) \rangle$. Let $w = \sum_i \alpha_i \Psi(z_i)$ and let $\nu = O(\frac{\epsilon}{L})$, then $\langle \Psi(x), w \rangle$ approximates $f(x)$ upto error $\epsilon$ and $N = \text{poly}(d, L, 1/\epsilon)$ and the norm of $w$ bounded by $B = \text{poly}(d, L, 1/\epsilon)$. The reasoning is the same for $f$ in the RKHS form, replacing sum with integral. By Theorem 2.2, $\langle \Psi(x), w \rangle$ has hinge-loss at least $p_0(1 - \frac{\sqrt{2}NB}{2^k})$. Thus, the function $f$ has loss at least $p_0(1 - \frac{\text{poly}(d,L,1/\epsilon)}{2^k}) - \epsilon$. Choose $\epsilon = \frac{1}{\text{poly}(d,L)}$, we get the bound. □

## A.6 Lower Bound for Learning without Input Structure

First recall the Statistical Query model (Kearns, 1998). In this model, the learning algorithm can only receive information about the data through statistical queries. A statistical query is specified by some property predicate $Q$ of labeled instances, and a tolerance parameter $\tau \in [0, 1]$. When the algorithm asks a statistical query $(Q, \tau)$, it receives a response $\hat{P}_Q \in [P_Q - \tau, P_Q + \tau]$, where $P_Q = \Pr[Q(x, y) \text{ is true}]$. $Q$ is also required to be polynomially computable, i.e., for any $(x, y)$ $Q(x, y)$ can be computed in polynomial time. Notice that a statistical query can be simulated by empirical average of a large random sample of data of size roughly $O(1/\tau^2)$ to assure the tolerance $\tau$ with high probability.

Blum et al. (1994) introduces the notion of Statistical Query dimension, which is convenient for our purpose.

**Definition A.25** (Definition 2 in Blum et al. (1994))**.** *For concept class* $C$ *and distribution* $\mathcal{D}$*, the statistical query dimension SQ-DIM*$(C, \mathcal{D})$ *is the largest number* $d$ *such that* $C$ *contains* $d$ *concepts* $c_1, \ldots, c_d$ *that are nearly pairwise uncorrelated: specifically, for all* $i \neq j$,

$$|\Pr_{x \sim \mathcal{D}}[c_i(x) = c_j(x)] - \Pr_{x \sim \mathcal{D}}[c_i(x) \neq c_j(x)]| \leqslant 1/d^3. \tag{A.287}$$

**Theorem A.26** (Theorem 12 in Blum et al. (1994))**.** *In order to learn* $\mathrm{C}$ *to error less than* $1/2 - 1/\mathrm{d}^3$ *in the Statistical Query model, where* $\mathrm{d} = \textit{SQ-DIM}(\mathrm{C}, \mathcal{D})$*, either the number of queries or* $1/\tau$ *must be at least* $\frac{1}{2}\mathrm{d}^{1/3}$*.*

We now use the above tools to prove our lower bound.

**Theorem A.27** (Restatement of Theorem 2.4)**.** *For any algorithm in the Statistical Query model that can learn over* $\mathcal{F}_{\Xi_0}$ *to error less than* $\frac{1}{2} - \frac{1}{\binom{D}{k}^3}$*, either the number of queries or* $1/\tau$ *must be at least* $\frac{1}{2}\binom{D}{k}^{1/3}$*.*

*Proof of Theorem 2.4.* Consider the following concept class and marginal distribution:

- Let $\mathcal{D}$ be the distribution over $\tilde{x}$, given by $\tilde{x} = M\tilde{\phi}$ and $\tilde{\phi}_j$ are i.i.d. with $\Pr[\tilde{\phi}_j = 0] = \Pr[\tilde{\phi}_j = 1] = 1/2$.
- Let C be the class of functions $y = g_{\mathbf{A}}(\tilde{\phi}) = \mathbb{I}[\sum_j (1 - \tilde{\phi}_j)$ is odd$]$ for different $\mathbf{A} \subseteq [D]$.

The distributions over $(\tilde{x}, y)$ induced by $(C, \mathcal{D})$ are a subset of $\mathcal{F}_{\Xi_0}$. It is then sufficient to show that SQ-DIM$(C, \mathcal{D}) \geqslant \binom{D}{k}$.

It is easy to see that C are essentially the sparse parity functions: if $z_j = 2\tilde{\phi}_j - 1$, then $g_{\mathbf{A}}(\tilde{\phi}) = \prod_{j \in \mathbf{A}} z_j$. This then implies that the $g_{\mathbf{A}}$'s are uncorrelated, so SQ-DIM$(C, \mathcal{D}) \geqslant \binom{D}{k}$. □

## A.7 Complete Experimental Results

Our experiments mainly focus on feature learning and the effect of the input structure. We first perform simulations on our learning problems to (1) verify our main theorems on the benefit of feature learning and the effect of input structure (2) verify our analysis of feature learning in networks. We then check if our insights carry over to real data: (3) whether similar feature learning is presented in real network/data; (4) whether damaging the input structure lowers the performance.

The results are consistent with our analysis and provide positive support for the theory.

The experiments were ran 5 times with different random seeds, and the average results (accuracy) are reported. The standard deviations of the results are smaller than 0.5% and thus we do not present them for clarity. The hardware specifications are 4 Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz, 16 GB RAM, and one NVIDIA GPU GTX1080.

### A.7.1 Simulation

We train a two-layer network following our learning process. We use two fixed feature methods: the NTK (Fang et al., 2021) and random feature (RF) methods based on the same network and random initialization as the network learning. More precisely, in the NTK method, we randomly initialize the network and take its NTK and learn a classifier on it. In the RF method, we freeze the first layer of the network, and train the second layer (on the random features given by the frozen neurons). The training step number is the same as that in network learning. We also test these three methods on the data distribution with input structure removed (i.e., $\mathcal{F}_{\Xi_0}$ in Theorem 2.4). For comparison, we take the representation of our two-layer network at step one/step two, named One Step/Two Step (fix the weight of the 1st layer after the first step/second step to train the weight of the second layer), and train the best classifiers on top of them.

Recall that our analysis is on the *directions* of the weights without considering their *scaling*, and thus it is important to choose cosine similarity rather than the typical $\ell_2$ distance. Thus, we use metric Cos Similarity $\max_{\{i \in [2m]\}} \cos(\mathbf{w}_i, \sum_{j \in \mathbf{A}} M_j)$ in our tables, and use Multidimensional Scaling to plot the weights distribution. The simulation dataset size is 50000. During training, the batch size is 1000, while for the first two steps we use the approximate full gradient (batch size is 50000). Each step is corresponding to one weights update.

### Parity Labeling

**Setting.** We generate data according to the parity function data distributions used in our proof of the lower bound for fixed features (Theorem 2.2), with $d = 500, D = 100, k = 5, p_o = 1/2$, with a randomly sampled $\mathbf{A}$. More precisely, we consider $\mathcal{D}$ defined as follows.

- Let $P = \{i \in [k] : i \text{ is odd}\}$. That is, if there are odd numbers of $1'$s in $\tilde{\phi}_{\mathbf{A}}$, then $y = +1$.
- Let $\mathcal{D}^{(0)}_{\tilde{\phi}}$ be a distribution where all entries $\tilde{\phi}_j$ are i.i.d. with $\Pr[\tilde{\phi}_j = 0] = \Pr[\tilde{\phi}_j = 1] = 1/2$. Let $\mathcal{D}^{(0)}$ be the distribution over $(\tilde{x}, y)$ induced by $\mathcal{D}^{(0)}_{\tilde{\phi}}$ and the above $P$.
- Let $\mathcal{D}^{(1)}_{\tilde{\phi}}$ be a distribution where all entries $\tilde{\phi}_j$ for $j \notin \mathbf{A}$ are i.i.d. with $\Pr[\tilde{\phi}_j = 1] = p_o/(2 - 2p_o)$, while $\Pr[\tilde{\phi}_{\mathbf{A}} = (0, 0, \dots, 0)] = \Pr[\tilde{\phi}_{\mathbf{A}} = (1, 1, \dots, 1)] = 1/2$. Let $\mathcal{D}^{(1)}$ be the distribution over $(\tilde{x}, y)$ induced by $\mathcal{D}^{(1)}_{\tilde{\phi}}$ and the above $P$.
- Let $\mathcal{D}^{\text{mix}}_{\mathbf{A}} = p_o\mathcal{D}^{(0)} + (1 - p_o)\mathcal{D}^{(1)}$.

The network and the training follow Section 2.3, where the network size is $m = 300$ and the training time $T = 600$ steps.

| Model | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc (%) | 100.0 | 84.0 | 74.7 | 51.3 | 100.0 | 100.0 |
| Test Acc (%) | 100.0 | 86.4 | 76.0 | 52.2 | 100.0 | 52.0 |
| Cos Similarity | 0.997 | NA | 0.114 | 0.848 | 0.997 | 0.253 |

Table A.1: Parity labeling results in six methods. The cosine similarity is computed between the ground-truth $\sum_{j \in \mathbf{A}} M_j$ and the closest neuron weight.

**Verification of the Main Results.** Figure A.1 shows that the results are consistent with our analysis. Network learning gets high test accuracy while the two fixed feature methods get significantly lower accuracy. Furthermore, when the input structure is removed, all three methods get test accuracy similar to random guessing.
**Feature Learning in Networks.** Figure A.2 shows that the results are as predicted by our analysis. After the first gradient step, some weights begin to cluster around

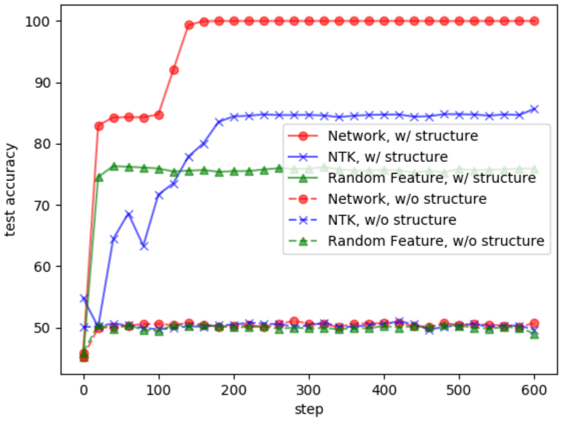


Figure A.1: Test accuracy on simulated data under parity labeling with or without input structure.

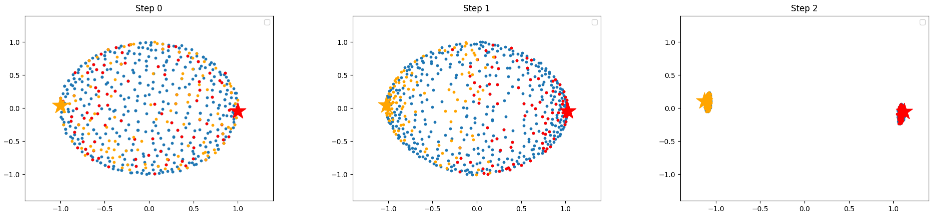


Figure A.2: Visualization of the weights $\mathbf{w}_i$'s after initialization/one gradient step/two gradient steps in network learning under parity labeling. The red star denotes the ground-truth $\sum_{j\in \mathbf{A}} M_j$; the orange star is $-\sum_{j\in \mathbf{A}} M_j$. The red dots are the weights closest to the red star after two steps; the orange ones are for the orange star.

the ground-truth $\sum_{j\in \mathbf{A}} M_j$ (or $-\sum_{j\in \mathbf{A}} M_j$ due to we have $a_i$ in the gradient update which can be positive or negative). After the second step the weights get improved and well-aligned with the ground-truth (with cosine similarity $> 0.99$).

Table A.1 shows the results for different methods. Recall that the Cos Similarity metric is $\max_{\{i\in[2m]\}} \cos(\mathbf{w}_i, \sum_{j\in \mathbf{A}} M_j)$, which reports the cosine value of the closest one. One Step refers to the method where we take the neurons after one gradient step, freeze their weights, and train a classifier on top; similar for Two Step. One

Step gets test accuracy about 52%, while Two Step gets accuracy about 100%. This demonstrates that while some effective feature emerge in the first step, they need to be improved in the second step for accurate prediction. NTK, random feature, One Step all failed, while Network and Two Step can achieve 100% test accuracy. Network w/o structure refers to training the network on data without the input structure. It overfits the training dataset with 52% test accuracy.

### Interval Labeling

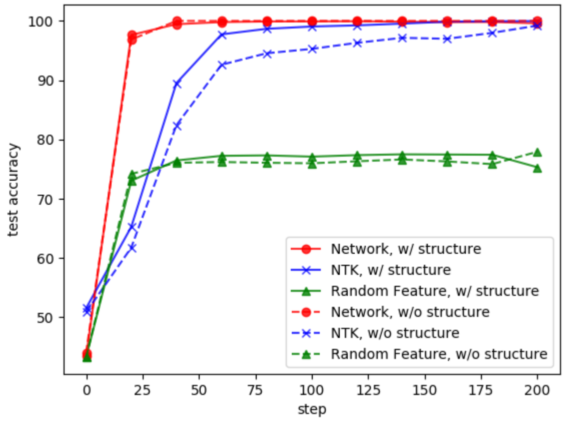


Figure A.3: Test accuracy on simulated data under interval labeling with or without input structure.

| Model | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc (%) | 100.0 | 100.0 | 76.4 | 44.1 | 100.0 | 100.0 |
| Test Acc (%) | 100.0 | 100.0 | 73.2 | 41.0 | 100.0 | 100.0 |
| Cos Similarity | 1.00 | NA | 0.153 | 0.901 | 0.994 | 0.965 |

Table A.2: Interval labeling results in six methods.

**Setting.** We also tried interval function, where $y = 1$ if $\sum_{i \in \mathbf{A}} \tilde{\phi}_i$ is in the range $[t_1, t_2]$ with $t_1 = 20$ and $t_2 = 30$, otherwise $y = -1$. We use $d = 500, D = 100, k = 30$. The $\tilde{\phi}_i$'s are independent, and $\Pr[\tilde{\phi}_i = 1] = 2/3$ for any $i \in A$, and $\Pr[\tilde{\phi}_i = 1] = 1/2$ otherwise. When the input structure is removed, we set $\Pr[\tilde{\phi}_i = 1] = 1/2$ for all $i$'s.

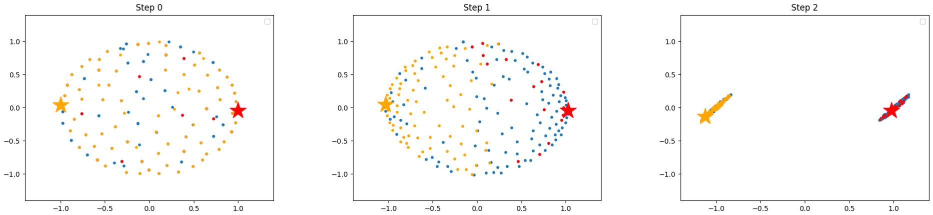


Figure A.4: Visualization of the weights $\mathbf{w}_i$'s after initialization/one gradient step/two gradient steps in network learning under interval labeling. The red star denotes the ground-truth $\sum_{j \in \mathbf{A}} M_j$; the orange star is $-\sum_{j \in \mathbf{A}} M_j$. The red dots are the weights closest to the red star after two steps; the orange ones are for the orange star.

The network and training again follows Section 2.3 with a network size $m = 100$ and the training time $T = 200$ steps.

**Verification of the Main Results.** Figure A.3 shows that network learning learns the fastest, NTK learns slower but reaches similar test accuracy, while random feature can only reach a decent but lower accuracy. This is because for such simpler labeling functions, fixed feature methods can still achieve good performance (note that the lower bound does not hold for such a case), while the performance depends on what fixed features to use.

Furthermore, when the input structure is removed, the methods still get similar (or only slightly worse) performance as with input structure. This shows that when the labeling function is simple, the help of the input structure for learning may not be needed. In the experiments on real data, we will show that when the input structure is changed, it indeed leads to lower performance which suggests that the labeling function in practice is typically more complicated than this interval labeling setting, and the help of the input structure is significant for learning.

**Feature Learning in Networks.** Figure A.4 shows the phenomenon of feature learning similar to that in the parity labeling setting. Table A.2 shows the test accuracy of six different methods. Random feature and One Step failed, while Network, NTK and Two Step succeed showing that interval labeling setting is a simpler case than parity labeling setting.

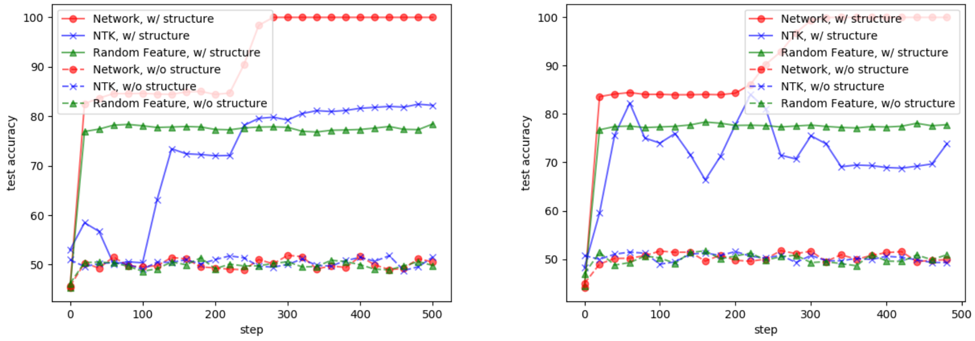
## A.7.2 More Simulation Result in Various Settings

We show the robustness of our simulation results by studying the learning behaviors in a variety of settings including different sample size, input data dimension and class imbalance. We reuse the same setting as the simulation in the main text (details in A.7.1), vary different parameters, and report the accuracy, the cosine similarities between the learned weights, and the visualization of the neuron weights.

### Varying Input Data Dimension

In the simulation experiments in the main text, the input data dimension d is 500. Here we change the input data dimension to 100 and 2000. All other configurations follow A.7.1.

**Verification of the Main Results.** Figure A.5 shows that our claim is robust under different input data dimensions. The performance of network learning is superior over NTK and random feature approaches on inputs with structure, and on inputs without structure, all three methods fail.



(a) d = 100 (b) d = 2000

Figure A.5: Test accuracy on simulated data under different input data dimensions.

**Feature Learning in Networks.** Figure A.6 visualizes the neuron weights. It shows similar results to that in A.7.1: the weights gets updated to to the effective feature in the first two steps, forming clusters. Table A.3 shows some quantitative results. In

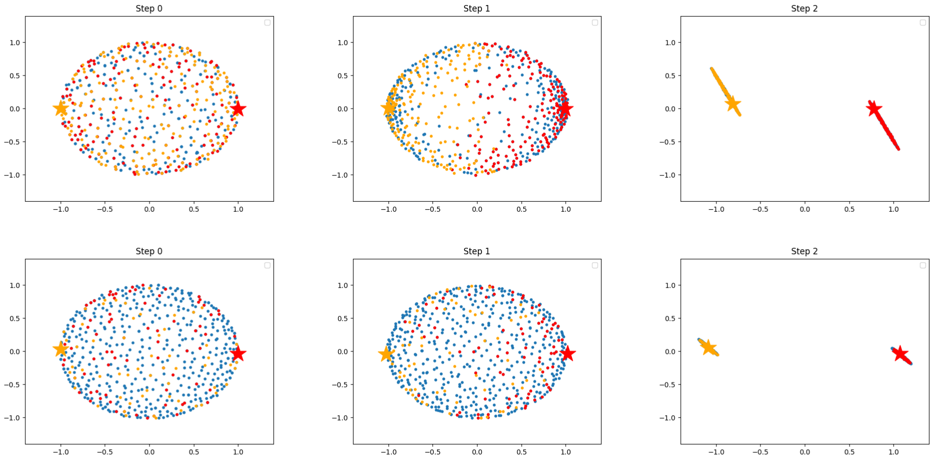


Figure A.6: Visualization of the weights $\mathbf{w}_i$'s in early steps under different input data dimensions. Upper row: input data dimension $d = 100$; lower row: $d = 2000$.

| $d = 100$ | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc | 100.0 | 83.1 | 78.9 | 53.0 | 100.0 | 100.0 |
| Test Acc | 100.0 | 81.5 | 78.3 | 51.1 | 100.0 | 51.0 |
| Cos Similarity | 1.000 | NA | 0.354 | 0.967 | 1.000 | 0.331 |

| $d = 2000$ | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc | 100.0 | 75.6 | 80.0 | 50.22 | 100.0 | 100.0 |
| Test Acc | 100.0 | 75.4 | 77.0 | 50.01 | 100.0 | 52.5 |
| Cos Similarity | 0.998 | NA | 0.056 | 0.560 | 0.998 | 0.309 |

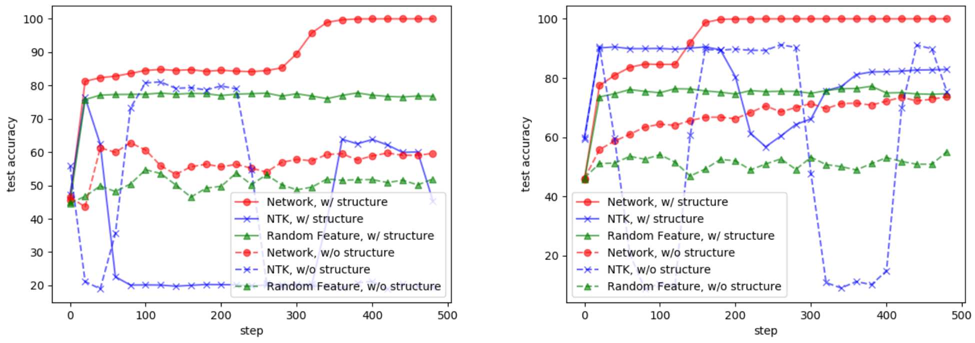
Table A.3: Results of six methods for different input data dimensions. The cosine similarity is computed between the ground-truth $\sum_{j \in \mathbf{A}} M_j$ and the closest neuron weight.

particular, the average cosine similarities between neuron weights and the effective features after two steps are close to 1, showing that they match the effective features.

### Varying Class Imbalance Ratio

The experiments in the main text has 25000 training samples for each class. Here we keep the total sample size 50000 but use different class imbalance ratios, which is the class $-1$ sample size divide by the total sample size.

**Verification of the Main Results.** Figure A.7 shows that our claim is robust under different class imbalance ratios. The results are similar to those for balanced classes, except that NTK becomes less stable.



(a) Negative class ratio $= 0.8$  (b) Negative class ratio $= 0.9$

Figure A.7: Test accuracy on simulated data under different negative class ratios.

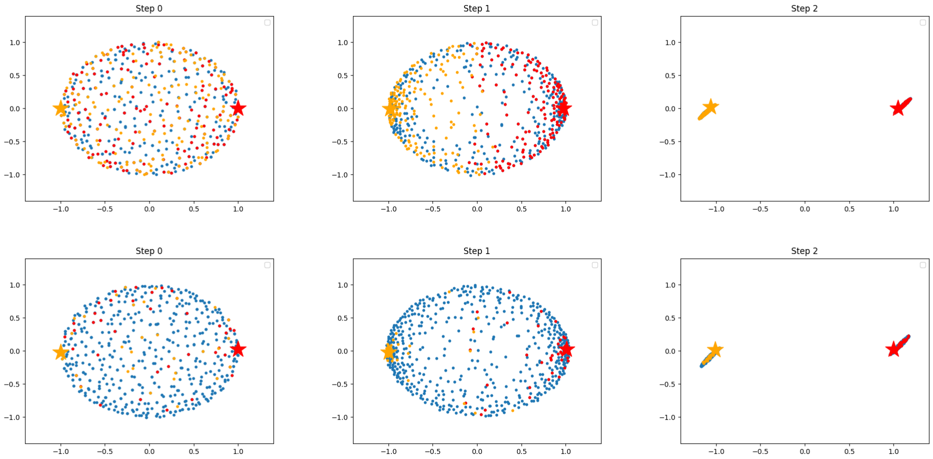


Figure A.8: Visualization of the weights $\mathbf{w}_i$'s in early steps under different class imbalance ratios. Upper row: negative class ratio 0.8; lower row: 0.9.

**Feature Learning in Networks.** Figure A.8 visualizes the neurons' weights. Again, the observation is similar to that for balanced classes. Table A.4 shows some quantitative results which are also similar to those for balanced classes.

| ratio = 0.8 | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc | 100.0 | 62.9 | 72.7 | 78.3 | 100.0 | 100.0 |
| Test Acc | 100.0 | 82.7 | 70.4 | 75.7 | 100.0 | 61.7 |
| Cos Similarity | 0.999 | NA | 0.293 | 0.950 | 0.999 | 0.218 |

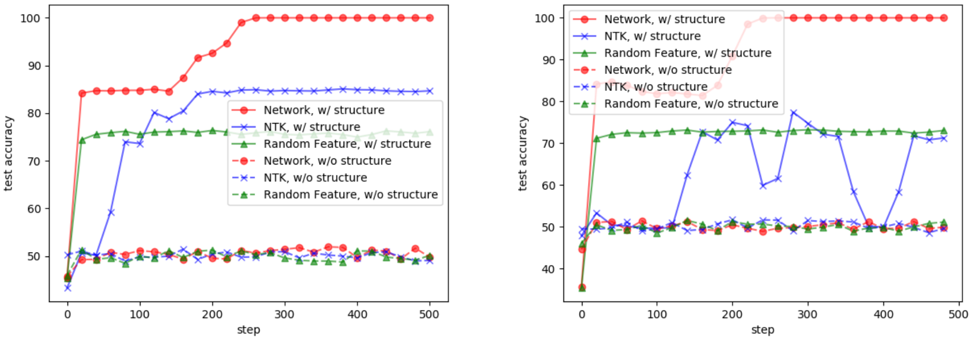
| ratio = 0.9 | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc | 100.0 | 84.0 | 73.6 | 92.3 | 100.0 | 100.0 |
| Test Acc | 100.0 | 81.7 | 72.4 | 89.2 | 100.0 | 71.8 |
| Cos Similarity | 0.997 | NA | 0.296 | 0.956 | 0.997 | 0.286 |

Table A.4: Results of six methods under different negative class ratios.

### Varying Sample Size

Here we change the sample size 50000 in Section A.7.1 to be 25000 and 10000. For sample size 25000, we observe similar results. For sample size 10000, we observe over-fitting (test accuracy much lower than train accuracy). Therefore, for sample size 10000 we reduces the size of the network (i.e., number of hidden neurons) from $m = 300$ to $m = 50$.

**Verification of the Main Results.** Figure A.9 shows that our claim is robust under different sample sizes. In particular, the network learning still outperforms the NTK and random feature approaches on structured inputs.



(a) $n = 25000$

(b) $n = 10000$

Figure A.9: Test accuracy on simulated data under different sample sizes $n$.

**Feature Learning in Networks.** Figure A.10 and Table A.5 show that the phenomenon of feature learning for different samples is similar to that in A.7.1.

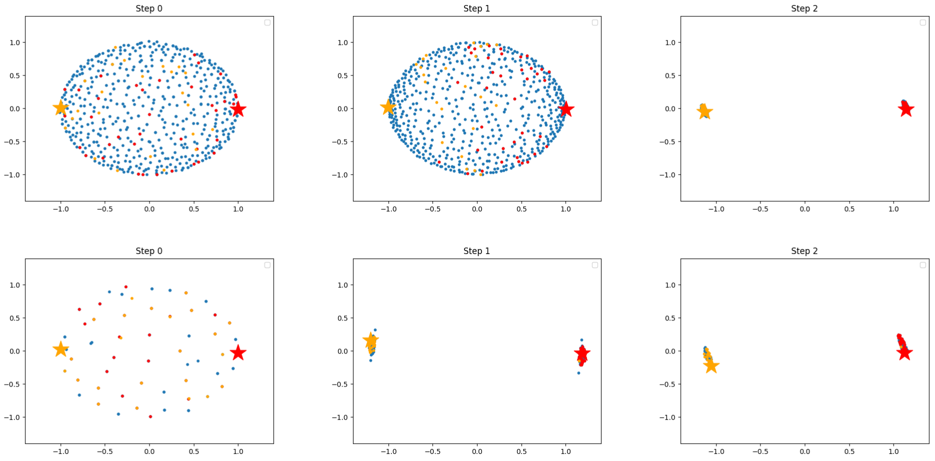


Figure A.10: Visualization of the weights $\mathbf{w}_i$'s in early steps under different sample sizes. Upper row: sample size 25000; lower row: 10000.

| $n = 25000$ | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc | 100.0 | 84.0 | 78.6 | 50.6 | 100.0 | 100 |
| Test Acc | 100.0 | 84.1 | 74.7 | 50.0 | 100.0 | 50.2 |
| Cos Similarity | 0.997 | NA | 0.105 | 0.851 | 0.997 | 0.230 |

| $n = 10000$ | Network | NTK | RF | One Step | Two Step | Network w/o structure |
|---|---|---|---|---|---|---|
| Train Acc | 100.0 | 73.9 | 71.6 | 50.7 | 100.0 | 100.0 |
| Test Acc | 100.0 | 75.0 | 74.3 | 50.3 | 100.0 | 52.2 |
| Cos Similarity | 0.995 | NA | 0.096 | 0.974 | 0.994 | 0.176 |

Table A.5: Results of six methods for different sample size.

### A.7.3 Experiments on More Data Generation Models

In this section we consider some additional data distributions and run the simulation experiments, in particular, focusing on the feature learning phenomenon. Note that our analysis is for the setting where the input distributions have structure revealing some information about the labeling function. (More precisely, the labeling function is specified by $\mathbf{A}$ and $P$, while the input distribution also depends on them.) We therefore consider two other data generation mechanisms where the labeling function also has connections to the input distributions.

### Hidden Representation Labeling

Here we consider the following data model: first uniformly at random select $\tilde{\phi}_{\mathbf{A}}$ from a set of binary vectors, and assign label 1 to some and -1 to others; sample irrelevant patterns $\tilde{\phi}_{-\mathbf{A}}$ uniformly at random; generate the input $\mathbf{x} = M\tilde{\phi}$. We randomly select 50 binary vectors for each label, with $d = 500, D = 250, k = 50, p_o = 1/2$.

This is a generalization of the distribution $\mathcal{D}^{(1)}$, a component in the distribution of our simulation experiments (see the proof of Theorem 2.2 for details). Recall the definition of $\mathcal{D}^{(1)}$: $\tilde{\phi}_A$ is uniform on only two values $[+1, \dots, +1]$ and $[0, \dots, 0]$, and uniform over irrelevant patterns; the value $[+1, \dots, +1]$ corresponds to one class and $[0, \dots, 0]$ correspond to another class. Our data model here generalizes $\mathcal{D}^{(1)}$ to more than 2 values.

The visualization is shown in Figure A.11. We can observe similar feature learning phenomena, and the neuron weights are updated to form clusters.

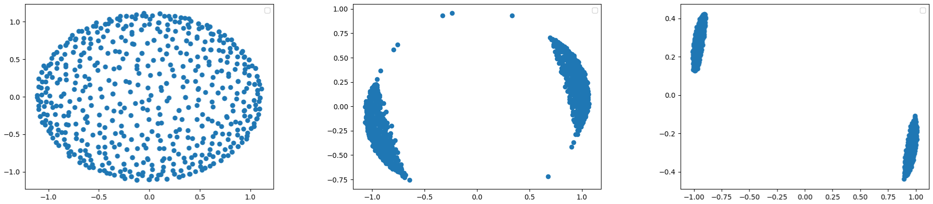


Figure A.11: Visualization of the weights $\mathbf{w}_i$'s after initialization/one gradient step/two gradient steps in network learning under hidden representation labeling.

### Two-layer Networks on Mixture of Gaussians

To further support our intuition of feature learning, we run experiments on mixture of Gaussians.

**Data.** Let $\mathbf{X} = \mathbb{R}^d$ be the input space, and $\mathcal{Y} = \{\pm 1\}$ be the label space. Suppose $M \in \mathbb{R}^{d \times k}$ is an dictionary with $k$ orthonormal columns. Let $\varepsilon_i, i = 1, \dots, k$ be i.i.d symmetric Bernoulli random variables, and $g \sim \mathcal{N}(0, \sigma_r^2 \frac{k}{d} \mathbb{I}_d)$. Then we generate the input $x$ and class label $y$ by:

$$\mathbf{x} = \sum_{i=1}^{k} \varepsilon_i M_{:i} + g, \quad y = \prod_{i=1}^{k} \varepsilon_i \tag{A.288}$$

In this case, $2^k$ Gaussian clusters will be created. The centers of the Gaussian clusters $\sum_{i=1}^{k} \pm M_{:i}$ lie on the vertices of a hyper cube, and the label of each Gaussian cluster is determined by the parity function on the vertices of the hyper cube.

Note that the labeling function is roughly equivalent to a network:

$$y = \sum_{i=1}^{n} a_i \mathrm{ReLU}(\langle c_i, \mathbf{x} \rangle)$$

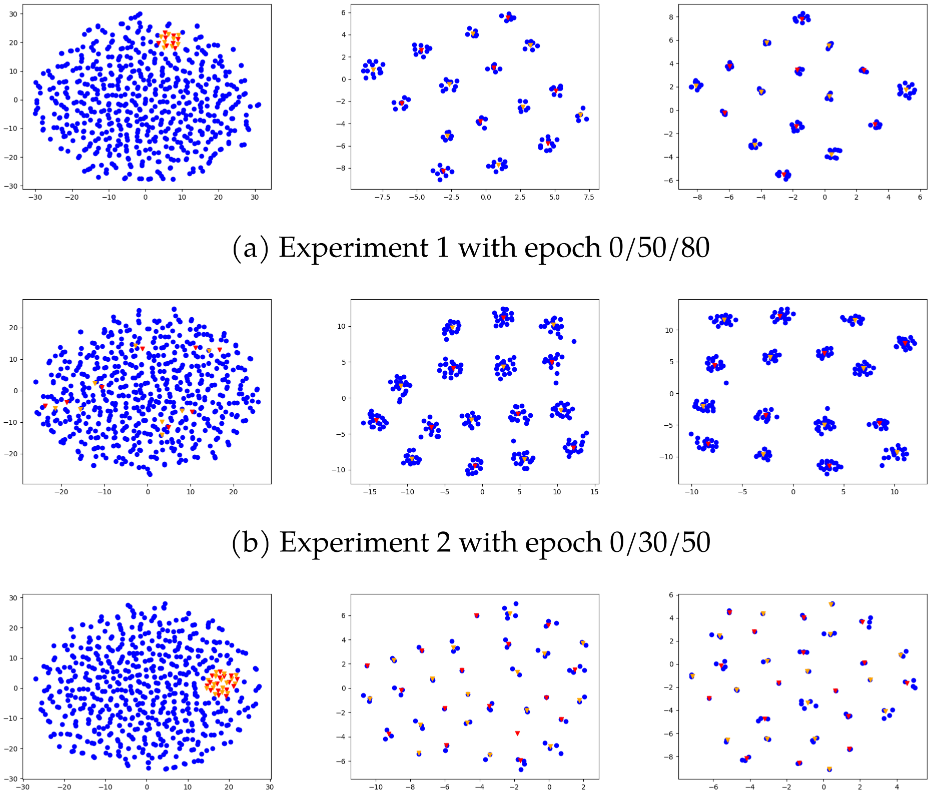
where $c_i$'s are the Gaussian centers, and $a_i \propto 1$ for Gaussian components with label 1 and $a_i \propto -1$ for those with label -1.

**Setting.** We then train a two-layer network with $m = 800$ hidden neurons on data sets generated as above with different chosen $k$'s and $d$'s. The training follows typical practice (not the hyperparameters in our analysis). In this setting, we expect the neural network to learn the effective features: the directions of Gaussian cluster centers.

**Result.** We run experiments with different settings. The parameters are shown in Table A.6. From Figure A.12 we can see that some neurons learn the directions of Gaussian centers, and each Gaussian center is covered by some neurons, which matches our expectation.

| Parameters | $d$ | $k$ | Number of Clusters | $\sigma_r$ |
|---|---|---|---|---|
| Experiment 1 | 100 | 4 | 16 | 1 |
| Experiment 2 | 25 | 4 | 16 | 0.7 |
| Experiment 3 | 100 | 5 | 32 | 1 |

Table A.6: Gaussian mixture setting.



(a) Experiment 1 with epoch 0/50/80

(b) Experiment 2 with epoch 0/30/50

(c) Experiment 3 with epoch 0/50/80

Figure A.12: Visualization of the weights $w_i$'s (blue dots) and Gaussian centers (red for positive labeled clusters and orange for negative labeled clusters).

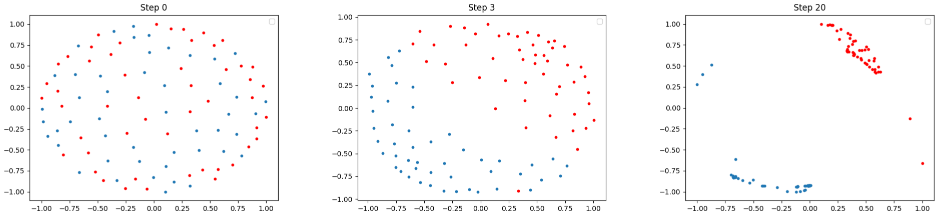


Figure A.13: Visualization of the neurons' weights in a two-layer network trained on the subset of MNIST data with label 0/1. The weights gradually form two clusters.

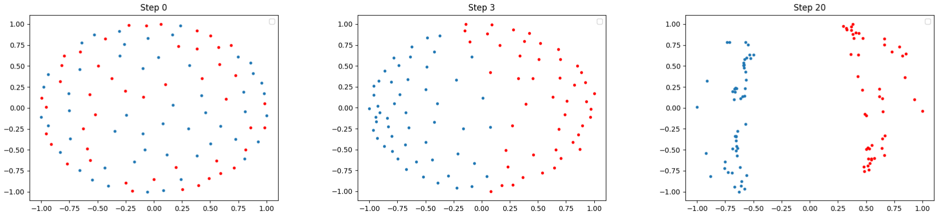


Figure A.14: Visualization of the neurons' weights in a two-layer network trained on the subset of CIFAR10 data with label airplane/automobile. The weights gradually form two clusters.

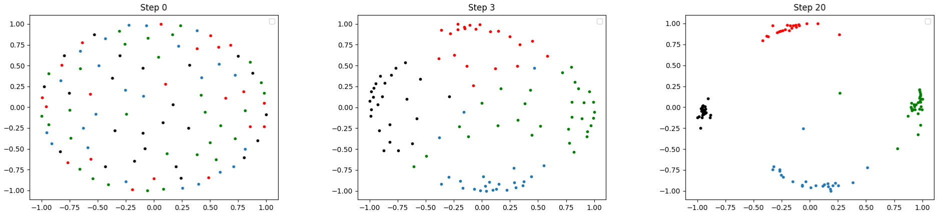


Figure A.15: Visualization of the neurons' weights in a two-layer network trained on the subset of SVHN data with label 0/1. The weights gradually form four clusters.

### A.7.4 Real Data: Feature Learning in Networks

We take the subset of MNIST (Deng, 2012) with labels 0/1, CIFAR10 (Krizhevsky, 2012) with labels airplane/automobile and SVHN (Netzer et al., 2011) with labels 0/1, and train a two-layer network with $m = 50$. We use traditional weight initialization method (random Gaussian) and training method (SGD with momentum $= 0.95$ without regularization) in this section, for our purpose of investigating the training dynamics in practice.

Then we visualize the neurons' weights following the same method in the simulation. Figure A.13, Figure A.14 and Figure A.15 show a similar feature learning phenomenon: effective features emerge after a few steps, and then get improved to form clusters. This shows the insights obtained on our learning problems are also applicable to the real data.

| | $\cos(\nu_1, \bar{\nu})$ | $\cos(\nu_2, \bar{\nu})$ | $\cos(\nu_3, \bar{\nu})$ | $\cos(\nu_1, \nu_2)$ | $\cos(\nu_1, \nu_3)$ | $\cos(\nu_2, \nu_3)$ |
|---|---|---|---|---|---|---|
| ResNet(128) | 0.9727 | 0.8655 | 0.6549 | 0.7454 | 0.5083 | 0.6533 |
| ResNet(256) | 0.8646 | 0.9665 | 0.9121 | 0.7087 | 0.6919 | 0.9135 |

Table A.7: Cosine similarities between the gradients in the early steps. We choose the neuron weight closest to the average weight of the green cluster at the end of the training (in Figure A.16 for ResNet(128) and Figure A.17 for ResNet(256)). We record the gradients of the first 30 steps and divide them to three trunks of 10 steps evenly and sequentially. For the three trunks, we get the average gradients $\nu_1, \nu_2, \nu_3$. We calculate their cosine similarities to their average $\bar{\nu} = (\nu_1 + \nu_2 + \nu_3)/3$ and those between them.

### CNNs on Binary Cifar10: Feature Learning in Networks

**Setting.** We use ResNet$(m)$, which is a ResNet-18 convolutional neural network (He et al., 2016) with $m$ filters in the first residual block. It is obtained by scaling the number of filters in each block proportionally from the standard ResNet-18 network which is ResNet(64). We use ResNet(128) and ResNet(256) in this experiment. We train our model on Binary CIFAR10 (Krizhevsky, 2012) with labels airplane/automobile for 20 epochs. The final test accuracy of ResNet(128) is 95.75% and that of ResNet(256) is 93.8%.

**Results.** Figure A.16 visualizes the filters' weights of different residual blocks in ResNet(128) at Epoch 0, 3, and 20, and Figure A.17 shows those in ResNet(256). They show that feature learning happens in the early stage, and show that there are some clusters of weights (e.g., the red and green points). These colored points are selected at Epoch 20. We first visualize the weights at Epoch 20, and then hand pick the points that roughly form two clusters (i.e., the points in the same cluster are close to each other while those in different clusters are far away). We assign red and green colors to the two clusters at Epoch 20, and then assign these weights with the same color in Epoch 0 and 3. Finally, we compute the cosine similarities and show that the hand picked points are indeed roughly clusters in the high-dimension.

In particular, we have the following three observations.

First, we can see that the filter weights change significantly during the early stage of the training, indicating feature learning happens in the early stage: the change between Epoch 0 and Epoch 3 is much more significant than that between Epoch 3 and Epoch 20.

Second, we can also verify that the feature learning is guided by the gradients: the gradients of a filter in the early gradient steps point to similar directions (and thus the updated filter will learn this direction). More precisely, for a selected filter, we average the gradients every 10 gradient steps (so to reduce the variance due to mini-batch), and get $v_1, v_2$ and $v_3$ for the first 30 steps and compute their cosine similarities and those to their average. Table A.7 shows the results. In general the similarities are high indicating they point to similar directions. (Note that a similarity of 0.6 is regarded as very significant as the filters are in a high dimension of $3 \times 3 \times 1024 = 9216$).

Third, we also observe some clustering effect of the filter weights, though not as significant as in our simulations. For example, in the red and green clusters in Figure A.16(a) for the first residual block, the average cosine similarity for filter weights in the red cluster is about 0.62 and that for the green is about 0.7, while the cosine similarity between the two clusters' centers is about -0.72. This shows significant similarities within the cluster while difference between clusters.

Note that the clustering is less significant than our simulation experiments. This is because practical data have more patterns (i.e., effective feature directions) to be learned than our synthetic data, and also the practical network is not as overparameterized as in our simulation. Then filters are likely to learn different patterns (or their mixtures) without forming significant clusters. The results of ResNet(256) show more significant clustering than ResNet(128), which supports our explanation. On the other hand, we emphasize that the key insight of our analysis is that the gradient guides the learning of effective features in the early stage of training (rather than the clustering), which is verified as discussed above.

### A.7.5 Real Data: The Effect of Input Structure

To study the influence of the input structure, we propose to keep the labeling function unchanged, vary the input distributions, and exam the change of the loss surface and the training dynamics. We first describe the detailed experimental methodology, which allows us to generate data with similar labeling function but different input distributions. Then we perform experiments on the generated datasets to investigate the change of the learning due to the change in the input distributions, and present the experimental results. Finally, we also perform experiments to verify the intuition behind our experimental method.

### Experimental Methodology

We consider the following experimental method. Given an original dataset $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^{n}$ (e.g., CIFAR10) and an unlabeled dataset $\mathcal{U} = \{\tilde{x}_i\}_{i=1}^{m}$ from a proposed distribution $P_{\mathcal{U}}$ (e.g., Gaussians), first extend the labeling function of $\mathcal{L}$ to $\mathcal{U}$, giving synthetic labels $\tilde{y}_i$ to $\tilde{x}_i$. Then train a neural network on the union of $\mathcal{L}$ and the synthetic data $\mathcal{L}_{\mathcal{U}} = \{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^{m}$. By investigating the new training dynamics, in particular the difference on the original part $\mathcal{L}$ and the synthetic part $\mathcal{L}_{\mathcal{U}}$, we can see the effect of the input structure. The original dataset should be from real-world data, since one of our goals is to compare them with synthetic data, and identify the properties of real-world data important for the success of learning.

A natural idea is to first learn a powerful network $f(x)$ (called the teacher) on $\mathcal{L}$ to approximate the true labeling function, then apply $f$ on $\mathcal{U}$ to generate synthetic labels, and finally train another network (called the student) on the synthetic data and original data. However, we found that naïvely implementation of this idea fails miserably: the support of $\mathcal{L}$ and $\mathcal{U}$ can be typically different, and the powerful network learned over $\mathcal{L}$ can have entirely different behavior on $\mathcal{U}$. Therefore, we need to control the size of the teacher $f$ so that the labeling on $\mathcal{U}$ has similar complexity as that on $\mathcal{L}$. For our purpose, we can define the complexity of the labeling on $\mathcal{L}$ as the minimum size of the teacher achieving an approximation error $\epsilon$ for a chosen $\epsilon$, if the ground-truth data distribution of $\mathcal{L}$ is known. However, given only limited data, we cannot faithfully estimate the needed size of the teacher, and need to take into account the variance introduced by the finite data.

Our key idea is to use the U-shaped curve of the bias-variance trade-off and select the size of the teacher at the minimum of the U-shaped curve. Since recent works (Belkin et al., 2019; Nakkiran et al., 2020) show that neural networks can have a double descent curve for the error v.s. model complexity, we thus plot the double descent curve, and find the minimum in the classical regime (corresponding to the traditional U-shape curve).

Our method is designed based on the following two reasons. First, on the U-shaped curve, the complexity of the network is still roughly controlled by that of the number of parameters. The local minimum of the U-shaped curve is a good measurement of the complexity of the data. If the ground-truth is much more complicated than the teacher, then increasing the teacher's size leads to a significant decrease in the approximation error (bias) compared to a small increase in the variance, that is, we will be on the left-hand side of the U-shaped. In contrast, on the right-hand side of the U-shaped, increasing the teacher's size leads to a small decrease in the bias compared to a significant increase in the variance. That is, the complexity of the ground-truth is comparable to or lower than the teacher. So the local minimum approximates the complexity of the ground-truth labeling function.

Second, the local minimum point is chosen to get the best approximation of the true labels. This helps to maintain the labeling from the real-world data and thus helps our investigation on the input, since too drastic change in the labeling can affect the training.

We note that the method is not perfect. First, the teacher at the local minimum of U-shape may not have very high accuracy, especially on more complicated data. To alleviate this, we also use the teacher to give synthetic labels $y_i'$ to $x_i$ in $\mathcal{L}$, and train the student network on $\mathcal{L}' = \{(x_i, y_i')\}_{i=1}^n$. Though this introduces some differences from the original labels, it is acceptable for our purpose of studying the inputs. Furthermore, ensuring the consistency of the labels on the original input in $\mathcal{L}$ and $\mathcal{U}$ is important in our experiments. Second, the measurement is an approximation due to variance. Since only limited labeled data is available, it's important and necessary to calibrate the measurement w.r.t. the level of variance on the given dataset.

**Method Description.** Algorithm 3 presents the details. For a fixed network architecture for the teacher $f$, it first varies the network size and plots the double descent curve. Then it selects the local minimum in the classic regime of U-shape and trains the teacher with the corresponding size. In practice, we observed that the teacher might have unbalanced probabilities for different classes on $\mathcal{U}$ if its training does not take into account $\mathcal{U}$. Therefore, we propose the following heuristic regularization using $x \in \mathcal{U}$, where $\lambda$ is a regularization weight, and $f(x)$ is the probabilities over classes given by the teacher:

$$R(x) = R_1(x) + \lambda R_2(x) \tag{A.289}$$

$$R_1(x) = \sum_j \left( \frac{\sum_i f(x)_j}{m} \ln \frac{\sum_i f(x)_j}{m} \right) \tag{A.290}$$

$$R_2(x) = -\frac{1}{m} \sum_i \sum_j (f(x)_j \ln(f(x)_j)). \tag{A.291}$$

Here, $R_1(x)$ guarantees that each kind of label has the same average probability to be generated, and $R_2(x)$ pushes the probability away from uniform to avoid the case that the class probabilities for each data point converge to uniform.

**Algorithm 3** Learning the teacher network to generate synthetic labels for studying the effect of the input structure

```
**Input:** teacher architecture $f$, labeled dataset $\mathcal{L} = \{(x_i, y_i)\}_{i=1}^n$, unlabeled dataset $\mathcal{U} = \{\tilde{x}_i\}_{i=1}^m$.
Let $i$ to be the size of $f$, $f_i$ to be the teacher of size $i$.
**for** $i = 1$ to $n$ **do**
    Train $f_i$ on $\mathcal{L}$ and let $l_i$ denote the test loss
**end for**
Plot $l_i$ v.s. $i$, identify the classical regime, and the size $i_t$ corresponding to the local minimum in classical regime.
Train $f_{i_t}$ on $\mathcal{L}$ with a regularizer $R(x)$ on $\mathcal{U}$ defined in (A.289).
**Output:** $f_{i_t}$
```

### Experimental Results

**Network models.** Here we use one-hidden-layer fully-connected networks with $m$ hidden units and quadratic activation functions. The network is denoted as FC$(m)$. We use ResNet$(m)$, which is a ResNet-18 convolutional neural network (He et al., 2016) with $m$ filters in the first residual block. It is obtained by scaling the number of filters in each block proportionally from the standard ResNet-18 network which is ResNet$(64)$.

**Datasets.** We use MNIST (Deng, 2012), CIFAR10 (Krizhevsky, 2012) and SVHN (Netzer et al., 2011) as $\mathcal{L}$, and use Gaussian and images in Tiny ImageNet (Le and Yang, 2015) as $\mathcal{U}$. We generate the mixture data, where the fraction of the unlabeled data is denoted as $\alpha$.

**Setup.** We first use Algorithm 3 on the labeled data $\mathcal{L}$ and the unlabeled data $\mathcal{U}$ to get a synthetic labeling function (the teacher network) and then use it to give synthetic labels on a mixture of inputs from $\mathcal{L}$ and $\mathcal{U}$. For MNIST, the teacher network learned is FC$(9)$, where the number of the hidden units is determined by Algorithm 3. See empirical verification in Figure A.22. For CIFAR10 and SVHN, the teacher networks are ResNet$(5)$ and ResNet$(2)$, respectively, as determined by our method. The student network for MNIST is FC$(9)$, and those for CIFAR10 and SVHN are ResNet$(9)$ and ResNet$(8)$, respectively. Finally, we train the student networks on these new datasets with perturbed input distributions.

Figure A.18 shows the results on an equal mixture of data and Gaussian. It presents the test accuracy of the student on the original data part, the Gaussian part, and the whole mixture. For example, for CIFAR10, the test accuracy on the whole mixture is lower than that of training on the original CIFAR10, showing that the input structure indeed has a significant impact on the learning. Furthermore, the network learns well over the CIFAR10 part (with accuracy similar to that on the original data) but learns slower with worse accuracy on the Gaussian part. This suggests that the CIFAR10 input structure is still helping the network to learn effective features. While the results on MNIST+Gaussian do not show a significant trend (possibly because the tasks there are simpler), the results on SVHN+Gaussian show similar significant trends as CIFAR10+Gaussian.

Figure A.20 shows the results when we vary the fraction of the Gaussian data $\alpha$. We observe that the test accuracy curve on the original part and that on the synthetic part have roughly the same trend for different $\alpha$ as before, further verifying our insights.

Figure A.19 shows the results when mixed with Tiny ImageNet data instead of Gaussians. It shows a similar trend, while the performance on the Tiny ImageNet part is higher than that on the Gaussian part. This suggests that compared to Gaussians, the Tiny ImageNet data has helpful input structures, though not as helpful as that on the original data for learning the particular labeling.

### Larger Network on MNIST for Checking The Effect of Input Structure

Here we perform the experiment on MNIST as in A.7.5, but for a network with $m = 50$ hidden neurons rather than $m = 9$. Figure A.21 shows similar results as those for $m = 9$: the learning on the MNIST input part is faster and better than that on the Gaussian input part. The separation between the two is actually more significant than that for $m = 9$. This then also supports our insight about the effect of input structures.

### Empirical Verification of Our Method

We also perform experiments to verify the intuition behind our methodology, i.e., the method gives a synthetic labeling function with roughly the same complexity on the original inputs and the injected inputs. We first use our method on MNIST and samples (of the same size as MNIST) from a Gaussian to get the teacher FC(9); the double descent curve is in Figure A.22(a). Then we train students on the Gaussian data with synthetic labels from the teacher, and plot the double descent curve for the students in Figure A.22(b). The local minimums of the two U-shapes are roughly the same, matching our reasoning. Then we also train larger teachers and plot the double descent curve for students on Gaussian data. Figure A.22(c) Teacher size

50. Figure A.22(d) Teacher size 500. The local minimum of the U-shape becomes larger when the teacher gets larger, again matching our reasoning.

## A.8 Provable Guarantees for Neural Networks in A More General Setting

This section provides the analysis in a more general setting. We first describe the learning problems, and then provide the proofs following similar intuitions as for the simpler settings in the main text.

### A.8.1 Problem Setup

Let $\mathbf{X} = \mathbb{R}^d$ be the input space, and $\mathcal{Y} = \{\pm 1\}$ be the label space. Suppose $M \in \mathbb{R}^{d \times D}$ is a dictionary with D elements, where each element $M_j$ can be regarded as a pattern. We assume quite general incoherent dictionary:

**(D)** M is $\mu$-incoherent, i.e., the columns of M are unit vectors, and for any $i \neq j, |\langle M_i, M_j \rangle| \leqslant \mu/\sqrt{d}$.

Note that the setting in the main text corresponds to $\mu = 0$.

Let $\tilde{\phi} \in \{0,1\}^D$ be a hidden vector that indicates the presence of each pattern, and $\mathcal{D}_{\tilde{\phi}}$ a distribution for $\tilde{\phi}$. Let $\mathbf{A} \subseteq [D]$ be a subset of size $k$ corresponding to the class relevant patterns. Let $P \subseteq [k]$. We first sample $\tilde{\phi}$ from $\mathcal{D}_{\tilde{\phi}}$, and then generate the input $\tilde{x}$ and the class label $y$ from $\tilde{\phi}, \mathbf{A}, P$ by:

$$\tilde{x} = M\tilde{\phi} + \zeta, \quad y = \begin{cases} +1, & \text{if } \sum_{i \in \mathbf{A}} \tilde{\phi}_i \in P, \\ -1, & \text{otherwise} \end{cases} \tag{A.292}$$

where the Gaussian noise $\zeta \sim \mathcal{N}(0, \sigma_\zeta^2 I_{d \times d})$ is independent from $\tilde{\phi}$. Note that the setting in the main text corresponds to $\sigma_\zeta = 0$.

We allow general $\mathcal{D}_{\tilde{\phi}}$ with the following assumptions:

**(A1)** The patterns in $\mathbf{A}$ are correlated with the labels: for any $i \in A$, for $v \in \{\pm 1\}$ let $\gamma_v = \mathbb{E}[y\tilde{\phi}_i | y = v]$, then $\gamma := (\gamma_{+1} + \gamma_{-1})/2 > 0$.

**(A2)** The patterns outside $\mathbf{A}$ are independent of the patterns in $\mathbf{A}$.

Note that we allow imbalanced classes. Let $p_{\min} := \min(\Pr[y = -1], \Pr[y = +1])$. If the classes are balanced, then the assumption **(A1)** implies the assumption **(A1)** in the main text, so the setting here is more general. **(A2)** is also more general, in particular, allowing dependence between irrelevant patterns and non-identical distributions for them.

Let $\mathcal{D}(\mathbf{A}, P, \mathcal{D}_{\tilde{\phi}})$ denote the distribution on $(\tilde{x}, y)$ corresponding to some $\mathbf{A}, P$, and $\mathcal{D}_{\tilde{\phi}}$. Given parameters $\Xi = (d, D, k, \gamma, p_o, \mu, \sigma_\zeta)$, the family $\mathcal{F}_\Xi$ of distributions for learning is the set of all $\mathcal{D}(\mathbf{A}, P, \mathcal{D}_{\tilde{\phi}})$ with $\mathbf{A} \subseteq [D]$, $P \subseteq [k]$, and $\mathcal{D}_{\tilde{\phi}}$ satisfying the above assumptions.

One special case is the mixture of two Gaussians.

*Example.* Suppose $M$ has one single column $v$, and $y = +1$ if $\tilde{\phi} = 1$ and $y = -1$ otherwise. Then the data distribution is simply a mixture of two Gaussians: $\tilde{x} \sim \frac{v}{2} + \mathcal{N}(y\frac{v}{2}, \sigma_\zeta^2 I_{d\times d})$.

### Neural Network Learning

Again, we will normalize the data for learning: we first compute $\mathbf{x} = (\tilde{x} - \mathbb{E}[\tilde{x}])/\tilde{\sigma}$ where $\tilde{\sigma}^2 := \sum_{i=1}^{d} (\tilde{x}_i - \mathbb{E}[\tilde{x}_i])^2 = \sum_{j\in[D]} \mathrm{Var}(\tilde{\phi}_j) + d\sigma_\zeta^2$ is the variance of the data, and then train on $(\mathbf{x}, y)$. This is equivalent to setting $\phi = (\tilde{\phi} - \mathbb{E}[\tilde{\phi}])/\tilde{\sigma}$ and generating $\mathbf{x} = M\phi + \zeta/\sigma_\zeta$. For $(\tilde{x}, y)$ from $\mathcal{D}$ and the normalized $(\mathbf{x}, y)$, we will simply say $(\mathbf{x}, y) \sim \mathcal{D}$.

The learning will be the same as that in the main text, except the following. We will use a small $\sigma_{\mathbf{w}}^2 = \tilde{\sigma}^2/\mathrm{poly}(Dm)$. And we will use a weighted loss to handle the imbalanced classes in the first two steps for feature learning, and then use the unweighted loss in the remaining steps. Formally, the weighted loss is:

$$L_{\mathcal{D}}^{\alpha}(g; \sigma_\xi) = \mathbb{E}_{(\mathbf{x},y)}[\alpha_y \ell(y, g(\mathbf{x}; \xi))], \tag{A.293}$$

where the class weights $\alpha_\nu = \frac{1}{2\Pr[y=\nu]}$ for $\nu \in \{\pm 1\}$.

## A.8.2 Main Result

In this setting, we have the following theorem:

**Theorem A.28.** *Set*

$$\eta^{(1)} = \frac{\gamma^2 p_{\min} \tilde{\sigma}}{k m^3}, \lambda_a^{(1)} = 0, \lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \sigma_\xi^{(1)} = 1/k^{3/2}, \tag{A.294}$$

$$\eta^{(2)} = 1, \lambda_a^{(2)} = \lambda_{\mathbf{w}}^{(2)} = 1/(2\eta^{(2)}), \sigma_\xi^{(2)} = 1/k^{3/2}, \tag{A.295}$$

$$\eta^{(t)} = \eta = \frac{k^2}{T m^{1/3}}, \lambda_a^{(t)} = \lambda_{\mathbf{w}}^{(t)} = \lambda \leqslant \frac{k^3}{\tilde{\sigma} m^{1/3}}, \sigma_\xi^{(t)} = 0, \text{ for } 2 < t \leqslant T. \tag{A.296}$$

*For any* $\delta \in (0, O(1/k^3))$*, if* $\mu \leqslant O(\sqrt{d}/D)$*,* $\sigma_\zeta \leqslant O(\min\{1/\tilde{\sigma}, \tilde{\sigma}/\sqrt{d}\})$*,* $k = \Omega\left(\log^2\left(\frac{Dmd}{\delta\gamma p_{\min}}\right)\right)$*,* $m \geqslant \max\{\Omega(k^4), D, d\}$*, then we have for any* $\mathcal{D} \in \mathcal{F}_\Xi$*, with probability at least* $1 - \delta$*, there exists* $t \in [T]$ *such that*

$$\Pr[\operatorname{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) = O\left(\frac{k^8}{m^{2/3}} + \frac{k^3 T}{m^2} + \frac{k^2 m^{2/3}}{T}\right). \tag{A.297}$$

*Consequently, for any* $\epsilon \in (0,1)$*, if* $T = m^{4/3}$*, and* $m \geqslant \max\{\Omega(k^{12}/\epsilon^{3/2}), D\}$*, then*

$$\Pr[\operatorname{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) \leqslant \epsilon. \tag{A.298}$$

The rest of the section is devoted to the proof of this theorem.

## A.8.3 Notations

Recall some notations that we will use throughout the analysis.

For a vector $\nu$ and an index set $I$, let $\nu_I$ denote the vector containing the entries of $\nu$ indexed by $I$, and $\nu_{-I}$ denote the vector containing the entries of $\nu$ with indices outside $I$.

Let $\rho := M^\top M$. Then we have $\rho_{jj} = 1$ for any $j$, and $|\rho_{j\ell}| \leqslant \mu/\sqrt{d}$ for any $j \neq \ell$.

By initialization, $\mathbf{w}_i^{(0)}$ for $i \in [m]$ are i.i.d. copies of the same random variable $\mathbf{w}^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 I_{d\times d})$; similar for $a^{(0)}$ and $b^{(0)}$. Let $\sigma_{\phi_j}^2 := p_{oj}(1-p_{oj})/\tilde{\sigma}^2$ denote the variance of $\phi_\ell$ for $\ell \notin \mathbf{A}$, where $p_{oj} = \Pr[\tilde{\phi}_j = 1]$. Let $p_o$ be the value such that with probability $1-\exp(-\Omega(k))$, $\sum_{j\notin \mathbf{A}} \tilde{\phi}_j \leqslant p_o(D-k)$ for some $p_o \in [0,1]$. That is, $p_o$ is an upper bound on the density of $\tilde{\phi}_j$ with high probability.

Let $q_\ell := \langle \mathbf{w}^{(0)}, M_\ell \rangle$. Similarly, define $q_{i,\ell}^{(t)} := \langle \mathbf{w}_i^{(t)}, M_\ell\rangle$.

We also define the following sets to denote typical initialization. For a fixed $\delta \in (0,1)$, define

$$\mathcal{G}_{\mathbf{w}}(\delta) := \Bigg\{ \mathbf{w} \in \mathbb{R}^d : q_\ell = \langle \mathbf{w}, M_\ell \rangle, \frac{\sigma_{\mathbf{w}}^2 d}{2} \leqslant \|\mathbf{w}^{(0)}\|_2^2 \leqslant \frac{3\sigma_{\mathbf{w}}^2 d}{2}, \tag{A.299}$$

$$\frac{\sigma_{\mathbf{w}}^2 (D-k)}{2} \leqslant \sum_{\ell \notin \mathbf{A}} q_\ell^2 \leqslant \frac{3\sigma_{\mathbf{w}}^2(D-k)}{2},$$

$$\max_\ell |q_\ell| \leqslant \sigma_{\mathbf{w}} \sqrt{2\log(Dm/\delta)} \Bigg\}, \tag{A.300}$$

$$\mathcal{G}_a(\delta) := \{a \in \mathbb{R} : |a| \leqslant \sigma_a \sqrt{2\log(m/\delta)}\}. \tag{A.301}$$

$$\mathcal{G}_b(\delta) := \{b \in \mathbb{R} : |b| \leqslant \sigma_b \sqrt{2\log(m/\delta)}\}. \tag{A.302}$$

### A.8.4 Existence of A Good Network

We first show that there exists a network that can fit the data distribution.

**Lemma A.29.** *Suppose $\frac{k\mu}{\sqrt{d}} \frac{p_o D}{\tilde{\sigma}} \leqslant \frac{1}{16}$. For any $\mathcal{D} \in \mathcal{F}_\Xi$, there exists a network $g^*(\mathbf{x}) = \sum_{i=1}^n a_i^* \sigma(\langle \mathbf{w}_i^*, \mathbf{x}\rangle + b_i^*)$ which satisfies*

$$\Pr_{(\mathbf{x},y)\sim\mathcal{D}}[y g^*(x) \leqslant 1] \leqslant \exp(-\Omega(k)) + \exp\left(-\Omega\left(\frac{1}{\sigma_\zeta^2(k + k^2\mu/\sqrt{d})}\right)\right).$$

*Furthermore, the number of neurons* $n = 3(k+1)$, $|a_i^*| \leqslant 64k, 1/(64k) \leqslant |b_i^*| \leqslant 1/4$, $\mathbf{w}_i^* = \tilde{\sigma} \sum_{j \in \mathbf{A}} M_j/(8k)$, *and* $|\langle \mathbf{w}_i^*, \mathbf{x} \rangle + b_i^*| \leqslant 1$ *for any* $i \in [n]$ *and* $(\mathbf{x}, y) \sim \mathcal{D}$.

*Consequently, if furthermore we have* $k\mu/\sqrt{d} < 1$ *and* $\sigma_\zeta < 1/k$, *then*

$$\Pr_{(\mathbf{x},y) \sim \mathcal{D}}[y g^*(x) \leqslant 1] \leqslant \exp(-\Omega(k)).$$

*Proof of Lemma A.29.* Let $\mathbf{w} = \tilde{\sigma} \sum_{j \in \mathbf{A}} M_j$ and let $u = \sum_{j \in \mathbf{A}} \mathbb{E}[\tilde{\phi}_j]$. We have

$$\begin{aligned}
\langle \mathbf{w}, \mathbf{x} \rangle &= \tilde{\sigma} \sum_{j \in \mathbf{A}} \langle M_j, M\phi \rangle + \langle \mathbf{w}, \zeta/\tilde{\sigma} \rangle && \text{(A.303)} \\
&= \sum_{j \in \mathbf{A}} \phi_j + \sum_{j \in \mathbf{A}, \ell \neq j} \rho_{j\ell} \phi_\ell + \langle \mathbf{w}, \zeta/\tilde{\sigma} \rangle && \text{(A.304)} \\
&= \sum_{j \in \mathbf{A}} \tilde{\phi}_j - u + \underbrace{\sum_{j \in \mathbf{A}, \ell \neq j} \rho_{j\ell} \phi_\ell + \langle \mathbf{w}, \zeta/\tilde{\sigma} \rangle}_{:= \epsilon_x}. && \text{(A.305)}
\end{aligned}$$

With probability $\geqslant 1 - \exp(-\Omega(k))$, among all $j \notin \mathbf{A}$, we have that at most $p_o(D-k)$ of $\phi_j$ are $(1-p_o)/\tilde{\sigma}$, while the others are $-p_o/\tilde{\sigma}$, and thus

$$\left| \sum_{j \in \mathbf{A}, \ell \neq j} \rho_{j\ell} \phi_\ell \right| \leqslant \frac{k\mu}{\sqrt{d}} \frac{p_o D}{\tilde{\sigma}} \leqslant \frac{1}{16}. \tag{A.306}$$

Furthermore, $\langle \mathbf{w}, \zeta \rangle \sim \mathcal{N}(0, \sigma_\zeta^2 \|w\|_2^2)$ and $\|w\|_2^2 \leqslant \tilde{\sigma}^2(k + k^2\mu/\sqrt{d})$, we have

$$\begin{aligned}
\Pr[|\langle \mathbf{w}, \zeta/\tilde{\sigma} \rangle| \leqslant 1/16] &\geqslant 1 - \exp\left(-\Theta\left(\frac{1}{\sigma_\zeta^2 \|w\|_2^2/\tilde{\sigma}^2}\right)\right) && \text{(A.307)} \\
&\geqslant 1 - \exp\left(-\Theta\left(\frac{1}{\sigma_\zeta^2(k + k^2\mu/\sqrt{d})}\right)\right). && \text{(A.308)}
\end{aligned}$$

For good data points with $\phi$ and $\zeta$ satisfying the above, we have $|\epsilon_\mathbf{x}| \leqslant 1/8$. By Lemma A.1,

$$g_1^*(x) := \sum_{p \in P} \delta_{p-\mu,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p-\mu,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) \tag{A.309}$$

$$= \sum_{p \in P} \delta_{p,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + u) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p,4,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + u) \tag{A.310}$$

$$= \sum_{p \in P} \delta_{p,4,1/2} \left( \sum_{j \in \mathbf{A}} \tilde{\phi}_j + \epsilon_{\mathbf{x}} \right) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p,4,1/2} \left( \sum_{j \in \mathbf{A}} \tilde{\phi}_j + \epsilon_{\mathbf{x}} \right). \tag{A.311}$$

Then for good data points, we have $y g_1^*(x) \geqslant 1$. Similarly,

$$g_2^*(x) := \sum_{p \in P} \delta_{p-\mu+1/4,8,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p-\mu+1/4,8,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle) \tag{A.312}$$

$$= \sum_{p \in P} \delta_{p+1/4,8,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + u) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p+1/4,8,1/2}(\langle \mathbf{w}, \mathbf{x} \rangle + u) \tag{A.313}$$

$$= \sum_{p \in P} \delta_{p+1/4,8,1/2} \left( \sum_{j \in \mathbf{A}} \tilde{\phi}_j + \epsilon_{\mathbf{x}} \right) - \sum_{p \notin P, 0 \leqslant p \leqslant k} \delta_{p+1/4,8,1/2} \left( \sum_{j \in \mathbf{A}} \tilde{\phi}_j + \epsilon_{\mathbf{x}} \right). \tag{A.314}$$

Then for good data points, we have $y g_2^*(x) \geqslant 1$.

Note that the bias terms in $g_1^*$ and $g_2^*$ have distance at least $1/4$, then at least one of them satisfies that all its bias terms have absolute value $\geqslant 1/8$. Pick that one and denote it as $g(\mathbf{x}) = \sum_{i=1}^{n} a_i \text{œ}_r(\langle \mathbf{w}_i, \mathbf{x} \rangle + b_i)$. By the positive homogeneity of $\text{œ}_r$, we have

$$g(\mathbf{x}) = \sum_{i=1}^{n} 8k a_i \text{œ}_r(\langle \mathbf{w}_i, \mathbf{x} \rangle / (8k) + b_i/(8k)). \tag{A.315}$$

Since for any good data points, $|\langle \mathbf{w}_i, \mathbf{x} \rangle / (8k) + b_i/(8k)| \leqslant 1$, then

$$g(\mathbf{x}) = \sum_{i=1}^{n} 8k a_i \sigma(\langle \mathbf{w}_i, \mathbf{x} \rangle / (8k) + b_i/(8k)) \tag{A.316}$$

where $\sigma$ is the truncated ReLU. Now we can set $a_i^* = 8k a_i, \mathbf{w}_i^* = \mathbf{w}_i/(8k), b_i^* = b_i/(8k)$, to get our final $g^*$. □

### A.8.5 Initialization

We first show that with high probability, the initial weights are in typical positions.

**Lemma A.30.** *Suppose* $D\mu/\sqrt{d} \leqslant 1/16$*. For any* $\delta \in (0, 1)$*, with probability at least* $1 - \delta - 2\exp\left(-\Theta(D-k)\right)$ *over* $\mathbf{w}^{(0)}$*,*

$$\sigma_{\mathbf{w}}^2 d/2 \leqslant \|\mathbf{w}^{(0)}\|_2^2 \leqslant 3\sigma_{\mathbf{w}}^2 d/2,$$

$$\sigma_{\mathbf{w}}^2 (D-k)/2 \leqslant \sum_{\ell \notin \mathbf{A}} q_\ell^2 \leqslant 3\sigma_{\mathbf{w}}^2 (D-k)/2,$$

$$\max_\ell |q_\ell| \leqslant \sigma_{\mathbf{w}} \sqrt{2\log(D/\delta)}.$$

*With probability at least* $1-\delta$ *over* $b^{(0)}$*,*

$$|b^{(0)}| \leqslant \sigma_b \sqrt{2\log(1/\delta)}.$$

*With probability at least* $1-\delta$ *over* $a^{(0)}$*,*

$$|a^{(0)}| \leqslant \sigma_a \sqrt{2\log(1/\delta)}.$$

*Proof of Lemma A.30.* The bound on $\|\mathbf{w}^{(0)}\|_2^2$ follows from the property of Gaussians.

Note that $q = M^\top \mathbf{w}^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 \rho)$ for the matrix $\rho = M^\top M$. We have with probability $\geqslant 1 - \delta/2$, $\max_\ell |q_\ell| \leqslant \sqrt{2\sigma_{\mathbf{w}}^2 \log \frac{D}{\delta}}$.

For any subset $S \subseteq [D]$, let $\rho_S$ denote the submatrix of $\rho$ containing the rows and columns indexed by $S$. Then $q_S = M^\top \mathbf{w}^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 \rho_S)$. By diagonalizing $\rho_S$ and then applying Bernstein's inequality, we have with probability $\geqslant 1 - 2\exp\left(-\Theta(|S|/\|\rho\|_2\right)$, $\|q_S\|_2^2 \in \left((\|\rho_S\|_F^2 - \frac{|S|}{4})\sigma_{\mathbf{w}}^2, (\|\rho_S\|_F^2 + \frac{|S|}{4})\sigma_{\mathbf{w}}^2\right)$. By Gershgorin circle theorem, we have

$$\|\rho\|_2 \leqslant 1 + (|S|-1)\mu/\sqrt{d} \leqslant 17/16.$$

Similarly, we have

$$\frac{3}{4}|S| \leqslant \left(\frac{15}{16}\right)^2 |S| \leqslant \|\rho_S\|_F^2 \leqslant \left(\frac{17}{16}\right)^2 |S| \leqslant \frac{5}{4}|S|.$$

The bounds on $q$ then follow.

The bounds on $b^{(0)}$ and $a^{(0)}$ follow from the property of Gaussians. □

**Lemma A.31.** *Suppose* $D\mu/\sqrt{d} \leqslant 1/16$*. We have:*

- *With probability* $\geqslant 1 - \delta - 2m\exp(-\Theta(D-k))$ *over* $\mathbf{w}_i^{(0)}$*'s, for all* $i \in [2m]$, $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta)$.
- *With probability* $\geqslant 1 - \delta$ *over* $b_i^{(0)}$*'s, for all* $i \in [2m]$, $b_i^{(0)} \in \mathcal{G}_b(\delta)$.
- *With probability* $\geqslant 1 - \delta$ *over* $a_i^{(0)}$*'s, for all* $i \in [2m]$, $a_i^{(0)} \in \mathcal{G}_a(\delta)$.

*Proof of Lemma A.31.* This follows from Lemma A.30 by union bound. □

### A.8.6 Some Auxiliary Lemmas

The expression of the gradients will be used frequently.

**Lemma A.32.**

$$\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}^{\alpha}(g;\sigma_\xi) = -a_i \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\{\alpha_y y \mathbb{I}[yg(\mathbf{x};\xi) \leqslant 1] \mathbb{E}_{\xi_i} \mathbb{I}[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i + \xi_i \in (0,1)]\mathbf{x}\}, \tag{A.317}$$

$$\frac{\partial}{\partial b_i} L_{\mathcal{D}}^{\alpha}(g;\sigma_\xi) = -a_i \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\{\alpha_y y \mathbb{I}[yg(\mathbf{x};\xi) \leqslant 1] \mathbb{E}_{\xi_i} \mathbb{I}[\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i \in (0,1)]\}, \tag{A.318}$$

$$\frac{\partial}{\partial a_i} L_{\mathcal{D}}^{\alpha}(g;\sigma_\xi) = -\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\{\alpha_y y \mathbb{I}[yg(\mathbf{x};\xi) \leqslant 1] \mathbb{E}_{\xi_i} \sigma(\langle \mathbf{w}_i, \mathbf{x}\rangle + b_i + \xi_i)\}. \tag{A.319}$$

*Proof of Lemma A.32.* It follows from straightforward calculation. □

We also have the following auxiliary lemma for later calculations.

**Lemma A.33.**

$$\begin{aligned}
\mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\} &= 0, && \text{(A.320)}\\
\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\} &= 1, && \text{(A.321)}\\
\mathbb{E}_{\phi_j}\{|\phi_j|\} &= 2\sigma^2_{\phi_j}\tilde{\sigma}, \textit{ for } j \notin \mathbf{A}, && \text{(A.322)}\\
\mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y \phi_j\} &= \frac{\gamma}{\tilde{\sigma}}, \textit{ for } j \in \mathbf{A}, && \text{(A.323)}\\
\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y \phi_j|\} &\leqslant \frac{1}{\tilde{\sigma}}, \textit{ for all } j \in [D]. && \text{(A.324)}
\end{aligned}$$

*Proof of Lemma A.33.*

$$\begin{aligned}
\mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\} &= \sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y | y = \nu\} \Pr[y = \nu] && \text{(A.325)}\\
&= \frac{1}{2}\sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{y | y = \nu\} && \text{(A.326)}\\
&= 0. && \text{(A.327)}\\
\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\} &= \sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\,| y = \nu\} \Pr[y = \nu] && \text{(A.328)}\\
&= \frac{1}{2}\sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{|y|\,| y = \nu\} && \text{(A.329)}\\
&= 1. && \text{(A.330)}\\
\mathbb{E}_{\phi_j}\{|\phi_j|\} &= \frac{|-p_{oj}|(1-p_{oj}) + |1-p_{oj}|p_{oj}}{\tilde{\sigma}} = 2\sigma^2_{\phi_j}\tilde{\sigma}. && \text{(A.331)}\\
\mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y \phi_j\} &= \sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y \phi_j |\, y = \nu\} \Pr[y = \nu] && \text{(A.332)}\\
&= \frac{1}{2}\sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{y \phi_j |\, y = \nu\} && \text{(A.333)}\\
&= \frac{1}{2}\sum_{\nu\in\{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\left\{ y \frac{\tilde{\phi}_j - \mathbb{E}[\tilde{\phi}_j]}{\tilde{\sigma}} \middle| y = \nu \right\} && \text{(A.334)}\\
&= \frac{1}{2\tilde{\sigma}}(\gamma_{+1} + \gamma_{-1}) = \frac{\gamma}{\tilde{\sigma}}. && \text{(A.335)}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y \phi_j|\} &= \sum_{\nu \in \{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_\nu y \phi_j| \, |y=\nu\} \Pr[y=\nu] && \text{(A.336)} \\
&\leqslant \frac{1}{2} \sum_{\nu \in \{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{|y \phi_j| \, |y=\nu\} && \text{(A.337)} \\
&\leqslant \frac{1}{2} \sum_{\nu \in \{\pm 1\}} \mathbb{E}_{\phi_{\mathbf{A}}}\{|y \phi_j| \, |y=\nu\} && \text{(A.338)} \\
&\leqslant \frac{1}{\tilde{\sigma}}. && \text{(A.339)}
\end{aligned}$$

□

### A.8.7 Feature Emergence: First Gradient Step

We will show that w.h.p. over the initialization, after the first gradient step, there are neurons that represent good features.

We begin with analyzing the gradients.

**Lemma A.34.** *Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta)$ *for all* $i \in [2m]$. *Let*

$$\epsilon_e := \frac{D\sigma_{\mathbf{w}}\sqrt{2\log(D/\delta)}}{\tilde{\sigma}^2 \sigma_\xi^{(1)}} + \frac{\sqrt{d}\sigma_\xi \sigma_{\mathbf{w}}\sqrt{2\log(D/\delta)}}{\tilde{\sigma}\sigma_\xi^{(1)}}, \epsilon_\nu := \epsilon_e.$$

*If* $\sigma_\zeta^2 \sigma_{\mathbf{w}}^2 d/\tilde{\sigma}^2 = O(1/k)$, $p_o = \Omega(k^2/D)$, $k = \Omega(\log^2(Dmd/\delta))$, *and* $\sigma_\xi^{(1)} = O(1/k)$, *then*

$$\frac{\partial}{\partial \mathbf{w}_i} L_{\mathcal{D}}^{\alpha}(g^{(0)}; \sigma_\xi^{(1)}) = -a_i^{(0)} \left( \sum_{j=1}^{D} M_j T_j + \nu \right) \tag{A.340}$$

*where* $T_j$ *satisfies:*

- *if* $j \in A$, *then* $|T_j - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_e/\tilde{\sigma})$, *where* $\beta \in [\Omega(1), 1]$ *and depends only on* $\mathbf{w}_i^{(0)}, b_i^{(0)}$;

- *if* $j \notin \mathbf{A}$, *then* $|T_j| \leqslant O(\sigma_{\phi_j}^2 \epsilon_e \tilde{\sigma})$;

- $|\nu_j| \leqslant O\left(\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}}\epsilon_\nu\right) + \frac{\sigma_\zeta d}{\tilde{\sigma}}e^{-\Theta(k)}$.

*Proof of Lemma A.34.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. Since the unbiased initialization leads to $g^{(0)}(\mathbf{x}; \xi^{(1)}) = 0$, we have

$$\frac{\partial}{\partial \mathbf{w}} L^{\alpha}_{\mathcal{D}}(g^{(0)}; \sigma^{(1)}_{\xi}) \tag{A.341}$$

$$= -a^{(0)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y \mathbb{I}[y g^{(0)}(\mathbf{x};\xi^{(1)}) \leqslant 1]\mathbb{E}_{\xi^{(1)}}\mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\mathbf{x}\right\} \tag{A.342}$$

$$= -a^{(0)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\alpha_y y \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\mathbf{x}\right\} \tag{A.343}$$

$$= -a^{(0)}\sum_{j=1}^{D} M_j \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\alpha_y y \phi_j \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\}}_{:=T_j} \tag{A.344}$$

$$- a^{(0)}\underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\frac{\alpha_y y \zeta}{\tilde{\sigma}} \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\}}_{:=\nu} \tag{A.345}$$

First, consider $j \in \mathbf{A}$.

$$T_j = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\alpha_y y \phi_j \mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} \tag{A.346}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}},\zeta}\left\{\alpha_y y \phi_j \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}\left[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)\right]\right\}. \tag{A.347}$$

where $\iota := \langle \mathbf{w}^{(0)}, \zeta/\tilde{\sigma}\rangle$.

Let

$$I_a := \Pr_{\xi^{(1)}}\left[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)\right], \tag{A.348}$$

$$I'_a := \Pr_{\xi^{(1)}}\left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)\right]. \tag{A.349}$$

Note that $|\langle \phi_{\mathbf{A}}, q_{\mathbf{A}}\rangle| = O(\frac{k\sigma_w\sqrt{2\log(D/\delta)}}{\tilde{\sigma}^2})$, and that $|\iota| = |\langle \mathbf{w}^{(0)}_i, \zeta/\tilde{\sigma}\rangle| = O(\frac{\sqrt{d}\sigma_\xi\sigma_w\sqrt{2\log(D/\delta)}}{\tilde{\sigma}})$, and that $|\langle \phi, q\rangle|, |\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle|$ are $O(\frac{D\sigma_w\sqrt{2\log(D/\delta)}}{\tilde{\sigma}^2})$. When $\sigma_\mathbf{w}$ is sufficiently small, by the property of the Gaussian $\xi^{(1)}$, we have

$$
\begin{aligned}
& |I_a - I'_a| && \text{(A.350)}\\
\leqslant & \left| \Pr_{\xi^{(1)}} \left[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \geqslant 0\right] - \Pr_{\xi^{(1)}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \geqslant 0\right] \right| && \text{(A.351)}\\
& + \Pr_{\xi^{(1)}} \left[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \geqslant 1\right] + \Pr_{\xi^{(1)}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \geqslant 1\right] && \text{(A.352)}\\
= & O(\epsilon_e). && \text{(A.353)}
\end{aligned}
$$

In summary,

$$
|\mathbb{E}_{\zeta, \phi_{-\mathbf{A}}}(I_a - I'_a)| = O(\epsilon_e). \tag{A.354}
$$

Then we have

$$
\begin{aligned}
& \left| T_j - \mathbb{E}_{\phi_{\mathbf{A}}, \zeta, \phi_{-\mathbf{A}}}\{\alpha_y y \phi_j I'_a\} \right| && \text{(A.355)}\\
\leqslant & \mathbb{E}_{\phi_{\mathbf{A}}} \left\{ |\alpha_y y \phi_j| \left| \mathbb{E}_{\zeta, \phi_{-\mathbf{A}}}(I_a - I'_a) \right| \right\} && \text{(A.356)}\\
\leqslant & O(\epsilon_e) \mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y \phi_j|\} && \text{(A.357)}\\
\leqslant & O(\epsilon_e / \tilde{\sigma}) && \text{(A.358)}
\end{aligned}
$$

where the last step is from Lemma A.33. Furthermore,

$$
\begin{aligned}
& \mathbb{E}_{\phi_{\mathbf{A}}, \zeta, \phi_{-\mathbf{A}}}\{\alpha_y y \phi_j I'_a\} && \text{(A.359)}\\
= & \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y \phi_j\} \mathbb{E}_{\zeta, \phi_{-\mathbf{A}}}[I'_a] && \text{(A.360)}\\
= & \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y \phi_j\} \Pr_{\phi_{-\mathbf{A}}, \zeta, \phi_{-\mathbf{A}}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0, 1)\right] && \text{(A.361)}
\end{aligned}
$$

When $\sigma_{\mathbf{w}}$ is sufficiently small, we have

$$
\Pr_{\phi_{-\mathbf{A}}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + b^{(0)} \in (0, 1/2)\right] \geqslant \Omega(1), \tag{A.362}
$$

$$\Pr_{\zeta,\xi^{(1)}}\left[\iota+\xi^{(1)}\in(0,1/2)\right]=1/2-\exp(-\Omega(k)), \tag{A.363}$$

This leads to

$$\beta:=\mathbb{E}_{\zeta,\phi_{-\mathbf{A}}}[I'_a]=\Pr_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\left[\langle\phi_{-\mathbf{A}},q_{-\mathbf{A}}\rangle+\iota+b^{(0)}+\xi^{(1)}\in(0,1)\right]\geqslant\Omega(1). \tag{A.364}$$

By Lemma A.33, $\mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\phi_j\}=\gamma/\tilde{\sigma}$. Therefore,

$$|T_j-\beta\gamma/\tilde{\sigma}|\leqslant O(\epsilon_e/\tilde{\sigma}). \tag{A.365}$$

Now, consider $j\notin\mathbf{A}$. Let B denote $\mathbf{A}\cup\{j\}$.

$$\begin{aligned}T_j&=\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\zeta,\xi^{(1)}}\left\{\alpha_y y\phi_j\mathbb{I}\left[\langle\phi,q\rangle+\iota+b^{(0)}+\xi^{(1)}\in(0,1)\right]\right\} &\text{(A.366)}\\&=\mathbb{E}_{\phi_B}\mathbb{E}_{\phi_{-B},\zeta,\xi^{(1)}}\left\{\alpha_y y\phi_j\mathbb{I}\left[\langle\phi,q\rangle+\iota+b^{(0)}+\xi^{(1)}\in(0,1)\right]\right\} &\text{(A.367)}\\&=\mathbb{E}_{\phi_B,\zeta}\left\{\alpha_y y\phi_j\Pr_{\phi_{-B},\xi^{(1)}}\left[\langle\phi,q\rangle+\iota+b^{(0)}+\xi^{(1)}\in(0,1)\right]\right\}. &\text{(A.368)}\end{aligned}$$

Let

$$\begin{aligned}I_b&:=\Pr_{\xi^{(1)}}\left[\langle\phi,q\rangle+\iota+b^{(0)}+\xi^{(1)}\in(0,1)\right], &\text{(A.369)}\\I'_b&:=\Pr_{\xi^{(1)}}\left[\langle\phi_{-B},q_{-B}\rangle+\iota+b^{(0)}+\xi^{(1)}\in(0,1)\right]. &\text{(A.370)}\end{aligned}$$

Similar as above, we have $|\mathbb{E}_{\zeta,\xi^{(1)}}(I_b-I'_b)|\leqslant O(\epsilon_e)$. Then by Lemma A.33,

$$\begin{aligned}&\left|T_j-\mathbb{E}_{\phi_B,\zeta,\phi_{-B}}\{\alpha_y y\phi_j I'_b\}\right| &\text{(A.371)}\\&\leqslant\mathbb{E}_{\phi_B}\left\{|\alpha_y y\phi_j||\mathbb{E}_{\zeta,\phi_{-B}}(I_b-I'_b)|\right\} &\text{(A.372)}\\&\leqslant O(\epsilon_e)\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\}\mathbb{E}_{\phi_j}\{|\phi_j|\} &\text{(A.373)}\\&\leqslant O(\epsilon_e)\times 1\times O(\sigma_{\phi_j}^2\tilde{\sigma}) &\text{(A.374)}\\&=O(\sigma_{\phi_j}^2\epsilon_e\tilde{\sigma}). &\text{(A.375)}\end{aligned}$$

Furthermore,

$$\mathbb{E}_{\phi_B,\zeta,\phi_{-B}}\{\alpha_y y \phi_j I'_b\} = \mathbb{E}_{\phi_\mathbf{A}}\{\alpha_y y\}\mathbb{E}_{\phi_j}\{\phi_j\}\mathbb{E}_{\zeta,\phi_{-B}}[I'_b] = 0. \tag{A.376}$$

Therefore,

$$|T_j| \leqslant O(\sigma_\phi^2 \epsilon_e \tilde{\sigma}). \tag{A.377}$$

Finally, consider $\nu_j$.

$$\begin{aligned}
\nu_j &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}\mathbb{I}[\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} && \text{(A.378)}\\
&= \mathbb{E}_{\phi_\mathbf{A},\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}\mathbb{I}[\langle \phi, q\rangle + \iota_j + \iota_{-j} + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} && \text{(A.379)}\\
&= \mathbb{E}_{\phi_\mathbf{A},\zeta}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}\Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi, q\rangle + \iota_j + \iota_{-j} + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} && \text{(A.380)}
\end{aligned}$$

where $\iota_j := \mathbf{w}_j^{(0)}\zeta_j/\tilde{\sigma}$ and $\iota_{-j} := \langle \mathbf{w}^{(0)}, \zeta/\tilde{\sigma}\rangle - \iota_j$.

With probability $\geqslant 1 - d\exp(-\Theta(k))$ over $\zeta$, for any j, $|\zeta_j| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$. Let $\mathcal{G}_\zeta$ denote this event.

Let

$$\begin{aligned}
I_j &:= \Pr_{\xi^{(1)}}\left[\langle \phi, q\rangle + \iota_j + \iota_{-j} + b^{(0)} + \xi^{(1)} \in (0,1)\right], && \text{(A.381)}\\
I'_j &:= \Pr_{\xi^{(1)}}\left[\langle \phi, q\rangle + \iota_{-j} + b^{(0)} + \xi^{(1)} \in (0,1)\right]. && \text{(A.382)}
\end{aligned}$$

Similar as above, we have $|\mathbb{E}_\zeta[I_j - I'_j|\mathcal{G}_\zeta]| \leqslant O(\epsilon_\nu)$. Then

$$\begin{aligned}
|\mathbb{E}_{\zeta,\phi_{-\mathbf{A}}}(I_j - I'_j)| &\leqslant |\mathbb{E}_{\zeta,\phi_{-\mathbf{A}}}[(I_j - I'_j)|\mathcal{G}_\zeta]| + \Pr[-\mathcal{G}_\zeta] && \text{(A.383)}\\
&\leqslant O(\epsilon_\nu + d\exp(-\Theta(k))). && \text{(A.384)}
\end{aligned}$$

$$\left|\nu_j - \mathbb{E}_{\phi_\mathbf{A},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}} I'_j\right\}\right| \tag{A.385}$$

$$= \left|\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}(I_j - I'_j)\right\}\right| \tag{A.386}$$

$$\leqslant \left|\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}(I_j - I'_j)|\mathcal{G}_\zeta\right\}\right| + \left|\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}(I_j - I'_j)| - \mathcal{G}_\zeta\right\}\right| \Pr[-\mathcal{G}_\zeta]. \tag{A.387}$$

The first term is bounded by

$$\left|\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}(I_j - I'_j)|\mathcal{G}_\zeta\right\}\right| \tag{A.388}$$

$$\leqslant \mathbb{E}_{\phi_{\mathbf{A}}}\left\{\frac{\alpha_y y \sigma_\zeta \sqrt{\log(k)}}{\tilde{\sigma}}|\mathbb{E}_{\zeta,\phi_{-\mathbf{A}}}[I_b - I'_b|\mathcal{G}_\zeta]|\right\} \tag{A.389}$$

$$\leqslant O(\epsilon_\nu)\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\}\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}} \tag{A.390}$$

$$\leqslant O(\epsilon_\nu) \times 1 \times \frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}} \tag{A.391}$$

$$= O\left(\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}}\epsilon_\nu\right). \tag{A.392}$$

The second term is bounded by

$$\left|\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}(I_j - I'_j)| - \mathcal{G}_\zeta\right\}\right| \Pr[-\mathcal{G}_\zeta] \tag{A.393}$$

$$\leqslant \left|\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}(I_j - I'_j)| - \mathcal{G}_\zeta\right\}\right| \times de^{-\Theta(k)} \tag{A.394}$$

$$\leqslant \mathbb{E}_{\phi_{\mathbf{A}}}\left|\frac{\alpha_y y}{\tilde{\sigma}}\right| \times \mathbb{E}_\zeta\{|\zeta_j|| - \mathcal{G}_\zeta\} \times de^{-\Theta(k)} \tag{A.395}$$

$$\leqslant \frac{\sigma_\zeta}{\tilde{\sigma}} \times de^{-\Theta(k)} \tag{A.396}$$

$$\leqslant \frac{\sigma_\zeta d}{\tilde{\sigma}}e^{-\Theta(k)}. \tag{A.397}$$

Furthermore,

$$\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\phi_{-\mathbf{A}}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}I'_j\right\} = \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\}\mathbb{E}_{\zeta_j}\left\{\frac{\zeta_j}{\tilde{\sigma}}\right\}\mathbb{E}_{\zeta_{-j}}[I'_j] = 0. \tag{A.398}$$

Therefore,

$$|v_j| \leqslant O\left(\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}}\epsilon_v\right) + \frac{\sigma_\zeta d}{\tilde{\sigma}}e^{-\Theta(k)}. \tag{A.399}$$

□

**Lemma A.35.** *Under the same assumptions as in Lemma A.34,*

$$\frac{\partial}{\partial b_i}L^{\alpha}_{\mathcal{D}}(g^{(0)};\sigma^{(1)}_{\xi}) = -a^{(0)}_i T_b \tag{A.400}$$

*where* $|T_b| \leqslant O(\epsilon_e)$.

*Proof of Lemma A.35.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. Since the unbiased initialization leads to $g^{(0)}(\mathbf{x};\xi^{(1)}) = 0$, we have

$$\frac{\partial}{\partial b}L^{\alpha}_{\mathcal{D}}(g^{(0)};\sigma^{(1)}_{\xi}) \tag{A.401}$$

$$= -a^{(0)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y\mathbb{I}[yg^{(0)}(\mathbf{x};\xi^{(1)}) \leqslant 1]\mathbb{E}_{\xi^{(1)}}\mathbb{I}[\langle \mathbf{w}^{(0)},\mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} \tag{A.402}$$

$$= -a^{(0)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\alpha_y y\mathbb{I}[\langle \mathbf{w}^{(0)},\mathbf{x}\rangle + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} \tag{A.403}$$

$$= -a^{(0)}\underbrace{\mathbb{E}_{\phi_{\mathbf{A}},\zeta,\xi^{(1)}}\left\{\alpha_y y\Pr_{\phi_{-\mathbf{A}}}\left[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)\right]\right\}}_{:=T_b}. \tag{A.404}$$

where $\iota := \langle \mathbf{w}^{(0)}, \zeta/\tilde{\sigma}\rangle$. Similar to the proof in Lemma A.34,

$$\left|\mathbb{E}_\zeta\left(\Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)]\right.\right. \tag{A.405}$$

$$\left.\left. - \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)]\right)\right| = O(\epsilon_e). \tag{A.406}$$

Then

$$\left|T_b - \mathbb{E}_{\phi_{\mathbf{A}},\zeta}\left\{\alpha_y y \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)]\right\}\right| \tag{A.407}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}},\zeta}\left\{ |\alpha_y y| \left| \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)] \right.\right. \tag{A.408}$$

$$\left.\left. - \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)]\right|\right\} \tag{A.409}$$

$$\leqslant O(\epsilon_e)\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\} \tag{A.410}$$

$$\leqslant O(\epsilon_e). \tag{A.411}$$

Also,

$$\mathbb{E}_{\phi_{\mathbf{A}},\zeta}\left\{\alpha_y y \Pr_{\phi_{-\mathbf{A}},\xi^{(1)}}[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)]\right\} \tag{A.412}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\} \Pr_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)] \tag{A.413}$$

$$= 0. \tag{A.414}$$

Therefore, $|T_b| \leqslant O(\epsilon_e)$. □

**Lemma A.36.** *We have*

$$\frac{\partial}{\partial a_i} L^{\alpha}_{\mathcal{D}}(g^{(0)}; \sigma^{(1)}_{\xi}) = -T_a \tag{A.415}$$

*where* $|T_a| \leqslant O(\max_\ell q^{(0)}_{i,\ell})$*. So if* $w^{(0)}_i \in \mathcal{G}(\delta)$, $|T_a| \leqslant O(\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)})$.

*Proof of Lemma A.36.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. Since the unbiased initialization leads to $g^{(0)}(\mathbf{x}; \xi^{(1)}) = 0$, we have

$$\frac{\partial}{\partial a} L^{\alpha}_{\mathcal{D}}(g^{(0)}; \sigma^{(1)}_{\xi}) \tag{A.416}$$

$$= -\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y \mathbb{I}[y g^{(0)}(\mathbf{x}; \xi^{(1)}) \leqslant 1]\mathbb{E}_{\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})\right\} \tag{A.417}$$

$$= -\underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(1)}}\left\{\alpha_y y\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})\right\}}_{:=T_a}. \tag{A.418}$$

Let $\phi'_\mathbf{A}$ be an independent copy of $\phi_\mathbf{A}$, $\phi'$ be the vector obtained by replacing in $\phi$ the entries $\phi_\mathbf{A}$ with $\phi'_\mathbf{A}$, and let $x' = M\phi' + \zeta/\tilde{\sigma}$ and its label is $y'$. Then

$$|T_a| = \left|\mathbb{E}_{\phi_\mathbf{A}}\left\{\alpha_y y \mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})\right\}\right| \tag{A.419}$$

$$\leqslant \frac{1}{2}\left|\mathbb{E}_{\phi_\mathbf{A}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})|y=1\right\}\right. \tag{A.420}$$

$$\left. - \mathbb{E}_{\phi_\mathbf{A}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})|y=-1\right\}\right| \tag{A.421}$$

$$\leqslant \frac{1}{2}\left|\mathbb{E}_{\phi_\mathbf{A}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}\rangle + b^{(0)} + \xi^{(1)})|y=1\right\}\right. \tag{A.422}$$

$$\left. - \mathbb{E}_{\phi'_\mathbf{A}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(1)}}\sigma(\langle \mathbf{w}^{(0)}, \mathbf{x}'\rangle + b^{(0)} + \xi^{(1)})|y'=-1\right\}\right|. \tag{A.423}$$

Since $\sigma$ is 1-Lipschitz,

$$|T_a| \leqslant \frac{1}{2}\mathbb{E}_{\phi_\mathbf{A},\phi'_\mathbf{A}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}}}\left|\langle \mathbf{w}^{(0)}, M\phi\rangle - \langle \mathbf{w}^{(0)}, M\phi'\rangle\right| |y=1, y'=-1\right\} \tag{A.424}$$

$$\leqslant \frac{1}{2}\mathbb{E}_{\phi_{-\mathbf{A}}}\left(\mathbb{E}_{\phi_\mathbf{A}}\left\{\left|\langle \mathbf{w}^{(0)}, M\phi\rangle\right| |y=1\right\} + \mathbb{E}_{\phi'_\mathbf{A}}\left\{\left|\langle \mathbf{w}^{(0)}, M\phi'\rangle\right| |y'=-1\right\}\right) \tag{A.425}$$

$$\leqslant \max_\ell q_{i,\ell}^{(0)}\sqrt{\mathbb{E}_\phi\left(\sum_{\ell\in[D]}\phi_\ell^2 + \sum_{j\neq\ell:j,\ell\in\mathbf{A}}|\phi_j\phi_\ell|\right)} \tag{A.426}$$

$$\leqslant \max_\ell q_{i,\ell}^{(0)}\sqrt{\mathbb{E}_\phi\left(1 + O(1)\right)} \tag{A.427}$$

$$= \Theta(\max_\ell q_{i,\ell}^{(0)}). \tag{A.428}$$

□

With the bounds on the gradient, we now summarize the results for the weights after the first gradient step.

**Lemma A.37.** *Set*

$$\lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \lambda_{a}^{(1)} = \lambda_{b}^{(1)} = 0, \sigma_{\xi}^{(1)} = 1/k^{3/2}.$$

*Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta)$ *for all* $i \in [2m]$. *If* $k = \Omega(\log^2(Dm/\delta))$, *then for all* $i \in [m]$, $\mathbf{w}_i^{(1)} = \sum_{\ell=1}^{D} q_{i,\ell}^{(1)} M_\ell + \upsilon$ *satisfying*

- *if* $\ell \in A$, *then* $|q_{i,\ell}^{(1)} - \eta^{(1)} a_i^{(0)} \beta\gamma/\tilde{\sigma}| \leqslant O\left(\frac{|\eta^{(1)} a_i^{(0)}|\epsilon_e}{\tilde{\sigma}}\right)$, *where* $\beta \in [\Omega(1), 1]$ *and depends only on* $\mathbf{w}_i^{(0)}, b_i^{(0)}$;
- *if* $\ell \notin \mathbf{A}$, *then* $|q_{i,\ell}^{(1)}| \leqslant O\left(|\eta^{(1)} a_i^{(0)}|\sigma_{\phi_\ell}^2 \epsilon_e \tilde{\sigma}\right)$;
- $|\upsilon_j| \leqslant O\left(|\eta^{(1)} a_i^{(0)}| \left(\frac{\sigma_\zeta \sqrt{\log(k)}}{\tilde{\sigma}} \epsilon_\nu + \frac{\sigma_\zeta d}{\tilde{\sigma}} e^{-\Theta(k)}\right)\right)$.

*and*

- $b_i^{(1)} = b_i^{(0)} + \eta^{(1)} a_i^{(0)} T_b$ *where* $|T_b| = O(\epsilon_e)$;
- $a_i^{(1)} = a_i^{(0)} + \eta^{(1)} T_a$ *where* $|T_a| = O(\sigma_{\mathbf{w}} \sqrt{\log(Dm/\delta)})$.

*Proof of Lemma A.37.* This follows from Lemma A.31 and Lemma A.34-A.36. □

### A.8.8 Feature Improvement: Second Gradient Step

We first show that with properly set $\eta^{(1)}$, for most $\mathbf{x}$, $|g^{(1)}(\mathbf{x}; \sigma_{\xi}^{(2)})| < 1$ and thus $y g^{(1)}(\mathbf{x}; \sigma_{\xi}^{(2)}) < 1$.

**Lemma A.38.** *Fix* $\delta \in (0,1)$ *and suppose* $\mathbf{w}_i^{(0)} \in \mathcal{G}_{\mathbf{w}}(\delta), b_i^{(0)} \in \mathcal{G}_b(\delta), a_i^{(0)} \in \mathcal{G}_a(\delta)$ *for all* $i \in [2m]$. *If* $D\mu/\sqrt{d} \leqslant 1/16$, $\sigma_\zeta \tilde{\sigma} = O(1)$, $\sigma_\zeta^2 d/\tilde{\sigma}^2 = O(1)$, $k = \Omega(\log^2(Dm/\delta))$, $\sigma_a \leqslant \tilde{\sigma}^2/(\gamma k^2)$, $\eta^{(1)} = O\left(\frac{\gamma}{km\sigma_a\tilde{\sigma}}\right)$, *and* $\sigma_{\xi}^{(2)} \leqslant 1/k$, *then with probability* $\geqslant 1 - (d+D)\exp(-\Omega(k))$ *over* $(\mathbf{x}, y)$, *we have* $y g^{(1)}(\mathbf{x}; \sigma_{\xi}^{(2)}) < 1$. *Furthermore, for any* $i \in [2m]$, $\left|\langle \mathbf{w}_i^{(1)}, \zeta/\tilde{\sigma}\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$, $\left|\langle q_i^{(1)}, \phi\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$, *and* $\left|\langle (q_i^{(1)})_{-\mathbf{A}}, \phi_{-\mathbf{A}}\rangle\right| =$

$O(\eta^{(1)}\tilde{\sigma}/\gamma)$, *and for any* $j \in [d], \ell \in [D]$, $|\zeta_j| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$ *and* $|\langle \zeta, D_\ell\rangle| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$.

*Proof of Lemma A.38.* Note that $\mathbf{w}_i^{(0)} = \mathbf{w}_{m+i}^{(0)}$, $b_i^{(0)} = b_{m+i}^{(0)}$, and $a_i^{(0)} = a_{m+i}^{(0)}$. Then the gradient for $\mathbf{w}_{m+i}$ is the negation of that for $\mathbf{w}_{m+i}$, the gradient for $b_{m+i}$ is the negation of that for $b_{m+i}$, and the gradient for $a_{m+i}$ is the same as that for $a_{m+i}$.

$$\left|g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})\right| \tag{A.429}$$

$$= \left|\sum_{i=1}^{2m} a_i^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)} + \xi_i^{(2)})\right| \tag{A.430}$$

$$= \left|\sum_{i=1}^{m}\left(a_i^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)} + \xi_i^{(2)}) + a_{m+i}^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_{m+i}^{(1)}, \mathbf{x}\rangle + b_{m+i}^{(1)} + \xi_{m+i}^{(2)})\right)\right| \tag{A.431}$$

$$\leqslant \left|\sum_{i=1}^{m}\left(a_i^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)} + \xi_i^{(2)}) + a_{m+i}^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)} + \xi_i^{(2)})\right)\right| \tag{A.432}$$

$$+ \left|\sum_{i=1}^{m}\left(-a_{m+i}^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)} + \xi_i^{(2)}) + a_{m+i}^{(1)}\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_{m+i}^{(1)}, \mathbf{x}\rangle + b_{m+i}^{(1)} + \xi_i^{(2)})\right)\right|. \tag{A.433}$$

Then we have

$$\begin{aligned}
\left|g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})\right| &\leqslant \sum_{i=1}^{m}\left|2\eta^{(1)}T_a\mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)} + \xi_i^{(2)})\right| && \text{(A.434)}\\
&\quad + \sum_{i=1}^{m}\left|a_{m+i}^{(1)}\right|\left(\left|\langle \mathbf{w}_i^{(1)} - \mathbf{w}_{m+i}^{(1)}, \mathbf{x}\rangle\right| + \left|b_i^{(1)} - b_{m+i}^{(1)}\right|\right) && \text{(A.435)}\\
&\leqslant \sum_{i=1}^{m}\left|2\eta^{(1)}T_a\right|\left(\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle + b_i^{(1)}\right| + \mathbb{E}_{\xi^{(2)}}\left|\xi_i^{(2)}\right|\right) && \text{(A.436)}\\
&\quad + \sum_{i=1}^{m}\left|a_{m+i}^{(1)}\right|\left(\left|\langle \mathbf{w}_i^{(1)} - \mathbf{w}_{m+i}^{(1)}, \mathbf{x}\rangle\right| + \left|b_i^{(1)} - b_{m+i}^{(1)}\right|\right). && \text{(A.437)}
\end{aligned}$$

With probability $\geqslant 1-\exp(-\Omega(k))$, among all $j \notin \mathbf{A}$, we have that at most $p_o(D-k)$ of $\phi_j$ are $(1-p_{oj})/\tilde{\sigma}$, while the others are $-p_{oj}/\tilde{\sigma}$. With probability $\geqslant 1-(d+D)\exp(-\Omega(k))$ over $\zeta$, for any $j$, $|\zeta_j| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$ and $|\langle \zeta, D_\ell\rangle| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$. For data points with $\phi$ and $\zeta$ satisfying these, we have:

**Claim A.39.** $\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle\right| \leqslant O(\eta^{(1)}/\gamma)(1+\tilde{\sigma}+\tilde{\sigma}/\sqrt{k})$.

*Proof of Claim A.39.*

$$\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle\right| = \left|\langle \sum_{\ell=1}^{D} q_{i,\ell}^{(1)} M_\ell + \upsilon, \sum_{j=1}^{D} \phi_j M_j + \zeta/\tilde{\sigma}\rangle\right| \tag{A.438}$$

$$\leqslant \left|\langle \sum_{\ell=1}^{D} q_{i,\ell}^{(1)} M_\ell, \sum_{j=1}^{D} \phi_j M_j\rangle\right| + \left|\langle \sum_{\ell=1}^{D} q_{i,\ell}^{(1)} M_\ell, \zeta/\tilde{\sigma}\rangle\right| + \left|\langle \upsilon, \sum_{j=1}^{D} \phi_j M_j\rangle\right| + |\langle \upsilon, \zeta/\tilde{\sigma}\rangle|. \tag{A.439}$$

For each term above we bound as follows. Note that when $\sigma_{\mathbf{w}}$ is sufficiently small, $\epsilon_e = O(k\log^{1/2}(Dm/\delta)/\sqrt{D})$. Let

$$B_1 := \beta\gamma/\tilde{\sigma} + \epsilon_e/\tilde{\sigma}, \tag{A.440}$$

$$B_2 := \sigma_\phi^2 \epsilon_e \tilde{\sigma} = O(\epsilon_e/\sqrt{D}), \tag{A.441}$$

$$C_1 = \frac{k}{\tilde{\sigma}}, \tag{A.442}$$

$$C_2 := p_o D/\tilde{\sigma} = O(D/\tilde{\sigma}). \tag{A.443}$$

Then

$$|a_i^{(0)}|B_1C_1 = O(\log(Dm/\delta)/k + \log(Dm/\delta)\epsilon_e/(\gamma k)) = O(1/\gamma), \tag{A.444}$$

$$|a_i^{(0)}|B_2C_2 = O(\tilde{\sigma}/\gamma), \tag{A.445}$$

$$|a_i^{(0)}|B_1C_2 = O(D/k + \sqrt{D}/\gamma), \tag{A.446}$$

$$|a_i^{(0)}|B_2C_1 = O(\epsilon_e/(\gamma\sqrt{k})) = O(1/\gamma), \tag{A.447}$$

Then by the assumption on $\mu$,

$$\left|\langle\sum_{\ell=1}^{D} q_{i,\ell}^{(1)} M_\ell, \sum_{j=1}^{D}\phi_j M_j\rangle\right| \tag{A.448}$$

$$= \left|\sum_{\ell\in\mathbf{A}}\langle q_{i,\ell}^{(1)}M_\ell, M_\ell\phi_\ell\rangle\right| + \left|\sum_{\ell\notin\mathbf{A}}\langle q_{i,\ell}^{(1)}M_\ell, M_\ell\phi_\ell\rangle\right| + \left|\sum_{\ell\neq j}\langle q_{i,\ell}^{(1)}M_\ell, M_j\phi_j\rangle\right| \tag{A.449}$$

$$\leqslant O(|\eta^{(1)}a_i^{(0)}|)\left(B_1C_1+B_2C_2+\frac{\mu}{\sqrt{d}}(kB_1(C_1+C_2)+DB_2(C_1+C_2))\right) \tag{A.450}$$

$$\leqslant O(|\eta^{(1)}a_i^{(0)}|)\left(B_1C_1+B_2C_2+\frac{k\mu}{\sqrt{d}}B_1C_2+B_2C_1\right) \tag{A.451}$$

$$\leqslant O(\eta^{(1)})(1/\gamma+\tilde{\sigma}/\gamma+1/\gamma+1/\gamma) \tag{A.452}$$

$$\leqslant O(\eta^{(1)}/\gamma)(1+\tilde{\sigma}). \tag{A.453}$$

By the assumption on $\sigma_\zeta$,

$$\left|\langle\sum_{\ell=1}^{D} q_{i,\ell}^{(1)} M_\ell, \zeta/\tilde{\sigma}\rangle\right| \tag{A.454}$$

$$\leqslant O(|\eta^{(1)}a_i^{(0)}|)(kB_1+DB_2)\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}} \tag{A.455}$$

$$\leqslant O(\eta^{(1)})(kB_1+DB_2)\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}}\frac{\tilde{\sigma}^2\sqrt{\log(Dm/\delta)}}{\gamma k^2} \tag{A.456}$$

$$\leqslant O(\eta^{(1)})\left(\sigma_\zeta\log(Dm/\delta)\left(\frac{1}{k}+\frac{\epsilon_e}{\gamma k}\right)+\sigma_\zeta\tilde{\sigma}\frac{\log(k)\log(Dm/\delta)}{\gamma k}\right) \tag{A.457}$$

$$\leqslant O(\eta^{(1)}/\gamma). \tag{A.458}$$

Also note that $|\nu_j| \leqslant O\left(\frac{\sigma_\zeta\log^2(Dm/\delta)}{\tilde{\sigma}\sqrt{D}}\right)$. Then by the assumption on $\sigma_\zeta$,

$$\left|\langle\upsilon, \sum_{j=1}^{D}\phi_j M_j\rangle\right| \tag{A.459}$$

$$\leqslant O(|\eta^{(1)}a_i^{(0)}|)\times\sqrt{d}\times O\left(\frac{\sigma_\zeta\log^2(Dm/\delta)}{\tilde{\sigma}\sqrt{D}}\right)\times(C_1+C_2) \tag{A.460}$$

$$\leqslant O(|\eta^{(1)}/\gamma). \tag{A.461}$$

Finally, we have

$$
\begin{aligned}
|\langle \upsilon, \zeta/\tilde{\sigma}\rangle| &\leqslant \sum_{j=1}^{d} |\upsilon_j||\zeta_j/\tilde{\sigma}| \qquad (A.462)\\
&\leqslant O(|\eta^{(1)} a_i^{(0)}|) \times d \times \frac{\sigma_\zeta \log^2(Dm/\delta)}{\tilde{\sigma}\sqrt{D}} \frac{\tilde{\sigma}\sqrt{\log(k)}}{\tilde{\sigma}} \qquad (A.463)\\
&\leqslant O(\eta^{(1)}/\gamma)\frac{\tilde{\sigma}}{\sqrt{k}}. \qquad (A.464)
\end{aligned}
$$

□

We also have:

$$
\begin{aligned}
|T_a| &= O(\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}) \qquad (A.465)\\
\left|b_i^{(1)}\right| &\leqslant \left|b_i^{(0)}\right| + \left|\eta^{(1)} a_i^{(0)} T_b\right| \qquad (A.466)\\
&\leqslant \frac{\sqrt{\log(m/\delta)}}{k^2} + \left|\eta^{(1)} a_i^{(0)} \epsilon_e\right|. \qquad (A.467)\\
\mathbb{E}_{\xi^{(2)}}\left|\xi_i^{(2)}\right| &\leqslant O(\sigma_\xi^{(2)}). \qquad (A.468)
\end{aligned}
$$

$$
\begin{aligned}
|a_{m+i}^{(1)}| \leqslant |a_i^{(0)}| + |\eta^{(1)} T_a| \leqslant |a_i^{(0)}| + O(\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}). \qquad (A.469)\\
\left|\langle \mathbf{w}_i^{(1)} - \mathbf{w}_{m+i}^{(1)}, \mathbf{x}\rangle\right| = 2\left|\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma). \qquad (A.470)\\
\left|b_i^{(1)} - b_{m+i}^{(1)}\right| = 2|\eta^{(1)} a_i^{(0)} T_b| = O(|\eta^{(1)} a_i^{(0)}|\epsilon_e). \qquad (A.471)
\end{aligned}
$$

Then we have

$$\left|g^{(1)}(\mathbf{x}; \sigma_\xi^{(2)})\right| \tag{A.472}$$

$$\leqslant O\left(m\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}\right)\left(\frac{\eta^{(1)}\tilde{\sigma}}{\gamma}+\frac{\sqrt{\log(m/\delta)}}{k^2}+\left|\eta^{(1)}a_i^{(0)}\epsilon_e\right|+\sigma_\xi^{(2)}\right) \tag{A.473}$$

$$+O\left(m(|a_i^{(0)}|+\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)})\right)\left(\frac{\eta^{(1)}\tilde{\sigma}}{\gamma}+\left|\eta^{(1)}a_i^{(0)}\epsilon_e\right|\right) \tag{A.474}$$

$$=O\left(m\eta^{(1)}\sigma_{\mathbf{w}}\frac{\log(Dm/\delta)}{k}+m|a_i^{(0)}|\left(\frac{\eta^{(1)}\tilde{\sigma}}{\gamma}+\left|\eta^{(1)}a_i^{(0)}\epsilon_e\right|\right)\right) \tag{A.475}$$

$$=O\left(m\eta^{(1)}\sigma_{\mathbf{w}}\frac{\log(Dm/\delta)}{k}+m|a_i^{(0)}|\frac{\eta^{(1)}\tilde{\sigma}}{\gamma}+m|a_i^{(0)}|\frac{\eta^{(1)}\tilde{\sigma}}{\gamma\sqrt{k}}\right) \tag{A.476}$$

$$<1. \tag{A.477}$$

Then $\left|yg^{(1)}(\mathbf{x};\sigma_\xi^{(2)})\right|<1$. Finally, the statement on $\left|\langle(q_i^{(1)})_{-\mathbf{A}},\phi_{-\mathbf{A}}\rangle\right|$ follows from a similar calculation on $\left|\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle\right|=\left|\langle q_i^{(1)},\phi\rangle\right|$. □

We are now ready to analyze the gradients in the second gradient step.

**Lemma A.40.** *Fix* $\delta\in(0,1)$ *and suppose* $\mathbf{w}_i^{(0)}\in\mathcal{G}_{\mathbf{w}}(\delta)$, $b_i^{(0)}\in\mathcal{G}_b(\delta)$, $a_i^{(0)}\in\mathcal{G}_a(\delta)$ *for all* $i\in[2m]$. *Let* $\epsilon_{e2}:=O\left(\frac{\eta^{(1)}|a_i^{(0)}|k(\gamma+\epsilon_e)}{\tilde{\sigma}^2\sigma_\xi^{(2)}}\right)+\exp(-\Theta(k))$. *If* $D\mu/\sqrt{d}\leqslant 1/16$, $\sigma_\zeta\tilde{\sigma}=O(1)$, $\sigma_\zeta^2 d/\tilde{\sigma}^2=O(1)$, $k=\Omega(\log^2(Dm/\delta))$, $\sigma_a\leqslant\tilde{\sigma}^2/(\gamma k^2)$, $\eta^{(1)}=O\left(\frac{\gamma}{km\sigma_a\tilde{\sigma}}\right)$, *and* $\sigma_\xi^{(2)}=1/k^{3/2}$, *then*

$$\frac{\partial}{\partial\mathbf{w}_i}L_{\mathcal{D}}(g^{(1)};\sigma_\xi^{(2)})=-a_i^{(1)}\left(\sum_{j=1}^{D}M_jT_j+\nu\right) \tag{A.478}$$

*where* $T_j$ *satisfies:*

- *if* $j\in A$, *then* $|T_j-\beta\gamma/\tilde{\sigma}|\leqslant O(\epsilon_{e2}/\tilde{\sigma}+\eta^{(1)}/\sigma_\xi^{(2)}+\eta^{(1)}|a_i^{(0)}|\epsilon_e/(\tilde{\sigma}\sigma_\xi^{(2)}))$, *where* $\beta\in[\Omega(1),1]$ *and depends only on* $\mathbf{w}_i^{(0)},b_i^{(0)}$;
- *if* $j\notin\mathbf{A}$, *then* $|T_j|\leqslant\frac{1}{\tilde{\sigma}}\exp(-\Omega(k))+O(\sigma_\phi^2\tilde{\sigma}\epsilon_{e2})$;
- $|\nu_j|\leqslant O\left(\frac{\eta^{(1)}\sigma_\zeta}{\gamma\sigma_\xi^{(2)}}\right)+\exp(-\Omega(k))$.

*Proof of Lemma A.40.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. By Lemma A.38, with probability at least $1-(d+D)\exp(-\Omega(k)) = 1-\exp(-\Omega(k))$ over $(\mathbf{x}, y)$, $yg^{(1)}(\mathbf{x};\xi^{(2)}) > 1$ and furthermore, for any $i \in [2m]$, $\left|\langle \mathbf{w}_i^{(1)}, \zeta/\tilde{\sigma}\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$, $\left|\langle q_i^{(1)}, \phi\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$, and $\left|\langle (q_i^{(1)})_{-\mathbf{A}}, \phi_{-\mathbf{A}}\rangle\right| = O(\eta^{(1)}\tilde{\sigma}/\gamma)$, and for any $j \in [d], \ell \in [D]$, $|\zeta_j| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$ and $|\langle \zeta, D_\ell\rangle| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$. Let $\mathbb{I}_\mathbf{x}$ be the indicator of this event.

$$\frac{\partial}{\partial \mathbf{w}} L_{\mathcal{D}}^{\alpha}(g^{(1)}; \sigma_\xi^{(2)}) \tag{A.479}$$

$$= -a^{(1)}\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y \mathbb{I}_\mathbf{x} \mathbb{E}_{\xi^{(2)}}\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\mathbf{x}\right\} \tag{A.480}$$

$$= -a^{(1)}\sum_{j=1}^{D} M_j \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\alpha_y y \mathbb{I}_\mathbf{x}\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j\right\}}_{:=T_j} \tag{A.481}$$

$$- a^{(1)}\underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\frac{\alpha_y y\zeta}{\tilde{\sigma}}\mathbb{I}_\mathbf{x}\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\right\}}_{:=\nu}. \tag{A.482}$$

Let $T_{j1} := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\alpha_y y\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j\right\}$. We have

$$|T_j - T_{j1}| \tag{A.483}$$

$$= \left|\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\alpha_y y(1-\mathbb{I}_\mathbf{x})\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j\right\}\right| \tag{A.484}$$

$$\leqslant \frac{1}{\tilde{\sigma}}\exp(-\Omega(k)). \tag{A.485}$$

Similarly, let $\nu' := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\frac{\alpha_y y\zeta}{\tilde{\sigma}}\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\right\}$. We have

$$|\nu - \nu'| \tag{A.486}$$

$$= \left|\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\frac{\alpha_y y\zeta}{\tilde{\sigma}}(1-\mathbb{I}_\mathbf{x})\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\right\}\right| \tag{A.487}$$

$$\leqslant \frac{\sigma_\zeta}{\tilde{\sigma}}\exp(-\Omega(k)). \tag{A.488}$$

So it is sufficient to bound $T_{j1}$ and $\nu'$. For simplicity, we use $q$ as a shorthand for $q_i^{(1)}$.

First, consider $j \in \mathbf{A}$.

$$T_{j1} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{\alpha_y y \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\phi_j\right\} \tag{A.489}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}}} \left\{\alpha_y y \phi_j \Pr_{\phi_{-\mathbf{A}},\xi^{(2)}} \left[\langle \phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\} \tag{A.490}$$

where $\iota := \langle \mathbf{w}^{(1)}, \zeta/\tilde{\sigma}\rangle$. Let

$$I_a := \Pr_{\xi^{(2)}} \left[\langle \phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right], \tag{A.491}$$

$$I'_a := \Pr_{\xi^{(2)}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]. \tag{A.492}$$

By the property of the Gaussian $\xi^{(2)}$, that $|\langle \phi_{\mathbf{A}}, q_{\mathbf{A}}\rangle| = O(\frac{\eta^{(1)}|a_i^{(0)}|k(\gamma+\epsilon_e)}{\tilde{\sigma}^2})$, and that $|\iota| = |\langle \mathbf{w}_i^{(1)}, \zeta/\tilde{\sigma}\rangle|, |\langle \phi, q\rangle|, |\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle|$ are all $O(\eta^{(1)}\tilde{\sigma}/\gamma) < O(1/k)$, we have

$$|I_a - I'_a| \tag{A.493}$$

$$\leqslant \left|\Pr_{\xi^{(2)}} \left[\langle \phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \geqslant 0\right] - \Pr_{\xi^{(2)}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(1)} + \xi^{(2)} \geqslant 0\right]\right| \tag{A.494}$$

$$+ \Pr_{\xi^{(2)}} \left[\langle \phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \geqslant 1\right] + \Pr_{\xi^{(2)}} \left[\langle \phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(1)} + \xi^{(2)} \geqslant 1\right] \tag{A.495}$$

$$= O\left(\frac{\eta^{(1)}|a_i^{(0)}|k(\gamma + \epsilon_e)}{\tilde{\sigma}^2\sigma_\xi^{(2)}}\right) + \exp(-\Theta(k)) = O(\epsilon_{e2}). \tag{A.496}$$

This leads to

$$\left|T_{j1} - \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}}\{\alpha_y y \phi_j I'_a\}\right| \tag{A.497}$$

$$\leqslant \mathbb{E}_{\phi_{\mathbf{A}}} \left\{|\alpha_y y \phi_j| \left|\mathbb{E}_{\phi_{-\mathbf{A}}}(I_a - I'_a)\right|\right\} \tag{A.498}$$

$$\leqslant O(\epsilon_{e2})\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y \phi_j|\} \tag{A.499}$$

$$\leqslant O(\epsilon_{e2}/\tilde{\sigma}) \tag{A.500}$$

where the last step is from Lemma A.33. Furthermore,

$$\mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}}\{\alpha_y y\phi_j I'_a\} \tag{A.501}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\phi_j\}\mathbb{E}_{\phi_{-\mathbf{A}}}[I'_a] \tag{A.502}$$

$$= \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\phi_j\} \Pr_{\phi_{-\mathbf{A}},\xi^{(2)}}\left[\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]. \tag{A.503}$$

By Lemma A.13, we have $|\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota| \leqslant O(\eta^{(1)}\tilde{\sigma}/\gamma)$. Also, $|b^{(1)} - b^{(0)}| \leqslant O(\eta^{(1)}|a_i^{(0)}|\epsilon_e)$. By the property of $\xi^{(2)}$,

$$\left|\Pr_{\xi^{(2)}}\left[\langle\phi_{-\mathbf{A}}, q_{-\mathbf{A}}\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right] - \Pr_{\xi^{(2)}}\left[b^{(0)} + \xi^{(2)} \in (0,1)\right]\right| \tag{A.504}$$

$$\leqslant O(\eta^{(1)}\tilde{\sigma}/(\gamma\sigma_\xi^{(2)})) + O(\eta^{(1)}|a_i^{(0)}|\epsilon_e/\sigma_\xi^{(2)}). \tag{A.505}$$

On the other hand,

$$\beta := \Pr_{\phi_{-\mathbf{A}},\xi^{(2)}}\left[b^{(0)} + \xi^{(2)} \in (0,1)\right] = \Pr_{\xi^{(2)}}\left[\xi^{(2)} \in (-b^{(0)}, 1 - b^{(0)})\right] \tag{A.506}$$

$$= \Omega(1) \tag{A.507}$$

and $\beta$ only depends on $b^{(0)}$. By Lemma A.33, $\mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\phi_j\} = \gamma/\tilde{\sigma}$. Therefore,

$$|T_{j1} - \beta\gamma/\tilde{\sigma}| \leqslant O(\epsilon_{e2}/\tilde{\sigma}) + O(\eta^{(1)}/\sigma_\xi^{(2)}) + O(\eta^{(1)}|a_i^{(0)}|\epsilon_e/(\tilde{\sigma}\sigma_\xi^{(2)})). \tag{A.508}$$

Now, consider $j \notin \mathbf{A}$. Let $B$ denote $\mathbf{A} \cup \{j\}$.

$$T_{j1} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\alpha_y y\phi_j \mathbb{I}\left[\langle\phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\} \tag{A.509}$$

$$= \mathbb{E}_{\phi_B}\mathbb{E}_{\phi_{-B},\zeta,\xi^{(2)}}\left\{\alpha_y y\phi_j \mathbb{I}\left[\langle\phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\} \tag{A.510}$$

$$= \mathbb{E}_{\phi_B}\left\{\alpha_y y\phi_j \Pr_{\phi_{-B},\zeta,\xi^{(2)}}\left[\langle\phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]\right\}. \tag{A.511}$$

Let

$$I_b := \Pr_{\xi^{(2)}}\left[\langle\phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right], \tag{A.512}$$

$$I'_b := \Pr_{\xi^{(2)}}\left[\langle \phi_{-B}, q_{-B}\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)\right]. \tag{A.513}$$

Similar as above, we have $|I_b - I'_b| \leqslant \epsilon_{e2}$. Then by Lemma A.33,

$$\begin{aligned}
&\left|T_{j1} - \mathbb{E}_{\phi_B,\phi_{-B},\zeta}\{\alpha_y y \phi_j I'_b\}\right| && \text{(A.514)}\\
&\leqslant \mathbb{E}_{\phi_B}\left\{|\alpha_y y \phi_j||\mathbb{E}_{\phi_{-B},\zeta}(I_b - I'_b)|\right\} && \text{(A.515)}\\
&\leqslant O(\epsilon_{e2})\mathbb{E}_{\phi_{\mathbf{A}}}\{|\alpha_y y|\}\mathbb{E}_{\phi_j}\{|\phi_j|\} && \text{(A.516)}\\
&\leqslant O(\epsilon_{e2}) \times O(\sigma_\phi^2 \tilde{\sigma}) && \text{(A.517)}\\
&= O(\sigma_\phi^2 \tilde{\sigma}\epsilon_{e2}). && \text{(A.518)}
\end{aligned}$$

Furthermore,

$$\mathbb{E}_{\phi_B,\phi_{-B},\zeta}\{\alpha_y y \phi_j I'_b\} = \mathbb{E}_{\phi_{\mathbf{A}}}\{\alpha_y y\}\mathbb{E}_{\phi_j}\{\phi_j\}\mathbb{E}_{\phi_{-B}}[I'_b] = 0. \tag{A.519}$$

Therefore,

$$|T_{j1}| \leqslant O(\sigma_\phi^2 \tilde{\sigma}\epsilon_{e2}). \tag{A.520}$$

Finally, consider $\nu'_j$.

$$\begin{aligned}
\nu'_j &= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}\mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)} \in (0,1)]\right\} && \text{(A.521)}\\
&= \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}},\zeta,\xi^{(2)}}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}\mathbb{I}[\langle \phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)]\right\} && \text{(A.522)}\\
&= \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}},\zeta}\left\{\frac{\alpha_y y \zeta_j}{\tilde{\sigma}}\Pr_{\xi^{(2)}}[\langle \phi, q\rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1)]\right\} && \text{(A.523)}
\end{aligned}$$

Let

$$\begin{aligned}
I_j &:= \Pr_{\xi^{(2)}}\left[\langle \phi, q\rangle + \iota + b^{(0)} + \xi^{(1)} \in (0,1)\right], && \text{(A.524)}\\
I'_j &:= \Pr_{\xi^{(2)}}\left[\langle \phi, q\rangle + b^{(0)} + \xi^{(1)} \in (0,1)\right]. && \text{(A.525)}
\end{aligned}$$

Since $|\iota| \leqslant O(\eta^{(1)}\tilde{\sigma}/\gamma)$, we have $|I_j - I'_j| \leqslant O(\eta^{(1)}\tilde{\sigma}/(\gamma\sigma_\xi^{(2)}))$. Then

$$\left|\nu'_j - \mathbb{E}_{\phi_A,\phi_{-A},\zeta}\left\{\frac{\alpha_y y\zeta_j}{\tilde{\sigma}} I'_j\right\}\right| \tag{A.526}$$

$$= \left|\mathbb{E}_{\phi_A,\phi_{-A},\zeta}\left\{\frac{\alpha_y y\zeta_j}{\tilde{\sigma}}(I_j - I'_j)\right\}\right| \tag{A.527}$$

$$\leqslant O(\eta^{(1)}\tilde{\sigma}/(\gamma\sigma_\xi^{(2)}))\mathbb{E}_{\phi_A,\phi_{-A},\zeta}\left|\frac{\alpha_y y\zeta_j}{\tilde{\sigma}}\right| \tag{A.528}$$

$$\leqslant O(\eta^{(1)}\tilde{\sigma}/(\gamma\sigma_\xi^{(2)}))\mathbb{E}_{\phi_A}|\alpha_y y|\,\mathbb{E}_\zeta\left|\frac{\zeta_j}{\tilde{\sigma}}\right| \tag{A.529}$$

$$\leqslant O(\eta^{(1)}\tilde{\sigma}/(\gamma\sigma_\xi^{(2)})) \times 1 \times \frac{\sigma_\zeta}{\tilde{\sigma}} \tag{A.530}$$

$$\leqslant O(\eta^{(1)}\sigma_\zeta/(\gamma\sigma_\xi^{(2)})). \tag{A.531}$$

Furthermore,

$$\mathbb{E}_{\phi_A,\phi_{-A},\zeta}\left\{\frac{\alpha_y y\zeta_j}{\tilde{\sigma}} I'_j\right\} = \mathbb{E}_{\phi_A,\phi_{-A}}\left\{\frac{\alpha_y y}{\tilde{\sigma}} I'_j\right\}\mathbb{E}_\zeta\{\zeta_j\} = 0. \tag{A.532}$$

Therefore,

$$|\nu_j| \leqslant O\left(\frac{\eta^{(1)}\sigma_\zeta}{\gamma\sigma_\xi^{(2)}}\right) + \exp(-\Omega(k)). \tag{A.533}$$

□

**Lemma A.41.** *Under the same assumptions as in Lemma A.40,*

$$\frac{\partial}{\partial b}L_{\mathcal{D}}(g^{(1)};\sigma_\xi^{(2)}) = -a_i^{(1)}T_b \tag{A.534}$$

*where* $|T_b| \leqslant \exp(-\Omega(k)) + O(\epsilon_{e2})$.

*Proof of Lemma A.41.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. By Lemma A.38, $\Pr[yg^{(1)}(\mathbf{x};\xi^{(2)}) > 1] \leqslant \exp(-\Omega(k))$. Let $\mathbb{I}_\mathbf{x} =$

$\mathbb{I}[yg^{(1)}(\mathbf{x};\xi^{(2)}) \leqslant 1]$.

$$\frac{\partial}{\partial b} L^{\alpha}_{\mathcal{D}}(g^{(1)}; \sigma^{(2)}_{\xi}) \tag{A.535}$$

$$= -a^{(1)} \underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} \left\{ \alpha_y y \mathbb{I}_{\mathbf{x}} \mathbb{E}_{\xi^{(2)}} \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x} \rangle + b^{(1)} + \xi^{(2)} \in (0,1)] \right\}}_{:=T_b}. \tag{A.536}$$

Let $T_{b1} := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ \alpha_y y \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x} \rangle + b^{(1)} + \xi^{(2)} \in (0,1)] \right\}$. We have

$$|T_b - T_{b1}| \tag{A.537}$$

$$= \left| \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ \alpha_y y (1 - \mathbb{I}_{\mathbf{x}}) \mathbb{I}[\langle \mathbf{w}^{(1)}, \mathbf{x} \rangle + b^{(1)} + \xi^{(2)} \in (0,1)] \right\} \right| \tag{A.538}$$

$$\leqslant \exp(-\Omega(k)). \tag{A.539}$$

So it is sufficient to bound $T_{b1}$. For simplicity, we use $q$ as a shorthand for $q^{(1)}_i$.

$$T_{b1} = \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D},\xi^{(2)}} \left\{ \alpha_y y \mathbb{I}\left[ \langle \phi, q \rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1) \right] \right\} \tag{A.540}$$

$$= \mathbb{E}_{\phi_A} \mathbb{E}_{\phi_{-A},\xi^{(2)}} \left\{ \alpha_y y \mathbb{I}\left[ \langle \phi, q \rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1) \right] \right\} \tag{A.541}$$

$$= \mathbb{E}_{\phi_A} \left\{ \alpha_y y \Pr_{\phi_{-A},\xi^{(2)}} \left[ \langle \phi, q \rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1) \right] \right\}. \tag{A.542}$$

Let

$$I_b := \Pr_{\xi^{(2)}} \left[ \langle \phi, q \rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1) \right], \tag{A.543}$$

$$I'_b := \Pr_{\xi^{(2)}} \left[ \langle \phi_{-\mathbf{A}}, q_{-A} \rangle + \iota + b^{(1)} + \xi^{(2)} \in (0,1) \right]. \tag{A.544}$$

Similar as in Lemma A.14, we have $|I_b - I'_b| \leqslant \epsilon_{e2}$. Then by Lemma A.33,

$$\left| T_{b1} - \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}} \{ \alpha_y y I'_b \} \right| \tag{A.545}$$

$$= \left| \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}} \{ \alpha_y y (I_b - I'_b) \} \right| \tag{A.546}$$

$$= O(\epsilon_{e2}) \mathbb{E}_{\phi_{\mathbf{A}},\phi_{-\mathbf{A}}} |\alpha_y y| \tag{A.547}$$

$$\leqslant O(\epsilon_{e2}). \tag{A.548}$$

Furthermore,

$$\mathbb{E}_{\phi_\mathbf{A},\phi_{-\mathbf{A}}}\{\alpha_y y I'_b\} = \mathbb{E}_{\phi_\mathbf{A}}\{\alpha_y y\}\mathbb{E}_{\phi_{-\mathbf{A}}}[I'_b] = 0. \tag{A.549}$$

Therefore, $|T_{b1}| \leqslant O(\epsilon_{e2})$ and the statement follows. □

**Lemma A.42.** *Under the same assumptions as in Lemma A.40,*

$$\frac{\partial}{\partial a_i} L_\mathcal{D}(g^{(1)}; \sigma_\xi^{(2)}) = -T_a \tag{A.550}$$

*where* $|T_a| = O\left(\frac{\eta^{(1)}\tilde{\sigma}}{\gamma p_{\min}}\right) + \exp(-\Omega(k))\mathrm{poly}\left(\frac{dD}{p_{\min}}\right)$.

*Proof of Lemma A.42.* Consider one neuron index $i$ and omit the subscript $i$ in the parameters. By Lemma A.38, $\Pr[yg^{(1)}(\mathbf{x};\xi^{(2)}) > 1] \leqslant \exp(-\Theta(k))$. Let $\mathbb{I}_\mathbf{x} = \mathbb{I}[yg^{(1)}(\mathbf{x};\xi^{(2)}) \leqslant 1]$.

$$\frac{\partial}{\partial a} L_\mathcal{D}^\alpha(g^{(1)}; \sigma_\xi^{(2)}) \tag{A.551}$$

$$= -\underbrace{\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y \mathbb{I}_\mathbf{x} \mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}}_{:=T_a}. \tag{A.552}$$

Let $T_{a1} := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y \mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}$. We have

$$|T_a - T_{a1}| \tag{A.553}$$

$$= \left|\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left\{\alpha_y y(1-\mathbb{I}_\mathbf{x}) \mathbb{E}_{\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}\right| \tag{A.554}$$

$$\leqslant \exp(-\Omega(k)). \tag{A.555}$$

So it is sufficient to bound $T_{a1}$. For simplicity, we use $q$ as a shorthand for $q_i^{(1)}$.

Let $\phi'_\mathbf{A}$ be an independent copy of $\phi_\mathbf{A}$, $\phi'$ be the vector obtained by replacing in $\phi$ the entries $\phi_\mathbf{A}$ with $\phi'_\mathbf{A}$, and let $x' = M\phi' + \zeta$ and its label is $y'$. Then

$$|T_{a1}| := \left|\mathbb{E}_{\phi_\mathbf{A}}\left\{\alpha_y y \mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(2)}}\sigma(\langle \mathbf{w}^{(1)}, \mathbf{x}\rangle + b^{(1)} + \xi^{(2)})\right\}\right| \tag{A.556}$$

$$\leqslant \frac{1}{2}\Bigg|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(2)}}\sigma(\langle\mathbf{w}^{(1)},\mathbf{x}\rangle + b^{(1)} + \xi^{(1)})|y=1\right\} \tag{A.557}$$

$$-\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(2)}}\sigma(\langle\mathbf{w}^{(1)},\mathbf{x}\rangle + b^{(1)} + \xi^{(2)})|y=-1\right\}\Bigg| \tag{A.558}$$

$$\leqslant \frac{1}{2}\Bigg|\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(2)}}\sigma(\langle\mathbf{w}^{(1)},\mathbf{x}\rangle + b^{(1)} + \xi^{(2)})|y=1\right\} \tag{A.559}$$

$$-\mathbb{E}_{\phi'_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}},\zeta,\xi^{(2)}}\sigma(\langle\mathbf{w}^{(1)},\mathbf{x}'\rangle + b^{(1)} + \xi^{(2)})|y'=-1\right\}\Bigg| \tag{A.560}$$

$$\leqslant \frac{1}{2}\mathbb{E}_{\phi_{\mathbf{A}},\phi'_{\mathbf{A}}}\left\{\mathbb{E}_{\phi_{-\mathbf{A}}}\left|\langle\mathbf{w}^{(1)},\mathbf{x}\rangle - \langle\mathbf{w}^{(1)},\mathbf{x}'\rangle\right| |y=1,y'=-1\right\} \tag{A.561}$$

$$\leqslant \frac{1}{2}\mathbb{E}_{\phi_{-\mathbf{A}}}\left(\mathbb{E}_{\phi_{\mathbf{A}}}\left\{\left|\langle\mathbf{w}^{(1)},M\phi\rangle\right| |y=1\right\} + \mathbb{E}_{\phi'_{\mathbf{A}}}\left\{\left|\langle\mathbf{w}^{(1)},M\phi'\rangle\right| |y'=-1\right\}\right) \tag{A.562}$$

$$\leqslant \mathbb{E}_{\phi_{-\mathbf{A}},\phi_{\mathbf{A}}}\left|\alpha_y\langle\mathbf{w}^{(1)},M\phi\rangle\right| \tag{A.563}$$

$$= \mathbb{E}_{\phi}\left|\alpha_y\langle\mathbf{w}^{(1)},M\phi\rangle\right| \tag{A.564}$$

$$= O(\eta^{(1)}\tilde{\sigma}/\gamma) + \exp(-\Omega(k)) \times \frac{\sqrt{dD}}{\tilde{\sigma}} \times \|\mathbf{w}^{(1)}\|_\infty \tag{A.565}$$

$$= O\left(\frac{\eta^{(1)}\tilde{\sigma}}{\gamma p_{\min}}\right) + \exp(-\Omega(k))\mathrm{poly}(dD/p_{\min}) \tag{A.566}$$

where the fourth step follows from that $\sigma$ is 1-Lipschitz, and the second to the last line from Lemma A.38 and that $\left|\langle\mathbf{w}^{(1)},M\phi\rangle\right| \leqslant \|\mathbf{w}^{(1)}\|_\infty\sqrt{d\|M\phi\|_2^2}$. □

With the above lemmas about the gradients, we are now ready to show that at the end of the second step, we get a good set of features for accurate prediction.

**Lemma A.43.** *Set*

$$\eta^{(1)} = \frac{\gamma^2 p_{\min}\tilde{\sigma}}{km^3}, \lambda_a^{(1)} = 0, \lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \sigma_\xi^{(1)} = 1/k^{3/2}, \tag{A.567}$$

$$\eta^{(2)} = 1, \lambda_a^{(2)} = \lambda_{\mathbf{w}}^{(2)} = 1/(2\eta^{(2)}), \sigma_\xi^{(2)} = 1/k^{3/2}. \tag{A.568}$$

*Fix* $\delta \in (0, O(1/k^3))$. *If* $D\mu/\sqrt{d} \leqslant 1/16$, $\sigma_\zeta\tilde{\sigma} = O(1)$, $\sigma_\zeta^2 d/\tilde{\sigma}^2 = O(1)$, $k =$

$\Omega\left(\log^2\left(\frac{Dmd}{\delta\gamma p_{\min}}\right)\right)$*, and* $m \geqslant \max\{\Omega(k^4), D, d\}$*, then with probability at least* $1-\delta$ *over the initialization, there exist* $\tilde{a}_i$*'s such that* $\tilde{g}(\mathbf{x}) := \sum_{i=1}^{2m} \tilde{a}_i \sigma(\langle \mathbf{w}_i^{(2)}, \mathbf{x}\rangle + b_i^{(2)})$ *satisfies* $L_{\mathcal{D}}(\tilde{g}) \leqslant \exp(-\Omega(k))$*. Furthermore,* $\|\tilde{a}\|_0 = O(m/k)$*,* $\|\tilde{a}\|_\infty = O(k^5/m)$*, and* $\|\tilde{a}\|_2^2 = O(k^9/m)$*. Finally,* $\|a^{(2)}\|_\infty = O\left(\frac{1}{km^2}\right)$*,* $\|\mathbf{w}_i^{(2)}\|_2 = O(\tilde{\sigma}/k)$*, and* $|b_i^{(2)}| = O(1/k^2)$ *for all* $i \in [2m]$*.*

*Proof of Lemma A.43.* By Lemma A.29, there exists a network

$$g^*(\mathbf{x}) = \sum_{\ell=1}^{3(k+1)} a_\ell^* \sigma(\langle \mathbf{w}_\ell^*, \mathbf{x}\rangle + b_\ell^*)$$

satisfying

$$\Pr_{(\mathbf{x},y)\sim\mathcal{D}}[yg^*(x) \leqslant 1] \leqslant \exp(-\Omega(k)).$$

Furthermore, the number of neurons $n = 3(k+1)$, $|a_i^*| \leqslant 64k, 1/(64k) \leqslant |b_i^*| \leqslant 1/4$, $\mathbf{w}_i^* = \tilde{\sigma}\sum_{j\in\mathbf{A}} M_j/(8k)$, and $|\langle \mathbf{w}_i^*, \mathbf{x}\rangle + b_i^*| \leqslant 1$ for any $i \in [n]$ and $(\mathbf{x}, y) \sim \mathcal{D}$. Now we fix an $\ell$, and show that with high probability there is a neuron in $g^{(2)}$ that can approximate the $\ell$-th neuron in $g^*$.

With probability $\geqslant 1 - \exp(-\Omega(\max\{2p_o(D-k), k\}))$, among all $j \notin \mathbf{A}$, we have that at most $2p_o(D-k)+k$ of $\phi_j$ are $(1-p_o)/\tilde{\sigma}$, while the others are $-p_o/\tilde{\sigma}$. With probability $\geqslant 1-(d+D)\exp(-\Omega(k))$ over $\zeta$, for any $j$, $|\zeta_j| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$ and $|\langle \zeta, D_\ell\rangle| \leqslant O(\sigma_\zeta\sqrt{\log(k)})$. Below we consider data points with $\phi$ and $\zeta$ satisfying these.

By Lemma A.31, with probability $1-2\delta$ over $\mathbf{w}_i^{(0)}$'s, they are all in $\mathcal{G}_{\mathbf{w}}(\delta)$; with probability $1-\delta$ over $a_i^{(0)}$'s, they are all in $\mathcal{G}_a(\delta)$; with probability $1-\delta$ over $b_i^{(0)}$'s, they are all in $\mathcal{G}_b(\delta)$. Under these events, by Lemma A.37, Lemma A.40 and A.41, for any neuron $i \in [2m]$, we have

$$\mathbf{w}_i^{(2)} = a_i^{(1)}\left(\sum_{j=1}^{D} M_j T_j + \nu\right), \tag{A.569}$$

$$b_i^{(2)} = b_i^{(1)} + a_i^{(1)} T_b. \tag{A.570}$$

where

- if $j \in A$, then $|T_j - \beta\gamma/\tilde{\sigma}| \leqslant \epsilon_{\mathbf{w}1} := O(\epsilon_{e2}/\tilde{\sigma} + \eta^{(1)}/\sigma_{\xi}^{(2)} + \eta^{(1)}|a_i^{(0)}|\epsilon_e/(\tilde{\sigma}\sigma_{\xi}^{(2)}))$, where $\beta \in [\Omega(1), 1]$ and depends only on $\mathbf{w}_i^{(0)}, b_i^{(0)}$;
- if $j \notin \mathbf{A}$, then $|T_j| \leqslant \epsilon_{\mathbf{w}2} := \frac{1}{\tilde{\sigma}}\exp(-\Omega(k)) + O(\sigma_{\phi}^2\tilde{\sigma}\epsilon_{e2})$;
- $|\nu_j| \leqslant \epsilon_{\nu} := O\left(\frac{\eta^{(1)}\sigma_{\zeta}}{\gamma\sigma_{\xi}^{(2)}}\right) + \exp(-\Omega(k))$.
- $|T_b| \leqslant \epsilon_b := \exp(-\Omega(k)) + O(\epsilon_{e2})$.

Given the initialization, with probability $\Omega(1)$ over $b_i^{(0)}$, we have

$$|b_i^{(0)}| \in \left[\frac{1}{2k^2}, \frac{2}{k^2}\right], \operatorname{sign}(b_i^{(0)}) = \operatorname{sign}(b_\ell^*). \tag{A.571}$$

Finally, since $\frac{8k|b_\ell^*|\beta\gamma}{|b_i^{(0)}|\tilde{\sigma}^2} \in [\Omega(k^2\gamma/\tilde{\sigma}^2), O(k^3\gamma/\tilde{\sigma}^2)]$ and depends only on $\mathbf{w}_i^{(0)}, b_i^{(0)}$, we have that for $\epsilon_a = \Theta(1/k^2)$, with probability $\Omega(\epsilon_a) > \delta$ over $a_i^{(0)}$,

$$\left|\frac{8k|b_\ell^*|\beta\gamma}{|b_i^{(0)}|\tilde{\sigma}^2}a_i^{(0)} - 1\right| \leqslant \epsilon_a, \quad |a_i^{(0)}| = O\left(\frac{\tilde{\sigma}^2}{k^2\gamma}\right). \tag{A.572}$$

Let $n_a = \epsilon_a m/4$. For the given value of $m$, by (A.569)-(A.572) we have with probability $\geqslant 1 - 5\delta$ over the initialization, for each $\ell$ there is a different set of neurons $I_\ell \subseteq [m]$ with $|I_\ell| = n_a$ and such that for each $i_\ell \in I_\ell$,

$$|b_{i_\ell}^{(0)}| \in \left[\frac{1}{2k^2}, \frac{2}{k^2}\right], \quad \operatorname{sign}(b_{i_\ell}^{(0)}) = \operatorname{sign}(b_\ell^*), \tag{A.573}$$

$$\left|\frac{8k|b_\ell^*|\beta\gamma}{|b_{i_\ell}^{(0)}|\tilde{\sigma}^2}a_{i_\ell}^{(0)} - 1\right| \leqslant \epsilon_a, \quad |a_{i_\ell}^{(0)}| = O\left(\frac{\tilde{\sigma}^2}{k^2\gamma}\right). \tag{A.574}$$

Now, construct $\tilde{a}$ such that $\tilde{a}_{i_\ell} = \frac{2a_\ell^*|b_\ell^*|}{|b_{i_\ell}^{(0)}|n_a}$ for each $\ell$ and each $i_\ell \in I_\ell$, and $\tilde{a}_i = 0$ elsewhere. To show that it gives accurate predictions, we first consider bounding some error terms.

For the given values of parameters, we have

$$\epsilon_{e2} = O\left(\frac{\gamma}{m^2}\right), \tag{A.575}$$

$$\epsilon_{\mathbf{w}1} = O\left(\frac{k\gamma}{m^2\tilde{\sigma}} + \frac{\gamma\epsilon_e}{km^2}\right), \tag{A.576}$$

$$\epsilon_{\mathbf{w}2} = O\left(\frac{\gamma}{m^2\tilde{\sigma}}\right), \tag{A.577}$$

$$\epsilon_{v} = O\left(\frac{\gamma k}{m^3}\right), \tag{A.578}$$

$$\epsilon_{b} = O\left(\frac{\gamma}{m^2}\right). \tag{A.579}$$

We also have the following useful claims.

**Claim A.44.** $\sum_{\ell \in \mathbf{A}} |\langle M_\ell, \mathbf{x}\rangle| \leqslant O\left(\frac{k}{\tilde{\sigma}}\right)$.

*Proof of Claim A.44.*

$$\sum_{\ell \in \mathbf{A}} |\langle M_\ell, \mathbf{x}\rangle| \tag{A.580}$$

$$\leqslant \sum_{\ell \in \mathbf{A}} \left( |\phi_j| + \left| \sum_{j \neq \ell} M_\ell^\top M_j \phi_j \right| + \left| M_\ell^\top \zeta / \tilde{\sigma} \right| \right) \tag{A.581}$$

$$\leqslant O\left(\frac{k}{\tilde{\sigma}}\right) + O\left(kD\frac{\mu}{\sqrt{d}\tilde{\sigma}}\right) + O\left(k\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}}\right) \tag{A.582}$$

$$\leqslant O\left(\frac{k}{\tilde{\sigma}}\right). \tag{A.583}$$

□

**Claim A.45.**

$$\left| \langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle \right| \leqslant O\left(\frac{1}{m}\right). \tag{A.584}$$

*Proof of Claim A.45.*

$$\left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle\right| = \left|\langle \sum_{\ell=1}^{D} a_{i_\ell}^{(1)} T_\ell M_\ell + \upsilon, \mathbf{x}\rangle\right| \tag{A.585}$$

$$\leqslant \left|\langle \sum_{\ell\in\mathbf{A}} a_{i_\ell}^{(1)} T_\ell M_\ell, \mathbf{x}\rangle\right| + \left|\langle \sum_{\ell\notin\mathbf{A}} a_{i_\ell}^{(1)} T_\ell M_\ell, \mathbf{x}\rangle\right| + |\langle \upsilon, \mathbf{x}\rangle|. \tag{A.586}$$

Then

$$\left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \tag{A.587}$$

$$\leqslant \left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| + \left|\frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \tag{A.588}$$

$$\leqslant \left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| + \left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right| \left|\frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right|. \tag{A.589}$$

The first term is

$$\left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(1)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \tag{A.590}$$

$$\leqslant \left|a_{i_\ell}^{(1)}\right| \left( \left|\langle \sum_{\ell\in\mathbf{A}} \left(T_\ell - \frac{\beta\gamma}{\tilde{\sigma}}\right) M_\ell, \mathbf{x}\rangle\right| + \left|\langle \sum_{\ell\notin\mathbf{A}} T_\ell M_\ell, \mathbf{x}\rangle\right| + |\langle \nu, \mathbf{x}\rangle| \right). \tag{A.591}$$

By Claim A.44,

$$\left|\langle \sum_{\ell\in\mathbf{A}} \left(T_\ell - \frac{\beta\gamma}{\tilde{\sigma}}\right) M_\ell, \mathbf{x}\rangle\right| \leqslant \sum_{\ell\in\mathbf{A}} \left|T_\ell - \frac{\beta\gamma}{\tilde{\sigma}}\right| |\langle M_\ell, \mathbf{x}\rangle| \tag{A.592}$$

$$\leqslant \epsilon_{w1} \sum_{\ell\in\mathbf{A}} |\langle M_\ell, \mathbf{x}\rangle| \tag{A.593}$$

$$\leqslant O\left(\frac{k\epsilon_{\mathbf{w}1}}{\tilde{\sigma}}\right). \tag{A.594}$$

$$\left|\langle\sum_{\ell\notin\mathbf{A}} T_\ell M_\ell, \mathbf{x}\rangle\right| \leqslant \left|\sum_{\ell\notin\mathbf{A}} T_\ell \phi_j\right| + \left|\sum_{\ell\notin\mathbf{A}, j\neq\ell} T_\ell M_\ell^\top M_j \phi_j\right| + \left|\sum_{\ell\notin\mathbf{A}} T_\ell M_\ell^\top \zeta/\tilde{\sigma}\right| \tag{A.595}$$

$$\leqslant O\left(\frac{D\epsilon_{\mathbf{w}2}}{\tilde{\sigma}}\right) + O\left(D^2\epsilon_{\mathbf{w}2}\frac{\mu}{\sqrt{d}\tilde{\sigma}}\right) + O\left(D\epsilon_{\mathbf{w}2}\frac{\sigma_\zeta\sqrt{\log(k)}}{\tilde{\sigma}}\right) \tag{A.596}$$

$$\leqslant O\left(\frac{D\epsilon_{\mathbf{w}2}}{\tilde{\sigma}}\right). \tag{A.597}$$

$$|\langle\nu, \mathbf{x}\rangle| \leqslant \left|\langle\nu, \sum_{\ell\in[D]} M_\ell\phi_\ell\rangle\right| + |\langle\nu, \zeta/\tilde{\sigma}\rangle| \tag{A.598}$$

$$\leqslant \sum_{\ell\in[D]} |\phi_\ell\langle\nu, M_\ell\rangle| + |\langle\nu, \zeta/\tilde{\sigma}\rangle| \tag{A.599}$$

$$\leqslant O\left(\frac{p_o D + k}{\tilde{\sigma}}\right)\epsilon_\nu\sqrt{d} + d\epsilon_\nu\frac{O(\sigma_\zeta\sqrt{\log(k)})}{\tilde{\sigma}} \tag{A.600}$$

$$\leqslant O\left(\frac{\epsilon_\nu\sqrt{d}}{\tilde{\sigma}}\right)\left(p_o D + \sqrt{d}\epsilon_\nu\sqrt{\log(k)}\right) \tag{A.601}$$

$$\leqslant O\left(\frac{\epsilon_\nu\sqrt{d}}{\tilde{\sigma}}\right)\left(p_o D + \tilde{\sigma}\sqrt{\log(k)}\right) \tag{A.602}$$

$$\leqslant O\left(\frac{p_o D\epsilon_\nu\sqrt{d}}{\tilde{\sigma}}\right). \tag{A.603}$$

Then by (A.575)-(A.579),

$$\epsilon_\phi := \left(\frac{k\epsilon_{\mathbf{w}1}}{\tilde{\sigma}} + \frac{D\epsilon_{\mathbf{w}2}}{\tilde{\sigma}} + \frac{p_o D\epsilon_\nu\sqrt{d}}{\tilde{\sigma}}\right) = O\left(\frac{k^2\gamma}{m^2\tilde{\sigma}^2} + \frac{\gamma\epsilon_e}{m^2\tilde{\sigma}} + \frac{\gamma}{m\tilde{\sigma}^2} + \frac{\gamma k}{m^{3/2}\tilde{\sigma}}\right). \tag{A.604}$$

We have $\left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right| = O(\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)})$. So the first term is bounded by

$$\left|\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)}\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \leqslant \left|a_{i_\ell}^{(1)}\right| \epsilon_\phi \tag{A.605}$$

$$\leqslant O\left(\frac{\tilde{\sigma}^2}{k^2\gamma} + \eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}\right)\left(\frac{k^2\gamma}{m^2\tilde{\sigma}^2} + \frac{\gamma\epsilon_e}{m^2\tilde{\sigma}} + \frac{\gamma}{m\tilde{\sigma}^2} + \frac{\gamma k}{m^{3/2}\tilde{\sigma}}\right) \leqslant O\left(\frac{1}{m}\right). \tag{A.606}$$

By Claim A.44, the second term is bounded by

$$\left|a_{i_\ell}^{(1)} - a_{i_\ell}^{(0)}\right| \left|\frac{\beta\gamma}{\tilde{\sigma}} \sum_{j\in\mathbf{A}} \langle M_j, \mathbf{x}\rangle\right| \leqslant O\left(\frac{k\eta^{(1)}\sigma_{\mathbf{w}}\sqrt{\log(Dm/\delta)}\gamma}{\tilde{\sigma}^2}\right) \leqslant O\left(\frac{\gamma^3}{m^3}\right). \tag{A.607}$$

Combining the bounds on the two terms leads to the claim. □

**Claim A.46.**

$$|b_{i_\ell}^{(2)} - b_{i_\ell}^{(0)}| \leqslant O\left(\frac{1}{k^2 m}\right). \tag{A.608}$$

*Proof of Claim A.46.* By Lemma A.37 and A.41:

$$|b_{i_\ell}^{(2)} - b_{i_\ell}^{(0)}| \leqslant |b_{i_\ell}^{(2)} - b_{i_\ell}^{(1)}| + |b_{i_\ell}^{(1)} - b_{i_\ell}^{(0)}| \tag{A.609}$$

$$\leqslant O\left(\eta^{(1)}|a_{i_\ell}^{(0)}|\epsilon_e + |a_{i_\ell}^{(1)}|\left(\exp(-\Omega(k)) + \epsilon_{e2}\right)\right) \tag{A.610}$$

$$\leqslant O\left(\frac{\gamma}{km^2} + \frac{1}{k^2 m}\right) \leqslant O\left(\frac{1}{k^2 m}\right). \tag{A.611}$$

□

We are now ready to show $\tilde{g}$ is close to $2g^*$.

$$|\tilde{g}(\mathbf{x}) - 2g^*(\mathbf{x})| \tag{A.612}$$

$$= \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \tilde{a}_{i_\ell} \sigma\left(\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle + b_{i_\ell}^{(2)}\right) - \sum_{\ell=1}^{3(k+1)} 2a_\ell^* \sigma\left(\langle \mathbf{w}_\ell^*, \mathbf{x}\rangle + b_\ell^*\right) \right| \tag{A.613}$$

$$= \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}| n_a} \sigma\left(\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle + b_{i_\ell}^{(2)}\right) - \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}| n_a} \sigma\left(\frac{|b_{i_\ell}^{(0)}|}{|b_\ell^*|} \langle \mathbf{w}_\ell^*, \mathbf{x}\rangle + b_{i_\ell}^{(0)}\right) \right| \tag{A.614}$$

$$\leqslant \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{1}{n_a} \left( \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left(\langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle + b_{i_\ell}^{(2)}\right) - \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle + b_{i_\ell}^{(0)} \right) \right) \right| \tag{A.615}$$

$$+ \left| \sum_{\ell=1}^{3(k+1)} \sum_{i_\ell \in I_\ell} \frac{1}{n_a} \left( \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle + b_{i_\ell}^{(0)} \right) - \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \sigma\left( \frac{|b_{i_\ell}^{(0)}|}{|b_\ell^*|} \langle \mathbf{w}_\ell^*, \mathbf{x}\rangle + b_{i_\ell}^{(0)} \right) \right) \right|. \tag{A.616}$$

Here the second equation follows from that $\sigma$ is positive-homogeneous in $[0,1]$, $|\langle \mathbf{w}_\ell^*, \mathbf{x}\rangle + b_\ell^*| \leqslant 1$, $|b_{i_\ell}^{(0)}|/|b_\ell^*| \leqslant 1$.

By Claim A.45 and A.46, the first term is bounded by:

$$3(k+1) \max_\ell \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \left( \left| \langle \mathbf{w}_{i_\ell}^{(2)}, \mathbf{x}\rangle - \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle \right| + |b_{i_\ell}^{(2)} - b_{i_\ell}^{(0)}| \right) \tag{A.617}$$

$$\leqslant 3(k+1) \max_\ell \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} O\left(\frac{1}{m}\right) \tag{A.618}$$

$$\leqslant O\left(\frac{k^4}{m}\right). \tag{A.619}$$

By Claim A.44, the second term is bounded by:

$$3(k+1) \max_\ell \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \left| \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle - \frac{|b_{i_\ell}^{(0)}|}{|b_\ell^*|} \langle \mathbf{w}_\ell^*, \mathbf{x}\rangle \right| \tag{A.620}$$

$$\leqslant 3(k+1) \max_\ell \frac{2a_\ell^* |b_\ell^*|}{|b_{i_\ell}^{(0)}|} \left| \frac{a_{i_\ell}^{(0)} \beta\gamma}{\tilde{\sigma}} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle - \frac{|b_{i_\ell}^{(0)}| \tilde{\sigma}}{8k |b_\ell^*|} \sum_{j \in \mathbf{A}} \langle M_j, \mathbf{x}\rangle \right| \tag{A.621}$$

$$\leqslant 3(k+1)\max_{\ell} \frac{2a_\ell^*|b_\ell^*|}{|b_{i_\ell}^{(0)}|} \frac{\tilde{\sigma}|b_{i_\ell}^{(0)}|}{8k|b_\ell^*|} \left| \frac{8ka_{i_\ell}\beta\gamma|b_\ell^*|}{\tilde{\sigma}^2|b_{i_\ell}^{(0)}|} - 1 \right| \left| \sum_{j\in \mathbf{A}} \langle M_j, \mathbf{x}\rangle \right| \tag{A.622}$$

$$\leqslant 3(k+1)\max_{\ell} O(a_\ell^* \epsilon_a) \tag{A.623}$$

$$\leqslant O\left(k^2\epsilon_a\right). \tag{A.624}$$

Then

$$|\tilde{g}(\mathbf{x}) - 2g^*(\mathbf{x})| = O\left(\frac{k^4}{m} + k^2\epsilon_a\right) \leqslant 1. \tag{A.625}$$

This guarantees $y\tilde{g}(\mathbf{x}) \geqslant 1$. Changing the scaling of $\delta$ leads to the statement.

Finally, the bounds on $\tilde{a}$ follow from the above calculation. The bound on $\|a^{(2)}\|_2$ follows from Lemma A.42, and those on $\|\mathbf{w}_i^{(2)}\|_2$ and $\|b_i^{(2)}\|_2$ follow from (A.569)(A.570) and the bounds on $a_i^{(1)}$ and $b_i^{(1)}$ in Lemma A.37. □

### A.8.9 Classifier Learning Stage and Main Theorem

Once we have a good set of features in Lemma A.43, we can follow exactly the same argument as in Section A.4.6 and A.4.7 for the simplified setting, and arrive at the main theorem for the general setting:

**Theorem A.47** (Restatement of Theorem A.28)**.** *Set*

$$\eta^{(1)} = \frac{\gamma^2 p_{\min}\tilde{\sigma}}{km^3}, \lambda_a^{(1)} = 0, \lambda_{\mathbf{w}}^{(1)} = 1/(2\eta^{(1)}), \sigma_\xi^{(1)} = 1/k^{3/2}, \tag{A.626}$$

$$\eta^{(2)} = 1, \lambda_a^{(2)} = \lambda_{\mathbf{w}}^{(2)} = 1/(2\eta^{(2)}), \sigma_\xi^{(2)} = 1/k^{3/2}, \tag{A.627}$$

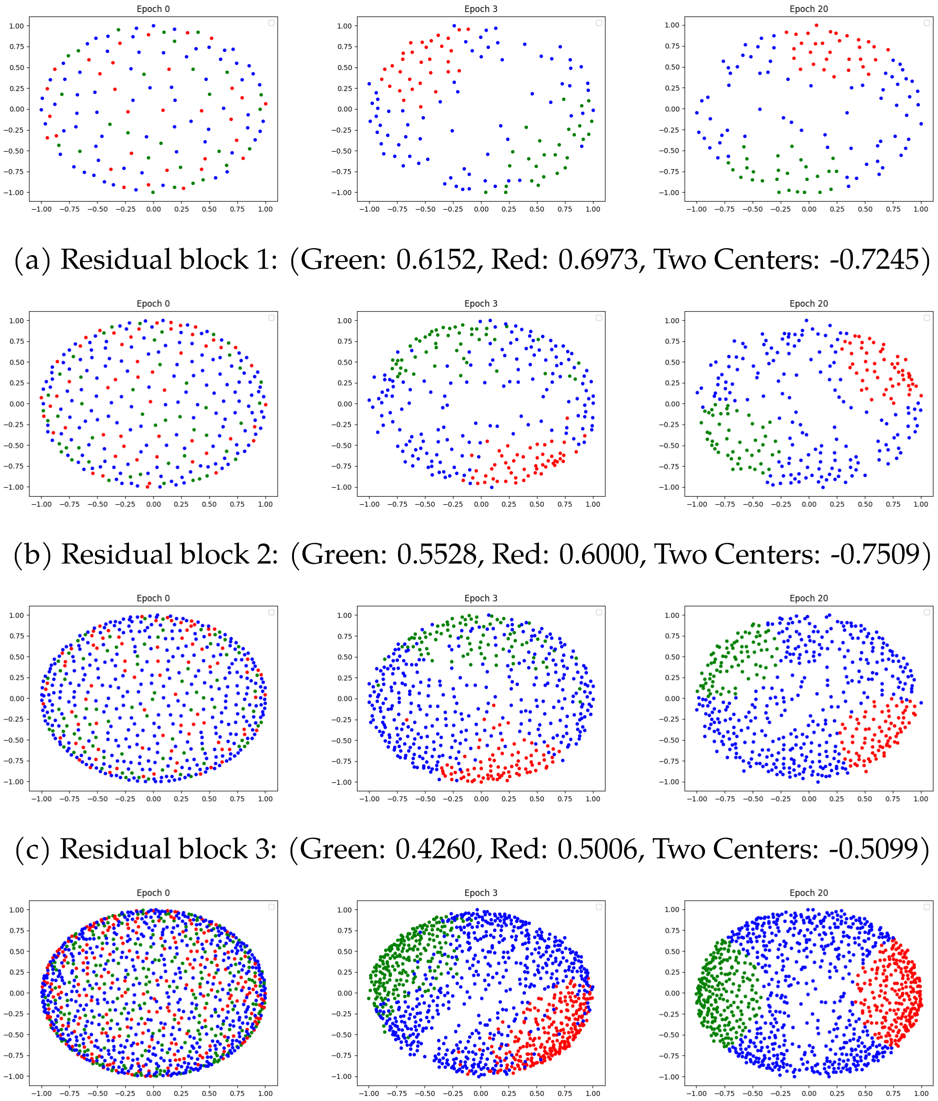
$$\eta^{(t)} = \eta = \frac{k^2}{Tm^{1/3}}, \lambda_a^{(t)} = \lambda_{\mathbf{w}}^{(t)} = \lambda \leqslant \frac{k^3}{\tilde{\sigma}m^{1/3}}, \sigma_\xi^{(t)} = 0, \text{ for } 2 < t \leqslant T. \tag{A.628}$$

*For any* $\delta \in (0, O(1/k^3))$, *if* $\mu \leqslant O(\sqrt{d}/D)$, $\sigma_\zeta \leqslant O(\min\{1/\tilde{\sigma}, \tilde{\sigma}/\sqrt{d}\})$, $k = \Omega\left(\log^2\left(\frac{Dmd}{\delta\gamma p_{\min}}\right)\right)$, $m \geqslant \max\{\Omega(k^4), D, d\}$, *then we have for any* $\mathcal{D} \in \mathcal{F}_\Xi$, *with probability at least* $1-\delta$, *there exists* $t \in [T]$ *such that*

$$\Pr[\mathrm{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) = O\left(\frac{k^8}{m^{2/3}} + \frac{k^3 T}{m^2} + \frac{k^2 m^{2/3}}{T}\right). \tag{A.629}$$

*Consequently, for any* $\epsilon \in (0,1)$*, if* $T = m^{4/3}$*, and* $m \geqslant \max\{\Omega(k^{12}/\epsilon^{3/2}), D\}$*, then*

$$\Pr[\mathrm{sign}(g^{(t)}(\mathbf{x})) \neq y] \leqslant L_{\mathcal{D}}(g^{(t)}) \leqslant \epsilon. \tag{A.630}$$



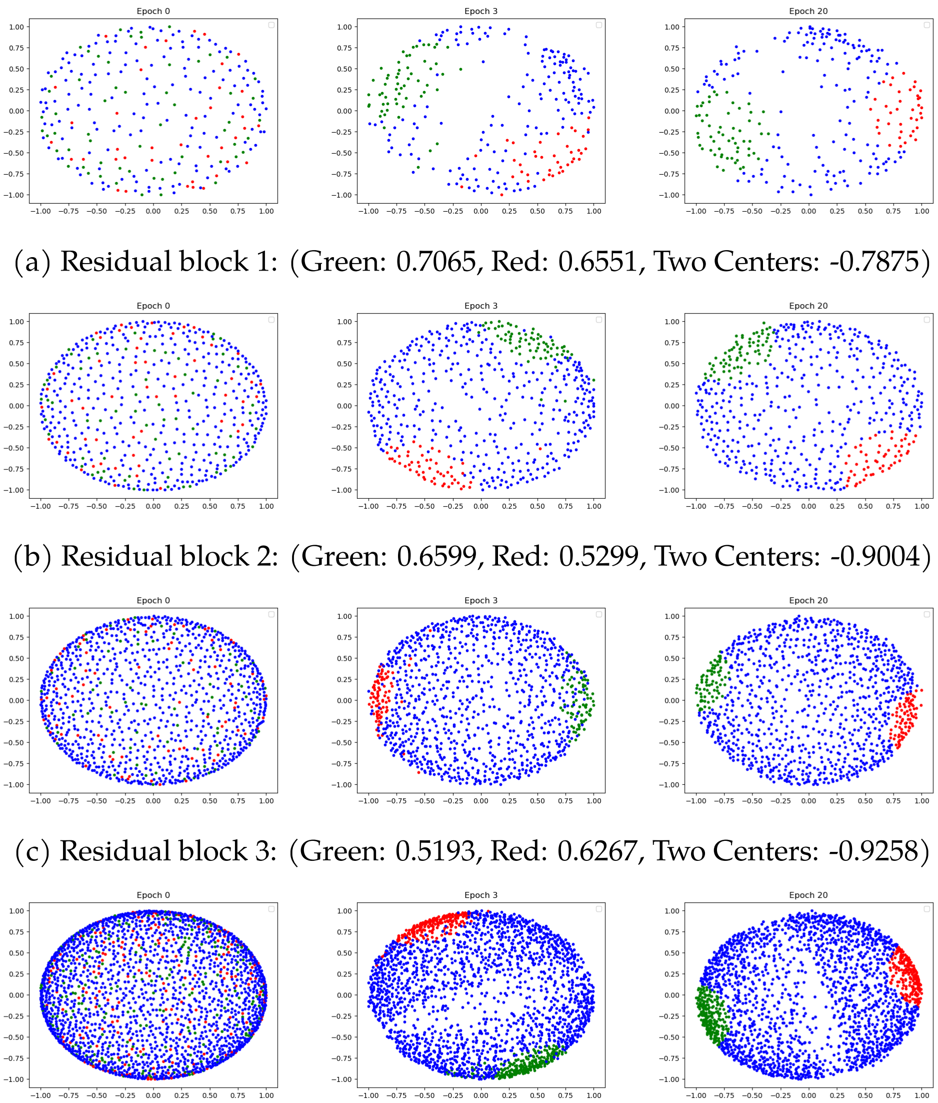
(a) Residual block 1: (Green: 0.6152, Red: 0.6973, Two Centers: -0.7245)

(b) Residual block 2: (Green: 0.5528, Red: 0.6000, Two Centers: -0.7509)

(c) Residual block 3: (Green: 0.4260, Red: 0.5006, Two Centers: -0.5099)

(d) Residual block 4: (Green: 0.5584, Red: 0.5697, Two Centers: -0.9074)

Figure A.16: Visualization of the normalized convolution weights in all Residual block of ResNet(128) trained on the subset of CIFAR10 data with labels airplane/automobile. We show the weights after 0/3/20 epochs in network learning. The weights gradually form two clusters in all Residual blocks. We also report average cosine similarity between the green/red points in the clusters to their centers and cosine similarity between two cluster centers as (Green, Red, Two Centers).



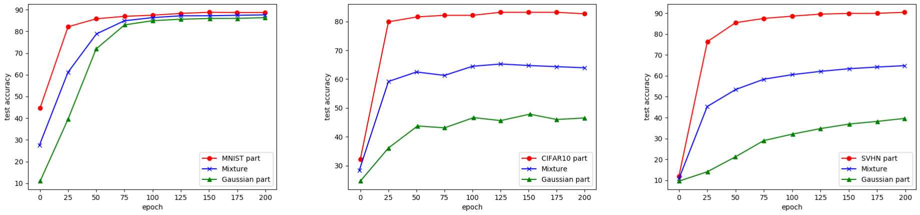
(a) Residual block 1: (Green: 0.7065, Red: 0.6551, Two Centers: -0.7875)

(b) Residual block 2: (Green: 0.6599, Red: 0.5299, Two Centers: -0.9004)

(c) Residual block 3: (Green: 0.5193, Red: 0.6267, Two Centers: -0.9258)

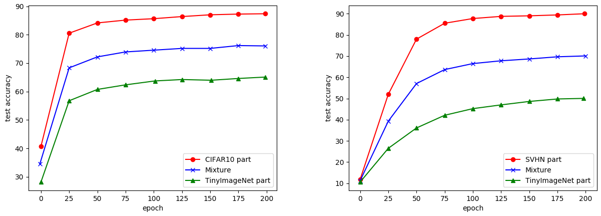
(d) Residual block 4: (Green: 0.7386, Red: 0.6839, Two Centers: -0.9740)

Figure A.17: Visualization of the normalized convolution weights in all Residual block of ResNet(256) trained on the subset of CIFAR10 data with labels airplane/automobile. We show the weights after 0/3/20 epochs in network learning. The weights gradually form two clusters in all Residual blocks. We also report average cosine similarity between the green/red points in the clusters to their centers and cosine similarity between two cluster centers as (Green, Red, Two Centers).



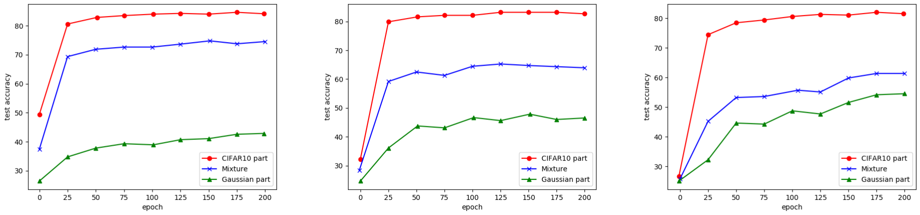
(a) (b) (c)

Figure A.18: Test accuracy at different steps for an equal mixture $\alpha = 0.5$ of Gaussian inputs with data: (a) MNIST, (b) CIFAR10, (c) SVHN.



(a) (b)

Figure A.19: Test accuracy at different steps for an equal mixture $\alpha = 0.5$ of Tiny ImageNet inputs with data: (a) CIFAR10, (b) SVHN.



(a) $\alpha = 0.25$ (b) $\alpha = 0.50$ (c) $\alpha = 0.75$

Figure A.20: Test accuracy at different steps for varying mixture $\alpha$ of Gaussian inputs with CIFAR10.

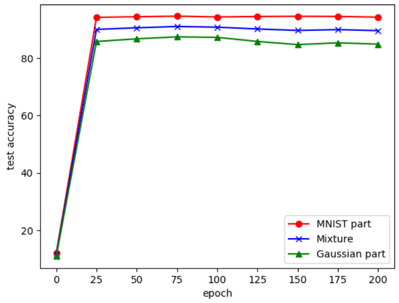
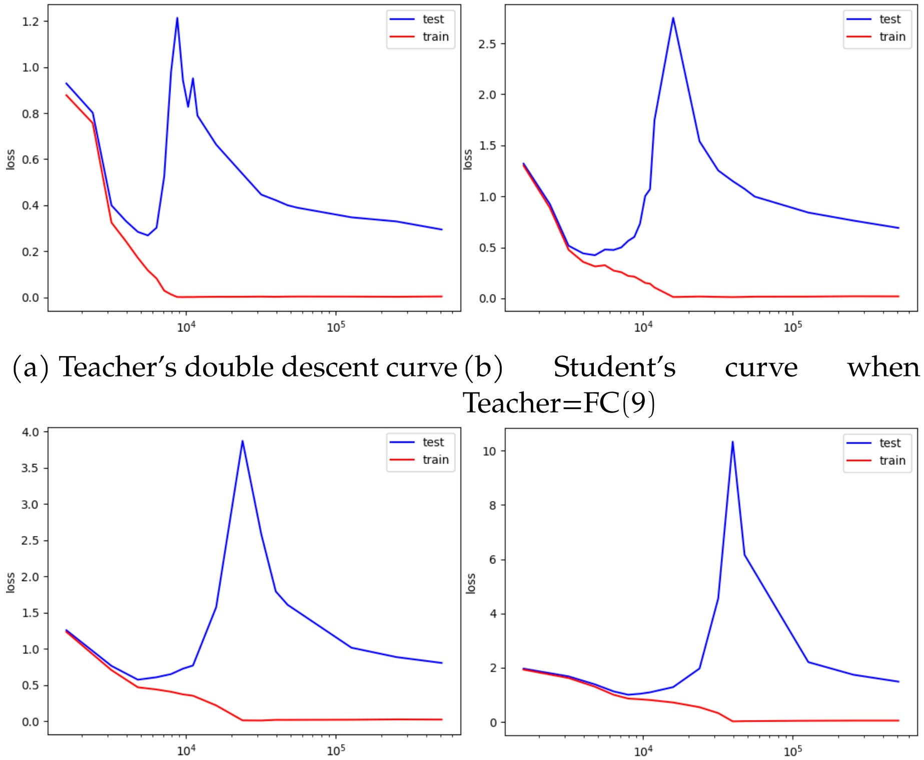


Figure A.21: Test accuracy at different steps for an equal mixture $\alpha = 0.5$ of Gaussian inputs with MNIST, where $m = 50$.



(a) Teacher's double descent curve

(b) Student's curve when Teacher=FC(9)

(c) Student's curve when Teacher=FC(50)

(d) Student's curve when Teacher=FC(500)

Figure A.22: Double descent curves of the students trained on data with synthetic labels (Loss v.s. Parameter number).

# B APPENDIX FOR CHAPTER 3

Section B.1 discusses the potential societal impact of our work. Section B.2 describes the limitations of our work. In Section B.3, we present our framework implications about simplicity bias and lottery ticket hypothesis. The complete proof of our main results is in Section B.4. We present the case study of linear data in Section B.5.1, mixtures of Gaussians in Section B.5.2 and Section B.5.3, parity functions in Section B.5.4, Section B.5.5 and Section B.5.6, and multiple-index models in Section B.5.7. We put auxiliary lemmas in Section B.6.

## B.1 Broader Impacts

Our paper is purely theoretical in nature and thus we foresee no immediate negative ethical impact. We provide a unified theoretical framework that can be applied to different theoretical problems. This may lead to a better understanding and inspire the development of improved network learning methods, which may have a positive impact on the theoretical machine learning community. On the other hand, it may be beneficial to engineering-inclined machine learning researchers as well.

## B.2 Limitations

The framework may or may not recover the width or sample complexity bounds in existing work.

1. The framework can give matching bounds as the existing work in some cases, like parities over uniform inputs (Section B.5.5).

2. In some other cases, it gives polynomial error bounds not the same as those in the existing work (e.g., for parities over structured inputs). This is because our work is analyzing general cases, and thus may not give better than or the same bounds as those in special cases, since special cases have more

properties that can be exploited to get potentially better bounds. On the other hand, our bounds can already show the advantage over kernel methods (e.g., Proposition 3.23).

We would like to emphasize that our contribution is providing an analysis framework that can (1) formalize the unifying principles of learning features from gradients in network training, and (2) give polynomial error bounds for prototypical problems. Our focus is not to recover the guarantees in existing work.

## B.3 Further Implications

Our general framework also sheds some light on several interesting phenomena in neural network learning observed in practice. Feature learning beyond the kernel regime has been discussed in Section 3.4.1 and 3.4.2. Below we discuss two other phenomena.

### B.3.1 Implicit Regularization/Simplicity Bias

It is now well known that practical neural networks are overparameterized and traditional uniform convergence bounds cannot adequately explain their generalization performance Zhang et al. (2017); Nagarajan and Kolter (2019); Jacot (2023). It is generally believed that the optimization has some *implicit regularization* effect that restricts learning dynamics to a subset of the whole hypothesis class, which is not of high capacity so can lead to good generalization Neyshabur (2017); Gidel et al. (2019). Furthermore, learning dynamics tend to first learn simple functions and then learn more and more sophisticated ones (referred to as simplicity bias) Nakkiran et al. (2019); Shah et al. (2020). However, it remains elusive to formalize such simplicity bias.

Our framework provides a candidate explanation: the learning dynamics first learn to approximate the best network in a smaller family of gradient feature induced networks $\mathcal{F}_{d,r,B_F,S}$ and then learn to approximate the best in a larger family. Consider the number of neurons $r$ for illustration. Let $r_1 \ll r_2$, and let $T_1$ and $T_2$ be their corresponding runtime bounds for $T$ in the main Theorem 3.12. Clearly, $T_1 \ll T_2$. Then, at time $T_1$, the theorem guarantees the learning dynamics learn to approximate the best in the family $\mathcal{F}_{d,r_1,B_F,S}$ with $r_1$ neurons, but not for the larger family $\mathcal{F}_{d,r_2,B_F,S}$. Later, at time $T_2$, the learning dynamics learn to approximate the best in the larger family $\mathcal{F}_{d,r_2,B_F,S}$. That is, the learning first learns simpler functions and then more sophisticated ones where the simplicity bias is measured by the size of the family of gradient feature-induced networks. The implicit regularization is then restricting to networks approximating smaller families of gradient feature-induced networks.

### B.3.2 Lottery Ticket Hypothesis (LTH)

Another interesting phenomenon is the LTH Frankle and Carbin (2018): randomly-initialized networks contain subnetworks that when trained in isolation reach test accuracy comparable to the original network in a similar number of iterations. Later studies (e.g., Frankle et al. (2019)) show that LTH is more stable when subnetworks are found in the network after a few gradient steps.

Our framework again provides some explanation for two-layer networks: the lottery ticket subnetwork contains exactly those neurons whose gradient feature approximates the weights of the "ground-truth" network $f^*$; they may not exist at initialization but can be found after the first gradient step. More precisely, Theorem 3.14 shows that after the first gradient step, there is a *sparse* second-layer weight $\tilde{a}$ with $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$, such that using this weight on the hidden neurons gives a network with a small loss. Let $U$ denote the support of $\tilde{a}$. Then equivalently, there is a small-loss subnetwork $g_{\Xi}^{U}$ with only neurons in $U$ and with second-layer weight $\tilde{a}_U$ on these neurons. Following the same proof of Theorem 3.12, we can show:

**Proposition B.1.** *In the same setting of Theorem 3.12 but only considering the subnetwork supported on $U$ after the first gradient step, with the same requirements on $m$ and $T$, with proper hyper-parameter values, we have the same guarantee: with probability $\geq 1-\delta$, there is $t \in [T]$ with* $\Pr[\text{sign}(g_{\Xi^{(t)}}^{U})(\mathbf{x}) \neq y] \leq \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}^{U}\right) \leq \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} +$

$$rB_{a1}B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G n^{\frac{1}{3}}}\right)} + \epsilon.$$

This essentially formally proves LTH for two-layer networks, showing (a) the existence of the subnetwork and (b) that gradient descent on the subnetwork can learn to similar loss in similar runtime as on the whole network. In particular, (b) is novel, not analyzed in existing work.

## B.4 Gradient Feature Learning Framework

We first prove a Simplified Gradient Feature Learning Framework in Section B.4.1, which only considers one-step gradient feature learning. Then, we prove our Gradient Feature Learning Framework, e.g., no freezing of the first layer. In Section B.4.2, we consider population loss to simplify the proof. Finally, we prove our Gradient Feature Learning Framework under empirical loss considering sample complexity in Section B.4.3.

### B.4.1 Simplified Gradient Feature Learning Framework

**Algorithm 4** Network Training via One Step Feature Learning

```
Initialize $g_{(a^{(0)},\mathbf{W}^{(0)},b)} \in \mathcal{F}_{d,m}$; Sample $\mathcal{Z} \sim \mathcal{D}^n$
Get $(a^{(1)}, \mathbf{W}^{(1)}, b)$ by one gradient step update and fix $\mathbf{W}^{(1)}, b$
for $t = 2$ to $T$ do
    $a^{(t)} = a^{(t-1)} - \eta^{(t)}\nabla_a \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{\Xi^{(t-1)}})$
end for
```

**Theorem B.2** (One Step Feature Learning: Main Result. Restatement of of Theorem 3.4)**.** *Assume* $\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(f_{(a,\mathbf{W}^{(1)},b)}\right)$ *is* $L$*-smooth to* $a$*. Let* $\eta^{(t)} = \frac{1}{L}, \lambda^{(t)} = 0$*, for all* $t \in \{2, 3, \ldots, T\}$*. Considering training by Algorithm 4, w.h.p., there exists* $t \in [T]$ *such that*

$$\mathcal{L}_{\mathcal{D}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) \leqslant \mathrm{OPT}_{\mathbf{W}^{(1)},b,B_{a2}}$$

$$+ O\left(\frac{L(\|a^{(1)}\|_2^2 + B_{a2}^2)}{T} + \sqrt{\frac{B_{a2}^2(\|\mathbf{W}^{(1)}\|_F^2 B_x^2 + \|b\|_2^2)}{n}}\right).$$

*Proof of Theorem B.2.* Recall that

$$\mathcal{F}_{\mathbf{W},b,B_{a2}} := \left\{g_{(a,\mathbf{W},b)} \in \mathcal{F}_{d,m} \;\middle|\; \|a\|_2 \leqslant B_{a2}\right\}, \qquad \mathrm{OPT}_{\mathbf{W},b,B_{a2}} := \min_{g \in \mathcal{F}_{\mathbf{W},b,B_{a2}}} \mathcal{L}_{\mathcal{D}}(f). \tag{B.1}$$

We denote $f^* = \arg\min_{g \in \mathcal{F}_{\mathbf{W},b,B_{a2}}} \mathcal{L}_{\mathcal{D}}(f)$ and $\tilde{f}^* = \arg\min_{g \in \mathcal{F}_{\mathbf{W},b,B_{a2}}} \widetilde{\mathcal{L}}_{\mathcal{Z}}(f)$. We use $a^*$ and $\tilde{a}^*$ to denote their second layer weights respectively. Then, we have

$$\begin{aligned}
\mathcal{L}_{\mathcal{D}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) =& \mathcal{L}_{\mathcal{D}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) - \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) && \text{(B.2)}\\
&+ \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) - \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(\tilde{a}^*,\mathbf{W}^{(1)},b)}) && \text{(B.3)}\\
&+ \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(\tilde{a}^*,\mathbf{W}^{(1)},b)}) - \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^*,\mathbf{W}^{(1)},b)}) && \text{(B.4)}\\
&+ \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^*,\mathbf{W}^{(1)},b)}) - \mathcal{L}_{\mathcal{D}}(g_{(a^*,\mathbf{W}^{(1)},b)}) && \text{(B.5)}\\
&+ \mathcal{L}_{\mathcal{D}}(g_{(a^*,\mathbf{W}^{(1)},b)}) && \text{(B.6)}\\
\leqslant& \left|\mathcal{L}_{\mathcal{D}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) - \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)})\right| && \text{(B.7)}\\
&+ \left|\widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^{(t)},\mathbf{W}^{(1)},b)}) - \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(\tilde{a}^*,\mathbf{W}^{(1)},b)})\right| && \text{(B.8)}\\
&+ 0 && \text{(B.9)}\\
&+ \left|\widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{(a^*,\mathbf{W}^{(1)},b)}) - \mathcal{L}_{\mathcal{D}}(g_{(a^*,\mathbf{W}^{(1)},b)})\right| && \text{(B.10)}\\
&+ \mathrm{OPT}_{\mathbf{W}^{(1)},b,B_{a2}}. && \text{(B.11)}
\end{aligned}$$

Fixing $\mathbf{W}^{(1)}$, $b$ and optimizing $a$ only is a convex optimization problem. Note that $\eta \leqslant \frac{1}{L}$, where $\widetilde{\mathcal{L}}_{\mathcal{Z}}$ is L-smooth to $a$. Thus with gradient descent, we have

$$\frac{1}{T}\sum_{t=1}^{T} \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a^{(t)},\mathbf{W}^{(1)},b)}\right) - \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a^*,\mathbf{W}^{(1)},b)}\right) \leqslant \frac{\|a^{(1)} - a^*\|_2^2}{2T\eta}. \tag{B.12}$$

Then our theorem gets proved by Theorem B.78 and generalization bounds based on Rademacher complexity. □

### B.4.2 Gradient Feature Learning Framework under Expected Risk

We consider the following training process under population loss to simplify the proof. We prove our Gradient Feature Learning Framework under empirical loss considering sample complexity in Section B.4.3.

**Algorithm 5** Network Training via Gradient Descent

```
Initialize $(\mathbf{a}^{(0)}, \mathbf{W}^{(0)}, \mathbf{b})$ as in Equation (3.9)
for $t = 1$ to $T$ do
    $\mathbf{a}^{(t)} = \mathbf{a}^{(t-1)} - \eta^{(t)} \nabla_{\mathbf{a}} \mathcal{L}_{\mathcal{D}}^{\lambda^{(t)}}(g_{\Xi^{(t-1)}})$
    $\mathbf{W}^{(t)} = \mathbf{W}^{(t-1)} - \eta^{(t)} \nabla_{\mathbf{W}} \mathcal{L}_{\mathcal{D}}^{\lambda^{(t)}}(g_{\Xi^{(t-1)}})$
end for
```

Given an input distribution, we can get a Gradient Feature set $S_{p,\gamma,B_G}$ and $g^*(\mathbf{x}) = \sum_{j=1}^{r} a_j^* \sigma(\langle \mathbf{w}_j^*, \mathbf{x} \rangle - b_j^*)$, where $f^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$ is a Gradient Feature Induced networks defined in Theorem 3.11. Considering training by Algorithm 5, we have the following results.

**Theorem B.3** (Gradient Feature Learning Framework under Expected Risk)**.** *Assume Assumption 3.1. For any* $\epsilon, \delta \in (0,1)$*, if* $m \leq e^d$ *and*

$$m = \Omega\left(\frac{1}{p}\left(\frac{rB_{a1}B_{x1}}{\epsilon}\sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{B.13}$$

$$T = \Omega\left(\frac{1}{\epsilon}\left(\frac{\sqrt{r}B_{a2}B_bB_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m}}{\sqrt{B_bB_G}} + \frac{1}{B_{x1}(mp)^{\frac{1}{4}}}\right)\right), \tag{B.14}$$

*then with proper hyper-parameter values, we have with probability* $\geq 1 - \delta$*, there exists* $t \in [T]$ *in Algorithm 5 with*

$$\Pr[\operatorname{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leq \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \leq \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}B_{x1}\sqrt{2\gamma} + \epsilon. \tag{B.15}$$

See the full statement and proof in Theorem B.11. Below, we show some Lemma used in the analysis under population loss.

### Feature Learning

We first show that a large subset of neurons has gradients at the first step as good features.

**Definition B.4** (Nice Gradients Set. Equivalent to Equation (3.14)). *We define*

$$G_{(D,+1),\text{Nice}} := \left\{ i \in [m] : \langle \mathbf{w}_i^{(1)}, D \rangle > (1-\gamma) \left\| \mathbf{w}_i^{(1)} \right\|_2, \left\| \mathbf{w}_i^{(1)} \right\|_2 \geqslant \left| \eta^{(1)} \ell'(0) a_i^{(0)} \right| B_G \right\}$$
$$G_{(D,-1),\text{Nice}} := \left\{ i \in [2m] \setminus [m] : \langle \mathbf{w}_i^{(1)}, D \rangle > (1-\gamma) \left\| \mathbf{w}_i^{(1)} \right\|_2, \left\| \mathbf{w}_i^{(1)} \right\|_2 \geqslant \left| \eta^{(1)} \ell'(0) a_i^{(0)} \right| B_G \right\}$$

*where $\gamma, B_G$ is the same in the Theorem 3.7.*

**Lemma B.5** (Feature Emergence. Full Statement of Theorem 3.13). *Let $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$. For any $r$ size subset $\{(D_1, s_1), \dots, (D_r, s_r)\} \subseteq S_{p,\gamma,B_G}$, with probability at least $1 - 2re^{-cmp}$ where $c > 0$ is a universal constant, we have that for all $j \in [r]$, $|G_{(D_j,s_j),\text{Nice}}| \geqslant \frac{mp}{4}$.*

*Proof of Theorem B.5.* By symmetric initialization and Theorem B.70, we have for all $i \in [2m]$

$$\mathbf{w}_i^{(1)} = -\eta^{(1)} \ell'(0) a_i^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ y \sigma' \left[ \langle \mathbf{w}_i^{(0)}, \mathbf{x} \rangle - b_i \right] \mathbf{x} \right] \tag{B.16}$$
$$= -\eta^{(1)} \ell'(0) a_i^{(0)} G(\mathbf{w}_i^{(0)}, b_i). \tag{B.17}$$

For all $j \in [r]$, as $(D_j, s_j) \in S_{p,\gamma,B_G}$, by Theorem B.72,
(1) if $s_j = +1$, for all $i \in [m]$, we have

$$\Pr \left[ i \in G_{(D_j,s_j),\text{Nice}} \right] \tag{B.18}$$
$$= \Pr \left[ \frac{\langle \mathbf{w}_i^{(1)}, D_j \rangle}{\left\| \mathbf{w}_i^{(1)} \right\|_2} > (1-\gamma), \left\| \mathbf{w}_i^{(1)} \right\|_2 \geqslant \left| \eta^{(1)} \ell'(0) a_i^{(0)} \right| B_G \right] \tag{B.19}$$

$$= \Pr\left[\frac{\langle \mathbf{w}_i^{(1)}, D_j\rangle}{\left\|\mathbf{w}_i^{(1)}\right\|_2} > (1-\gamma),\ \left\|\mathbf{w}_i^{(1)}\right\|_2 \geqslant \left|\eta^{(1)}\ell'(0)a_i^{(0)}\right| B_G,\ \frac{b_i}{|b_i|} = s_j\right] \tag{B.20}$$

$$\geqslant \Pr\left[G(\mathbf{w}_i^{(0)}, b_i) \in \mathcal{C}_{D_j,\gamma},\ \|G(\mathbf{w}_i^{(0)}, b_i)\|_2 \geqslant B_G,\ \frac{b_i}{|b_i|} = s_j,\ a_i^{(0)}\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle > 0\right] \tag{B.21}$$

$$\geqslant \frac{p}{2}, \tag{B.22}$$

(2) if $s_j = -1$, for all $i \in [2m] \setminus [m]$, similarly we have

$$\Pr\left[i \in G_{(D_j,s_j),\mathrm{Nice}}\right] \geqslant \frac{p}{2}. \tag{B.23}$$

By concentration inequality, (Chernoff's inequality under small deviations), we have

$$\Pr\left[|G_{(D_j,s_j),\mathrm{Nice}}| < \frac{mp}{4}\right] \leqslant 2e^{-cmp}. \tag{B.24}$$

We complete the proof by union bound. □

### Good Network Exists

Then, the gradients allow for obtaining a set of neurons approximating the "ground-truth" network with comparable loss.

**Lemma B.6** (Existence of Good Networks. Full Statement of Theorem 3.14)**.** *Let* $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$. *For any* $B_\epsilon \in (0, B_b)$, *let* $\sigma_a = \Theta\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)}B_G B_\epsilon}\right)$ *and* $\delta = 2re^{-\sqrt{mp}}$. *Then, with probability at least* $1-\delta$ *over the initialization, there exists* $\tilde{a}_i$*'s such that* $g_{(\tilde{a},\mathbf{W}^{(1)},\mathbf{b})}(\mathbf{x}) = \sum_{i=1}^{4m} \tilde{a}_i \sigma\left(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle - b_i\right)$ *satisfies*

$$\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},\mathbf{b})}) \leqslant rB_{a1}\left(\frac{B_{x1}^2 B_b}{\sqrt{mp}B_G B_\epsilon} + B_{x1}\sqrt{2\gamma} + B_\epsilon\right) + \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}, \tag{B.25}$$

*and* $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$, $\|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right)$, $\|\tilde{a}\|_\infty = O\left(\frac{B_{a1}B_b}{\tilde{b}(mp)^{\frac{1}{2}}}\right)$.

*Proof of Theorem B.6.* Recall $g^*(\mathbf{x}) = \sum_{j=1}^{r} a_j^* \sigma(\langle \mathbf{w}_j^*, \mathbf{x}\rangle - b_j^*)$, where $f^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$ is defined in Theorem 3.11 and let $s_j^* = \frac{b_j^*}{|b_j^*|}$. By Theorem B.5, with probability at least $1-\delta_1$, $\delta_1 = 2re^{-cmp}$, for all $j \in [r]$, we have $|G_{(\mathbf{w}_j^*, s_j^*),Nice}| \geqslant \frac{mp}{4}$. Then for all $i \in G_{(\mathbf{w}_j^*, s_j^*),Nice} \subseteq [2m]$, we have $-\ell'(0)\eta^{(1)} G(\mathbf{w}_i^{(0)}, b_i)\frac{b_j^*}{\tilde{b}}$ only depend on $\mathbf{w}_i^{(0)}$ and $b_i$, which is independent of $a_i^{(0)}$. Given Theorem 3.7, we have

$$-\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{b_j^*}{\tilde{b}} \in \left[\ell'(0)\eta^{(1)}B_{x1}\frac{B_b}{\tilde{b}}, -\ell'(0)\eta^{(1)}B_{x1}\frac{B_b}{\tilde{b}}\right]. \tag{B.26}$$

We split $[r]$ into $\Gamma = \{j \in [r] : |b_j^*| < B_\epsilon\}$, $\Gamma_- = \{j \in [r] : b_j^* \leqslant -B_\epsilon\}$ and $\Gamma_+ = \{j \in [r] : b_j^* \geqslant B_\epsilon\}$. Let $\epsilon_a = \frac{B_{x1}B_b}{\sqrt{mp}B_G B_\epsilon}$. Then we know that for all $j \in \Gamma_+ \cup \Gamma_-$, for all $i \in G_{(\mathbf{w}_j^*, s_j^*),Nice}$, we have

$$\Pr_{a_i^{(0)}\sim\mathcal{N}(0,\sigma_a^2)}\left[\left|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{|b_j^*|}{\tilde{b}} - 1\right| \leqslant \epsilon_a\right] \tag{B.27}$$

$$= \Pr_{a_i^{(0)}\sim\mathcal{N}(0,\sigma_a^2)}\left[1-\epsilon_a \leqslant -a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{|b_j^*|}{\tilde{b}} \leqslant 1+\epsilon_a\right] \tag{B.28}$$

$$= \Pr_{g\sim\mathcal{N}(0,1)}\left[1-\epsilon_a \leqslant g\Theta\left(\frac{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}{B_G B_\epsilon}\right) \leqslant 1+\epsilon_a\right] \tag{B.29}$$

$$= \Pr_{g\sim\mathcal{N}(0,1)}\left[(1-\epsilon_a)\Theta\left(\frac{B_G B_\epsilon}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}\right) \leqslant g \leqslant (1+\epsilon_a)\Theta\left(\frac{B_G B_\epsilon}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}\right)\right] \tag{B.30}$$

$$=\Theta\left(\frac{\epsilon_a B_G B_\epsilon}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}\right) \tag{B.31}$$

$$\geqslant\Omega\left(\frac{\epsilon_a B_G B_\epsilon}{B_{x1}B_b}\right) \tag{B.32}$$

$$=\Omega\left(\frac{1}{\sqrt{mp}}\right). \tag{B.33}$$

Thus, with probability $\Omega\left(\frac{1}{\sqrt{mp}}\right)$ over $a_i^{(0)}$, we have

$$\left|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)},b_i)\|_2\frac{|b_j^*|}{\tilde{b}}-1\right| \leqslant \epsilon_a, \quad \left|a_i^{(0)}\right| = O\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)}B_G B_\epsilon}\right). \tag{B.34}$$

Similarly, for $j \in \Gamma$, for all $i \in G_{(\mathbf{w}_j^*,s_j^*),\mathrm{Nice}}$, with probability $\Omega\left(\frac{1}{\sqrt{mp}}\right)$ over $a_i^{(0)}$, we have

$$\left|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)},b_i)\|_2\frac{B_\epsilon}{\tilde{b}}-1\right| \leqslant \epsilon_a, \quad \left|a_i^{(0)}\right| = O\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)}B_G B_\epsilon}\right). \tag{B.35}$$

For all $j \in [r]$, let $\Lambda_j \subseteq G_{(\mathbf{w}_j^*,s_j^*),\mathrm{Nice}}$ be the set of $i$'s such that condition Equation (B.34) or Equation (B.35) are satisfied. By Chernoff bound and union bound, with probability at least $1-\delta_2$, $\delta_2 = re^{-\sqrt{mp}}$, for all $j \in [r]$ we have $|\Lambda_j| \geqslant \Omega(\sqrt{mp})$.

We have for $\forall j \in \Gamma_+ \cup \Gamma_-, \forall i \in \Lambda_j$,

$$\left|\frac{|b_j^*|}{\tilde{b}}\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle - \langle \mathbf{w}_j^*,\mathbf{x}\rangle\right| \tag{B.36}$$

$$\leqslant \left\|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)},b_i)\|_2\frac{|b_j^*|}{\tilde{b}}\frac{\mathbf{w}_i^{(1)}}{\|\mathbf{w}_i^{(1)}\|_2} - \frac{\mathbf{w}_i^{(1)}}{\|\mathbf{w}_i^{(1)}\|_2} + \frac{\mathbf{w}_i^{(1)}}{\|\mathbf{w}_i^{(1)}\|_2} - \mathbf{w}_j^*\right\| \|\mathbf{x}\|_2 \tag{B.37}$$

$$\leqslant (\epsilon_a + \sqrt{2\gamma})\|\mathbf{x}\|_2. \tag{B.38}$$

Similarly, for $\forall j \in \Gamma, \forall i \in \Lambda_j$,

$$\left|\frac{B_\epsilon}{\tilde{b}}\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle - \langle \mathbf{w}_j^*,\mathbf{x}\rangle\right| \leqslant (\epsilon_a + \sqrt{2\gamma})\|\mathbf{x}\|_2. \tag{B.39}$$

If $i \in \Lambda_j, j \in \Gamma_+ \cup \Gamma_-$, set $\tilde{a}_i = a_j^* \frac{|b_j^*|}{|\Lambda_j|\tilde{b}}$, if $i \in \Lambda_j, j \in \Gamma$, set $\tilde{a}_i = a_j^* \frac{B_\epsilon}{|\Lambda_j|\tilde{b}}$, otherwise set $\tilde{a}_i = 0$, we have $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$, $\|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right)$, $\|\tilde{a}\|_\infty = O\left(\frac{B_{a1}B_b}{\tilde{b}(mp)^{\frac{1}{2}}}\right)$.

Finally, we have

$$\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)}) \tag{B.40}$$

$$=\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)}) - \mathcal{L}_{\mathcal{D}}(g^*) + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.41}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|g_{(\tilde{a},\mathbf{W}^{(1)},b)}(\mathbf{x}) - g^*(\mathbf{x})\right|\right] + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.42}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{i=1}^{m}\tilde{a}_i\sigma\left(\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle - \tilde{b}\right) + \sum_{i=m+1}^{2m}\tilde{a}_i\sigma\left(\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle + \tilde{b}\right) - \sum_{j=1}^{r}a_j^*\sigma(\langle\mathbf{w}_j^*,\mathbf{x}\rangle - b_j^*)\right|\right] \tag{B.43}$$

$$+\mathcal{L}_{\mathcal{D}}(g^*) \tag{B.44}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_+}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\sigma\left(\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle - \tilde{b}\right) - \sigma(\langle\mathbf{w}_j^*,\mathbf{x}\rangle - b_j^*)\right|\right|\right] \tag{B.45}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_-}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\sigma\left(\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle + \tilde{b}\right) - \sigma(\langle\mathbf{w}_j^*,\mathbf{x}\rangle - b_j^*)\right|\right|\right] \tag{B.46}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{B_\epsilon}{\tilde{b}}\sigma\left(\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle - \tilde{b}\right) - \sigma(\langle\mathbf{w}_j^*,\mathbf{x}\rangle - b_j^*)\right|\right|\right] + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.47}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_+}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle - \langle\mathbf{w}_j^*,\mathbf{x}\rangle\right|\right|\right] \tag{B.48}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_-}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle - \langle\mathbf{w}_j^*,\mathbf{x}\rangle\right|\right|\right] \tag{B.49}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{B_\epsilon}{\tilde{b}}\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle + B_\epsilon - \langle\mathbf{w}_j^*,\mathbf{x}\rangle\right|\right|\right] + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.50}$$

$$\leqslant r\|a^*\|_\infty(\epsilon_a + \sqrt{2\gamma})\mathbb{E}_{(\mathbf{x},y)}\|\mathbf{x}\|_2 + |\Gamma|\|a^*\|_\infty B_\epsilon + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.51}$$

$$\leqslant rB_{x1}B_{a1}(\epsilon_a + \sqrt{2\gamma}) + |\Gamma|B_{a1}B_\epsilon + \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}. \tag{B.52}$$

We finish the proof by union bound and $\delta \geqslant \delta_1 + \delta_2$. □

**Learning an Accurate Classifier**

We will use the following theorem from existing work to prove that gradient descent learns a good classifier (Theorem B.11). Theorem B.3 is simply a direct corollary of Theorem B.11.

**Theorem B.7** (Theorem 13 in Daniely and Malach (2020))**.** *Fix some* $\eta$*, and let* $f_1, \dots, f_T$ *be some sequence of convex functions. Fix some* $\theta_1$*, and assume we update* $\theta_{t+1} = \theta_t - \eta \nabla f_t(\theta_t)$*. Then for every* $\theta^*$ *the following holds:*

$$\frac{1}{T}\sum_{t=1}^{T} f_t(\theta_t) \leqslant \frac{1}{T}\sum_{t=1}^{T} f_t(\theta^*) + \frac{1}{2\eta T}\|\theta^*\|_2^2 + \|\theta_1\|_2 \frac{1}{T}\sum_{t=1}^{T}\|\nabla f_t(\theta_t)\|_2 + \eta\frac{1}{T}\sum_{t=1}^{T}\|\nabla f_t(\theta_t)\|_2^2. \tag{B.53}$$

To apply the theorem we first present a few lemmas bounding the change in the network during steps.

**Lemma B.8** (Bound of $\Xi^{(0)}, \Xi^{(1)}$)**.** *Assume the same conditions as in Theorem B.6, and* $d \geqslant \log m$*, with probability at least* $1 - \delta - \frac{1}{m^2}$ *over the initialization,* $\|a^{(0)}\|_\infty = O\left(\frac{\tilde{b}\sqrt{\log m}}{-\ell'(0)\eta^{(1)} B_G B_\epsilon}\right)$*, and for all* $i \in [4m]$*, we have* $\|\mathbf{w}_i^{(0)}\|_2 = O\left(\sigma_\mathbf{w}\sqrt{d}\right)$*. Finally,* $\|a^{(1)}\|_\infty = O\left(-\eta^{(1)}\ell'(0)(B_{x1}\sigma_\mathbf{w}\sqrt{d} + \tilde{b})\right)$*, and for all* $i \in [4m]$*,* $\|\mathbf{w}_i^{(1)}\|_2 = O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon}\right)$*.*

*Proof of Theorem B.8.* By Theorem B.73, we have $\|a^{(0)}\|_\infty = O\left(\frac{\tilde{b}\sqrt{\log m}}{-\ell'(0)\eta^{(1)} B_G B_\epsilon}\right)$ with probability at least $1 - \frac{1}{2m^2}$ by property of maximum i.i.d Gaussians. For any $i \in [4m]$, by Theorem B.74 and $d \geqslant \log m$, we have

$$\Pr\left(\frac{1}{\sigma_\mathbf{w}^2}\left\|\mathbf{w}_i^{(0)}\right\|_2^2 \geqslant d + 2\sqrt{4d\log(m)} + 8\log(m)\right) \leqslant O\left(\frac{1}{m^4}\right). \tag{B.54}$$

Thus, by union bound, with probability at least $1 - \frac{1}{2m^2}$, for all $i \in [4m]$, we have $\|\mathbf{w}_i^{(0)}\|_2 = O\left(\sigma_\mathbf{w}\sqrt{d}\right)$.

For all $i \in [4m]$, we have

$$\begin{aligned}
|a_i^{(1)}| &= -\eta^{(1)}\ell'(0)\left|\mathbb{E}_{(\mathbf{x},y)}\left[y\left[\sigma\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right]\right]\right| && \text{(B.55)}\\
&\leqslant -\eta^{(1)}\ell'(0)(\|\mathbf{w}_i^{(0)}\|_2\mathbb{E}_{(\mathbf{x},y)}[\|\mathbf{x}\|_2] + \tilde{b}) && \text{(B.56)}\\
&\leqslant O\left(-\eta^{(1)}\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d} + \tilde{b})\right). && \text{(B.57)}
\end{aligned}$$

$$\begin{aligned}
\|\mathbf{w}_i^{(1)}\|_2 &= -\eta^{(1)}\ell'(0)\left\|a_i^{(0)}\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right]\mathbf{x}\right]\right\|_2 && \text{(B.58)}\\
&\leqslant O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon}\right). && \text{(B.59)}
\end{aligned}$$

□

**Lemma B.9** (Bound of $\Xi^{(t)}$)**.** *Assume the same conditions as in Theorem B.8, and let* $\eta = \eta^{(t)}$ *for all* $t \in \{2, 3, \ldots, T\}$, $0 < T\eta B_{x1} \leqslant o(1)$, *and* $0 = \lambda = \lambda^{(t)}$ *for all* $t \in \{2, 3, \ldots, T\}$, *for all* $i \in [4m]$, *we have*

$$\begin{aligned}
|a_i^{(t)}| &\leqslant O\left(|a_i^{(1)}| + \|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{B_{x1}} + \eta\tilde{b}\right) && \text{(B.60)}\\
\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 &\leqslant O\left(t\eta B_{x1}|a_i^{(1)}| + t\eta^2 B_{x1}^2\|\mathbf{w}_i^{(1)}\|_2 + t\eta^2 B_{x1}\tilde{b}\right). && \text{(B.61)}
\end{aligned}$$

*Proof of Theorem B.9.* For all $i \in [4m]$, by Theorem B.8,

$$\begin{aligned}
|a_i^{(t)}| &= \left|(1-\eta\lambda)a_i^{(t-1)} - \eta\mathbb{E}_{(\mathbf{x},y)}\left[\ell'(yg_{\Xi^{(t-1)}(\mathbf{x})})y\left[\sigma\left(\langle \mathbf{w}_i^{(t-1)}, \mathbf{x}\rangle - b_i\right)\right]\right]\right| && \text{(B.62)}\\
&\leqslant \left|(1-\eta\lambda)a_i^{(t-1)}\right| + \eta\left|\mathbb{E}_{(\mathbf{x},y)}\left[\left[\sigma\left(\langle \mathbf{w}_i^{(t-1)}, \mathbf{x}\rangle - b_i\right)\right]\right]\right| && \text{(B.63)}\\
&\leqslant \left|a_i^{(t-1)}\right| + \eta(B_{x1}\|\mathbf{w}_i^{(t-1)}\|_2 + \tilde{b}) && \text{(B.64)}\\
&\leqslant \left|a_i^{(t-1)}\right| + \eta B_{x1}\|\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(1)}\|_2 + \eta B_{x1}\|\mathbf{w}_i^{(1)}\|_2 + \eta\tilde{b} && \text{(B.65)}\\
&= \left|a_i^{(t-1)}\right| + \eta B_{x1}\|\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(1)}\|_2 + \eta Z_i, && \text{(B.66)}
\end{aligned}$$

where we denote $Z_i = B_{x1}\|\mathbf{w}_i^{(1)}\|_2 + \tilde{b}$. Then we give a bound of the first layer's weights change,

$$\begin{aligned}
&\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 && \text{(B.67)}\\
=&\left\|(1-\eta\lambda)\mathbf{w}_i^{(t-1)} - \eta a_i^{(t-1)}\mathbb{E}_{(\mathbf{x},y)}\left[\ell'(yg_{\Xi^{(t-1)}(\mathbf{x})})y\sigma'\left[\langle\mathbf{w}_i^{(t-1)},\mathbf{x}\rangle - b_i\right]\mathbf{x}\right] - \mathbf{w}_i^{(1)}\right\|_2 && \text{(B.68)}\\
\leqslant&\|\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(1)}\|_2 + \eta B_{x1}|a_i^{(t-1)}|. && \text{(B.69)}
\end{aligned}$$

Combine two bounds, we can get

$$\begin{aligned}
|a_i^{(t)}| \leqslant&|a_i^{(t-1)}| + \eta Z_i + (\eta B_{x1})^2\sum_{l=1}^{t-2}|a_i^{(l)}| && \text{(B.70)}\\
\Leftrightarrow \sum_{l=1}^{t}|a_i^{(l)}| \leqslant&2\left(\sum_{l=1}^{t-1}|a_i^{(l)}|\right) - (1-(\eta B_{x1})^2)\left(\sum_{l=1}^{t-2}|a_i^{(l)}|\right) + \eta Z_i. && \text{(B.71)}
\end{aligned}$$

Let $h(1) = |a_i^{(1)}|$, $h(2) = 2|a_i^{(1)}|+\eta Z_i$ and $h(t+2) = 2h(t+1)-(1-(\eta B_{x1})^2)h(t)+\eta Z_i$ for $n \in \mathbb{N}_+$, by Theorem B.77, we have

$$\begin{aligned}
h(t) =& -\frac{Z_i}{\eta B_{x1}^2} + c_1(1-\eta B_{x1})^{(t-1)} + c_2(1+\eta B_{x1})^{(t-1)} && \text{(B.72)}\\
c_1 =&\frac{1}{2}\left(|a_i^{(1)}| + \frac{Z_i}{\eta B_{x1}^2} - \frac{|a_i^{(1)}| + \eta Z_i}{\eta B_{x1}}\right) && \text{(B.73)}\\
c_2 =&\frac{1}{2}\left(|a_i^{(1)}| + \frac{Z_i}{\eta B_{x1}^2} + \frac{|a_i^{(1)}| + \eta Z_i}{\eta B_{x1}}\right). && \text{(B.74)}
\end{aligned}$$

Thus, by $|c_1| \leqslant c_2$, and $0 < T\eta B_{x1} \leqslant o(1)$, we have

$$\begin{aligned}
|a_i^{(t)}| \leqslant&h(t) - h(t-1) && \text{(B.75)}\\
=& -\eta B_{x1}c_1(1-\eta B_{x1})^{(t-2)} + \eta B_{x1}c_2(1+\eta B_{x1})^{(t-2)} && \text{(B.76)}\\
\leqslant&2\eta B_{x1}c_2(1+\eta B_{x1})^t && \text{(B.77)}\\
\leqslant&O(2\eta B_{x1}c_2). && \text{(B.78)}
\end{aligned}$$

Similarly, by binomial approximation, we also have

$$\begin{aligned}
\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 \leqslant& \eta B_{x1} h(t-1) && \text{(B.79)}\\
=& \eta B_{x1}\left(-\frac{Z_i}{\eta B_{x1}^2} + c_1(1-\eta B_{x1})^{(t-2)} + c_2(1+\eta B_{x1})^{(t-2)}\right) && \text{(B.80)}\\
\leqslant& \eta B_{x1} O\left(-\frac{Z_i}{\eta B_{x1}^2} + c_1(1-(t-2)\eta B_{x1}) + c_2(1+(t-2)\eta B_{x1})\right) && \text{(B.81)}\\
\leqslant& \eta B_{x1} O\left(-\frac{Z_i}{\eta B_{x1}^2} + c_1 + c_2 + (c_2 - c_1)t\eta B_{x1}\right) && \text{(B.82)}\\
\leqslant& \eta B_{x1} O\left(|a_i^{(1)}| + \frac{|a_i^{(1)}| + \eta Z_i}{\eta B_{x1}} t\eta B_{x1}\right) && \text{(B.83)}\\
\leqslant& O\left((\eta |a_i^{(1)}| + \eta^2 Z_i) t B_{x1}\right). && \text{(B.84)}
\end{aligned}$$

We finish the proof by plugging $Z_i, c_2$ into the bound. □

**Lemma B.10** (Bound of Loss Gap and Gradient). *Assume the same conditions as in Theorem B.9, for all* $t \in [T]$, *we have*

$$|\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(t)},b)}) - \mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})| \leqslant B_{x1}\|\tilde{a}\|_2\sqrt{\|\tilde{a}\|_0}\max_{i\in[4m]}\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 \quad \text{(B.85)}$$

*and for all* $t \in [T]$, *for all* $i \in [4m]$, *we have*

$$\left|\frac{\partial \mathcal{L}_{\mathcal{D}}(g_{\Xi^{(t)}})}{\partial a_i^{(t)}}\right| \leqslant B_{x1}(\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 + \|\mathbf{w}_i^{(1)}\|_2) + \tilde{b}. \quad \text{(B.86)}$$

*Proof of Theorem B.10.* It follows from that

$$\begin{aligned}
&|\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(t)},b)}) - \mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})| && \text{(B.87)}\\
&\leqslant \mathbb{E}_{(\mathbf{x},y)}|g_{(\tilde{a},\mathbf{W}^{(t)},b)}(\mathbf{x}) - g_{(\tilde{a},\mathbf{W}^{(1)},b)}(\mathbf{x})| && \text{(B.88)}\\
&\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\|\tilde{a}\|_2\sqrt{\|\tilde{a}\|_0}\max_{i\in[4m]}\left|\sigma\left[\langle\mathbf{w}_i^{(t)},\mathbf{x}\rangle - b_i\right] - \sigma\left[\langle\mathbf{w}_i^{(1)},\mathbf{x}\rangle - b_i\right]\right|\right] && \text{(B.89)}
\end{aligned}$$

$$\leqslant B_{x1}\|\tilde{a}\|_2\sqrt{\|\tilde{a}\|_0}\max_{i\in[4m]}\|\mathbf{w}_i^{(t)}-\mathbf{w}_i^{(1)}\|_2. \tag{B.90}$$

Also, we have

$$\left|\frac{\partial\mathcal{L}_{\mathcal{D}}(g_{\Xi^{(t)}})}{\partial a_i^{(t)}}\right| = \left|\mathbb{E}_{(\mathbf{x},y)}\left[\ell'(yg_{\Xi^{(t)}}(\mathbf{x}))y\left[\sigma\left(\langle\mathbf{w}_i^{(t)},\mathbf{x}\rangle-b_i\right)\right]\right]\right| \tag{B.91}$$

$$\leqslant B_{x1}\|\mathbf{w}_i^{(t)}\|_2+\tilde{b} \tag{B.92}$$

$$\leqslant B_{x1}(\|\mathbf{w}_i^{(t)}-\mathbf{w}_i^{(1)}\|_2+\|\mathbf{w}_i^{(1)}\|_2)+\tilde{b}. \tag{B.93}$$

□

We are now ready to prove the main theorem.

**Theorem B.11** (Online Convex Optimization. Full Statement of Theorem B.3)**.** *Consider training by Algorithm 5, and any* $\delta\in(0,1)$*. Assume* $d\geqslant\log m$*. Set*

$$\sigma_{\mathbf{w}}>0,\quad \tilde{b}>0,\quad \eta^{(t)}=\eta,\ \lambda^{(t)}=0 \textit{ for all } t\in\{2,3,\dots,T\}, \tag{B.94}$$

$$\eta^{(1)}=\Theta\left(\frac{\min\{O(\eta),O(\eta\tilde{b})\}}{-\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d}+\tilde{b})}\right),\ \lambda^{(1)}=\frac{1}{\eta^{(1)}},\quad \sigma_a=\Theta\left(\frac{\tilde{b}(mp)^{\frac{1}{4}}}{-\ell'(0)\eta^{(1)}B_{x1}\sqrt{B_GB_b}}\right). \tag{B.95}$$

*Let* $0<T\eta B_{x1}\leqslant o(1)$*,* $m=\Omega\left(\frac{1}{\sqrt{\delta}}+\frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right)$*. With probability at least* $1-\delta$ *over the initialization, there exists* $t\in[T]$ *such that*

$$\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right)\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}+rB_{a1}\left(\frac{2B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}}+B_{x1}\sqrt{2\gamma}\right) \tag{B.96}$$

$$+\eta\left(\sqrt{r}B_{a2}B_bT\eta B_{x1}^2+m\tilde{b}\right)O\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}}+1\right) \tag{B.97}$$

$$+O\left(\frac{B_{a2}^2B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right). \tag{B.98}$$

*Furthermore, for any* $\epsilon \in (0,1)$*, set*

$$\tilde{b} = \Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{r B_{a1}}}\right), \quad m = \Omega\left(\frac{1}{p\epsilon^4}\left(r B_{a1} B_{x1} \sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{B.99}$$

$$\eta = \Theta\left(\frac{\epsilon}{\left(\frac{\sqrt{r} B_{a2} B_b B_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m} B_{x1} (mp)^{\frac{1}{4}}}{\sqrt{B_b B_G}} + 1\right)}\right), \quad T = \Theta\left(\frac{1}{\eta B_{x1} (mp)^{\frac{1}{4}}}\right), \tag{B.100}$$

*we have there exists* $t \in [T]$ *with*

$$\Pr[\text{sign}(g_{\Xi^{(t)}})(\mathbf{x}) \neq y] \leqslant \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \leqslant \text{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + r B_{a1} B_{x1} \sqrt{2\gamma} + \epsilon. \tag{B.101}$$

*Proof of Theorem B.11.* By $m = \Omega\left(\frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right)$ we have $2re^{-\sqrt{mp}} + \frac{1}{m^2} \leqslant \delta$. For any $B_\epsilon \in (0, B_b)$, when $\sigma_a = \Theta\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)} B_G B_\epsilon}\right)$, by Theorem B.7, Theorem B.6, Theorem B.10, with probability at least $1-\delta$ over the initialization, we have

$$\frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.102}$$

$$\leqslant \frac{1}{T}\sum_{t=1}^{T} |(\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(t)},b)}) - \mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})| + \mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})) \tag{B.103}$$

$$+ \frac{\|\tilde{a}\|_2^2}{2\eta T} + (2\|a^{(1)}\|_2\sqrt{m} + 4\eta m) \max_{i \in [4m]} \left|\frac{\partial \mathcal{L}_{\mathcal{D}}(g_{\Xi^{(T)}})}{\partial a_i^{(T)}}\right| \tag{B.104}$$

$$\leqslant \text{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + r B_{a1}\left(\frac{B_{x1}^2 B_b}{\sqrt{mp} B_G B_\epsilon} + B_{x1}\sqrt{2\gamma} + B_\epsilon\right) \tag{B.105}$$

$$+ B_{x1}\|\tilde{a}\|_2 \sqrt{\|\tilde{a}\|_0} \max_{i \in [4m]} \|\mathbf{w}_i^{(T)} - \mathbf{w}_i^{(1)}\|_2 \tag{B.106}$$

$$+\frac{\|\tilde{a}\|_2^2}{2\eta T} + 4mB_{x1}(\|a^{(1)}\|_\infty + \eta)\left(\max_{i\in[4m]}\|\mathbf{w}_i^{(T)} - \mathbf{w}_i^{(1)}\|_2 + \max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{B_{x1}}\right). \tag{B.107}$$

By Theorem B.6, Theorem B.8, Theorem B.9, when $\eta^{(1)} = \Theta\left(\frac{\min\{O(\eta), O(\eta\tilde{b})\}}{-\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d}+\tilde{b})}\right)$, we have

$$\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right), \quad \|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right) \tag{B.108}$$

$$\|a^{(1)}\|_\infty = O\left(-\eta^{(1)}\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d} + \tilde{b})\right) \tag{B.109}$$

$$= \min\{O(\eta), O(\eta\tilde{b})\} \tag{B.110}$$

$$\max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 = O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon}\right) \tag{B.111}$$

$$\max_{i\in[4m]}\|\mathbf{w}_i^{(T)} - \mathbf{w}_i^{(1)}\|_2 = O\left(T\eta B_{x1}\|a^{(1)}\|_\infty + T\eta^2 B_{x1}^2 \max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 + T\eta^2 B_{x1}\tilde{b}\right) \tag{B.112}$$

$$= O\left(T\eta^2 B_{x1}^2\left(\max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{B_{x1}}\right)\right). \tag{B.113}$$

Set $B_\epsilon = \frac{B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}}$, we have $\sigma_a = \Theta\left(\frac{\tilde{b}(mp)^{\frac{1}{4}}}{-\ell'(0)\eta^{(1)}B_{x1}\sqrt{B_G B_b}}\right)$ which satisfy the requirements. Then,

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.114}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}\left(\frac{2B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma}\right) \tag{B.115}$$

$$+\left(\sqrt{r}B_{a2}B_b T\eta^2 B_{x1}^2\frac{B_{x1}}{\tilde{b}} + m\eta B_{x1}\right) O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon} + \frac{\tilde{b}}{B_{x1}}\right) + O\left(\frac{B_{a2}^2 B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right) \tag{B.116}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}\left(\frac{2B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma}\right) \tag{B.117}$$

$$+\eta\left(\sqrt{r}B_{a2}B_bT\eta B_{x1}^2 + m\tilde{b}\right)O\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right) + O\left(\frac{B_{a2}^2B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right). \tag{B.118}$$

Furthermore, for any $\epsilon \in (0,1)$, set

$$\tilde{b} = \Theta\left(\frac{B_G^{\frac{1}{4}}B_{a2}B_b^{\frac{3}{4}}}{\sqrt{r}B_{a1}}\right), \quad m = \Omega\left(\frac{1}{p\epsilon^4}\left(rB_{a1}B_{x1}\sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{B.119}$$

$$\eta = \Theta\left(\frac{\epsilon}{\left(\frac{\sqrt{r}B_{a2}B_bB_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right)}\right), \quad T = \Theta\left(\frac{1}{\eta B_{x1}(mp)^{\frac{1}{4}}}\right), \tag{B.120}$$

we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}\left(\frac{2B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma}\right) + \frac{\epsilon}{2} \tag{B.121}$$

$$+ O\left(\frac{B_{x1}B_{a2}^2B_b^2}{\tilde{b}^2(mp)^{\frac{1}{4}}}\right) \tag{B.122}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}B_{x1}\sqrt{2\gamma} + \epsilon. \tag{B.123}$$

We finish the proof as the 0-1 classification error is bounded by the loss function, e.g., $\mathbb{I}[\mathrm{sign}(g(\mathbf{x})) \neq y] \leqslant \frac{\ell(yg(\mathbf{x}))}{\ell(0)}$, where $\ell(0) = 1$. □

**Remark B.12.** *In practice, the value of* $\sigma_{\mathbf{w}}$ *cannot be arbitrary, because its choice will have an effect on the Gradient Feature set* $S_{p,\gamma,B_G}$. *On the other hand,* $d \geqslant \log m$ *is a natural assumption, otherwise, the two-layer neural networks may fall in the NTK regime.*

**Remark B.13** (Parameter Choice)**.** *We use $\lambda = 1/\eta$ in the first step so that the neural network will totally forget its initialization, leading to the feature emergence here. This is a common setting for analysis convenience in previous work, e.g., Daniely and Malach (2020); Shi et al. (2022c); Damian et al. (2022). We can extend this to other choices (e.g., small initialization and large step size for the first few steps), as long as after the gradient update, the gradient dominates the neuron weights. We use $\lambda = 0$ afterward as the regularization effect is weak in our analysis. We can extend our analysis to $\lambda$ being a small value.*

**Remark B.14** (Early Stopping)**.** *Our analysis divides network learning into two stages: the feature learning stage, and then classifier learning over the good features. The feature learning stage is simplified to one gradient step for the convenience of analysis, while in practice feature learning can happen in multiple steps. The current framework focuses on the gradient features in the early gradient steps, while feature learning can also happen in later steps, in particular for more complicated data. It is an interesting direction to extend the analysis to a longer training horizon.*

### B.4.3 Gradient Feature Learning Framework under Empirical Risk with Sample Complexity

In this section, we consider training with empirical risk. Intuitively, the proof is straightforward from the proof for population loss. We can simply replace the population loss with the empirical loss, which will introduce an error term in the gradient analysis. We use concentration inequality to control the error term and show that the error term depends inverse-polynomially on the sample size $n$.

**Definition B.15** (Empirical Simplified Gradient Vector)**.** *Recall $\mathcal{Z} = \{(\mathbf{x}^{(l)}, y^{(l)})\}_{l\in[n]}$, for any $\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$, an Empirical Simplified Gradient Vector is defined as*

$$\widetilde{G}(\mathbf{w}, b) := \frac{1}{n} \sum_{l\in[n]} [y^{(l)}\mathbf{x}^{(l)}\mathbb{I}[\mathbf{w}^\top \mathbf{x}^{(l)} > b]]. \tag{B.124}$$

**Definition B.16** (Empirical Gradient Feature)**.** *Recall* $\mathcal{Z} = \{(\mathbf{x}^{(l)}, y^{(l)})\}_{l \in [n]}$*, let* $\mathbf{w} \in \mathbb{R}^d$*,* $b \in \mathbb{R}$ *be random variables drawn from some distribution* $\mathcal{W}, \mathcal{B}$*. An Empirical Gradient Feature set with parameters* $p, \gamma, B_G$ *is defined as:*

$$\widetilde{S}_{p,\gamma,B_G}(\mathcal{W}, \mathcal{B}) := \left\{ (D, s) \;\middle|\; \Pr_{\mathbf{w},b}\left[ \widetilde{G}(\mathbf{w}, b) \in \mathcal{C}_{D,\gamma} \text{ and } \|\widetilde{G}(\mathbf{w}, b)\|_2 \geqslant B_G \text{ and } s = \frac{b}{|b|} \right] \geqslant p \right\}. \tag{B.125}$$

*When clear from context, write it as* $\widetilde{S}_{p,\gamma,B_G}$*.*

Considering training by Algorithm 1, we have the following results.

**Theorem B.17** (Gradient Feature Learning Framework under Empirical Risk with Sample Complexity. Restatement of Theorem 3.12)**.** *Assume Assumption 3.1. For any* $\epsilon, \delta \in (0, 1)$*, if* $m \leqslant e^d$ *and*

$$m = \Omega\left( \frac{1}{p\epsilon^4}\left( rB_{a1}B_{x1}\sqrt{\frac{B_b}{B_G}} \right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left( \log\left( \frac{r}{\delta} \right) \right)^2 \right), \tag{B.126}$$

$$T = \Omega\left( \frac{1}{\epsilon}\left( \frac{\sqrt{r}B_{a2}B_bB_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b} \right)\left( \frac{\sqrt{\log m}}{\sqrt{B_bB_G}} + \frac{1}{B_{x1}(mp)^{\frac{1}{4}}} \right) \right), \tag{B.127}$$

$$n = \Omega\left( \left( \frac{mB_xB_{a2}^2\sqrt{B_b}(mp)^{\frac{1}{2}}\log m}{\epsilon r B_{a1}\sqrt{B_G}} \right)^3 + \left( \frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G})\frac{\sqrt{B_{x2}}}{|\ell'(0)|} \right)^3 \right), \tag{B.128}$$

*then with initialization* (3.9) *and proper hyper-parameter values, we have with probability* $\geqslant 1 - \delta$ *over the initialization and training samples, there exists* $t \in [T]$ *in Algorithm 1 with*

$$\Pr[\text{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.129}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}B_{x1}\sqrt{2\gamma + O\left( \frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|n^{\frac{1}{3}}} \right)} + \epsilon. \tag{B.130}$$

See the full statement and proof in Theorem B.23. Below, we show some Lemma used in the analysis under empirical loss.

**Lemma B.18** (Empirical Gradient Concentration Bound)**.** *When* $n > \left(\frac{B_x}{\sqrt{B_{x2}}}\right)^3$*, with probability at least* $1 - O\left(\exp\left(-n^{\frac{1}{3}}\right)\right)$ *over training samples, for all* $i \in [4m]$*, we have*

$$\left\|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_\Xi)}{\partial \mathbf{w}_i} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_i}\right\|_2 \leqslant O\left(\frac{|a_i|\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right) \tag{B.131}$$

$$\left|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_\Xi)}{\partial a_i} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial a_i}\right| \leqslant O\left(\frac{\|\mathbf{w}_i\|_2\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right) \tag{B.132}$$

$$\left|\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_\Xi\right) - \mathcal{L}_{\mathcal{D}}\left(g_\Xi\right)\right| \leqslant O\left(\frac{\|a\|_0\|a\|_\infty(\max_{i\in[4m]}\|\mathbf{w}_i\|_2 B_x + \tilde{b}) + 1}{n^{\frac{1}{3}}}\right). \tag{B.133}$$

*Proof of Theorem B.18.* First, we define,

$$\mathbf{z}^{(l)} = \ell'(y^{(l)}g_\Xi(\mathbf{x}^{(l)}))y^{(l)}\left[\sigma'\left(\langle \mathbf{w}_i, \mathbf{x}^{(l)}\rangle - b_i\right)\mathbf{x}^{(l)}\right] \tag{B.134}$$

$$- \mathbb{E}_{(\mathbf{x},y)}\left[\ell'(yg_\Xi(\mathbf{x}))y\left[\sigma'\left(\langle \mathbf{w}_i, \mathbf{x}\rangle - b_i\right)\right]\mathbf{x}\right]. \tag{B.135}$$

As $|\ell'(z)| \leqslant 1, |y| \leqslant 1, |\sigma'(z)| \leqslant 1$, we have $\mathbf{z}^{(l)}$ is zero-mean random vector with $\left\|\mathbf{z}^{(l)}\right\|_2 \leqslant 2B_x$ as well as $\mathbb{E}\left[\left\|\mathbf{z}^{(l)}\right\|_2^2\right] \leqslant B_{x2}$. Then by Vector Bernstein Inequality, Lemma 18 in Kohler and Lucchi (2017), for $0 < z < \frac{B_{x2}}{B_x}$ we have

$$\Pr\left(\left\|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_\Xi)}{\partial \mathbf{w}_i} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_i}\right\|_2 \geqslant |a_i|z\right) = \Pr\left(\left\|\frac{1}{n}\sum_{l\in[n]}\mathbf{z}^{(l)}\right\|_2 \geqslant z\right) \tag{B.136}$$

$$\leqslant \exp\left(-n \cdot \frac{z^2}{8B_{x2}} + \frac{1}{4}\right). \tag{B.137}$$

Thus, when $n > \left(\frac{B_x}{\sqrt{B_{x2}}}\right)^3$, with probability at least $1 - O\left(\exp\left(-n^{\frac{1}{3}}\right)\right)$, we have

$$\left\|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_\Xi)}{\partial \mathbf{w}_i} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_i}\right\|_2 \leqslant O\left(\frac{|a_i|\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right). \tag{B.138}$$

On the other hand, by Bernstein Inequality, for $z > 0$ we have

$$\Pr\left(\left|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_\Xi)}{\partial a_i} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial a_i}\right| > z\|\mathbf{w}_i\|_2\right) \tag{B.139}$$

$$= \Pr\left(\left|\frac{1}{n}\sum_{l\in[n]}\left(\ell'(y^{(l)}g_\Xi(\mathbf{x}^{(l)}))y^{(l)}\left[\sigma\left(\langle \mathbf{w}_i, \mathbf{x}^{(l)}\rangle - b_i\right)\right]\right.\right.\right. \tag{B.140}$$

$$\left.\left.\left. - \mathbb{E}_{(\mathbf{x},y)}\left[\ell'(yg_\Xi(\mathbf{x}))y\left[\sigma\left(\langle \mathbf{w}_i, \mathbf{x}\rangle - b_i\right)\right]\right]\right)\right| > z\|\mathbf{w}_i\|_2\right) \tag{B.141}$$

$$\leqslant 2\exp\left(-\frac{\frac{1}{2}nz^2}{B_{x2} + \frac{1}{3}B_x z}\right). \tag{B.142}$$

Thus, when $n > \left(\frac{B_x}{\sqrt{B_{x2}}}\right)^3$, with probability at least $1 - O\left(\exp\left(-n^{\frac{1}{3}}\right)\right)$, we have

$$\left|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_\Xi)}{\partial a_i} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial a_i}\right| \leqslant O\left(\frac{\|\mathbf{w}_i\|_2\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right). \tag{B.143}$$

Finally, we have

$$\left|\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_\Xi\right) - \mathcal{L}_{\mathcal{D}}\left(g_\Xi\right)\right| \tag{B.144}$$

$$= \left|\frac{1}{n}\sum_{l=1}^{n}\left(\ell\left(y^{(l)}\mathbf{a}^\top\left[\sigma(\mathbf{W}^\top\mathbf{x}^{(l)} - \mathbf{b})\right]\right) - \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\ell\left(y\mathbf{a}^\top\left[\sigma(\mathbf{W}^\top\mathbf{x} - \mathbf{b})\right]\right)\right]\right)\right|. \tag{B.145}$$

By Assumption 3.1, we have $\ell\left(y^{(l)}\mathbf{a}^\top\left[\sigma(\mathbf{W}^\top\mathbf{x}^{(l)} - \mathbf{b})\right]\right) - \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}\left[\ell\left(y\mathbf{a}^\top\left[\sigma(\mathbf{W}^\top\mathbf{x} - \mathbf{b})\right]\right)\right]$ is a zero mean random variable, with bound $2\|\mathbf{a}\|_0\|\mathbf{a}\|_\infty(\max_{i\in[4m]}\|\mathbf{w}_i\|_2 B_x + \tilde{b}) + 2$.

By Hoeffding's inequality, for all $z > 0$, we have

$$\Pr\left(\left|\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{\Xi}\right) - \mathcal{L}_{\mathcal{D}}\left(g_{\Xi}\right)\right| \geqslant z\right) \leqslant 2\exp\left(-\frac{z^2 n}{(\|a\|_0\|a\|_\infty(\max_{i\in[4m]}\|\mathbf{w}_i\|_2 B_x + \tilde{b}) + 1)^2}\right). \tag{B.146}$$

Thus, with probability at least $1 - O\left(\exp\left(-n^{\frac{1}{3}}\right)\right)$, we have

$$\left|\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{\Xi}\right) - \mathcal{L}_{\mathcal{D}}\left(g_{\Xi}\right)\right| \leqslant O\left(\frac{\|a\|_0\|a\|_\infty(\max_{i\in[4m]}\|\mathbf{w}_i\|_2 B_x + \tilde{b}) + 1}{n^{\frac{1}{3}}}\right). \tag{B.147}$$

□

The gradients allow for obtaining a set of neurons approximating the "ground-truth" network with comparable loss.

**Lemma B.19** (Existence of Good Networks under Empirical Risk). *Suppose* $n > \Omega\left(\left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{1}{p} + \frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|}\right)^3\right)$. *Let* $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$. *For any* $B_\epsilon \in (0, B_b)$, *let* $\sigma_a = \Theta\left(\frac{\tilde{b}}{-|\ell'(0)|\eta^{(1)}B_G B_\epsilon}\right)$ *and* $\delta = 2re^{-\sqrt{\frac{mp}{2}}}$. *Then, with probability at least* $1-\delta$ *over the initialization and training samples, there exists* $\tilde{a}_i$*'s such that*

$$g_{(\tilde{a},\mathbf{W}^{(1)},b)}(\mathbf{x}) = \sum_{i=1}^{4m} \tilde{a}_i \sigma\left(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle - b_i\right)$$

*satisfies*

$$\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)}) \tag{B.148}$$

$$\leqslant r B_{a1}\left(\frac{2B_{x1}^2 B_b}{\sqrt{mp}B_G B_\epsilon} + B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)} + B_\epsilon\right) + \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}, \tag{B.149}$$

*and* $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$, $\|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right)$, $\|\tilde{a}\|_\infty = O\left(\frac{B_{a1}B_b}{\tilde{b}(mp)^{\frac{1}{2}}}\right)$.

*Proof of Theorem B.19.* Denote $\rho = O\left(\exp\left(-n^{\frac{1}{3}}\right)\right)$ and $\beta = O\left(\frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)$. Note that by symmetric initialization, we have $\ell'(y g_{\Xi^{(0)}}(\mathbf{x})) = |\ell'(0)|$ for any $\mathbf{x} \in \mathcal{X}$, so that, by Theorem B.18, we have $\left\|\widetilde{G}(\mathbf{w}_i^{(0)}, b_i) - G(\mathbf{w}_i^{(0)}, b_i)\right\|_2 \leqslant \frac{\beta}{|\ell'(0)|}$ with probability at least $1-\rho$. Thus, by union bound, we can see that $S_{p,\gamma,B_G} \subseteq \widetilde{S}_{p-\rho,\gamma+\frac{\beta}{B_G|\ell'(0)|},B_G-\frac{\beta}{|\ell'(0)|}}$. Consequently, we have $\mathrm{OPT}_{d,r,B_F,\widetilde{S}_{p-\rho,\gamma+\frac{\beta}{B_G|\ell'(0)|},B_G-\frac{\beta}{|\ell'(0)|}}} \leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}$. Exactly follow the proof in Theorem B.6 by replacing $S_{p,\gamma,B_G}$ to $\widetilde{S}_{p-\rho,\gamma+\frac{\beta}{B_G|\ell'(0)|},B_G-\frac{\beta}{|\ell'(0)|}}$. Then, we finish the proof by $\rho \leqslant \frac{p}{2}$, $\frac{\beta}{|\ell'(0)|} \leqslant (1-1/\sqrt{2})B_G$. □

We will use Theorem B.7 to prove that gradient descent learns a good classifier (Theorem B.23). Theorem B.17 is simply a direct corollary of Theorem B.23. To apply the theorem we first present a few lemmas bounding the change in the network during steps.

**Lemma B.20** (Bound of $\Xi^{(0)}, \Xi^{(1)}$ under Empirical Risk). *Assume the same conditions as in Theorem B.19, and* $d \geqslant \log m$, *with probability at least* $1-\delta-\frac{1}{m^2}-O\left(m\exp\left(-n^{\frac{1}{3}}\right)\right)$ *over the initialization and training samples,* $\|a^{(0)}\|_\infty = O\left(\frac{\tilde{b}\sqrt{\log m}}{|\ell'(0)|\eta^{(1)}B_G B_\epsilon}\right)$, *and for all* $i \in [4m]$, *we have* $\|\mathbf{w}_i^{(0)}\|_2 = O\left(\sigma_{\mathbf{w}}\sqrt{d}\right)$. *Finally,*

$$\|a^{(1)}\|_\infty = O\left(\eta^{(1)}|\ell'(0)|(B_{x1}\sigma_{\mathbf{w}}\sqrt{d}+\tilde{b}) + \eta^{(1)}\frac{\sigma_{\mathbf{w}}\sqrt{d}B_{x2}}{n^{\frac{1}{3}}}\right),$$

*and for all* $i \in [4m]$, $\|\mathbf{w}_i^{(1)}\|_2 = O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon} + \frac{\tilde{b}\sqrt{\log m}\sqrt{B_{x2}}}{|\ell'(0)|B_G B_\epsilon n^{\frac{1}{3}}}\right)$.

*Proof of Theorem B.20.* The proof exactly follows the proof of Theorem B.8 with Theorem B.18. □

**Lemma B.21** (Bound of $\Xi^{(t)}$ under Empirical Risk). *Assume the same conditions as in Theorem B.20, and let* $\eta = \eta^{(t)}$ *for all* $t \in \{2, 3, \ldots, T\}$, $0 < T\eta B_{x1} \leqslant o(1)$, *and*

$0 = \lambda = \lambda^{(t)}$ *for all* $t \in \{2, 3, \ldots, T\}$*. With probability at least* $1 - O\left(Tm \exp\left(-n^{\frac{1}{3}}\right)\right)$ *over training samples, for all* $i \in [4m]$*, for all* $t \in \{2, 3, \ldots, T\}$*, we have*

$$|a_i^{(t)}| \leqslant O\left(|a_i^{(1)}| + \|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{\left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)} + \eta\tilde{b}\right) \tag{B.150}$$

$$\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 \leqslant O\Bigg(t\eta\left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)|a_i^{(1)}| + t\eta^2\left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)^2\|\mathbf{w}_i^{(1)}\|_2 \tag{B.151}$$

$$+ t\eta^2\left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)\tilde{b}\Bigg). \tag{B.152}$$

*Proof of Theorem B.21.* The proof exactly follows the proof of Theorem B.9 with Theorem B.18. Note that, we have

$$|a_i^{(t)}| \leqslant \left|a_i^{(t-1)}\right| + \eta(B_{x1}\|\mathbf{w}_i^{(t-1)}\|_2 + \tilde{b}) + \eta\frac{\|\mathbf{w}_i^{(t-1)}\|_2\sqrt{B_{x2}}}{n^{\frac{1}{3}}} \tag{B.153}$$

$$\leqslant \left|a_i^{(t-1)}\right| + \eta\left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)\|\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(1)}\|_2 + \eta Z_i, \tag{B.154}$$

where we denote $Z_i = \left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)\|\mathbf{w}_i^{(1)}\|_2 + \tilde{b}$. Similarly, we have

$$\|\mathbf{w}_i^{(t)} - \mathbf{w}_i^{(1)}\|_2 \leqslant \|\mathbf{w}_i^{(t-1)} - \mathbf{w}_i^{(1)}\|_2 + \eta\left(B_{x1} + \frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)|a_i^{(t-1)}|. \tag{B.155}$$

We finish the proof by following the same arguments in the proof of Theorem B.9 and union bound. □

**Lemma B.22** (Bound of Loss Gap and Gradient under Empirical Risk)**.** *Assume the same conditions as in Theorem B.21. With probability at least* $1 - O\left(T \exp\left(-n^{\frac{1}{3}}\right)\right)$*, for all* $t \in [T]$*, we have*

$$\left|\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{(\tilde{a},\mathbf{W}^{(t)},b)}\right) - \mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})\right| \tag{B.156}$$

$$\leqslant O\left(\frac{\|\tilde{a}\|_0\|\tilde{a}\|_\infty(\max_{i\in[4m]}\|\mathbf{w}_i^{(t)}\|_2 B_x+\tilde{b})}{n^{\frac{1}{3}}}\right)+B_{x1}\|\tilde{a}\|_2\sqrt{\|\tilde{a}\|_0}\max_{i\in[4m]}\|\mathbf{w}_i^{(t)}-\mathbf{w}_i^{(1)}\|_2. \tag{B.157}$$

*With probability at least* $1-O\left(Tm\exp\left(-n^{\frac{1}{3}}\right)\right)$*, for all* $t\in[T]$, $i\in[4m]$ *we have*

$$\left|\frac{\partial\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}(g_{\Xi^{(t)}})}{\partial a_i^{(t)}}\right|\leqslant B_{x1}(\|\mathbf{w}_i^{(t)}-\mathbf{w}_i^{(1)}\|_2+\|\mathbf{w}_i^{(1)}\|_2)+\tilde{b}+O\left(\frac{\|\mathbf{w}_i^{(t)}\|_2\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right). \tag{B.158}$$

*Proof of Theorem B.22.* By Theorem B.10 and Theorem B.18, with probability at least $1-O\left(T\exp\left(-n^{\frac{1}{3}}\right)\right)$, for all $t\in[T]$, we have

$$\left|\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{(\tilde{a},\mathbf{W}^{(t)},b)}\right)-\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})\right| \tag{B.159}$$

$$\leqslant\left|\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{(\tilde{a},\mathbf{W}^{(t)},b)}\right)-\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(t)},b)})\right|+\left|\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(t)},b)})-\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})\right| \tag{B.160}$$

$$\leqslant O\left(\frac{\|\tilde{a}\|_0\|\tilde{a}\|_\infty(\max_{i\in[4m]}\|\mathbf{w}_i^{(t)}\|_2 B_x+\tilde{b})}{n^{\frac{1}{3}}}\right)+B_{x1}\|\tilde{a}\|_2\sqrt{\|\tilde{a}\|_0}\max_{i\in[4m]}\|\mathbf{w}_i^{(t)}-\mathbf{w}_i^{(1)}\|_2. \tag{B.161}$$

By Theorem B.10 and Theorem B.18, with probability at least $1-O\left(Tm\exp\left(-n^{\frac{1}{3}}\right)\right)$, for all $t\in[T]$, $i\in[4m]$ we have

$$\left|\frac{\partial\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}(g_{\Xi^{(t)}})}{\partial a_i^{(t)}}\right|\leqslant B_{x1}(\|\mathbf{w}_i^{(t)}-\mathbf{w}_i^{(1)}\|_2+\|\mathbf{w}_i^{(1)}\|_2)+\tilde{b}+O\left(\frac{\|\mathbf{w}_i^{(t)}\|_2\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right). \tag{B.162}$$

□

We are now ready to prove the main theorem.

**Theorem B.23** (Online Convex Optimization under Empirical Risk. Full Statement of Theorem B.17)**.** *Consider training by Algorithm 1, and any* $\delta \in (0,1)$*. Assume* $d \geqslant \log m$*. Set*

$$\sigma_{\mathbf{w}} > 0, \quad \tilde{b} > 0, \quad \eta^{(t)} = \eta,\ \lambda^{(t)} = 0 \textit{ for all } t \in \{2,3,\dots,T\}, \tag{B.163}$$

$$\eta^{(1)} = \Theta\left(\frac{\min\{O(\eta), O(\eta\tilde{b})\}}{-\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d} + \tilde{b})}\right), \lambda^{(1)} = \frac{1}{\eta^{(1)}}, \quad \sigma_a = \Theta\left(\frac{\tilde{b}(mp)^{\frac{1}{4}}}{-\ell'(0)\eta^{(1)}B_{x1}\sqrt{B_G B_b}}\right). \tag{B.164}$$

*Let* $m = \Omega\left(\frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right)$*,* $n > \Omega\left(\left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G})\frac{\sqrt{B_{x2}}}{|\ell'(0)|}\right)^3\right)$ *and* $0 < T\eta B_{x1} \leqslant o(1)$*. With probability at least* $1-\delta$ *over the initialization and training samples, there exists* $t \in [T]$ *such that*

$$\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.165}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}\left(\frac{2\sqrt{2}B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)}\right) \tag{B.166}$$

$$+\eta\left(\sqrt{r}B_{a2}B_bT\eta B_{x1}^2 + m\tilde{b}\right)O\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right) + O\left(\frac{B_{a2}^2B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right) \tag{B.167}$$

$$+\frac{1}{n^{\frac{1}{3}}}O\Bigg(\left(\frac{rB_{a1}B_b}{\tilde{b}} + m\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + \frac{\tilde{b}}{B_{x1}}\right)\right) \tag{B.168}$$

$$\cdot\left(\left(\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + T\eta^2B_{x1}\tilde{b}\right)B_x + \tilde{b}\right) + 1\right) \tag{B.169}$$

$$+\frac{1}{n^{\frac{1}{3}}}O\left(m\eta\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + T\eta^2B_{x1}\tilde{b}\right)\sqrt{B_{x2}}\right). \tag{B.170}$$

*Furthermore, for any* $\epsilon \in (0, 1)$*, set*

$$\tilde{b} = \Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{r B_{a1}}}\right), \quad m = \Omega\left(\frac{1}{p\epsilon^4}\left(r B_{a1} B_{x1}\sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{B.171}$$

$$\eta = \Theta\left(\frac{\epsilon}{\left(\frac{\sqrt{r} B_{a2} B_b B_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m} B_{x1} (mp)^{\frac{1}{4}}}{\sqrt{B_b B_G}} + 1\right)}\right), \quad T = \Theta\left(\frac{1}{\eta B_{x1} (mp)^{\frac{1}{4}}}\right), \tag{B.172}$$

$$n = \Omega\left(\left(\frac{m B_x B_{a2}^2 \sqrt{B_b} (mp)^{\frac{1}{2}} \log m}{\epsilon r B_{a1} \sqrt{B_G}}\right)^3 + \left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G})\frac{\sqrt{B_{x2}}}{|\ell'(0)|}\right)^3\right), \tag{B.173}$$

*we have there exists* $t \in [T]$ *with*

$$\Pr[\text{sign}(g_{\Xi^{(t)}})(\mathbf{x}) \neq y] \leq \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.174}$$

$$\leq \text{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + r B_{a1} B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G |\ell'(0)| n^{\frac{1}{3}}}\right)} + \epsilon. \tag{B.175}$$

*Proof of Theorem B.23.* We follow the proof in Theorem B.11. By $m = \Omega\left(\frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right)$ and $n > \Omega\left(\left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G})\frac{\sqrt{B_{x2}}}{|\ell'(0)|}\right)^3\right)$, we have

$$2re^{-\sqrt{\frac{mp}{2}}} + \frac{1}{m^2} + O\left(Tm\exp\left(-n^{\frac{1}{3}}\right)\right) \leq \delta.$$

For any $B_\epsilon \in (0, B_b)$, when $\sigma_a = \Theta\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)} B_G B_\epsilon}\right)$, by Theorem B.7, Theorem B.18, Theorem B.19, Theorem B.22, with probability at least $1 - \delta$ over the initialization and training samples, we have

$$
\begin{aligned}
&\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) && \text{(B.176)}\\
\leqslant&\frac{1}{T}\sum_{t=1}^{T}\left|\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right)-\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{\Xi^{(t)}}\right)\right|+\frac{1}{T}\sum_{t=1}^{T}\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{\Xi^{(t)}}\right) && \text{(B.177)}\\
\leqslant&\frac{1}{T}\sum_{t=1}^{T}\left|\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right)-\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{\Xi^{(t)}}\right)\right|+\frac{1}{T}\sum_{t=1}^{T}\left|\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}\left(g_{(\tilde{a},\mathbf{W}^{(t)},b)}\right)-\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})\right| && \text{(B.178)}\\
&+\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a},\mathbf{W}^{(1)},b)})+\frac{\|\tilde{a}\|_2^2}{2\eta T}+(2\|a^{(1)}\|_2\sqrt{m}+4\eta m)\max_{t\in[T],i\in[4m]}\left|\frac{\partial\widetilde{\mathcal{L}}_{\mathcal{Z}^{(t)}}(g_{\Xi^{(t)}})}{\partial a_i^{(t)}}\right| && \text{(B.179)}\\
\leqslant&B_{x1}\|\tilde{a}\|_2\sqrt{\|\tilde{a}\|_0}\max_{i\in[4m]}\|\mathbf{w}_i^{(T)}-\mathbf{w}_i^{(1)}\|_2 && \text{(B.180)}\\
&+O\left(\frac{(\|\tilde{a}\|_0\|\tilde{a}\|_\infty+m\|a^{(T)}\|_\infty)(\max_{i\in[4m]}\|\mathbf{w}_i^{(T)}\|_2B_x+\tilde{b})+1}{n^{\frac{1}{3}}}\right) && \text{(B.181)}\\
&+\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}}+rB_{a1}\left(\frac{2B_{x1}^2B_b}{\sqrt{mp}B_GB_\epsilon}+B_{x1}\sqrt{2\gamma+O\left(\frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)}+B_\epsilon\right) && \text{(B.182)}\\
&+\frac{\|\tilde{a}\|_2^2}{2\eta T}+4mB_{x1}(\|a^{(1)}\|_\infty+\eta) && \text{(B.183)}\\
&\cdot\left(\max_{i\in[4m]}\|\mathbf{w}_i^{(T)}-\mathbf{w}_i^{(1)}\|_2+\max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2+\frac{\tilde{b}}{B_{x1}}+O\left(\frac{\max_{i\in[4m]}\|\mathbf{w}_i^{(T)}\|_2\sqrt{B_{x2}}}{B_{x1}n^{\frac{1}{3}}}\right)\right). && \text{(B.184)}
\end{aligned}
$$

Set $B_\epsilon=\frac{B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{2B_b}{B_G}}$, we have $\sigma_a=\Theta\left(\frac{\tilde{b}(mp)^{\frac{1}{4}}}{-\ell'(0)\eta^{(1)}B_{x1}\sqrt{B_GB_b}}\right)$ which satisfy the requirements. By Theorem B.19, Theorem B.20, Theorem B.21,

$$n > \Omega\left(\left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1+\frac{1}{B_G})\frac{\sqrt{B_{x2}}}{|\ell'(0)|}\right)^3\right),$$

when $\eta^{(1)} = \Theta\left(\frac{\min\{O(\eta),O(\eta\tilde{b})\}}{-\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d}+\tilde{b})}\right)$, we have

$$\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right), \quad \|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right), \quad \|\tilde{a}\|_\infty = O\left(\frac{B_{a1}B_b}{\tilde{b}(mp)^{\frac{1}{2}}}\right) \tag{B.185}$$

$$\|a^{(1)}\|_\infty = O\left(\eta^{(1)}|\ell'(0)|(B_{x1}\sigma_{\mathbf{w}}\sqrt{d}+\tilde{b}) + \eta^{(1)}\frac{\sigma_{\mathbf{w}}\sqrt{dB_{x2}}}{n^{\frac{1}{3}}}\right) \tag{B.186}$$

$$= \min\{O(\eta), O(\eta\tilde{b})\} \tag{B.187}$$

$$\|a^{(T)}\|_\infty \leqslant O\left(\|a^{(1)}\|_\infty + \max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{\left(B_{x1}+\frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)} + \eta\tilde{b}\right) \tag{B.188}$$

$$\leqslant O\left(\max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{B_{x1}}\right) \tag{B.189}$$

$$\max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 = O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon} + \frac{\tilde{b}\sqrt{\log m}\sqrt{B_{x2}}}{|\ell'(0)|B_G B_\epsilon n^{\frac{1}{3}}}\right) \tag{B.190}$$

$$= O\left(\frac{\tilde{b}\sqrt{\log m}B_{x1}}{B_G B_\epsilon}\right) = O\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_b B_G}}\right) \tag{B.191}$$

$$\max_{i\in[4m]}\|\mathbf{w}_i^{(T)} - \mathbf{w}_i^{(1)}\|_2 = O\Bigg(T\eta\left(B_{x1}+\frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)|a_i^{(1)}| + T\eta^2\left(B_{x1}+\frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)^2\|\mathbf{w}_i^{(1)}\|_2 \tag{B.192}$$

$$+ T\eta^2\left(B_{x1}+\frac{\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right)\tilde{b}\Bigg) \tag{B.193}$$

$$= O\left(T\eta^2 B_{x1}^2\left(\max_{i\in[4m]}\|\mathbf{w}_i^{(1)}\|_2 + \frac{\tilde{b}}{B_{x1}}\right)\right). \tag{B.194}$$

Then, following the proof in Theorem B.11, we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.195}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}\left(\frac{2\sqrt{2}B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)}\right) \tag{B.196}$$

$$+\eta\left(\sqrt{r}B_{a2}B_bT\eta B_{x1}^2 + m\tilde{b}\right)O\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right) + O\left(\frac{B_{a2}^2B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right) \tag{B.197}$$

$$+O\left(\frac{(\|\tilde{a}\|_0\|\tilde{a}\|_\infty + m\|a^{(T)}\|_\infty)(\max_{i\in[4m]}\|\mathbf{w}_i^{(T)}\|_2B_x + \tilde{b}) + 1}{n^{\frac{1}{3}}}\right) \tag{B.198}$$

$$+O\left(\frac{m\eta\max_{i\in[4m]}\|\mathbf{w}_i^{(T)}\|_2\sqrt{B_{x2}}}{n^{\frac{1}{3}}}\right) \tag{B.199}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + rB_{a1}\left(\frac{2\sqrt{2}B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)}\right) \tag{B.200}$$

$$+\eta\left(\sqrt{r}B_{a2}B_bT\eta B_{x1}^2 + m\tilde{b}\right)O\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right) + O\left(\frac{B_{a2}^2B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right) \tag{B.201}$$

$$+\frac{1}{n^{\frac{1}{3}}}O\Bigg(\left(\frac{rB_{a1}B_b}{\tilde{b}} + m\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + \frac{\tilde{b}}{B_{x1}}\right)\right) \tag{B.202}$$

$$\cdot\left(\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + T\eta^2B_{x1}\tilde{b}\right)B_x + \tilde{b}\right) + 1\Bigg) \tag{B.203}$$

$$+\frac{1}{n^{\frac{1}{3}}}O\left(m\eta\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + T\eta^2B_{x1}\tilde{b}\right)\sqrt{B_{x2}}\right). \tag{B.204}$$

Furthermore, for any $\epsilon \in (0,1)$, set

$$\tilde{b} =\Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{r B_{a1}}}\right), \quad m = \Omega\left(\frac{1}{p\epsilon^4}\left(r B_{a1} B_{x1}\sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{B.205}$$

$$\eta =\Theta\left(\frac{\epsilon}{\left(\frac{\sqrt{r} B_{a2} B_b B_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m} B_{x1} (mp)^{\frac{1}{4}}}{\sqrt{B_b B_G}} + 1\right)}\right), \quad T = \Theta\left(\frac{1}{\eta B_{x1} (mp)^{\frac{1}{4}}}\right), \tag{B.206}$$

$$n =\Omega\left(\left(\frac{m B_x B_{a2}^2 \sqrt{B_b} (mp)^{\frac{1}{2}} \log m}{\epsilon r B_{a1}\sqrt{B_G}}\right)^3 + \left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G})\frac{\sqrt{B_{x2}}}{|\ell'(0)|}\right)^3\right), \tag{B.207}$$

and note that $B_G \leqslant B_{x_1} \leqslant B_x$ and $\sqrt{B_{x_2}} \leqslant B_x$ naturally, we have

$$\frac{1}{T}\sum_{t=1}^{T} \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.208}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + r B_{a1}\left(\frac{2\sqrt{2} B_{x1}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G |\ell'(0)| n^{\frac{1}{3}}}\right)}\right) \tag{B.209}$$

$$+ \frac{\epsilon}{2} + O\left(\frac{B_{x1} B_{a2}^2 B_b^2}{\tilde{b}^2 (mp)^{\frac{1}{4}}}\right) + O\left(\frac{m B_x B_{a2}^2 \sqrt{B_b} (mp)^{\frac{1}{2}} \log m}{n^{\frac{1}{3}} r B_{a1}\sqrt{B_G}}\right) \tag{B.210}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} + r B_{a1} B_{x1}\sqrt{2\gamma + O\left(\frac{\sqrt{B_{x2}}}{B_G |\ell'(0)| n^{\frac{1}{3}}}\right)} + \epsilon. \tag{B.211}$$

We finish the proof as the 0-1 classification error is bounded by the loss function, e.g., $\mathbb{I}[\mathrm{sign}(g(\mathbf{x})) \neq y] \leqslant \frac{\ell(y g(\mathbf{x}))}{\ell(0)}$, where $\ell(0) = 1$.

□

## B.5 Applications in Special Cases

We present the case study of linear data in Section B.5.1, mixtures of Gaussians in Section B.5.2 and Section B.5.3, parity functions in Section B.5.4, Section B.5.5 and Section B.5.6, and multiple-index models in Section B.5.7.

In special case applications, we consider binary classification with hinge loss, e.g., $\ell(z) = \max\{1 - z, 0\}$. Let $\mathbf{X} = \mathbb{R}^d$ be the input space, and $\mathcal{Y} = \{\pm 1\}$ be the label space.

**Remark B.24** (Hinge Loss and Logistic Loss)**.** *Both hinge loss and logistic loss can be used in special cases and general cases. For convenience, we use hinge loss in special cases, where we can directly get the ground-truth NN close form of the optimal solution which has zero loss. For logistic loss, there is no zero-loss solution. We can still show that the OPT value has an exponentially small upper bound at the cost of more computation.*

### B.5.1 Linear Data

**Data Distributions.** Suppose two labels are equiprobable, i.e., $\mathbb{E}[y = -1] = \mathbb{E}[y = +1] = \frac{1}{2}$. The input data are linearly separable and there is a ground truth direction $\mathbf{w}^*$, where $\|\mathbf{w}^*\|_2 = 1$, such that $y\langle \mathbf{w}^*, \mathbf{x}\rangle > 0$. We also assume $\mathbb{E}[yP_{\mathbf{w}^{*\perp}}\mathbf{x}] = 0$, where $P_{\mathbf{w}^{*\perp}}$ is the projection operator on the complementary space of the ground truth, i.e., the components of input data being orthogonal with the ground truth are independent of the label $y$. We define the input data signal level as $\rho := \mathbb{E}[y\langle \mathbf{w}^*, \mathbf{x}\rangle] > 0$ and the margin as $\beta := \min_{(\mathbf{x},y)} y\langle \mathbf{w}^*, \mathbf{x}\rangle > 0$.

We call this data distribution $\mathcal{D}_{\mathtt{linear}}$.

**Lemma B.25** (Linear Data: Gradient Feature Set)**.** *Let* $\tilde{b} = d^{\tau} B_{x1} \sigma_{\mathbf{w}}$, *where* $\tau$ *is any number large enough to satisfy* $d^{\tau/2 - \frac{1}{4}} > \Omega\left(\frac{\sqrt{B_{x2}}}{\rho}\right)$. *For* $\mathcal{D}_{\mathtt{linear}}$ *setting, we have* $(\mathbf{w}^*, -1) \in S_{p,\gamma,B_G}$ *where*

$$p = \frac{1}{2}, \quad \gamma = \Theta\left(\frac{\sqrt{B_{x2}}}{\rho d^{\tau/2 - \frac{1}{4}}}\right), \quad B_G = \rho - \Theta\left(\frac{\sqrt{B_{x2}}}{d^{\tau/2 - \frac{1}{4}}}\right). \tag{B.212}$$

*Proof of Theorem B.25.* By data distribution, we have

$$\mathbb{E}_{(\mathbf{x},y)}[y\mathbf{x}] = \rho\mathbf{w}^*. \tag{B.213}$$

Define $S_{Sure} : \{i \in [m] : \|\mathbf{w}_i^{(0)}\|_2 \leqslant 2\sqrt{d}\sigma_w\}$. For all $i \in [m]$, we have

$$\Pr[i \in S_{Sure}] = \Pr[\|\mathbf{w}_i^{(0)}\|_2 \leqslant 2\sqrt{d}\sigma_w] \geqslant \frac{1}{2}. \tag{B.214}$$

For all $i \in S_{Sure}$, by Markov's inequality and considering neuron $i + m$, we have

$$\begin{aligned}
\Pr_{\mathbf{x}}\left[\langle \mathbf{w}_{i+m}^{(0)}, \mathbf{x}\rangle - b_{i+m} < 0\right] =& \Pr_{\mathbf{x}}\left[\langle \mathbf{w}_{i}^{(0)}, \mathbf{x}\rangle + b_{i} < 0\right] && \text{(B.215)}\\
\leqslant& \Pr_{\mathbf{x}}\left[\|\mathbf{w}_i^{(0)}\|_2\|\mathbf{x}\|_2 \geqslant b_i\right] && \text{(B.216)}\\
\leqslant& \Pr_{\mathbf{x}}\left[\|\mathbf{x}\|_2 \geqslant \frac{d^{\tau-\frac{1}{2}}B_{x1}}{2}\right] && \text{(B.217)}\\
\leqslant& \Theta\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right). && \text{(B.218)}
\end{aligned}$$

For all $i \in S_{Sure}$, by Hölder's inequality, we have

$$\begin{aligned}
&\left\|\mathbb{E}_{(\mathbf{x},y)}\left[y\left(1-\sigma'\left[\langle \mathbf{w}_{i+m}^{(0)}, \mathbf{x}\rangle - b_{i+m}\right]\right)\mathbf{x}\right]\right\|_2 && \text{(B.219)}\\
=&\left\|\mathbb{E}_{(\mathbf{x},y)}\left[y\left(1-\sigma'\left[\langle \mathbf{w}_{i}^{(0)}, \mathbf{x}\rangle + b_{i}\right]\right)\mathbf{x}\right]\right\|_2 && \text{(B.220)}\\
\leqslant&\sqrt{\mathbb{E}[\|\mathbf{x}\|_2^2]\mathbb{E}\left[\left(1-\sigma'\left[\langle \mathbf{w}_{i}^{(0)}, \mathbf{x}\rangle + b_{i}\right]\right)^2\right]} && \text{(B.221)}\\
\leqslant&\Theta\left(\frac{\sqrt{B_{x2}}}{d^{\tau/2-\frac{1}{4}}}\right). && \text{(B.222)}
\end{aligned}$$

We have

$$1 - \frac{\left|\langle G(\mathbf{w}_{i+m}^{(0)}, b_{i+m}), \mathbf{w}^*\rangle\right|}{\|G(\mathbf{w}_{i+m}^{(0)}, b_{i+m})\|_2} = 1 - \frac{\left|\langle G(\mathbf{w}_{i}^{(0)}, -b_{i}), \mathbf{w}^*\rangle\right|}{\|G(\mathbf{w}_{i}^{(0)}, -b_{i})\|_2} \tag{B.223}$$

$$\leqslant 1 - \frac{\rho - \Theta\left(\frac{\sqrt{B_{x2}}}{d^{\tau/2-\frac{1}{4}}}\right)}{\rho + \Theta\left(\frac{\sqrt{B_{x2}}}{d^{\tau/2-\frac{1}{4}}}\right)} \tag{B.224}$$

$$= \Theta\left(\frac{\sqrt{B_{x2}}}{\rho d^{\tau/2-\frac{1}{4}}}\right) = \gamma. \tag{B.225}$$

We finish the proof by $\frac{b_{i+m}}{|b_{i+m}|} = -1$. □

**Lemma B.26** (Linear Data: Existence of Good Networks)**.** *Assume the same conditions as in Theorem B.25. Define*

$$g^*(\mathbf{x}) = \frac{1}{\beta}\sigma(\langle \mathbf{w}^*, \mathbf{x}\rangle) - \frac{1}{\beta}\sigma(\langle -\mathbf{w}^*, \mathbf{x}\rangle). \tag{B.226}$$

*For $\mathcal{D}_{\mathrm{linear}}$ setting, we have $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$, where $r = 2, B_F = (B_{a1}, B_{a2}, B_b) = \left(\frac{1}{\beta}, \frac{\sqrt{2}}{\beta}, \frac{1}{B_{x1}^2}\right), p = \frac{1}{2}, \gamma = \Theta\left(\frac{\sqrt{B_{x2}}}{\rho d^{\tau/2-\frac{1}{4}}}\right), B_G = \rho - \Theta\left(\frac{\sqrt{B_{x2}}}{d^{\tau/2-\frac{1}{4}}}\right)$. We also have $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} = 0$.*

*Proof of Theorem B.26.* By Theorem B.25 and Theorem B.72, we have $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$. We also have

$$\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} \leqslant \mathcal{L}_{\mathcal{D}_{\mathrm{linear}}}(g^*) \tag{B.227}$$

$$= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\mathrm{linear}}} \mathcal{L}_{(\mathbf{x},y)}(g^*) \tag{B.228}$$

$$= 0. \tag{B.229}$$

□

**Theorem B.27** (Linear Data: Main Result)**.** *For $\mathcal{D}_{\mathrm{linear}}$ setting, for any $\delta \in (0,1)$ and for any $\epsilon \in (0,1)$ when*

$$m = \mathrm{poly}\left(\frac{1}{\delta}, \frac{1}{\epsilon}, \frac{1}{\beta}, \frac{1}{\rho}\right) \leqslant e^d, \quad T = \mathrm{poly}\left(m, B_{x1}\right), \quad n = \mathrm{poly}\left(m, B_x, \frac{1}{\delta}, \frac{1}{\epsilon}, \frac{1}{\beta}, \frac{1}{\rho}\right), \tag{B.230}$$

*trained by Algorithm 1 with hinge loss, with probability at least* $1-\delta$ *over the initialization, with proper hyper-parameters, there exists* $t \in [T]$ *such that*

$$\Pr[\operatorname{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \epsilon. \tag{B.231}$$

*Proof of Theorem B.27.* Let $\tilde{b} = d^\tau B_{x1} \sigma_{\mathbf{w}}$, where $\tau$ is a number large enough to satisfy $d^{\tau/2-\frac{1}{4}} > \Omega\left(\frac{\sqrt{B_{x2}}}{\rho}\right)$ and $O\left(\frac{B_{x1} B_{x2}^{\frac{1}{4}}}{\beta\sqrt{\rho} d^{\tau/4-\frac{1}{8}}}\right) \leqslant \frac{\epsilon}{2}$. By Theorem B.26, we have $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$, where $r = 2, B_F = (B_{a1}, B_{a2}, B_b) = \left(\frac{1}{\beta}, \frac{\sqrt{2}}{\beta}, \frac{1}{B_{x1}^2}\right)$, $p = \frac{1}{2}$ , $\gamma = \Theta\left(\frac{\sqrt{B_{x2}}}{\rho d^{\tau/2-\frac{1}{4}}}\right)$, $B_G = \rho - \Theta\left(\frac{\sqrt{B_{x2}}}{d^{\tau/2-\frac{1}{4}}}\right)$. We also have $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} = 0$.

Adjust $\sigma_{\mathbf{w}}$ such that $\tilde{b} = d^\tau B_{x1}\sigma_{\mathbf{w}} = \Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{r B_{a1}}}\right)$. Injecting above parameters into Theorem 3.12, we have with probability at least $1-\delta$ over the initialization, with proper hyper-parameters, there exists $t \in [T]$ such that

$$\Pr[\operatorname{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant O\left(\frac{B_{x1} B_{x2}^{\frac{1}{4}}}{\beta\sqrt{\rho} d^{\tau/4-\frac{1}{8}}}\right) + O\left(\frac{B_{x1} B_{x2}{}^{\frac{1}{4}}}{\beta\sqrt{\rho} n^{\frac{1}{6}}}\right) + \epsilon/2 \leqslant \epsilon. \tag{B.232}$$

□

### B.5.2 Mixture of Gaussians

We recap the problem setup in Section 3.4.1 for readers' convenience.

#### Problem Setup

**Data Distributions.** We follow the notations from Refinetti et al. (2021). The data are from a mixture of $r$ high-dimensional Gaussians, and each Gaussian is assigned to one of two possible labels in $\mathcal{Y} = \{\pm 1\}$. Let $\mathcal{S}(y) \subseteq [r]$ denote the set of indices of the Gaussians associated with the label $y$. The data distribution is then:

$$q(\mathbf{x}, y) = q(y) q(\mathbf{x}|y), \quad q(\mathbf{x}|y) = \sum_{j \in \mathcal{S}(y)} p_j \mathcal{N}_j(\mathbf{x}), \tag{B.233}$$

where $\mathcal{N}_j(\mathbf{x})$ is a multivariate normal distribution with mean $\mu_j$ and covariance $\Sigma_j$, and $p_j$ are chosen such that $q(\mathbf{x}, y)$ is correctly normalized.

We call this data distribution $\mathcal{D}_{\mathrm{mixture}}$.

We will make some assumptions about the Gaussians, for which we first introduce some notations. For all $j \in [r]$, let $y_{(j)} \in \{+1, -1\}$ be the label for $\mathcal{N}_j(\mathbf{x})$.

$$D_j := \frac{\mu_j}{\|\mu_j\|_2}, \quad \tilde{\mu}_j := \mu_j/\sqrt{d}, \quad B_{\mu 1} := \min_{j \in [r]} \|\tilde{\mu}_j\|_2, \quad B_{\mu 2} := \max_{j \in [r]} \|\tilde{\mu}_j\|_2, \quad p_B := \min_{j \in [r]} p_j.$$

**Assumption B.28** (Mixture of Gaussians. Recap of Assumption 3.15)**.** *Let* $8 \leqslant \tau \leqslant d$ *be a parameter that will control our final error guarantee. Assume*

- *Equiprobable labels:* $q(-1) = q(+1) = 1/2$.
- *For all* $j \in [r]$, $\Sigma_j = \sigma_j I_{d \times d}$. *Let* $\sigma_B := \max_{j \in [r]} \sigma_j$ *and* $\sigma_{B+} := \max\{\sigma_B, B_{\mu 2}\}$.
- $r \leqslant 2d, \quad p_B \geqslant \frac{1}{2d}, \quad \Omega\left(\frac{1}{d} + \sqrt{\frac{\tau {\sigma_{B+}}^2 \log d}{d}}\right) \leqslant B_{\mu 1} \leqslant B_{\mu 2} \leqslant d.$
- *The Gaussians are well-separated: for all* $i \neq j \in [r]$, *we have* $-1 \leqslant \langle D_i, D_j \rangle \leqslant \theta$, *where* $0 \leqslant \theta \leqslant \min\left\{\frac{1}{2r}, \frac{\sigma_{B+}}{B_{\mu 2}}\sqrt{\frac{\tau \log d}{d}}\right\}$.

Below, we define a sufficient condition that randomly initialized weights will fall in nice gradients set after the first gradient step update.

**Definition B.29** (Mixture of Gaussians: Subset of Nice Gradients Set)**.** *Recall* $\mathbf{w}_i^{(0)}$ *is the weight for the* $i$*-th neuron at initialization. For all* $j \in [r]$, *let* $S_{D_j, \mathrm{Sure}} \subseteq [m]$ *be those neurons that satisfy*

- $\langle \mathbf{w}_i^{(0)}, \mu_j \rangle \geqslant C_{\mathrm{Sure},1} b_i$,
- $\langle \mathbf{w}_i^{(0)}, \mu_{j'} \rangle \leqslant C_{\mathrm{Sure},2} b_i$, *for all* $j' \neq j, j' \in [r]$.
- $\left\|\mathbf{w}_i^{(0)}\right\|_2 \leqslant \Theta(\sqrt{d}\sigma_{\mathbf{w}})$.

**Mixture of Gaussians: Feature Learning**

We show the important Theorem B.30 first and defer other Lemmas after it.

**Lemma B.30** (Mixture of Gaussians: Gradient Feature Set. Part statement of Theorem 3.17)**.** *Let* $C_{\mathrm{Sure},1} = \frac{3}{2}$, $C_{\mathrm{Sure},2} = \frac{1}{2}$, $\tilde{b} = C_b\sqrt{\tau d \log d}\sigma_{\mathbf{w}}\sigma_{B+}$, *where* $C_b$ *is a large enough universal constant. For* $\mathcal{D}_{\mathrm{mixture}}$ *setting, we have* $(D_j, +1) \in S_{p,\gamma,B_G}$ *for all* $j \in [r]$, *where*

$$p = \Theta\left(\frac{B_{\mu 1}}{\sqrt{\tau \log d}\sigma_{B+} \cdot d^{\left(9C_b^2\tau\sigma_{B+}{}^2/(2B_{\mu 1}^2)\right)}}\right), \quad \gamma = \frac{1}{d^{0.9\tau-1.5}}, \tag{B.234}$$

$$B_G = p_B B_{\mu 1}\sqrt{d} - O\left(\frac{\sigma_{B+}}{d^{0.9\tau}}\right). \tag{B.235}$$

*Proof of Theorem B.30.* For all $j \in [r]$, by Theorem B.33, for all $i \in S_{D_j,\mathrm{Sure}}$,

$$1 - \frac{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2} \tag{B.236}$$

$$\leqslant 1 - \frac{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|}{\sqrt{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|^2 + \max_{D_j^\top D_j^\perp=0, \|D_j^\perp\|_2=1}\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j^\perp\rangle\right|^2}} \tag{B.237}$$

$$\leqslant 1 - \frac{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|}{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right| + \max_{D_j^\top D_j^\perp=0, \|D_j^\perp\|_2=1}\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j^\perp\rangle\right|} \tag{B.238}$$

$$\leqslant 1 - \frac{1}{1 + \frac{B_{\mu 2}O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) + \sigma_{B+}O\left(\frac{1}{d^{0.9\tau}}\right)}{p_j B_{\mu 1}\sqrt{d}\left(1-O\left(\frac{1}{d^\tau}\right)\right) - B_{\mu 2}O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) - \sigma_{B+}O\left(\frac{1}{d^{0.9\tau}}\right)}} \tag{B.239}$$

$$\leqslant \frac{\sigma_{B+}O\left(\frac{1}{d^{0.9\tau}}\right)}{p_j B_{\mu 1}\sqrt{d} - \sigma_{B+}O\left(\frac{1}{d^{0.9\tau}}\right)} \tag{B.240}$$

$$< \frac{1}{d^{0.9\tau-1.5}} = \gamma, \tag{B.241}$$

where the last inequality follows $B_{\mu 1} \geqslant \Omega\left(\sigma_{B+}\sqrt{\frac{\tau \log d}{d}}\right)$.

Thus, we have $G(\mathbf{w}_i^{(0)}, b_i) \in \mathcal{C}_{D_j,\gamma}$ and $\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right| \leqslant \|G(\mathbf{w}_i^{(0)}, b_i)\|_2 \leqslant B_{x1}$, $\frac{b_i}{|b_i|} = +1$. Thus, by Theorem B.31, we have

$$\Pr_{\mathbf{w},b}\left[G(\mathbf{w}, b) \in \mathcal{C}_{D_j,\gamma} \text{ and } \|G(\mathbf{w}, b)\|_2 \geqslant B_G \text{ and } \frac{b}{|b|} = +1\right] \tag{B.242}$$

$$\geqslant \Pr\left[i \in S_{D_j,Sure}\right] \tag{B.243}$$

$$\geqslant p. \tag{B.244}$$

Thus, $(D_j, +1) \in S_{p,\gamma,B_G}$. We finish the proof. □

Below are Lemmas used in the proof of Theorem B.30. In Theorem B.31, we calculate $p$ used in $S_{p,\gamma,B_G}$.

**Lemma B.31** (Mixture of Gaussians: Geometry at Initialization. Lemma B.2 in Allen-Zhu and Li (2022))**.** *Assume the same conditions as in Theorem B.30, recall for all* $i \in [m]$, $\mathbf{w}_i^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 I_{d\times d})$, *over the random initialization, we have for all* $i \in [m], j \in [r]$,

$$\Pr\left[i \in S_{D_j,Sure}\right] \geqslant \Theta\left(\frac{B_{\mu 1}}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(9C_b^2 \tau \sigma_{B+}{}^2/(2B_{\mu 1}^2)\right)}}\right). \tag{B.245}$$

*Proof of Theorem B.31.* Recall for all $l \in [r]$, $\tilde{\mu}_l = \mu_l/\sqrt{d}$.

WLOG, let $j = r$. For all $l \in [r-1]$. We define $Z_1 = \{l \in [r-1] : \langle D_l, D_r\rangle \geqslant -\theta\}$ and $Z_2 = \{l \in [r-1] : -1 < \langle D_l, D_r\rangle < -\theta\}$. WLOG, let $Z_1 = [r_1]$, $Z_2 = \{r_1+1, \dots, r_2\}$, where $0 \leqslant r_1 \leqslant r_2 \leqslant r-1$. We define the following events

$$\zeta_l = \left\{\langle \mathbf{w}_i^{(0)}, \mu_l\rangle \leqslant C_{Sure,2} b_i\right\}, \hat{\zeta}_l = \left\{\left|\langle \mathbf{w}_i^{(0)}, \mu_l\rangle\right| \leqslant C_{Sure,2} b_i\right\}. \tag{B.246}$$

We define space $A = \text{span}(\mu_1, \dots, \mu_{r_1})$ and $\hat{\mu}_r = P_{A^\perp}\mu_r$, where $P_{A^\perp}$ is the projection operator on the complementary space of $A$. For $l \in Z_2$, we also define $\dot{\mu}_l =$

$\mu_l - \frac{\langle \mu_l, \mu_r \rangle \mu_r}{\|\mu_r\|_2^2}$ , and the event

$$\dot{\zeta}_l = \left\{ \langle \mathbf{w}_i^{(0)}, \dot{\mu}_l \rangle \leqslant C_{\mathcal{S}ure,2} b_i \right\}, \hat{\dot{\zeta}}_l = \left\{ \left| \langle \mathbf{w}_i^{(0)}, \dot{\mu}_l \rangle \right| \leqslant C_{\mathcal{S}ure,2} b_i \right\}. \tag{B.247}$$

For $l \in Z_2$, we have $\mu_l = \dot{\mu}_l - \rho\mu_r$, where $\rho \geqslant 0$. So $\langle \mathbf{w}, \mu_l \rangle = \langle \mathbf{w}, \dot{\mu}_l \rangle - \rho \langle \mathbf{w}, \mu_r \rangle \leqslant \langle \mathbf{w}, \dot{\mu}_l \rangle$ when $\langle \mathbf{w}, \mu_r \rangle \geqslant 0$. As a result, we have

$$\dot{\zeta}_l \cap \left\{ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \right\} \subseteq \zeta_l \cap \left\{ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \right\}. \tag{B.248}$$

By Assumption B.28, we have

$$\frac{1}{2} \leqslant 1 - r\theta \leqslant 1 - r_1\theta \leqslant \frac{\|\hat{\mu}_r\|_2}{\|\mu_r\|_2} \leqslant 1. \tag{B.249}$$

We also have,

$$\begin{aligned}
& \Pr\left[ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i, \zeta_1, \ldots, \zeta_{r-1} \right] && \text{(B.250)} \\
= & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i, \zeta_1, \ldots, \zeta_{r_2} \right] && \text{(B.251)} \\
\geqslant & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i, \zeta_1, \ldots, \zeta_{r_1}, \dot{\zeta}_{r_1+1}, \ldots, \dot{\zeta}_{r_2} \right] && \text{(B.252)} \\
\geqslant & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i, \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}, \hat{\dot{\zeta}}_{r_1+1}, \ldots, \hat{\dot{\zeta}}_{r_2} \right] && \text{(B.253)} \\
= & \underbrace{\Pr\left[ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \middle| \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}, \hat{\dot{\zeta}}_{r_1+1}, \ldots, \hat{\dot{\zeta}}_{r_2} \right]}_{p_r} \underbrace{\Pr\left[ \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}, \hat{\dot{\zeta}}_{r_1+1}, \ldots, \hat{\dot{\zeta}}_{r_2} \right]}_{\Pi_{l \in [r_2]} p_l}. && \\
& && \text{(B.254)}
\end{aligned}$$

For the first condition in Theorem B.29, we have,

$$\begin{aligned}
p_r = & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \mu_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \middle| \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}, \hat{\dot{\zeta}}_{r_1+1}, \ldots, \hat{\dot{\zeta}}_{r_2} \right] && \text{(B.255)} \\
= & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \hat{\mu}_r + \mu_r - \hat{\mu}_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \middle| \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1} \right] && \text{(B.256)} \\
\geqslant & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \hat{\mu}_r + \mu_r - \hat{\mu}_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i, \ \langle \mathbf{w}_i^{(0)}, \mu_r - \hat{\mu}_r \rangle \geqslant 0 \middle| \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1} \right] && \text{(B.257)} \\
= & \Pr\left[ \langle \mathbf{w}_i^{(0)}, \hat{\mu}_r + \mu_r - \hat{\mu}_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \middle| \ \langle \mathbf{w}_i^{(0)}, \mu_r - \hat{\mu}_r \rangle \geqslant 0, \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1} \right] && \text{(B.258)}
\end{aligned}$$

$$\cdot \Pr\left[\langle \mathbf{w}_i^{(0)}, \mu_r - \hat{\mu}_r \rangle \geqslant 0 \middle| \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}\right] \tag{B.259}$$

$$= \frac{1}{2} \Pr\left[\langle \mathbf{w}_i^{(0)}, \hat{\mu}_r + \mu_r - \hat{\mu}_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \middle| \langle \mathbf{w}_i^{(0)}, \mu_r - \hat{\mu}_r \rangle \geqslant 0, \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}\right] \tag{B.260}$$

$$\geqslant \frac{1}{2} \Pr\left[\langle \mathbf{w}_i^{(0)}, \hat{\mu}_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i \middle| \langle \mathbf{w}_i^{(0)}, \mu_r - \hat{\mu}_r \rangle \geqslant 0, \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}\right] \tag{B.261}$$

$$= \frac{1}{2} \Pr\left[\langle \mathbf{w}_i^{(0)}, \hat{\mu}_r \rangle \geqslant C_{\mathcal{S}ure,1} b_i\right] \tag{B.262}$$

$$\geqslant \Theta\left(\frac{\|\tilde{\mu}_r\|_2}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(9C_b^2 \tau \sigma_{B+}{}^2/(2\|\tilde{\mu}_r\|_2^2)\right)}}\right), \tag{B.263}$$

where the last equality following that $\hat{\mu}_r$ is orthogonal with $\mu_1, \ldots, \mu_{r_1}$ and the property of the standard Gaussian vector, and the last inequality follows Theorem B.75.

For the second condition in Theorem B.29, by Theorem B.75, we have,

$$p_1 = \Pr\left[\hat{\zeta}_1\right] = 1 - \Theta\left(\frac{\|\tilde{\mu}_1\|_2}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(C_b^2 \tau \sigma_{B+}{}^2/(8\|\tilde{\mu}_1\|_2^2)\right)}}\right) \tag{B.264}$$

$$p_2 = \Pr\left[\hat{\zeta}_2 \middle| \hat{\zeta}_1\right] \geqslant \Pr\left[\hat{\zeta}_2\right] \geqslant 1 - \Theta\left(\frac{\|\tilde{\mu}_2\|_2}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(C_b^2 \tau \sigma_{B+}{}^2/(8\|\tilde{\mu}_2\|_2^2)\right)}}\right) \tag{B.265}$$

$$\vdots \tag{B.266}$$

$$p_{r-1} = \Pr\left[\hat{\dot{\zeta}}_{r_2} \middle| \hat{\zeta}_1, \ldots, \hat{\zeta}_{r_1}, \hat{\dot{\zeta}}_{r_1+1}, \ldots, \hat{\dot{\zeta}}_{r_2}\right] \geqslant \Pr\left[\hat{\dot{\zeta}}_{r_2}\right] \geqslant \Pr\left[\hat{\zeta}_{r_2}\right] \tag{B.267}$$

$$\geqslant 1 - \Theta\left(\frac{\|\tilde{\mu}_{r-1}\|_2}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(C_b^2 \tau \sigma_{B+}{}^2/(8\|\tilde{\mu}_{r_2}\|_2^2)\right)}}\right). \tag{B.268}$$

On the other hand, if X is a $\chi^2(k)$ random variable. Then we have

$$\Pr(X \geqslant k + 2\sqrt{kx} + 2x) \leqslant e^{-x}. \tag{B.269}$$

Therefore, by assumption $B_{\mu 1} \geqslant \Omega\left(\sigma_{B+}\sqrt{\frac{\tau \log d}{d}}\right)$, we have

$$\Pr\left(\frac{1}{\sigma_{\mathbf{w}}^2}\left\|\mathbf{w}_i^{(0)}\right\|_2^2 \geqslant d + 2\sqrt{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)+2\right) d\log d}\right. \tag{B.270}$$

$$\left. + 2\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)+2\right)\log d\right) \tag{B.271}$$

$$\leqslant O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)\right)}}\right). \tag{B.272}$$

Recall $B_{\mu 1} = \min_{j \in [r]} \|\tilde{\mu}_j\|_2, B_{\mu 2} = \max_{j \in [r]} \|\tilde{\mu}_j\|_2$. Thus, by union bound, we have

$$\Pr\left[i \in S_{D_j,Sure}\right] \tag{B.273}$$

$$\geqslant \Pi_{l \in [r]} p_l - O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)\right)}}\right) \tag{B.274}$$

$$\geqslant \Theta\left(\frac{B_{\mu 1}}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)\right)}} \cdot \left(1 - \frac{rB_{\mu 2}}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(C_b^2\tau{\sigma_{B+}}^2/(8B_{\mu 2}^2)\right)}}\right)\right) \tag{B.275}$$

$$- O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)\right)}}\right) \tag{B.276}$$

$$\geqslant \Theta\left(\frac{B_{\mu 1}}{\sqrt{\tau \log d}\,\sigma_{B+} \cdot d^{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)\right)}}\right). \tag{B.277}$$

□

In Theorem B.32, we compute the activation pattern for the neurons in $S_{D_j,Sure}$.

**Lemma B.32** (Mixture of Gaussians: Activation Pattern)**.** *Assume the same conditions as in Theorem B.30, for all* $j \in [r], i \in S_{D_j,Sure}$*, we have*

*(1) When* $\mathbf{x} \sim \mathcal{N}_j(\mu_j, \sigma_j I_{d \times d})$*, the activation probability satisfies,*

$$\Pr_{\mathbf{x} \sim \mathcal{N}_j(\mu_j, \sigma_j I_{d \times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i \geqslant 0\right] \geqslant 1 - O\left(\frac{1}{d^\tau}\right). \tag{B.278}$$

(2) *For all* $j' \neq j, j' \in [r]$, *when* $\mathbf{x} \sim \mathcal{N}_{j'}(\mu_{j'}, \Sigma_{j'})$, *the activation probability satisfies,*

$$\Pr_{\mathbf{x}\sim\mathcal{N}_{j'}(\mu_{j'},\sigma_{j'} I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i \geqslant 0\right] \leqslant O\left(\frac{1}{d^{\tau}}\right). \tag{B.279}$$

*Proof of Theorem B.32.* In the proof, we need $\tilde{b} = C_b\sqrt{\tau d \log d}\sigma_{\mathbf{w}}\sigma_{B+}$, where $C_b$ is a large enough universal constant. For the first statement, when $\mathbf{x} \sim \mathcal{N}_j(\mu_j, \sigma_j I_{d\times d})$, by $C_{Sure,1} \geqslant \frac{3}{2}$, we have

$$\Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i \geqslant 0\right] \geqslant \Pr_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle \geqslant (1 - C_{Sure,1})b_i\right] \tag{B.280}$$

$$\geqslant \Pr_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle \geqslant -\frac{b_i}{2}\right] \tag{B.281}$$

$$=1 - \Pr_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle \leqslant -\frac{b_i}{2}\right] \tag{B.282}$$

$$\geqslant 1 - \exp\left(-\frac{{b_i}^2}{\Theta(d\sigma_{\mathbf{w}}^2\sigma_j^2)}\right) \tag{B.283}$$

$$\geqslant 1 - O\left(\frac{1}{d^{\tau}}\right), \tag{B.284}$$

where the third inequality follows the Chernoff bound and symmetricity of the Gaussian vector.

For the second statement, we prove similarly by $0 < C_{Sure,2} \leqslant \frac{1}{2}$. □

Then, Theorem B.33 gives gradients of neurons in $S_{D_j,Sure}$. It shows that these gradients are highly aligned with $D_j$.

**Lemma B.33** (Mixture of Gaussians: Feature Emergence)**.** *Assume the same conditions as in Theorem B.30, for all* $j \in [r]$, $i \in S_{D_j,Sure}$, *we have*

$$\langle \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right], y_{(j)}D_j\rangle \tag{B.285}$$

$$\geqslant p_j B_{\mu 1}\sqrt{d}\left(1 - O\left(\frac{1}{d^{\tau}}\right)\right) - B_{\mu 2}O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) - \sigma_{B+}O\left(\frac{1}{d^{0.9\tau}}\right). \tag{B.286}$$

*For any unit vector $D_j^\perp$ which is orthogonal with $D_j$, we have*

$$\left|\langle \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right], D_j^\perp\rangle\right| \leqslant B_{\mu 2} O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) + \sigma_{B+} O\left(\frac{1}{d^{0.9\tau}}\right). \tag{B.287}$$

*Proof of Theorem B.33.* For all $j \in [r]$, $i \in S_{D_j,Sure}$, we have

$$\begin{align*}
& \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right] \tag{B.288}\\
= & \sum_{l\in[r]} p_l \mathbb{E}_{\mathbf{x}\sim\mathcal{N}_l(\mathbf{x})}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right] \tag{B.289}\\
= & \sum_{l\in[r]} p_l y_{(l)} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I_{d\times d})}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_l\rangle - b_i\right)(\mathbf{x}+\mu_l)\right]. \tag{B.290}
\end{align*}$$

Thus, by Theorem B.76 and Theorem B.32,

$$\begin{align*}
& \langle \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right], y_{(j)}D_j\rangle \tag{B.291}\\
= & p_j \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I_{d\times d})}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_j\rangle - b_i\right)(\mathbf{x}+\mu_j)^\top D_j\right] \tag{B.292}\\
& + \sum_{l\in[r],l\neq j} p_l y_{(l)} y_{(j)} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I_{d\times d})}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_l\rangle - b_i\right)(\mathbf{x}+\mu_l)^\top D_j\right] \tag{B.293}\\
\geqslant & p_j \mu_j^\top D_j\left(1 - O\left(\frac{1}{d^\tau}\right)\right) - \sum_{l\in[r],l\neq j} p_l|\mu_l^\top D_j| O\left(\frac{1}{d^\tau}\right) \tag{B.294}\\
& - p_j\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I)}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_j\rangle - b_i\right)\mathbf{x}^\top D_j\right]\right| \tag{B.295}\\
& - \sum_{l\in[r],l\neq j} p_l\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_l\rangle - b_i\right)\mathbf{x}^\top D_j\right]\right| \tag{B.296}\\
\geqslant & p_j B_{\mu 1}\sqrt{d}\left(1 - O\left(\frac{1}{d^\tau}\right)\right) - B_{\mu 2} O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) \tag{B.297}\\
& - p_j\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I)}\left[\left(1 - \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_j\rangle - b_i\right) - 1\right)\mathbf{x}^\top D_j\right]\right| \tag{B.298}\\
& - \sum_{l\in[r],l\neq j} p_l\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_l\rangle - b_i\right)\mathbf{x}^\top D_j\right]\right| \tag{B.299}\\
= & p_j B_{\mu 1}\sqrt{d}\left(1 - O\left(\frac{1}{d^\tau}\right)\right) - B_{\mu 2} O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) \tag{B.300}
\end{align*}$$

$$
\begin{aligned}
&- p_j \left| \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I)} \left[ \left(1 - \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} + \mu_j \rangle - b_i\right)\right) \mathbf{x}^\top D_j \right] \right| && \text{(B.301)} \\
&- \sum_{l\in[r], l\neq j} p_l \left| \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)} \left[ \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} + \mu_l \rangle - b_i\right) \mathbf{x}^\top D_j \right] \right| && \text{(B.302)} \\
\geqslant & p_j B_{\mu 1}\sqrt{d}\left(1 - O\left(\frac{1}{d^\tau}\right)\right) - B_{\mu 2} O\left(\frac{1}{d^{\tau - \frac{1}{2}}}\right) - \sigma_{B+} O\left(\frac{1}{d^{0.9\tau}}\right). && \text{(B.303)}
\end{aligned}
$$

For any unit vector $D_j^\perp$ which is orthogonal with $D_j$, similarly, we have

$$
\begin{aligned}
&\left| \langle \mathbb{E}_{(\mathbf{x},y)} \left[ y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} \rangle - b_i\right) \mathbf{x} \right], D_j^\perp \rangle \right| && \text{(B.304)} \\
\leqslant & p_j \left| \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I)} \left[ \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} + \mu_j \rangle - b_i\right) \mathbf{x}^\top D_j^\perp \right] \right| && \text{(B.305)} \\
&+ \sum_{l\in[r], l\neq j} p_l \left| \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)} \left[ \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} + \mu_l \rangle - b_i\right) (\mathbf{x} + \mu_l)^\top D_j^\perp \right] \right| && \text{(B.306)} \\
\leqslant & B_{\mu 2} O\left(\frac{1}{d^{\tau - \frac{1}{2}}}\right) + p_j \left| \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_j I)} \left[ \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} + \mu_j \rangle - b_i\right) \mathbf{x}^\top D_j^\perp \right] \right| && \text{(B.307)} \\
&+ \sum_{l\in[r], l\neq j} p_l \left| \mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)} \left[ \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x} + \mu_l \rangle - b_i\right) \mathbf{x}^\top D_j^\perp \right] \right| && \text{(B.308)} \\
\leqslant & B_{\mu 2} O\left(\frac{1}{d^{\tau - \frac{1}{2}}}\right) + \sigma_{B+} O\left(\frac{1}{d^{0.9\tau}}\right). && \text{(B.309)}
\end{aligned}
$$

□

### Mixture of Gaussians: Final Guarantee

**Lemma B.34** (Mixture of Gaussians: Existence of Good Networks. Part statement of Theorem 3.17)**.** *Assume the same conditions as in Theorem B.30. Define*

$$
g^*(\mathbf{x}) = \sum_{j=1}^{r} \frac{y_{(j)}}{\sqrt{\tau \log d}\,\sigma_{B+}} \left[ \sigma\left(\langle D_j, \mathbf{x} \rangle - 2\sqrt{\tau \log d}\,\sigma_{B+}\right)\right]. \qquad \text{(B.310)}
$$

*For* $\mathcal{D}_{\mathrm{mixture}}$ *setting, we have* $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$*, where*

$$
B_F = (B_{a1}, B_{a2}, B_b) = \left( \frac{1}{\sqrt{\tau \log d}\,\sigma_{B+}}, \frac{\sqrt{r}}{\sqrt{\tau \log d}\,\sigma_{B+}}, 2\sqrt{\tau \log d}\,\sigma_{B+} \right),
$$

$p = \Theta\left(\frac{B_{\mu 1}}{\sqrt{\tau \log d}\sigma_{B+} \cdot d^{\left(9C_b^2 \tau \sigma_{B+}{}^2/(2B_{\mu 1}^2)\right)}}\right)$, $\gamma = \frac{1}{d^{0.9\tau - 1.5}}$, $B_G = p_B B_{\mu 1}\sqrt{d} - O\left(\frac{\sigma_{B+}}{d^{0.9\tau}}\right)$ *and* $B_{x1} = (B_{\mu 2} + \sigma_{B+})\sqrt{d}, B_{x2} = (B_{\mu 2} + \sigma_{B+})^2 d$. *We also have* $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} \leqslant \frac{3}{d^\tau} + \frac{4}{d^{0.9\tau-1}\sqrt{\tau \log d}}$.

*Proof of Theorem B.34.* We can check $B_{x1} = (B_{\mu 2} + \sigma_{B+})\sqrt{d}, B_{x2} = (B_{\mu 2} + \sigma_{B+})^2 d$ by direct calculation. By Theorem B.30, we have $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$.

For any $j \in [r]$, by $B_{\mu 1} \geqslant \Omega\left(\sigma_{B+}\sqrt{\frac{\tau \log d}{d}}\right) \geqslant 4\sigma_{B+}\sqrt{\frac{\tau \log d}{d}}$, we have

$$\Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\langle D_j, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+} \geqslant \sqrt{\tau \log d}\sigma_{B+}\right] \tag{B.311}$$

$$= \Pr_{\mathbf{x}\sim\mathcal{N}_j(0,\sigma_j I_{d\times d})}\left[\langle D_j, \mathbf{x}\rangle + \|\mu_j\|_2 - 2\sqrt{\tau \log d}\sigma_{B+} \geqslant \sqrt{\tau \log d}\sigma_{B+}\right] \tag{B.312}$$

$$\geqslant \Pr_{\mathbf{x}\sim\mathcal{N}_j(0,\sigma_j I_{d\times d})}\left[\langle D_j, \mathbf{x}\rangle + \sqrt{d}B_{\mu 1} - 2\sqrt{\tau \log d}\sigma_{B+} \geqslant \sqrt{\tau \log d}\sigma_{B+}\right] \tag{B.313}$$

$$\geqslant \Pr_{\mathbf{x}\sim\mathcal{N}_j(0,\sigma_j I_{d\times d})}\left[\langle D_j, \mathbf{x}\rangle \geqslant -\sqrt{\tau \log d}\sigma_{B+}\right] \tag{B.314}$$

$$\geqslant 1 - \frac{1}{d^\tau}, \tag{B.315}$$

where the last inequality follows Chernoff bound.

For any $l \neq j, l \in [r]$, by $\theta \leqslant \frac{\sigma_{B+}}{B_{\mu 2}}\sqrt{\frac{\tau \log d}{d}}$, we have

$$\Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\langle D_l, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+} \geqslant 0\right] \tag{B.316}$$

$$\leqslant \Pr_{\mathbf{x}\sim\mathcal{N}_j(0,\sigma_j I_{d\times d})}\left[\langle D_l, \mathbf{x}\rangle + \theta B_{\mu 2}\sqrt{d} - 2\sqrt{\tau \log d}\sigma_{B+} \geqslant 0\right] \tag{B.317}$$

$$\leqslant \Pr_{\mathbf{x}\sim\mathcal{N}_j(0,\sigma_j I_{d\times d})}\left[\langle D_l, \mathbf{x}\rangle \geqslant \sqrt{\tau \log d}\sigma_{B+}\right] \tag{B.318}$$

$$\leqslant \frac{1}{d^\tau}. \tag{B.319}$$

Thus, we have

$$\Pr_{(\mathbf{x},y)\sim\mathcal{D}_{mixture}}[y g^*(\mathbf{x}) > 1] \tag{B.320}$$

$$\geqslant \sum_{j\in[r]} p_j \left( \Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\langle D_j, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+} \geqslant \sqrt{\tau \log d}\sigma_{B+}\right]\right) \tag{B.321}$$

$$- \sum_{j\in[r]} p_j \left( \sum_{l\neq j, l\in[r]} \Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\langle D_l, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+} < 0\right]\right) \tag{B.322}$$

$$\geqslant 1 - \frac{2}{d^\tau}. \tag{B.323}$$

We also have

$$\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{mixture}}[\mathbb{I}[yg^*(\mathbf{x}) \leqslant 1]|yg^*(\mathbf{x})|] \tag{B.324}$$

$$\leqslant \sum_{j\in[r]} p_j \left( \Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\langle D_j, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+} < \sqrt{\tau \log d}\sigma_{B+}\right] \frac{y_{(j)}^2\sqrt{\tau \log d}\sigma_{B+}}{\sqrt{\tau \log d}\sigma_{B+}}\right) \tag{B.325}$$

$$+ \sum_{j\in[r]} p_j \left( \sum_{l\neq j, l\in[r]} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_j I_{d\times d})}\left[\sigma'\left[\langle D_l, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+} > 0\right] \frac{\langle D_l, \mathbf{x}\rangle - 2\sqrt{\tau \log d}\sigma_{B+}}{\sqrt{\tau \log d}\sigma_{B+}}\right]\right) \tag{B.326}$$

$$\leqslant \frac{1}{d^\tau} + \sum_{j\in[r]} p_j \left( \sum_{l\neq j, l\in[r]} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}_j(0,\sigma_j I_{d\times d})}\left[\sigma'\left[\langle D_l, \mathbf{x}\rangle > \sqrt{\tau \log d}\sigma_{B+}\right] \frac{\langle D_l, \mathbf{x}\rangle - \sqrt{\tau \log d}\sigma_{B+}}{\sqrt{\tau \log d}\sigma_{B+}}\right]\right) \tag{B.327}$$

$$\leqslant \frac{1}{d^\tau} + \frac{1}{\sqrt{\tau \log d}} \sum_{j\in[r]} p_j \left( \sum_{l\neq j, l\in[r]} \mathbb{E}_{\mathbf{x}\sim\mathcal{N}_j(0,I_{d\times d})}\left[\sigma'\left[\langle D_l, \mathbf{x}\rangle > \sqrt{\tau \log d}\right] \langle D_l, \mathbf{x}\rangle\right]\right) \tag{B.328}$$

$$\leqslant \frac{1}{d^\tau} + \frac{4}{d^{0.9\tau-1}\sqrt{\tau \log d}}, \tag{B.329}$$

where the second last inequality follows Theorem B.76 and $r \leqslant 2d$. Thus, we have

$$\mathrm{OPT}_{d,r,B_F,\mathcal{S}_{p,\gamma,B_G}} \tag{B.330}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{mixture}}[\ell(yg^*(\mathbf{x}))] \tag{B.331}$$

$$
\begin{aligned}
&= \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\mathtt{mixture}}}[\mathbb{I}[yg^*(\mathbf{x}) \leqslant 1](1 - yg^*(\mathbf{x}))] && \text{(B.332)}\\
&\leqslant \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\mathtt{mixture}}}[\mathbb{I}[yg^*(\mathbf{x}) \leqslant 1]|yg^*(\mathbf{x})|] + \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\mathtt{mixture}}}[\mathbb{I}[yg^*(\mathbf{x}) \leqslant 1]] && \text{(B.333)}\\
&\leqslant \frac{3}{d^{\tau}} + \frac{4}{d^{0.9\tau-1}\sqrt{\tau\log d}}. && \text{(B.334)}
\end{aligned}
$$

□

**Theorem B.35** (Mixture of Gaussians: Main Result. Restatement of Theorem 3.18)**.** *For $\mathcal{D}_{\mathtt{mixture}}$ setting with Assumption B.28, for any $\delta \in (0,1)$ and for any $\epsilon \in (0,1)$ when*

$$
m = \mathrm{poly}\left(\frac{1}{\delta}, \frac{1}{\epsilon}, d^{\Theta\left(\tau{\sigma_{B+}}^2/B_{\mu 1}^2\right)}, r, \frac{1}{p_B}\right) \leqslant e^d, \quad T = \mathrm{poly}\left(m\right), \quad n = \mathrm{poly}\left(m\right) \tag{B.335}
$$

*trained by Algorithm 1 with hinge loss, with probability at least $1-\delta$ over the initialization, with proper hyper-parameters, there exists $t \in [T]$ such that*

$$
\Pr[\mathrm{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \frac{\sqrt{2}r}{d^{0.4\tau-0.8}} + \epsilon. \tag{B.336}
$$

*Proof of Theorem B.35.* Let $\tilde{b} = C_b\sqrt{\tau d\log d}\sigma_{\mathbf{w}}\sigma_{B+}$, where $C_b$ is a large enough universal constant.

By Theorem B.34, we have $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$, where $B_F = (B_{a1}, B_{a2}, B_b) = \left(\frac{1}{\sqrt{\tau\log d}\sigma_{B+}}, \frac{\sqrt{r}}{\sqrt{\tau\log d}\sigma_{B+}}, 2\sqrt{\tau\log d}\sigma_{B+}\right)$, $p = \Theta\left(\frac{B_{\mu 1}}{\sqrt{\tau\log d}\sigma_{B+}\cdot d^{\left(9C_b^2\tau{\sigma_{B+}}^2/(2B_{\mu 1}^2)\right)}}\right)$, $\gamma = \frac{1}{d^{0.9\tau-1.5}}$, $B_G = p_B B_{\mu 1}\sqrt{d} - O\left(\frac{\sigma_{B+}}{d^{0.9\tau}}\right)$ and $B_{x1} = (B_{\mu 2} + \sigma_{B+})\sqrt{d}$, $B_{x2} = (B_{\mu 2} + \sigma_{B+})^2 d$. We also have $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} \leqslant \frac{3}{d^{\tau}} + \frac{4}{d^{0.9\tau-1}\sqrt{\tau\log d}}$.

Adjust $\sigma_{\mathbf{w}}$ such that $\tilde{b} = C_b\sqrt{\tau d\log d}\sigma_{\mathbf{w}}\sigma_{B+} = \Theta\left(\frac{B_G^{\frac{1}{4}}B_{a2}B_b^{\frac{3}{4}}}{\sqrt{r}B_{a1}}\right)$. Injecting above parameters into Theorem 3.12, we have with probability at least $1-\delta$ over the initialization, with proper hyper-parameters, there exists $t \in [T]$ such that

$$
\Pr[\mathrm{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \tag{B.337}
$$

$$\leqslant \frac{3}{d^{\tau}} + \frac{4}{d^{0.9\tau-1}\sqrt{\tau \log d}} + \frac{\sqrt{2} r B_{\mu 2}}{d^{(0.9\tau-1.5)/2}\sqrt{\tau \log d}\,\sigma_{B+}} + O\left(\frac{r B_{a1} B_{x1} B_{x2}{}^{\frac{1}{4}}}{\sqrt{B_G} n^{\frac{1}{6}}}\right) + \epsilon/2 \tag{B.338}$$

$$\leqslant \frac{\sqrt{2} r}{d^{0.4\tau-0.8}} + \epsilon. \tag{B.339}$$

□

### B.5.3 Mixture of Gaussians - XOR

We consider a special Mixture of Gaussians distribution studied in Refinetti et al. (2021). Consider the same data distribution in Section B.5.2 and Theorem B.29 with the following assumptions.

**Assumption B.36** (Mixture of Gaussians in Refinetti et al. (2021))**.** *Assume four Gaussians cluster with XOR-like pattern, for any* $\tau > 0$*,*

- $r = 4$ *and* $p_1 = p_2 = p_3 = p_4 = \frac{1}{4}$.
- $\mu_1 = -\mu_2$, $\mu_3 = -\mu_4$ *and* $\|\mu_1\|_2 = \|\mu_2\|_2 = \|\mu_3\|_2 = \|\mu_4\|_2 = \sqrt{d}$ *and* $\langle \mu_1, \mu_3 \rangle = 0$.
- *For all* $j \in [4]$, $\Sigma_j = \sigma_B I_{d \times d}$ *and* $1 \leqslant \sigma_B \leqslant \sqrt{\frac{d}{\tau \log \log d}}$.
- $y_{(1)} = y_{(2)} = 1$ *and* $y_{(3)} = y_{(4)} = -1$.

We denote this data distribution as $\mathcal{D}_{\mathrm{mixture-xor}}$ setting.

#### Mixture of Gaussians - XOR: Feature Learning

**Lemma B.37** (Mixture of Gaussians in Refinetti et al. (2021): Gradient Feature Set)**.** *Let* $C_{\mathrm{Sure},1} = \frac{6}{5}$, $C_{\mathrm{Sure},2} = \frac{\sqrt{2}}{\sqrt{\tau \log \log d}}$, $\tilde{b} = \sqrt{\tau d \log \log d}\,\sigma_{\mathbf{w}} \sigma_B$ *and* $d$ *is large enough. For* $\mathcal{D}_{\mathrm{mixture-xor}}$ *setting, we have* $(D_j, +1) \in S_{p,\gamma,B_G}$ *for all* $j \in [4]$*, where*

$$p = \Theta\left(\frac{1}{\sqrt{\tau \log \log d}\,\sigma_B \cdot (\log d)^{\frac{18\tau\sigma_B^2}{25}}}\right), \quad \gamma = \frac{\sigma_B}{\sqrt{d}}, \tag{B.340}$$

$$B_G = \frac{\sqrt{d}}{4}\left(1 - O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right) - \sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right). \tag{B.341}$$

*Proof of Theorem B.37.* For all $j \in [r]$, by Theorem B.40, for all $i \in S_{D_j,Sure}$,

$$\begin{aligned}
&1 - \frac{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2} && \text{(B.342)}\\
\leqslant & 1 - \frac{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|}{\sqrt{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|^2 + \max_{D_j^\top D_j^\perp = 0, \|D_j^\perp\|_2 = 1}\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j^\perp\rangle\right|^2}} && \text{(B.343)}\\
\leqslant & 1 - \frac{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right|}{\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right| + \max_{D_j^\top D_j^\perp = 0, \|D_j^\perp\|_2 = 1}\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j^\perp\rangle\right|} && \text{(B.344)}\\
\leqslant & 1 - \frac{1}{1 + \frac{\sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right)}{\frac{1}{4}\sqrt{d}\left(1 - O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right) - \sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right)}} && \text{(B.345)}\\
\leqslant & \frac{\sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right)}{\frac{1}{4}\sqrt{d}\left(1 - O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right) - \sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right)} && \text{(B.346)}\\
< & \frac{\sigma_B}{\sqrt{d}} = \gamma. && \text{(B.347)}
\end{aligned}$$

Thus, we have $G(\mathbf{w}_i^{(0)}, b_i) \in \mathcal{C}_{D_j,\gamma}$ and $\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right| \leqslant \|G(\mathbf{w}_i^{(0)}, b_i)\|_2 \leqslant B_{x1}$, $\frac{b_i}{|b_i|} = +1$. Thus, by Theorem B.38, we have

$$\begin{aligned}
&\Pr_{\mathbf{w},b}\left[G(\mathbf{w}, b) \in \mathcal{C}_{D_j,\gamma} \text{ and } \|G(\mathbf{w}, b)\|_2 \geqslant B_G \text{ and } \frac{b}{|b|} = +1\right] && \text{(B.348)}\\
\geqslant & \Pr\left[i \in S_{D_j,Sure}\right] && \text{(B.349)}\\
\geqslant & p. && \text{(B.350)}
\end{aligned}$$

Thus, $(D_j, +1) \in S_{p,\gamma,B_G}$. We finish the proof. □

**Lemma B.38** (Mixture of Gaussians in Refinetti et al. (2021): Geometry at Initialization)**.** *Assume the same conditions as in Theorem B.37. Recall for all* $i \in [m]$*,* $\mathbf{w}_i^{(0)} \sim \mathcal{N}(0, \sigma_{\mathbf{w}}^2 I_{d \times d})$*, over the random initialization, we have for all* $i \in [m], j \in [4]$*,*

$$\Pr\left[i \in S_{D_j,Sure}\right] \geqslant \Theta\left(\frac{1}{\sqrt{\tau \log\log d}\sigma_B \cdot (\log d)^{\frac{18\tau\sigma_B^2}{25}}}\right). \tag{B.351}$$

*Proof of Theorem B.38.* WLOG, let $j = 1$. By Assumption B.36, for the first condition in Theorem B.29, we have,

$$\Pr\left[\langle \mathbf{w}_i^{(0)}, \mu_1 \rangle \geqslant C_{Sure,1} b_i\right] \geqslant \Theta\left(\frac{1}{\sqrt{\tau \log\log d}\sigma_B \cdot (\log d)^{\frac{18\tau\sigma_B^2}{25}}}\right), \tag{B.352}$$

where the the last inequality follows Theorem B.75.

For the second condition in Theorem B.29, by Theorem B.75, we have,

$$\Pr\left[\left|\langle \mathbf{w}_i^{(0)}, \mu_2 \rangle\right| \leqslant C_{Sure,2} b_i\right] \geqslant 1 - \frac{1}{2\sqrt{\pi}} \frac{1}{\sigma_B \cdot e^{\sigma_B^2}}, \tag{B.353}$$

On the other hand, if X is a $\chi^2(k)$ random variable. Then we have

$$\Pr(X \geqslant k + 2\sqrt{kx} + 2x) \leqslant e^{-x}. \tag{B.354}$$

Therefore, we have

$$\Pr\left(\frac{1}{\sigma_{\mathbf{w}}^2}\left\|\mathbf{w}_i^{(0)}\right\|_2^2 \geqslant d + 2\sqrt{\left(\frac{18\tau\sigma_B^2}{25} + 2\right) d\log\log d} + 2\left(\frac{18\tau\sigma_B^2}{25} + 2\right)\log\log d\right) \tag{B.355}$$

$$\leqslant O\left(\frac{1}{(\log d)^2 \cdot (\log d)^{\frac{18\tau\sigma_B^2}{25}}}\right). \tag{B.356}$$

Thus, by union bound, we have

$$\Pr\left[i \in S_{D_j,\text{Sure}}\right] \geqslant \Theta\left(\frac{1}{\sqrt{\tau \log\log d}\sigma_B \cdot (\log d)^{\frac{18\tau\sigma_B^2}{25}}}\right). \tag{B.357}$$

□

**Lemma B.39** (Mixture of Gaussians in Refinetti et al. (2021): Activation Pattern). *Assume the same conditions as in Theorem B.37, for all* $j \in [4], i \in S_{D_j,\text{Sure}}$*, we have*

(1) *When* $\mathbf{x} \sim \mathcal{N}_j(\mu_j, \sigma_B I_{d\times d})$*, the activation probability satisfies,*

$$\Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_B I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i \geqslant 0\right] \geqslant 1 - \frac{1}{(\log d)^{\frac{\tau}{50}}}. \tag{B.358}$$

(2) *For all* $j' \neq j, j' \in [4]$*, when* $\mathbf{x} \sim \mathcal{N}_{j'}(\mu_{j'}, \sigma_B I_{d\times d})$*, the activation probability satisfies,*

$$\Pr_{\mathbf{x}\sim\mathcal{N}_{j'}(\mu_{j'},\sigma_B I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i \geqslant 0\right] \leqslant O\left(\frac{1}{(\log d)^{\frac{\tau}{2}}}\right). \tag{B.359}$$

*Proof of Theorem B.39.* In the proof, we need $\tilde{b} = \sqrt{\tau d \log\log d}\sigma_{\mathbf{w}}\sigma_B$. For the first statement, when $\mathbf{x} \sim \mathcal{N}_j(\mu_j, \sigma_B I_{d\times d})$, by $C_{\text{Sure},1} \geqslant \frac{6}{5}$, we have

$$\Pr_{\mathbf{x}\sim\mathcal{N}_j(\mu_j,\sigma_B I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i \geqslant 0\right] \geqslant \Pr_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle \geqslant (1 - C_{\text{Sure},1})b_i\right] \tag{B.360}$$

$$\geqslant \Pr_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle \geqslant -\frac{b_i}{5}\right] \tag{B.361}$$

$$=1 - \Pr_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I_{d\times d})}\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle \leqslant -\frac{b_i}{5}\right] \tag{B.362}$$

$$\geqslant 1 - \exp\left(-\frac{{b_i}^2}{50 d\sigma_{\mathbf{w}}^2\sigma_B^2}\right) \tag{B.363}$$

$$\geqslant 1 - \frac{1}{(\log d)^{\frac{\tau}{50}}}, \tag{B.364}$$

where the third inequality follows the Chernoff bound and symmetricity of the Gaussian vector.

For the second statement, we prove similarly by $0 < C_{\mathrm{Sure},2} \leqslant \frac{\sqrt{2}}{\sqrt{\tau \log \log d}}$. □

Then, Theorem B.40 gives gradients of neurons in $S_{D_j,\mathrm{Sure}}$. It shows that these gradients are highly aligned with $D_j$.

**Lemma B.40** (Mixture of Gaussians in Refinetti et al. (2021): Feature Emergence)**.** *Assume the same conditions as in Theorem B.37, for all* $j \in [4]$, $i \in S_{D_j,\mathrm{Sure}}$, *we have*

$$\begin{aligned}
&\langle \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right], y_{(j)}D_j\rangle && \text{(B.365)}\\
\geqslant&\frac{1}{4}\sqrt{d}\left(1 - O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right) - \sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right). && \text{(B.366)}
\end{aligned}$$

*For any unit vector* $D_j^\perp$ *which is orthogonal with* $D_j$, *we have*

$$\left|\langle \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right], D_j^\perp\rangle\right| \leqslant \sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right). \quad \text{(B.367)}$$

*Proof of Theorem B.40.* For all $j \in [4]$, $i \in S_{D_j,\mathrm{Sure}}$, we have

$$\begin{aligned}
&\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right] && \text{(B.368)}\\
=&\sum_{l\in[4]}\frac{1}{4}\mathbb{E}_{\mathbf{x}\sim\mathcal{N}_l(\mathbf{x})}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right] && \text{(B.369)}\\
=&\sum_{l\in[4]}\frac{1}{4}y_{(l)}\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I_{d\times d})}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_l\rangle - b_i\right)(\mathbf{x}+\mu_l)\right]. && \text{(B.370)}
\end{aligned}$$

Thus, by Theorem B.76 and Theorem B.39,

$$\begin{aligned}
&\langle \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\mathbf{x}\right], y_{(j)}D_j\rangle && \text{(B.371)}\\
=&\frac{1}{4}\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I_{d\times d})}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_j\rangle - b_i\right)(\mathbf{x}+\mu_j)^\top D_j\right] && \text{(B.372)}\\
&+\sum_{l\in[4],l\neq j}\frac{1}{4}y_{(l)}y_{(j)}\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I_{d\times d})}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}+\mu_l\rangle - b_i\right)(\mathbf{x}+\mu_l)^\top D_j\right] && \text{(B.373)}
\end{aligned}$$

$$
\begin{aligned}
\geqslant & \frac{1}{4}\mu_j^\top D_j\left(1-O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right)-\sum_{l\in[4],l\neq j}\frac{1}{4}|\mu_l^\top D_j|O\left(\frac{1}{d^{\frac{\tau}{2}}}\right) && \text{(B.374)}\\
& -\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_j\rangle-b_i\right)\mathbf{x}^\top D_j\right]\right| && \text{(B.375)}\\
& -\sum_{l\in[4],l\neq j}\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_l\rangle-b_i\right)\mathbf{x}^\top D_j\right]\right| && \text{(B.376)}\\
\geqslant & \frac{1}{4}\sqrt{d}\left(1-O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right) && \text{(B.377)}\\
& -\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I)}\left[\left(1-\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_j\rangle-b_i\right)-1\right)\mathbf{x}^\top D_j\right]\right| && \text{(B.378)}\\
& -\sum_{l\in[4],l\neq j}\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_l\rangle-b_i\right)\mathbf{x}^\top D_j\right]\right| && \text{(B.379)}\\
= & \frac{1}{4}\sqrt{d}\left(1-O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right)- && \text{(B.380)}\\
& \frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I)}\left[\left(1-\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_j\rangle-b_i\right)\right)\mathbf{x}^\top D_j\right]\right| && \text{(B.381)}\\
& -\sum_{l\in[4],l\neq j}\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_l\rangle-b_i\right)\mathbf{x}^\top D_j\right]\right| && \text{(B.382)}\\
\geqslant & \frac{1}{4}\sqrt{d}\left(1-O\left(\frac{1}{(\log d)^{\frac{\tau}{50}}}\right)\right)-\sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right). && \text{(B.383)}
\end{aligned}
$$

For any unit vector $D_j^\perp$ which is orthogonal with $D_j$, similarly, we have

$$
\begin{aligned}
& \left|\langle\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}\rangle-b_i\right)\mathbf{x}\right],D_j^\perp\rangle\right| && \text{(B.384)}\\
\leqslant & \frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_j\rangle-b_i\right)\mathbf{x}^\top D_j^\perp\right]\right| && \text{(B.385)}\\
& +\sum_{l\in[4],l\neq j}\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_l\rangle-b_i\right)(\mathbf{x}+\mu_l)^\top D_j^\perp\right]\right| && \text{(B.386)}\\
\leqslant & \frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_B I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_j\rangle-b_i\right)\mathbf{x}^\top D_j^\perp\right]\right| && \text{(B.387)}\\
& +\sum_{l\in[4],l\neq j}\frac{1}{4}\left|\mathbb{E}_{\mathbf{x}\sim\mathcal{N}(0,\sigma_l I)}\left[\sigma'\left(\langle\mathbf{w}_i^{(0)},\mathbf{x}+\mu_l\rangle-b_i\right)\mathbf{x}^\top D_j^\perp\right]\right| && \text{(B.388)}
\end{aligned}
$$

$$\leqslant \sigma_B O\left(\frac{1}{(\log d)^{0.018\tau}}\right). \tag{B.389}$$

□

### Mixture of Gaussians - XOR: Final Guarantee

**Lemma B.41** (Mixture of Gaussians in Refinetti et al. (2021): Existence of Good Networks)**.** *Assume the same conditions as in Theorem B.37 and let* $\tau = 1$ *and when* $0 < \tilde{\tau} \leqslant O\left(\frac{d}{\sigma_B^2 \log d}\right)$*. Define*

$$g^*(\mathbf{x}) = \sum_{j=1}^{4} \frac{y_{(j)}}{\sqrt{\tilde{\tau}\log d}\sigma_B}\left[\sigma\left(\langle D_j, \mathbf{x}\rangle - 2\sqrt{\tilde{\tau}\log d}\sigma_B\right)\right]. \tag{B.390}$$

*For* $\mathcal{D}_{\mathrm{mixture-xor}}$ *setting, we have* $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$*, where* $B_F = (B_{a1}, B_{a2}, B_b) = \left(\frac{1}{\sqrt{\tilde{\tau}\log d}\sigma_B}, \frac{2}{\sqrt{\tilde{\tau}\log d}\sigma_B}, 2\sqrt{\tilde{\tau}\log d}\sigma_B\right)$, $p = \Omega\left(\frac{1}{\sigma_B \cdot (\log d)^{\sigma_B^2}}\right)$, $\gamma = \frac{\sigma_B}{\sqrt{d}}$, $B_G = \frac{1}{5}\sqrt{d}$ *and* $B_{x1} = (1+\sigma_B)\sqrt{d}$, $B_{x2} = (1+\sigma_B)^2 d$*. We also have* $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} \leqslant \frac{3}{d^{\tilde{\tau}}} + \frac{4}{d^{0.9\tilde{\tau}-1}\sqrt{\tilde{\tau}\log d}}$.

*Proof of Theorem B.41.* We finish the proof by following the proof of Theorem B.34 □

**Theorem B.42** (Mixture of Gaussians in Refinetti et al. (2021): Main Result)**.** *For* $\mathcal{D}_{\mathrm{mixture-xor}}$ *setting with Assumption B.36, when* $d$ *is large enough, for any* $\delta \in (0, 1)$ *and for any* $\epsilon \in (0, 1)$ *when*

$$m = \Omega\left(\sigma_B (\log d)^{\sigma_B^2}\left(\left(\log\left(\frac{1}{\delta}\right)\right)^2 + \frac{1}{\epsilon^4}\right) + \frac{1}{\sqrt{\delta}}\right) \leqslant e^d, \tag{B.391}$$

$$T = \mathrm{poly}(\sigma_B, 1/\epsilon, 1/\delta, \log d), \tag{B.392}$$

$$n = \tilde{\Omega}\left(\left(\frac{(m\sigma_B)^{\frac{3}{2}}(\log d)^{\frac{\sigma_B^2}{2}}}{\epsilon} + \sigma_B\sqrt{d}\right)^3\right), \tag{B.393}$$

*trained by Algorithm 1 with hinge loss, with probability at least* $1-\delta$ *over the initialization and training samples, with proper hyper-parameters, there exists* $t \in [T]$ *such that*

$$\Pr[\text{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant O\left(\sigma_B^{\frac{3}{2}}\left(\frac{1}{d^{\frac{1}{4}}}+\frac{1}{n^{\frac{1}{6}}}\right)\right)+\epsilon. \tag{B.394}$$

*Proof of Theorem B.42.* Let $\tilde{b} = \sqrt{d \log\log d}\,\sigma_{\mathbf{w}}\sigma_B$.

By Theorem B.41, let $\tau = 1$ and when $\tilde{\tau} = O\left(\frac{d}{\sigma_B^2 \log d}\right)$, we have $g^* \in \mathcal{F}_{d,r,B_F,S_{p,\gamma,B_G}}$, where

$$B_F = (B_{a1}, B_{a2}, B_b) = \left(\frac{1}{\sqrt{\tilde{\tau}\log d}\,\sigma_B}, \frac{2}{\sqrt{\tilde{\tau}\log d}\,\sigma_B}, 2\sqrt{\tilde{\tau}\log d}\,\sigma_B\right),$$

$p = \Omega\left(\frac{1}{\sigma_B \cdot (\log d)^{\sigma_B^2}}\right)$, $\gamma = \frac{\sigma_B}{\sqrt{d}}$, $B_G = \frac{1}{5}\sqrt{d}$ and $B_{x1} = (1+\sigma_B)\sqrt{d}$, $B_{x2} = (1+\sigma_B)^2 d$. We also have $\mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} \leqslant \frac{3}{d^{\tilde{\tau}}} + \frac{4}{d^{0.9\tilde{\tau}-1}\sqrt{\tilde{\tau}\log d}}$.

Adjust $\sigma_{\mathbf{w}}$ such that $\tilde{b} = \sqrt{d\log\log d}\,\sigma_{\mathbf{w}}\sigma_B = \Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{r B_{a1}}}\right)$. Injecting above parameters into Theorem 3.12, we have with probability at least $1-\delta$ over the initialization, with proper hyper-parameters, there exists $t \in [T]$ such that

$$\Pr[\text{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant O\left(\sigma_B^{\frac{3}{2}}\left(\frac{1}{d^{\frac{1}{4}}}+\frac{1}{n^{\frac{1}{6}}}\right)\right)+\epsilon. \tag{B.395}$$

□

## B.5.4 Parity Functions

We recap the problem setup in Section 3.4.2 for readers' convenience.

### Problem Setup

**Data Distributions.** Suppose $M \in \mathbb{R}^{d\times D}$ is an unknown dictionary with $D$ columns that can be regarded as patterns. For simplicity, assume $d = D$ and $M$ is orthonormal. Let $\phi \in \mathbb{R}^d$ be a hidden representation vector. Let $\mathbf{A} \subseteq [D]$ be a subset of size

$\mathrm{rk}$ corresponding to the class relevant patterns and $r$ is an odd number. Then the input is generated by $M\phi$, and some function on $\phi_{\mathbf{A}}$ generates the label. WLOG, let $\mathbf{A} = \{1, \dots, rk\}$, $\mathbf{A}^{\perp} = \{rk+1, \dots, d\}$. Also, we split $\mathbf{A}$ such that for all $j \in [r]$, $\mathbf{A}_j = \{(j-1)k+1, \dots, jk\}$. Then the input $\mathbf{x}$ and the class label $y$ are given by:

$$\mathbf{x} = M\phi, \; y = g^*(\phi_{\mathbf{A}}) = \operatorname{sign}\left(\sum_{j=1}^{r} \mathrm{XOR}(\phi_{\mathbf{A}_j})\right), \tag{B.396}$$

where $g^*$ is the ground-truth labeling function mapping from $\mathbb{R}^{rk}$ to $\mathcal{Y} = \{\pm 1\}$, $\phi_{\mathbf{A}}$ is the sub-vector of $\phi$ with indices in $\mathbf{A}$, and $\mathrm{XOR}(\phi_{\mathbf{A}_j}) = \prod_{l \in \mathbf{A}_j} \phi_l$ is the parity function.

We still need to specify the distribution of $\phi$, which determines the structure of the input distribution:

$$\mathbf{X} := (1 - 2rp_A)\mathbf{X}_U + \sum_{j \in [r]} p_A(\mathbf{X}_{j,+} + \mathbf{X}_{j,-}). \tag{B.397}$$

For all corresponding $\phi_{\mathbf{A}^{\perp}}$ in $\mathbf{X}$, we have $\forall l \in \mathbf{A}^{\perp}$, independently:

$$\phi_l = \begin{cases} +1, & \text{w.p. } p_o \\ -1, & \text{w.p. } p_o \\ 0, & \text{w.p. } 1 - 2p_o \end{cases}$$

where $p_o$ controls the signal noise ratio: if $p_o$ is large, then there are many nonzero entries in $\mathbf{A}^{\perp}$ which are noise interfering with the learning of the ground-truth labeling function on $\mathbf{A}$.

For corresponding $\phi_{\mathbf{A}}$, any $j \in [r]$, we have

- In $\mathbf{X}_{j,+}$, $\phi_{\mathbf{A}_j} = [+1, +1, \dots, +1]^{\top}$ and $\phi_{\mathbf{A} \setminus \mathbf{A}_j}$ only have zero elements.
- In $\mathbf{X}_{j,-}$, $\phi_{\mathbf{A}_j} = [-1, -1, \dots, -1]^{\top}$ and $\phi_{\mathbf{A} \setminus \mathbf{A}_j}$ only have zero elements.
- In $\mathbf{X}_U$, we have $\phi_{\mathbf{A}}$ draw from $\{+1, -1\}^{rk}$ uniformly.

We call this data distribution $\mathcal{D}_{parity}$.

**Assumption B.43** (Parity Functions. Recap of Assumption 3.19)**.** *Let* $8 \leqslant \tau \leqslant d$ *be a parameter that will control our final error guarantee. Assume* $k$ *is an odd number and:*

$$k \geqslant \Omega(\tau \log d), \quad d \geqslant rk + \Omega(\tau r \log d), \quad p_o = O\left(\frac{rk}{d - rk}\right), \quad p_A \geqslant \frac{1}{d}. \tag{B.398}$$

**Remark B.44.** *The assumptions require* $k, d,$ *and* $p_A$ *to be sufficiently large so as to provide enough large signals for learning. When* $p_o = \Theta(\frac{rk}{d-rk})$ *means that the signal noise ratio is constant: the expected norm of* $\phi_{\mathbf{A}}$ *and that of* $\phi_{\mathbf{A}^\perp}$ *are comparable.*

To apply our framework, again we only need to compute the parameters in the Gradient Feature set and the corresponding optimal approximation loss. To this end, we first define the gradient features: For all $j \in [r]$, let

$$D_j = \frac{\sum_{l \in \mathbf{A}_j} M_l}{\|\sum_{l \in \mathbf{A}_j} M_l\|_2}. \tag{B.399}$$

**Remark B.45.** *Our data distribution is symmetric, which means for any* $\phi \in \mathbb{R}^d$*:*

- $-y = g^*(-\phi_{\mathbf{A}})$ *and* $-x = M(-\phi)$*,*
- $\mathbb{P}(\phi) = \mathbb{P}(-\phi)$*,*
- $\mathbb{E}_{(\mathbf{x},y)}[y\mathbf{x}] = \mathbf{0}$.

Below, we define a sufficient condition that randomly initialized weights will fall in nice gradients set after the first gradient step update.

**Definition B.46** (Parity Functions: Subset of Nice Gradients Set)**.** *Recall* $\mathbf{w}_i^{(0)}$ *is the weight for the* $i$*-th neuron at initialization. For all* $j \in [r]$*, let* $S_{D_j,Sure} \subseteq [m]$ *be those neurons that satisfy*

- $\langle \mathbf{w}_i^{(0)}, D_j \rangle \geqslant \frac{C_{Sure,1}}{\sqrt{k}} b_i$*,*
- $\left|\langle \mathbf{w}_i^{(0)}, D_{j'} \rangle\right| \leqslant \frac{C_{Sure,2}}{\sqrt{k}} b_i$*, for all* $j' \neq j, j' \in [r]$*,*

- $\left\| P_{\mathbf{A}} \mathbf{w}_i^{(0)} \right\|_2 \leqslant \Theta(\sqrt{rk}\sigma_{\mathbf{w}})$,
- $\left\| P_{\mathbf{A}^\perp} \mathbf{w}_i^{(0)} \right\|_2 \leqslant \Theta(\sqrt{d - rk}\sigma_{\mathbf{w}})$,

*where* $P_{\mathbf{A}}, P_{\mathbf{A}^\perp}$ *are the projection operator on the space* $M_{\mathbf{A}}$ *and* $M_{\mathbf{A}^\perp}$.

**Parity Functions: Feature Learning**

We show the important Theorem B.47 first and defer other Lemmas after it.

**Lemma B.47** (Parity Functions: Gradient Feature Set. Part statement of Theorem 3.21)**.** *Let* $C_{\mathrm{Sure},1} = \frac{3}{2}$, $C_{\mathrm{Sure},2} = \frac{1}{2}$, $\tilde{b} = C_b\sqrt{\tau r k \log d}\sigma_{\mathbf{w}}$, *where* $C_b$ *is a large enough universal constant. For* $\mathcal{D}_{\mathrm{parity}}$ *setting, we have* $(D_j, +1), (D_j, -1) \in S_{p,\gamma,B_G}$ *for all* $j \in [r]$, *where*

$$p = \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\left(9C_b^2 \tau r/8\right)}}\right), \quad \gamma = \frac{1}{d^{\tau-2}}, \quad B_G = \sqrt{k}p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right). \tag{B.400}$$

*Proof of Theorem B.47.* Note that for all $l \in [d]$, we have $M_l^\top \mathbf{x} = \phi_l$. For all $j \in [r]$, by Theorem B.50, for all $i \in S_{D_j,\mathrm{Sure}}$, when $\gamma = \frac{1}{d^{\tau-2}}$,

$$\begin{aligned}
&\left|\langle G(\mathbf{w}_i^{(0)}, b_i), D_j\rangle\right| - (1-\gamma)\|G(\mathbf{w}_i^{(0)}, b_i)\|_2 && \text{(B.401)}\\
=&\left|\langle G(\mathbf{w}_i^{(0)}, b_i), \frac{\sum_{l\in\mathbf{A}_j} M_l}{\sqrt{k}}\rangle\right| - (1-\gamma)\|G(\mathbf{w}_i^{(0)}, b_i)\|_2 && \text{(B.402)}\\
\geqslant&\sqrt{k}p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right) - \left(1 - \frac{1}{d^{\tau-2}}\right)\sqrt{kp_A^2 + \sum_{l\in[d]} O\left(\frac{1}{d^\tau}\right)^2} && \text{(B.403)}\\
\geqslant&\sqrt{k}p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right) - \left(1 - \frac{1}{d^{\tau-2}}\right)\left(\sqrt{k}p_A + O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right)\right) && \text{(B.404)}\\
\geqslant&\frac{\sqrt{k}p_A}{d^{\tau-2}} - O\left(\frac{\sqrt{k}}{d^\tau}\right) - O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right) && \text{(B.405)}\\
>&0. && \text{(B.406)}
\end{aligned}$$

Thus, we have $G(\mathbf{w}_i^{(0)}, b_i) \in \mathcal{C}_{D_j,\gamma}$ and $\sqrt{k}p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right) \leqslant \|G(\mathbf{w}_i^{(0)}, b_i)\|_2 \leqslant \sqrt{k}p_A + O\left(\frac{1}{d^{\tau-\frac{1}{2}}}\right)$, $\frac{b_i}{|b_i|} = +1$. Thus, by Theorem B.48, we have

$$\Pr_{\mathbf{w},b}\left[G(\mathbf{w}, b) \in \mathcal{C}_{D_j,\gamma} \text{ and } \|G(\mathbf{w}, b)\|_2 \geqslant B_G \text{ and } \frac{b}{|b|} = +1\right] \tag{B.407}$$

$$\geqslant \Pr\left[i \in S_{D_j,Sure}\right] \tag{B.408}$$

$$\geqslant p. \tag{B.409}$$

Thus, $(D_j, +1) \in S_{p,\gamma,B_G}$. Since $\mathbb{E}_{(\mathbf{x},y)}[y\mathbf{x}] = \mathbf{0}$, by Theorem B.71 and considering $i \in [2m] \setminus [m]$, we have $(D_j, -1) \in S_{p,\gamma,B_G}$. We finish the proof. □

Below are Lemmas used in the proof of Theorem B.47. In Theorem B.48, we calculate p used in $S_{p,\gamma,B_G}$.

**Lemma B.48** (Parity Functions: Geometry at Initialization. Lemma B.2 in Allen-Zhu and Li (2022))**.** *Assume the same conditions as in Theorem B.47, recall for all* $i \in [m]$, $\mathbf{w}_i^{(0)} \sim \mathcal{N}(0, \sigma_\mathbf{w}^2 I_{d\times d})$, *over the random initialization, we have for all* $i \in [m], j \in [r]$,

$$\Pr\left[i \in S_{D_j,Sure}\right] \geqslant \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\left(9C_b^2\tau r/8\right)}}\right). \tag{B.410}$$

*Proof of Theorem B.48.* For every $i \in [m]$, $j, j' \in [r]$, $j \neq j'$, by Theorem B.75,

$$p_1 = \Pr\left[\langle \mathbf{w}_i^{(0)}, D_j\rangle \geqslant \frac{C_{Sure,1}}{\sqrt{k}} b_i\right] = \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\left(9C_b^2\tau r/8\right)}}\right) \tag{B.411}$$

$$p_2 = \Pr\left[\left|\langle \mathbf{w}_i^{(0)}, D_{j'}\rangle\right| \geqslant \frac{C_{Sure,2}}{\sqrt{k}} b_i\right] = \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\left(C_b^2\tau r/8\right)}}\right). \tag{B.412}$$

On the other hand, if X is a $\chi^2(k)$ random variable, by Theorem B.74, we have

$$\Pr(X \geqslant k + 2\sqrt{kx} + 2x) \leqslant e^{-x}. \tag{B.413}$$

Therefore, by assumption $rk \geqslant \Omega(\tau r \log d), d - rk \geqslant \Omega(\tau r \log d)$ , we have

$$\Pr\left(\frac{1}{\sigma_{\mathbf{w}}^2}\left\|P_A \mathbf{w}_i^{(0)}\right\|_2^2 \geqslant rk + 2\sqrt{(9C_b^2\tau r/8+2)rk\log d} + 2\left(9C_b^2\tau r/8+2\right)\log d\right) \tag{B.414}$$

$$\leqslant O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau r/8\right)}}\right), \tag{B.415}$$

$$\Pr\left(\frac{1}{\sigma_{\mathbf{w}}^2}\left\|P_A \mathbf{w}_i^{(0)}\right\|_2^2 \geqslant (d-rk) + 2\sqrt{(9C_b^2\tau r/8+2)(d-rk)\log d} + 2\left(9C_b^2\tau r/8+2\right)\log d\right) \tag{B.416}$$

$$\leqslant O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau r/8\right)}}\right). \tag{B.417}$$

Thus, by union bound, and $D_1, \ldots, D_r$ being orthogonal with each other, we have

$$\Pr\left[i \in S_{D_j,Sure}\right] \geqslant p_1(1-p_2)^{r-1} - O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau r/8\right)}}\right) \tag{B.418}$$

$$= \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\left(9C_b^2\tau r/8\right)}} \cdot \left(1 - \frac{r}{\sqrt{\tau r \log d} \cdot d^{\left(C_b^2\tau r/8\right)}}\right)\right) \tag{B.419}$$

$$- O\left(\frac{1}{d^2 \cdot d^{\left(9C_b^2\tau r/8\right)}}\right) \tag{B.420}$$

$$= \Theta\left(\frac{1}{\sqrt{\tau r \log d} \cdot d^{\left(9C_b^2\tau r/8\right)}}\right). \tag{B.421}$$

□

In Theorem B.49, we compute the activation pattern for the neurons in $S_{D_j,Sure}$.

**Lemma B.49** (Parity Functions: Activation Pattern)**.** *Assume the same conditions as in Theorem B.47, for all* $j \in [r], i \in S_{D_j,Sure}$*, we have*

*(1) When* $\mathbf{x} \sim \mathbf{X}$*, we have*

$$\Pr_{\mathbf{x}\sim\mathbf{X}}\left[\left|\sum_{l\in A^{\perp}}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle\right| \geqslant t\right] \leqslant \exp\left(-\frac{t^2}{\Theta\left(rk\sigma_{\mathbf{w}}^2\right)}\right). \tag{B.422}$$

*(2) When* $\mathbf{x} \sim \mathbf{X}_U$*, we have*

$$\Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\left|\sum_{l\in A}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle\right| \geqslant t\right] \leqslant \exp\left(-\frac{t^2}{\Theta(rk\sigma_{\mathbf{w}}^2)}\right). \tag{B.423}$$

*(3) When* $\mathbf{x} \sim \mathbf{X}_U$*, the activation probability satisfies,*

$$\Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\sum_{l\in[d]}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle - b_i \geqslant 0\right] \leqslant O\left(\frac{1}{d^{\tau}}\right). \tag{B.424}$$

*(4) When* $\mathbf{x} \sim \mathbf{X}_{j,+}$*, the activation probability satisfies,*

$$\Pr_{\mathbf{x}\sim\mathbf{X}_{j,+}}\left[\sum_{l\in[d]}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle - b_i \geqslant 0\right] \geqslant 1 - O\left(\frac{1}{d^{\tau}}\right). \tag{B.425}$$

*(5) For all* $j' \neq j, j' \in [r]$*,* $s \in \{+,-\}$*, when* $\mathbf{x} \sim \mathbf{X}_{j',s}$*, or* $\mathbf{x} \sim \mathbf{X}_{j,-}$*, the activation probability satisfies,*

$$\Pr\left[\sum_{l\in[d]}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle - b_i \geqslant 0\right] \leqslant O\left(\frac{1}{d^{\tau}}\right). \tag{B.426}$$

*Proof of Theorem B.49.* For the first statement, when $\mathbf{x} \sim \mathbf{X}$, note that $\langle \mathbf{w}_i^{(0)}, M_l\rangle\phi_l$ is a mean-zero sub-Gaussian random variable with sub-Gaussion norm $\Theta\left(\left|\langle \mathbf{w}_i^{(0)}, M_l\rangle\right|\sqrt{p_o}\right)$.

$$\Pr_{\mathbf{x}\sim\mathbf{X}}\left[\left|\sum_{l\in\mathbf{A}^{\perp}}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle\right| \geqslant t\right] = \Pr_{\mathbf{x}\sim\mathbf{X}}\left[\left|\sum_{l\in\mathbf{A}^{\perp}}\langle \mathbf{w}_i^{(0)}, M_l\rangle\phi_l\right| \geqslant t\right] \tag{B.427}$$

$$\leqslant \exp\left(-\frac{t^2}{\sum_{l\in\mathbf{A}^\perp}\Theta\left(\langle \mathbf{w}_i^{(0)}, M_l\rangle^2 p_o\right)}\right) \tag{B.428}$$

$$\leqslant \exp\left(-\frac{t^2}{\Theta\left((d-rk)\sigma_{\mathbf{w}}^2 p_o\right)}\right) \tag{B.429}$$

$$\leqslant \exp\left(-\frac{t^2}{\Theta(rk\sigma_{\mathbf{w}}^2)}\right), \tag{B.430}$$

where the inequality follows general Hoeffding's inequality.

For the second statement, when $\mathbf{x} \sim \mathbf{X}_U$, by Hoeffding's inequality,

$$\Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\left|\sum_{l\in\mathbf{A}}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle\right| \geqslant t\right] = \Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\left|\sum_{l\in\mathbf{A}}\langle \mathbf{w}_i^{(0)}, M_l\rangle\phi_l\right| \geqslant t\right] \tag{B.431}$$

$$\leqslant 2\exp\left(-\frac{t^2}{2\sum_{l\in\mathbf{A}}\langle \mathbf{w}_i^{(0)}, M_l\rangle^2}\right) \tag{B.432}$$

$$\leqslant \exp\left(-\frac{t^2}{\Theta(rk\sigma_{\mathbf{w}}^2)}\right). \tag{B.433}$$

In the proof of the third to the last statement, we need $\tilde{b} = C_b\sqrt{\tau rk\log d}\sigma_{\mathbf{w}}$, where $C_b$ is a large enough universal constant.

For the third statement, when $\mathbf{x} \sim \mathbf{X}_U$, by union bound and previous statements,

$$\Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\sum_{l\in[d]}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle - b_i \geqslant 0\right] \tag{B.434}$$

$$\leqslant \Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\sum_{l\in\mathbf{A}}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle \geqslant \frac{b_i}{2}\right] + \Pr_{\mathbf{x}\sim\mathbf{X}_U}\left[\sum_{l\in\mathbf{A}^\perp}\langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle \geqslant \frac{b_i}{2}\right] \tag{B.435}$$

$$\leqslant O\left(\frac{1}{d^\tau}\right). \tag{B.436}$$

For the forth statement, when $\mathbf{x} \sim \mathbf{X}_{j,+}$, by $C_{\text{Sure},1} \geqslant \frac{3}{2}$ and previous statements,

$$\Pr_{\mathbf{x}\sim\mathbf{X}_{j,+}}\left[\sum_{l\in[d]} \langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle - b_i \geqslant 0\right] \tag{B.437}$$

$$= \Pr_{\mathbf{x}\sim\mathbf{X}_{j,+}}\left[\sum_{l\in\mathbf{A}_j} \langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle + \sum_{l\in A\setminus\mathbf{A}_j} \langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle + \sum_{l\in\mathbf{A}^\perp} \langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle \geqslant b_i\right] \tag{B.438}$$

$$\geqslant \Pr_{\mathbf{x}\sim\mathbf{X}_{j,+}}\left[\sum_{l\in\mathbf{A}^\perp} \langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle \geqslant (1 - C_{\text{Sure},1})b_i\right] \tag{B.439}$$

$$\geqslant \Pr_{\mathbf{x}\sim\mathbf{X}_{j,+}}\left[\sum_{l\in\mathbf{A}^\perp} \langle \mathbf{w}_i^{(0)}, M_l\phi_l\rangle \geqslant -\frac{b_i}{2}\right] \tag{B.440}$$

$$\geqslant 1 - O\left(\frac{1}{d^\tau}\right). \tag{B.441}$$

For the last statement, we prove similarly by $0 < C_{\text{Sure},2} \leqslant \frac{1}{2}$. □

Then, Theorem B.50 gives gradients of neurons in $S_{D_j,\text{Sure}}$. It shows that these gradients are highly aligned with $D_j$.

**Lemma B.50** (Parity Functions: Feature Emergence)**.** *Assume the same conditions as in Theorem B.47, for all $j \in [r]$, $i \in S_{D_j,\text{Sure}}$, we have the following holds:*

(1) *For all $l \in \mathbf{A}_j$, we have*

$$p_A - O\left(\frac{1}{d^\tau}\right) \leqslant \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] \leqslant p_A + O\left(\frac{1}{d^\tau}\right). \tag{B.442}$$

(2) *For all $l \in \mathbf{A}_{j'}$, any $j' \neq j, j' \in [r]$, we have*

$$\left|\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right| \leqslant O\left(\frac{1}{d^\tau}\right). \tag{B.443}$$

*(3) For all* $l \in \mathbf{A}^{\perp}$*, we have*

$$\left|\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right| \leqslant O\left(\frac{1}{d^{\tau}}\right). \tag{B.444}$$

*Proof of Theorem B.50.* For all $l \in [d]$, we have

$$\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] \tag{B.445}$$

$$=p_A \sum_{l\in[r]}\left(\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{l,+}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] - \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{l,-}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right) \tag{B.446}$$

$$+(1-2rp_A)\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_U}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]. \tag{B.447}$$

For the first statement, for all $l \in \mathbf{A}_j$, by Theorem B.49 (3) and (4), we have

$$\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] \tag{B.448}$$

$$=p_A\left(\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,+}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right] + \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,-}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right]\right) \tag{B.449}$$

$$+(1-2rp_A)\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_U}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] \tag{B.450}$$

$$\geqslant p_A\left(1 - O\left(\frac{1}{d^{\tau}}\right)\right) - O\left(\frac{1}{d^{\tau}}\right) \tag{B.451}$$

$$\geqslant p_A - O\left(\frac{1}{d^{\tau}}\right), \tag{B.452}$$

and we also have

$$\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] \tag{B.453}$$

$$=p_A\left(\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,+}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right] + \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,-}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right]\right) \tag{B.454}$$

$$+(1-2rp_A)\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_U}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] \tag{B.455}$$

$$\leqslant p_A + O\left(\frac{1}{d^{\tau}}\right). \tag{B.456}$$

Similarly, for the second statement, for all $l \in \mathbf{A}_{j'}$, any $j' \neq j, j' \in [r]$, by Theorem B.49 (3) and (5), we have

$$\left|\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right| \tag{B.457}$$

$$\leqslant \left|p_A\left(\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j',+}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right] + \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j',-}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right]\right)\right| + O\left(\frac{1}{d^\tau}\right) \tag{B.458}$$

$$\leqslant O\left(\frac{1}{d^\tau}\right). \tag{B.459}$$

For the third statement, for all $l \in \mathbf{A}^\perp$, by Theorem B.49 (3), (4), (5), we have

$$\left|\mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right| \tag{B.460}$$

$$\leqslant p_A \sum_{l\in[r]} \left|\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{l,+}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] - \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{l,-}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right| \tag{B.461}$$

$$+ O\left(\frac{1}{d^\tau}\right) \tag{B.462}$$

$$\leqslant p_A \left|\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,+}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right] - \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,-}}\left[\sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\phi_l\right]\right| + O\left(\frac{1}{d^\tau}\right) \tag{B.463}$$

$$\leqslant p_A \left|\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,+}}\left[\left(1 - \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right)\phi_l\right] - \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,-}}\left[\left(1 - \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right)\phi_l\right]\right| \tag{B.464}$$

$$+ p_A \left|\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,+}}[\phi_l] - \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,-}}[\phi_l]\right| + O\left(\frac{1}{d^\tau}\right) \tag{B.465}$$

$$= p_A \left|\mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,+}}\left[\left(1 - \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right)\phi_l\right] - \mathbb{E}_{\mathbf{x}\sim \mathcal{X}_{j,-}}\left[\left(1 - \sigma'\left(\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right)\right)\phi_l\right]\right| \tag{B.466}$$

$$+ O\left(\frac{1}{d^\tau}\right) \tag{B.467}$$

$$\leqslant O\left(\frac{1}{d^\tau}\right), \tag{B.468}$$

where the second inequality follows $2rp_A \leqslant 1$ and the third inequality follows the triangle inequality. □

### Parity Functions: Final Guarantee

**Lemma B.51** (Parity Functions: Existence of Good Networks. Part statement of Theorem 3.21)**.** *Assume the same conditions as in Theorem B.47. Define*

$$g^*(\mathbf{x}) = \sum_{j=1}^{r} \sum_{i=0}^{k} (-1)^{i+1}\sqrt{k} \tag{B.469}$$

$$\cdot \left[\sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k-1}{\sqrt{k}}\right) - 2\sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k}{\sqrt{k}}\right) + \sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k+1}{\sqrt{k}}\right)\right]. \tag{B.470}$$

*For* $\mathcal{D}_{parity}$ *setting, we have* $g^* \in \mathcal{F}_{d,3r(k+1),B_F,S_{p,\gamma,B_G}}$*, where* $B_F = (B_{a1}, B_{a2}, B_b) = \left(2\sqrt{k}, 2\sqrt{rk(k+1)}, \frac{k+1}{\sqrt{k}}\right)$*,* $p = \Theta\left(\frac{1}{\sqrt{\tau r \log d \cdot d^{\left(9C_b^2 \tau r/8\right)}}}\right)$*,* $\gamma = \frac{1}{d^{\tau-2}}$*,* $B_G = \sqrt{k}p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right)$ *and* $B_{x1} = \sqrt{d}, B_{x2} = d$*. We also have* $\mathrm{OPT}_{d,3r(k+1),B_F,S_{p,\gamma,B_G}} = 0$.

*Proof of Theorem B.51.* We can check $B_{x1} = \sqrt{d}, B_{x2} = d$ by direct calculation. By Theorem B.47, we have $g^* \in \mathcal{F}_{d,3r(k+1),B_F,S_{p,\gamma,B_G}}$. We note that

$$\sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k-1}{\sqrt{k}}\right) - 2\sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k}{\sqrt{k}}\right) + \sigma\left(\langle D_j, \mathbf{x}\rangle - \frac{2i-k+1}{\sqrt{k}}\right) \tag{B.471}$$

is a bump function for $\langle D_j, \mathbf{x}\rangle$ at $\frac{2i-k}{\sqrt{k}}$. We can check that $yg^*(\mathbf{x}) \geqslant 1$. Thus, we have

$$\begin{aligned} \mathrm{OPT}_{d,3r(k+1),B_F,S_{p,\gamma,B_G}} \leqslant & \mathcal{L}_{\mathcal{D}_{parity}}(g^*) & \text{(B.472)} \\ = & \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{parity}} \mathcal{L}_{(\mathbf{x},y)}(g^*) & \text{(B.473)} \\ = & 0. & \text{(B.474)} \end{aligned}$$

□

**Theorem B.52** (Parity Functions: Main Result. Full statement of Theorem 3.22)**.** *For $\mathcal{D}_{\mathrm{parity}}$ setting with Assumption B.43, for any $\delta \in (0,1)$ and for any $\epsilon \in (0,1)$ when*

$$m = \mathrm{poly}\left(\frac{1}{\delta}, \frac{1}{\epsilon}, d^{\Theta(\tau r)}, k, \frac{1}{p_A}\right) \leqslant e^d, \quad T = \mathrm{poly}\,(m), \quad n = \mathrm{poly}\,(m) \tag{B.475}$$

*trained by Algorithm 1 with hinge loss, with probability at least $1-\delta$ over the initialization, with proper hyper-parameters, there exists $t \in [T]$ such that*

$$\Pr[\mathrm{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \frac{3r\sqrt{k}}{d^{(\tau-3)/2}} + \epsilon. \tag{B.476}$$

*Proof of Theorem B.52.* Let $\tilde{b} = C_b\sqrt{\tau r k \log d}\,\sigma_{\mathbf{w}}$, where $C_b$ is a large enough universal constant. By Theorem B.51, we have $g^* \in \mathcal{F}_{d,3r(k+1),B_F,S_{p,\gamma,B_G}}$, where $B_F = (B_{a1}, B_{a2}, B_b) = \left(2\sqrt{k}, 2\sqrt{rk(k+1)}, \frac{k+1}{\sqrt{k}}\right)$, $p = \Theta\left(\frac{1}{\sqrt{\tau r \log d}\cdot d^{\left(9C_b^2\tau r/8\right)}}\right)$, $\gamma = \frac{1}{d^{\tau-2}}$, $B_G = \sqrt{k}p_A - O\left(\frac{\sqrt{k}}{d^\tau}\right)$ and $B_{x1} = \sqrt{d}, B_{x2} = d$. We also have $\mathrm{OPT}_{d,3r(k+1),B_F,S_{p,\gamma,B_G}} = 0$.

Adjust $\sigma_{\mathbf{w}}$ such that $\tilde{b} = C_b\sqrt{\tau r k \log d}\,\sigma_{\mathbf{w}} = \Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{r B_{a1}}}\right)$. Injecting above parameters into Theorem 3.12, we have with probability at least $1-\delta$ over the initialization, with proper hyper-parameters, there exists $t \in [T]$ such that

$$\Pr[\mathrm{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \frac{2\sqrt{2}r\sqrt{k}}{d^{(\tau-3)/2}} + O\left(\frac{r B_{a1} B_{x1} B_{x2}^{\frac{1}{4}}}{\sqrt{B_G} n^{\frac{1}{6}}}\right) + \epsilon/2 \leqslant \frac{3r\sqrt{k}}{d^{(\tau-3)/2}} + \epsilon. \tag{B.477}$$

□

### B.5.5 Uniform Parity Functions

We consider the sparse parity problem over the uniform data distribution studied in Barak et al. (2022). We use the properties of the problem to prove the key lemma (i.e., the existence of good networks) in our framework and then derive the final guarantee from our theorem of the simple setting (Theorem 3.4).

Consider the same data distribution in Section B.5.4 and Theorem B.46 with the following assumptions.

**Assumption B.53** (Uniform Parity Functions)**.** *We follow the data distribution in Section B.5.4. Let* $r = 1, p_A = 0, p_o = \frac{1}{2}$, $M = I_{d\times d}$ *and* $d \geqslant 2k^2$*, and k is an even number.*

We denote this data distribution as $\mathcal{D}_{parity-uniform}$ setting.

To apply our framework, again we only need to compute the parameters in the Gradient Feature set and the corresponding optimal approximation loss. To this end, we first define the gradient features: let

$$D = \frac{\sum_{l\in\mathbf{A}} M_l}{\|\sum_{l\in\mathbf{A}} M_l\|_2}. \tag{B.478}$$

We follow the initialization and training dynamic in Barak et al. (2022).

**Initialization and Loss.** We use hinge loss and we have unbiased initialization, for all $i \in [m]$,

$$a_i^{(0)} \sim \text{Unif}(\{\pm 1\}), \mathbf{w}_i^{(0)} \sim \text{Unif}(\{\pm 1\}^d), b_i = \text{Unif}(\{-1+1/k, \dots, 1-1/k\}). \tag{B.479}$$

**Training Process.** We use the following one-step training algorithm for this specific data distribution.

**Algorithm 6** Network Training via Gradient Descent Barak et al. (2022). Special case of Algorithm 4

```
Initialize $(a^{(0)}, \mathbf{W}^{(0)}, b)$ as in Equation (3.9) and Equation (B.479); Sample $\mathcal{Z} \sim \mathcal{D}^n_{parity-uniform}$
$\mathbf{W}^{(1)} = \mathbf{W}^{(0)} - \eta^{(1)}(\nabla_{\mathbf{W}}\widetilde{\mathcal{L}}_{\mathcal{Z}}(f_{\Xi^{(0)}}) + \lambda^{(1)}\mathbf{W}^{(0)})$
$a^{(1)} = a^{(0)} - \eta^{(1)}(\nabla_a\widetilde{\mathcal{L}}_{\mathcal{Z}}(f_{\Xi^{(0)}}) + \lambda^{(1)}a^{(0)})$
for $t = 2$ to $T$ do
    $a^{(t)} = a^{(t-1)} - \eta^{(t)}\nabla_a\widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{\Xi^{(t-1)}})$
end for
```

Use the notation in Section 5.3 of O'Donnell (2014), for every $S \in [n]$, s.t. $|S| = k$, we define

$$\xi_k := \widehat{\mathrm{Maj}}(S) = (-1)^{\frac{k-1}{2}} \frac{\binom{\frac{d-1}{2}}{\frac{d-1}{2}}}{\binom{d-1}{k-1}} \cdot 2^{-(d-1)} \binom{d-1}{\frac{d-1}{2}}. \tag{B.480}$$

**Lemma B.54** (Uniform Parity Functions: Existence of Good Networks. Rephrase of Lemma 5 in Barak et al. (2022))**.** *For every $\epsilon, \delta \in (0, 1/2)$, denoting $\tau = \frac{|\xi_{k-1}|}{16k\sqrt{2d\log(32k^3d/\epsilon)}}$, let $\eta^{(1)} = \frac{1}{k|\xi_{k-1}|}$, $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$, $m \geqslant k \cdot 2^k \log(k/\delta)$, $n \geqslant \frac{2}{\tau^2}\log(4dm/\delta)$ and $d \geqslant \Omega\left(k^4 \log(kd/\epsilon)\right)$, w.p. at least $1 - 2\delta$ over the initialization and the training samples, there exists $\tilde{a} \in \mathbb{R}^m$ with $\|\tilde{a}\|_\infty \leqslant 8k$ and $\|\tilde{a}\|_2 \leqslant 8k\sqrt{k}$ such that $f_{(\tilde{a}, \mathbf{W}^{(1)}, b)}$ satisfies*

$$\mathcal{L}_{\mathcal{D}_{\text{parity-uniform}}}\left(f_{(\tilde{a}, \mathbf{W}^{(1)}, b)}\right) \leqslant \epsilon. \tag{B.481}$$

*Additionally, it holds that $\|\sigma({\mathbf{W}^{(1)}}^\top \mathbf{x} - b)\|_\infty \leqslant d + 1$.*

**Remark B.55.** *In Barak et al. (2022), they update the bias term in the first gradient step. However, if we check the proof carefully, we can see that the fixed bias still goes through all their analysis.*

### Uniform Parity Functions: Final Guarantee

Considering training by Algorithm 6, we have the following results.

**Theorem B.56** (Uniform Parity Functions: Main Result)**.** *Fix $\epsilon \in (0, 1/2)$ and let $m \geqslant \Omega\left(k \cdot 2^k \log(k/\epsilon)\right)$, $n \geqslant \Omega\left(k^{7/6} d\binom{d}{k-1}\log(kd/\epsilon)\log(dm/\epsilon) + \frac{k^3 m d^2}{\epsilon^2}\right)$, $d \geqslant \Omega\left(k^4 \log(kd/\epsilon)\right)$. Let $\eta^{(1)} = \frac{1}{k|\xi_{k-1}|}$, $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$, and $\eta = \eta^{(t)} = \Theta\left(\frac{1}{d^2 m}\right)$, for all $t \in \{2, 3, \ldots, T\}$. If $T \geqslant \Omega\left(\frac{k^3 m d^2}{\epsilon}\right)$, then training by Algorithm 6 with hinge loss, w.h.p. over the initialization and the training samples, there exists $t \in [T]$ such that*

$$\Pr[\mathrm{sign}(g_{\Xi^{(t)}})(\mathbf{x}) \neq y] \leqslant \mathcal{L}_{\mathcal{D}_{\text{parity-uniform}}} g_{(a^{(t)}, \mathbf{W}^{(1)}, b)} \leqslant \epsilon. \tag{B.482}$$

*Proof of Theorem B.56.* By Theorem B.54, w.h.p., we have for properly chosen hyperparameters,

$$\mathrm{OPT}_{\mathbf{W}^{(1)},b,B_{a2}} \leqslant \mathcal{L}_{\mathcal{D}_{parity-uniform}}\left(f_{(\tilde{a},\mathbf{W}^{(1)},b)}\right) \leqslant \frac{\epsilon}{3}. \tag{B.483}$$

We compute the L-smooth constant of $\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(f_{(a,\mathbf{W}^{(1)},b)}\right)$ to $a$.

$$\begin{aligned}
&\left\|\nabla_a \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a_1,\mathbf{W}^{(1)},b)}\right) - \nabla_a \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a_2,\mathbf{W}^{(1)},b)}\right)\right\|_2 && \text{(B.484)}\\
=&\left\|\frac{1}{n}\sum_{\mathbf{x}\in\mathcal{Z}}\left[\left(\ell'\left(yg_{(a_1,\mathbf{W}^{(1)},b)}(\mathbf{x})\right) - \ell'\left(yg_{(a_2,\mathbf{W}^{(1)},b)}(\mathbf{x})\right)\right)\sigma(\mathbf{W}^{(1)\top}\mathbf{x}-b)\right]\right\|_2 && \text{(B.485)}\\
\leqslant&\left\|\frac{1}{n}\sum_{\mathbf{x}\in\mathcal{Z}}\left[\left|g_{(a_1,\mathbf{W}^{(1)},b)}(\mathbf{x}) - g_{(a_2,\mathbf{W}^{(1)},b)}(\mathbf{x})\right|\sigma(\mathbf{W}^{(1)\top}\mathbf{x}-b)\right]\right\|_2 && \text{(B.486)}\\
\leqslant&\frac{1}{n}\sum_{\mathbf{x}\in\mathcal{Z}}\left[\|a_1-a_2\|_2\left\|\sigma(\mathbf{W}^{(1)\top}\mathbf{x}-b)\right\|_2^2\right]. && \text{(B.487)}
\end{aligned}$$

By the Theorem B.54, we have $\|\sigma({\mathbf{W}^{(1)}}^\top\mathbf{x}-b)\|_\infty \leqslant d+1$. Thus, we have,

$$\begin{aligned}
L &= O\left(\frac{1}{n}\sum_{\mathbf{x}\in\mathcal{Z}}\left\|\sigma(\mathbf{W}^{(1)\top}\mathbf{x}-b)\right\|_2^2\right) && \text{(B.488)}\\
&\leqslant O(d^2m). && \text{(B.489)}
\end{aligned}$$

This means that we can let $\eta = \Theta\left(\frac{1}{d^2m}\right)$ and we will get our convergence result. Note that we have $a^{(1)} = \mathbf{0}$ and $\|\tilde{a}\|_2 = O\left(k\sqrt{k}\right)$. So, if we choose $T \geqslant \Omega\left(\frac{k^3}{\epsilon\eta}\right)$, there exists $t \in [T]$ such that $\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a^{(t)},\mathbf{W}^{(1)},b)}\right) - \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(\tilde{a},\mathbf{W}^{(1)},b)}\right) \leqslant O\left(\frac{L\|a^{(1)}-\tilde{a}\|_2^2}{T}\right) \leqslant \epsilon/3$.

We also have $\sqrt{\frac{\|\tilde{a}\|_2^2(\|\mathbf{W}^{(1)}\|_F^2 B_x^2+\|b\|_2^2)}{n}} \leqslant \frac{\epsilon}{3}$. Then our theorem gets proved by Theorem 3.4. □

### B.5.6 Uniform Parity Functions: Alternative Analysis

It is also possible to unify Barak et al. (2022) into our general Gradient Feature Learning Framework by mildly modifying the framework.

In order to do that, we first need to use a different metric in the definition of gradient features.

#### Modified General Feature Learning Framework for Uniform Parity Functions

**Definition B.57** (Gradient Feature with Infinity Norm). *For a unit vector* $D \in \mathbb{R}^d$ *with* $\|D\|_2 = 1$*, and a* $\gamma_\infty \in (0, 1)$*, a direction neighborhood (cone)* $\mathcal{C}^\infty_{D,\gamma_\infty}$ *is defined as:* $\mathcal{C}^\infty_{D,\gamma_\infty} := \left\{\mathbf{w} \;\middle|\; \left\|\frac{\mathbf{w}}{\|\mathbf{w}\|} - D\right\|_\infty < \gamma_\infty\right\}$. *Let* $\mathbf{w} \in \mathbb{R}^d$, $b \in \mathbb{R}$ *be random variables drawn from some distribution* $\mathcal{W}, \mathcal{B}$. *A Gradient Feature set with parameters* $p, \gamma_\infty, B_G, B_{G1}$ *is defined as:*

$$S^\infty_{p,\gamma_\infty,B_G,B_{G1}}(\mathcal{W},\mathcal{B}) := \left\{(D,s) \;\middle|\; \Pr_{\mathbf{w},b}\left[G(\mathbf{w},b) \in \mathcal{C}^\infty_{D,\gamma_\infty}\,,\, B_{G1} \geqslant \|G(\mathbf{w},b)\|_2 \geqslant B_G\,,\, s = \frac{b}{|b|}\right] \geqslant p\right\}. \tag{B.490}$$

*When clear from context, write it as* $S^\infty_{p,\gamma_\infty,B_G,B_{G1}}$.

**Definition B.58** (Optimal Approximation via Gradient Features with Infinity Norm). *The Optimal Approximation network and loss using gradient feature induced networks* $\mathcal{F}_{d,r,B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}}$ *are defined as:*

$$g^* := \operatorname*{arg\,min}_{g\in\mathcal{F}_{d,r,B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}}} \mathcal{L}_\mathcal{D}(f), \tag{B.491}$$

$$\mathrm{OPT}_{d,r,B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}} := \min_{g\in\mathcal{F}_{d,r,B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}}} \mathcal{L}_\mathcal{D}(f). \tag{B.492}$$

We consider the data distribution in Section B.5.4 with Assumption B.53, i.e., $\mathcal{D}_{parity-uniform}$ in Section B.5.5. Note that with this dataset, we have $\|\mathbf{x}\|_\infty \leqslant$

$B_{x\infty} = 1$. We use the following unbiased initialization:

$$\begin{aligned}
&\text{for } i \in \{1, \dots, m\}: \quad a_i^{(0)} \sim \mathcal{N}(0, \sigma_a^2), \mathbf{w}_i^{(0)} \sim \{\pm 1\}^d, b_i = \tilde{b} \leqslant 1, \\
&\text{for } i \in \{m+1, \dots, 2m\}: \quad a_i^{(0)} = -a_{i-m}^{(0)}, \mathbf{w}_i^{(0)} = -\mathbf{w}_{i-m}^{(0)}, b_i = -b_{i-m}, \\
&\text{for } i \in \{2m+1, \dots, 4m\}: \quad a_i^{(0)} = -a_{i-2m}^{(0)}, \mathbf{w}_i^{(0)} = \mathbf{w}_{i-2m}^{(0)}, b_i = b_{i-2m} \qquad \text{(B.493)}
\end{aligned}$$

Let $\nabla_i$ denote the gradient of the i-th neuron $\nabla_{\mathbf{w}_i} \mathcal{L}_{\mathcal{D}}(g_{\Xi^{(0)}})$. Denote the subset of neurons with nice gradients approximating feature $(D, s)$ as:

$$G^{\infty}_{(D,s),Nice} := \left\{ i \in [2m] : s = \frac{b_i}{|b_i|}, \left\| \frac{\nabla_i}{\|\nabla_i\|} - D \right\|_{\infty} \leqslant \gamma_{\infty}, \left| a_i^{(0)} \right| B_{G1} \geqslant \|\nabla_i\|_2 \geqslant \left| a_i^{(0)} \right| B_G \right\}. \qquad \text{(B.494)}$$

**Lemma B.59** (Existence of Good Networks. Modified Version of Theorem 3.14 Under Uniform Parity Setting). *Let* $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$. *For any* $B_{\epsilon} \in (0, B_b)$, *let* $\sigma_a = \Theta\left( \frac{\tilde{b}}{-\ell'(0)\eta^{(1)} B_G B_{\epsilon}} \right)$ *and* $\delta = 2re^{-\sqrt{mp}} + \frac{1}{d^2}$. *Then, with probability at least* $1 - \delta$ *over the initialization, there exists* $\tilde{a}_i$*'s such that* $g_{(\tilde{a}, \mathbf{W}^{(1)}, \mathbf{b})}(\mathbf{x}) = \sum_{i=1}^{4m} \tilde{a}_i \sigma\left( \langle \mathbf{w}_i^{(1)}, \mathbf{x} \rangle - b_i \right)$ *satisfies*

$$\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a}, \mathbf{W}^{(1)}, \mathbf{b})}) \leqslant rB_{a1} \left( \frac{B_{x1} B_{G1} B_b}{\sqrt{mp} B_G B_{\epsilon}} + \sqrt{2\log(d) d \gamma_{\infty}} + B_{\epsilon} \right) + \mathrm{OPT}_{d,r,B_F,S^{\infty}_{p,\gamma_{\infty},B_G,B_{G1}}}, \qquad \text{(B.495)}$$

*and* $\|\tilde{a}\|_0 = O\left( r(mp)^{\frac{1}{2}} \right)$, $\|\tilde{a}\|_2 = O\left( \frac{B_{a2} B_b}{\tilde{b}(mp)^{\frac{1}{4}}} \right)$, $\|\tilde{a}\|_{\infty} = O\left( \frac{B_{a1} B_b}{\tilde{b}(mp)^{\frac{1}{2}}} \right)$.

*Proof of Theorem B.59.* Recall $g^*(\mathbf{x}) = \sum_{j=1}^{r} a_j^* \sigma(\langle \mathbf{w}_j^*, \mathbf{x} \rangle - b_j^*)$, where $f^* \in \mathcal{F}_{d,r,B_F,S^{\infty}_{p,\gamma_{\infty},B_G,B_{G1}}}$ is defined in Theorem B.58 and let $s_j^* = \frac{b_j^*}{|b_j^*|}$. By Theorem B.5, with probability at least $1 - \delta_1$, $\delta_1 = 2re^{-cmp}$, for all $j \in [r]$, we have $|G^{\infty}_{(\mathbf{w}_j^*, s_j^*),Nice}| \geqslant \frac{mp}{4}$. Then for all $i \in G^{\infty}_{(\mathbf{w}_j^*, s_j^*),Nice} \subseteq [2m]$, we have $-\ell'(0)\eta^{(1)} G(\mathbf{w}_i^{(0)}, b_i) \frac{b_j^*}{\tilde{b}}$ only depend on $\mathbf{w}_i^{(0)}$ and

$b_i$, which is independent of $a_i^{(0)}$. Given Theorem B.57, we have

$$-\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2 \frac{b_j^*}{\tilde{b}} \in \left[\ell'(0)\eta^{(1)}B_{x1}\frac{B_b}{\tilde{b}}, -\ell'(0)\eta^{(1)}B_{x1}\frac{B_b}{\tilde{b}}\right]. \tag{B.496}$$

We split $[r]$ into $\Gamma = \{j \in [r] : |b_j^*| < B_\epsilon\}$, $\Gamma_- = \{j \in [r] : b_j^* \leqslant -B_\epsilon\}$ and $\Gamma_+ = \{j \in [r] : b_j^* \geqslant B_\epsilon\}$. Let $\epsilon_a = \frac{B_{G1}B_b}{\sqrt{mp}B_G B_\epsilon}$. Then we know that for all $j \in \Gamma_+ \cup \Gamma_-$, for all $i \in G^\infty_{(\mathbf{w}_j^*, s_j^*),Nice}$, we have

$$\Pr_{a_i^{(0)} \sim \mathcal{N}(0,\sigma_a^2)}\left[\left|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{|b_j^*|}{\tilde{b}} - 1\right| \leqslant \epsilon_a\right] \tag{B.497}$$

$$= \Pr_{a_i^{(0)} \sim \mathcal{N}(0,\sigma_a^2)}\left[1 - \epsilon_a \leqslant -a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{|b_j^*|}{\tilde{b}} \leqslant 1 + \epsilon_a\right] \tag{B.498}$$

$$= \Pr_{g \sim \mathcal{N}(0,1)}\left[1 - \epsilon_a \leqslant g\Theta\left(\frac{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}{B_G B_\epsilon}\right) \leqslant 1 + \epsilon_a\right] \tag{B.499}$$

$$= \Pr_{g \sim \mathcal{N}(0,1)}\left[(1 - \epsilon_a)\Theta\left(\frac{B_G B_\epsilon}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}\right) \leqslant g \leqslant (1 + \epsilon_a)\Theta\left(\frac{B_G B_\epsilon}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}\right)\right] \tag{B.500}$$

$$=\Theta\left(\frac{\epsilon_a B_G B_\epsilon}{\|G(\mathbf{w}_i^{(0)}, b_i)\|_2|b_j^*|}\right) \tag{B.501}$$

$$\geqslant\Omega\left(\frac{\epsilon_a B_G B_\epsilon}{B_{G1}B_b}\right) \tag{B.502}$$

$$=\Omega\left(\frac{1}{\sqrt{mp}}\right). \tag{B.503}$$

Thus, with probability $\Omega\left(\frac{1}{\sqrt{mp}}\right)$ over $a_i^{(0)}$, we have

$$\left|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{|b_j^*|}{\tilde{b}} - 1\right| \leqslant \epsilon_a, \quad \left|a_i^{(0)}\right| = O\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)}B_G B_\epsilon}\right). \tag{B.504}$$

Similarly, for $j \in \Gamma$, for all $i \in G^\infty_{(\mathbf{w}_j^*, s_j^*),Nice}$, with probability $\Omega\left(\frac{1}{\sqrt{mp}}\right)$ over $a_i^{(0)}$, we have

$$\left|-a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{B_\epsilon}{\tilde{b}} - 1\right| \leqslant \epsilon_a, \quad \left|a_i^{(0)}\right| = O\left(\frac{\tilde{b}}{-\ell'(0)\eta^{(1)}B_G B_\epsilon}\right). \tag{B.505}$$

For all $j \in [r]$, let $\Lambda_j \subseteq G^{\infty}_{(\mathbf{w}_j^*, s_j^*), Nice}$ be the set of $i$'s such that condition Equation (B.504) or Equation (B.505) are satisfied. By Chernoff bound and union bound, with probability at least $1 - \delta_2$, $\delta_2 = re^{-\sqrt{mp}}$, for all $j \in [r]$ we have $|\Lambda_j| \geqslant \Omega(\sqrt{mp})$.

We have for $\forall j \in \Gamma_+ \cup \Gamma_-, \forall i \in \Lambda_j$,

$$\left|\frac{|b_j^*|}{\tilde{b}}\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle - \langle \mathbf{w}_j^*, \mathbf{x}\rangle\right| \tag{B.506}$$

$$\leqslant \left|\langle -a_i^{(0)}\ell'(0)\eta^{(1)}\|G(\mathbf{w}_i^{(0)}, b_i)\|_2\frac{|b_j^*|}{\tilde{b}}\frac{\mathbf{w}_i^{(1)}}{\|\mathbf{w}_i^{(1)}\|_2} - \frac{\mathbf{w}_i^{(1)}}{\|\mathbf{w}_i^{(1)}\|_2}, \mathbf{x}\rangle + \langle \frac{\mathbf{w}_i^{(1)}}{\|\mathbf{w}_i^{(1)}\|_2} - \mathbf{w}_j^*, \mathbf{x}\rangle\right| \tag{B.507}$$

$$\leqslant \epsilon_a\|\mathbf{x}\|_2 + \sqrt{2\log(d)d}\gamma_\infty. \tag{B.508}$$

With probability $1 - \frac{1}{d^2}$ by Hoeffding's inequality. Similarly, for $\forall j \in \Gamma, \forall i \in \Lambda_j$,

$$\left|\frac{B_\epsilon}{\tilde{b}}\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle - \langle \mathbf{w}_j^*, \mathbf{x}\rangle\right| \leqslant \epsilon_a\|\mathbf{x}\|_2 + \sqrt{2\log(d)d}\gamma_\infty. \tag{B.509}$$

If $i \in \Lambda_j, j \in \Gamma_+ \cup \Gamma_-$, set $\tilde{a}_i = a_j^* \frac{|b_j^*|}{|\Lambda_j|\tilde{b}}$, if $i \in \Lambda_j, j \in \Gamma$, set $\tilde{a}_i = a_j^* \frac{B_\epsilon}{|\Lambda_j|\tilde{b}}$, otherwise set $\tilde{a}_i = 0$, we have $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$, $\|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right)$, $\|\tilde{a}\|_\infty = O\left(\frac{B_{a1}B_b}{\tilde{b}(mp)^{\frac{1}{2}}}\right)$.

Finally, we have

$$\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a}, \mathbf{W}^{(1)}, \mathbf{b})}) \tag{B.510}$$

$$= \mathcal{L}_{\mathcal{D}}(g_{(\tilde{a}, \mathbf{W}^{(1)}, \mathbf{b})}) - \mathcal{L}_{\mathcal{D}}(g^*) + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.511}$$

$$\leqslant \mathbb{E}_{(\mathbf{x}, y)}\left[\left|g_{(\tilde{a}, \mathbf{W}^{(1)}, \mathbf{b})}(\mathbf{x}) - g^*(\mathbf{x})\right|\right] + \mathcal{L}_{\mathcal{D}}(g^*) \tag{B.512}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{i=1}^{m}\tilde{a}_i\sigma\left(\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle-\tilde{b}\right)+\sum_{i=m+1}^{2m}\tilde{a}_i\sigma\left(\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle+\tilde{b}\right)-\sum_{j=1}^{r}a_j^*\sigma(\langle \mathbf{w}_j^*,\mathbf{x}\rangle-b_j^*)\right|\right] \tag{B.513}$$

$$+\mathcal{L}_{\mathcal{D}}(g^*) \tag{B.514}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_+}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\sigma\left(\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle-\tilde{b}\right)-\sigma(\langle \mathbf{w}_j^*,\mathbf{x}\rangle-b_j^*)\right|\right|\right] \tag{B.515}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_-}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\sigma\left(\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle+\tilde{b}\right)-\sigma(\langle \mathbf{w}_j^*,\mathbf{x}\rangle-b_j^*)\right|\right|\right] \tag{B.516}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{B_\epsilon}{\tilde{b}}\sigma\left(\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle-\tilde{b}\right)-\sigma(\langle \mathbf{w}_j^*,\mathbf{x}\rangle-b_j^*)\right|\right|\right]+\mathcal{L}_{\mathcal{D}}(g^*) \tag{B.517}$$

$$\leqslant \mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_+}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle-\langle \mathbf{w}_j^*,\mathbf{x}\rangle\right|\right|\right] \tag{B.518}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma_-}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{|b_j^*|}{\tilde{b}}\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle-\langle \mathbf{w}_j^*,\mathbf{x}\rangle\right|\right|\right] \tag{B.519}$$

$$+\mathbb{E}_{(\mathbf{x},y)}\left[\left|\sum_{j\in\Gamma}\sum_{i\in\Lambda_j}a_j^*\frac{1}{|\Lambda_j|}\left|\frac{B_\epsilon}{\tilde{b}}\langle \mathbf{w}_i^{(1)},\mathbf{x}\rangle+B_\epsilon-\langle \mathbf{w}_j^*,\mathbf{x}\rangle\right|\right|\right]+\mathcal{L}_{\mathcal{D}}(g^*) \tag{B.520}$$

$$\leqslant r\|a^*\|_\infty(\epsilon_a\mathbb{E}_{(\mathbf{x},y)}\|\mathbf{x}\|_2+\sqrt{2\log(d)d}\gamma_\infty)+|\Gamma|\|a^*\|_\infty B_\epsilon+\mathcal{L}_{\mathcal{D}}(g^*) \tag{B.521}$$

$$\leqslant rB_{a1}(\epsilon_a B_{x1}+\sqrt{2\log(d)d}\gamma_\infty)+|\Gamma|B_{a1}B_\epsilon+\mathrm{OPT}_{d,r,B_F,S^\infty_{p,\gamma,B_G,B_{G1}}}. \tag{B.522}$$

We finish the proof by union bound and $\delta \geqslant \delta_1+\delta_2+\frac{1}{d^2}$. □

**Lemma B.60** (Empirical Gradient Concentration Bound for Single coordinate)**.** *For* $i \in [m]$, *when* $n \geqslant (\log(d))^6$, *with probability at least* $1-O\left(\exp\left(-n^{\frac{1}{3}}\right)\right)$ *over training* *samples, we have*

$$\left|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{\Xi})}{\partial \mathbf{w}_{i,j}} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_{\Xi})}{\partial \mathbf{w}_{i,j}}\right| \leqslant O\left(\frac{|a_i|B_{x\infty}}{n^{\frac{1}{3}}}\right), \quad \forall j \in [d]. \tag{B.523}$$

*Proof of Theorem B.60.* First, we define,

$$z_{i,j}^{(l)} = \ell'(y^{(l)} g_{\Xi}(\mathbf{x}^{(l)}))y^{(l)} \left[\sigma'\left(\langle \mathbf{w}_i, \mathbf{x}^{(l)}\rangle - b_i\right) \mathbf{x}_j^{(l)}\right] \tag{B.524}$$

$$- \mathbb{E}_{(\mathbf{x},y)} \left[\ell'(y g_{\Xi}(\mathbf{x}))y \left[\sigma'\left(\langle \mathbf{w}_i, \mathbf{x}\rangle - b_i\right)\right] \mathbf{x}_j\right]. \tag{B.525}$$

As $|\ell'(z)| \leqslant 1, |y| \leqslant 1, |\sigma'(z)| \leqslant 1$, we have $z_{i,j}^{(l)}$ is zero-mean random variable with $\left|z_{i,j}^{(l)}\right| \leqslant 2B_{x\infty}$ as well as $\mathbb{E}\left[\left|z_{i,j}^{(l)}\right|_2^2\right] \leqslant 4B_{x\infty}^2$. Then by Bernstein Inequality, for $0 < z < 2B_{x\infty}$, we have

$$\Pr\left(\left|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{\Xi})}{\partial \mathbf{w}_{i,j}} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_{\Xi})}{\partial \mathbf{w}_{i,j}}\right| \geqslant |a_i| z\right) = \Pr\left(\left|\frac{1}{n}\sum_{l\in[n]} z_{i,j}^{(l)}\right| \geqslant z\right) \tag{B.526}$$

$$\leqslant \exp\left(-n \cdot \frac{z^2}{8B_{x\infty}}\right). \tag{B.527}$$

Thus, for some $i \in [m]$, when $n \geqslant (\log(d))^6$, with probability at least $1 - O\left(\exp\Theta\left(-n^{\frac{1}{3}}\right)\right)$, from a union bound over $j \in [d]$, we have

$$\left|\frac{\partial \widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{\Xi})}{\partial \mathbf{w}_{i,j}} - \frac{\partial \mathcal{L}_{\mathcal{D}}(g_{\Xi})}{\partial \mathbf{w}_{i,j}}\right| \leqslant O\left(\frac{|a_i|B_{x\infty}}{n^{\frac{1}{3}}}\right) \tag{B.528}$$

for $\forall j \in [d]$. □

**Lemma B.61** (Existence of Good Networks under Empirical Risk. Modified version of Theorem B.19 Under Uniform Parity Setting)**.** *Suppose*

$$n > \Omega\left(\left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{1}{p} + \frac{B_{x\infty}}{B_G|\ell'(0)|} + \frac{B_{x\infty}}{B_{G1}|\ell'(0)|}\right)^3 + (\log(d))^6\right).$$

*Let* $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$*. For any* $B_\epsilon \in (0, B_b)$*, let* $\sigma_a = \Theta\left(\frac{\tilde{b}}{-|\ell'(0)|\eta^{(1)} B_G B_\epsilon}\right)$ *and* $\delta = 2re^{-\sqrt{\frac{mp}{2}}} + \frac{1}{d^2}$*. Then, with probability at least* $1 - \delta$ *over the initialization and training samples, there exists* $\tilde{a}_i$*'s such that* $g_{(\tilde{a}, \mathbf{W}^{(1)}, b)}(\mathbf{x}) = \sum_{i=1}^{4m} \tilde{a}_i \sigma\left(\langle \mathbf{w}_i^{(1)}, \mathbf{x}\rangle - b_i\right)$ *satisfies*

$$\begin{aligned}
&\mathcal{L}_{\mathcal{D}}(g_{(\tilde{a}, \mathbf{W}^{(1)}, b)}) && \text{(B.529)}\\
\leqslant & rB_{a1}\left(\frac{2B_{x1}B_{G1}B_b}{\sqrt{mp}B_G B_\epsilon} + \sqrt{2\log(d)d}\left(\gamma_\infty + O\left(\frac{B_{x\infty}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)\right) + B_\epsilon\right) && \text{(B.530)}\\
&+ \mathrm{OPT}_{d,r,B_F,S^\infty_{p,\gamma,B_G,B_{G1}}}, && \text{(B.531)}
\end{aligned}$$

*and* $\|\tilde{a}\|_0 = O\left(r(mp)^{\frac{1}{2}}\right)$, $\|\tilde{a}\|_2 = O\left(\frac{B_{a2}B_b}{\tilde{b}(mp)^{\frac{1}{4}}}\right)$, $\|\tilde{a}\|_\infty = O\left(\frac{B_{a1}B_b}{\tilde{b}(mp)^{\frac{1}{2}}}\right)$.

*Proof of Theorem B.61.* Denote $\rho = O\left(\exp\Theta\left(-n^{\frac{1}{3}}\right)\right)$ and $\beta = O\left(\frac{B_{x\infty}}{n^{\frac{1}{3}}}\right)$. Note that by symmetric initialization, we have $\ell'(yg_{\Xi^{(0)}}(\mathbf{x})) = |\ell'(0)|$ for any $\mathbf{x} \in \mathcal{X}$, so that, by Theorem B.60, we have $\left|\widetilde{G}(\mathbf{w}_i^{(0)}, b_i)_j - G(\mathbf{w}_i^{(0)}, b_i)_j\right| \leqslant \frac{\beta}{|\ell'(0)|}$ with probability at least $1 - \rho$. Thus, by union bound, we can see that

$$S^\infty_{p,\gamma_\infty,B_G,B_{G1}} \subseteq \widetilde{S}^\infty_{p-\rho,\gamma_\infty+\frac{\beta}{B_G|\ell'(0)|},B_G-\frac{\beta}{|\ell'(0)|},B_{G1}+\frac{\beta}{|\ell'(0)|}}.$$

Consequently, we have $\mathrm{OPT}_{d,r,B_F,\widetilde{S}^\infty_{p-\rho,\gamma_\infty+\frac{\beta}{B_G|\ell'(0)|},B_G-\frac{\beta}{|\ell'(0)|},B_{G1}+\frac{\beta}{|\ell'(0)|}}} \leqslant \mathrm{OPT}_{d,r,B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}}$. Exactly follow the proof in Theorem B.6 by replacing $S^\infty_{p,\gamma_\infty,B_G,B_{G1}}$ to

$$\widetilde{S}^\infty_{p-\rho,\gamma_\infty+\frac{\beta}{B_G|\ell'(0)|},B_G-\frac{\beta}{|\ell'(0)|},B_{G1}+\frac{\beta}{|\ell'(0)|}}.$$

Then, we finish the proof by $\rho \leqslant \frac{p}{2}$, $\frac{\beta}{|\ell'(0)|} \leqslant (1 - 1/\sqrt{2})B_G$, $\frac{\beta}{|\ell'(0)|} \leqslant (\sqrt{2} - 1)B_{G1}$. □

**Theorem B.62** (Online Convex Optimization under Empirical Risk. Modified version of Theorem B.23 Under Uniform Parity Setting )**.** *Consider training by Algorithm 1, and any* $\delta \in (0, 1)$*. Assume* $d \geqslant \log m, \delta \leqslant O(\frac{1}{d^2})$*. Set*

$$\sigma_{\mathbf{w}} > 0, \quad \tilde{b} > 0, \quad \eta^{(t)} = \eta, \ \lambda^{(t)} = 0 \textit{ for all } t \in \{2, 3, \ldots, T\}, \qquad \text{(B.532)}$$

$$\eta^{(1)} = \Theta\left(\frac{\min\{O(\eta), O(\eta\tilde{b})\}}{-\ell'(0)(B_{x1}\sigma_{\mathbf{w}}\sqrt{d}+\tilde{b})}\right),\ \lambda^{(1)} = \frac{1}{\eta^{(1)}},\quad \sigma_a = \Theta\left(\frac{\tilde{b}(mp)^{\frac{1}{4}}}{-\ell'(0)\eta^{(1)}B_{x1}\sqrt{B_G B_b}}\right). \tag{B.533}$$

*Let* $0 < T\eta B_{x1} \leqslant o(1)$, $m = \Omega\left(\frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right)$ *and*

$$n > \Omega\left(\left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G} + \frac{1}{B_{G1}})\frac{B_{x\infty}}{|\ell'(0)|}\right)^3\right).$$

*With probability at least* $1-\delta$ *over the initialization and training samples, there exists* $t \in [T]$ *such that*

$$\mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.534}$$

$$\leqslant \mathrm{OPT}_{d,r,B_F,S_{p,\gamma,B_G}} \tag{B.535}$$

$$+ rB_{a1}\left(\frac{2\sqrt{2}\sqrt{B_{x1}B_{G1}}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}} + \sqrt{2\log(d)d}\left(\gamma_\infty + O\left(\frac{B_{x\infty}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)\right)\right) \tag{B.536}$$

$$+ \eta\left(\sqrt{r}B_{a2}B_bT\eta B_{x1}^2 + m\tilde{b}\right) O\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right) + O\left(\frac{B_{a2}^2B_b^2}{\eta T\tilde{b}^2(mp)^{\frac{1}{2}}}\right) \tag{B.537}$$

$$+ \frac{1}{n^{\frac{1}{3}}}O\left(\left(\frac{rB_{a1}B_b}{\tilde{b}} + m\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + \frac{\tilde{b}}{B_{x1}}\right)\right)\right. \tag{B.538}$$

$$\left.\cdot\left(\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + T\eta^2B_{x1}\tilde{b}\right)B_x + \tilde{b}\right) + 1\right) \tag{B.539}$$

$$+ \frac{1}{n^{\frac{1}{3}}}O\left(m\eta\left(\frac{\tilde{b}\sqrt{\log m}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + T\eta^2B_{x1}\tilde{b}\right)\sqrt{B_{x2}}\right). \tag{B.540}$$

*Furthermore, for any* $\epsilon \in (0,1)$*, set*

$$\tilde{b} = \Theta\left(\frac{B_G^{\frac{1}{4}} B_{a2} B_b^{\frac{3}{4}}}{\sqrt{rB_{a1}}}\right), \quad m = \Omega\left(\frac{1}{p\epsilon^4}\left(rB_{a1}\sqrt{B_{x1}B_{G1}}\sqrt{\frac{B_b}{B_G}}\right)^4 + \frac{1}{\sqrt{\delta}} + \frac{1}{p}\left(\log\left(\frac{r}{\delta}\right)\right)^2\right), \tag{B.541}$$

$$\eta = \Theta\left(\frac{\epsilon}{\left(\frac{\sqrt{r}B_{a2}B_bB_{x1}}{(mp)^{\frac{1}{4}}} + m\tilde{b}\right)\left(\frac{\sqrt{\log m}B_{x1}(mp)^{\frac{1}{4}}}{\sqrt{B_bB_G}} + 1\right)}\right), \quad T = \Theta\left(\frac{1}{\eta B_{x1}(mp)^{\frac{1}{4}}}\right), \tag{B.542}$$

$$n = \Omega\left(\left(\frac{mB_xB_{a2}^2\sqrt{B_b}(mp)^{\frac{1}{2}}\log m}{\epsilon rB_{a1}\sqrt{B_G}}\right)^3 + \left(\frac{B_x}{\sqrt{B_{x2}}} + \log\frac{Tm}{p\delta} + (1 + \frac{1}{B_G} + \frac{1}{B_{G1}})\frac{B_{x\infty}}{|\ell'(0)|}\right)^3\right), \tag{B.543}$$

*we have there exists* $t \in [T]$ *with*

$$\Pr[\text{sign}(g_{\Xi^{(t)}})(\mathbf{x}) \neq y] \leqslant \mathcal{L}_{\mathcal{D}}\left(g_{\Xi^{(t)}}\right) \tag{B.544}$$

$$\leqslant \text{OPT}_{d,r,B_F,S^{\infty}_{p,\gamma_\infty,B_G,B_{G1}}} + rB_{a1}\sqrt{2\log(d)d}\left(\gamma_\infty + O\left(\frac{B_{x\infty}}{B_G|\ell'(0)|n^{\frac{1}{3}}}\right)\right) + \epsilon. \tag{B.545}$$

*Proof of Theorem B.62.* Proof of the theorem and parameter choices remain the same as Theorem B.23 except for setting $B_\epsilon = \frac{\sqrt{B_{x1}B_{G1}}}{(mp)^{\frac{1}{4}}}\sqrt{\frac{B_b}{B_G}}$ and apply Theorem B.61. □

**Feature Learning of Uniform Parity Functions**

Denote

$$g_{i,j} = \mathbb{E}_{(\mathbf{x},y)}\left[y\sigma'\left[\langle \mathbf{w}_i^{(0)}, \mathbf{x}\rangle - b_i\right]\mathbf{x}_j\right] \tag{B.546}$$

$$\xi_k = (-1)^{\frac{k-1}{2}} \frac{\binom{\frac{n-1}{2}}{\frac{k-1}{2}}}{\binom{n-1}{k-1}} \cdot 2^{-(n-1)} \binom{n-1}{\frac{n-1}{2}} \tag{B.547}$$

**Lemma B.63** (Uniform Parity Functions: Gradient Feature Learning. Corollary of Lemma 3 in Barak et al. (2022))**.** *Assume that* $n \geqslant 2(k+1)^2$*. Then, the following holds:*

*If* $j \in A$*, then*

$$g_{i,j} = \xi_{k-1} \prod_{l \in A \setminus \{j\}} (\mathbf{w}_{i,l}^{(0)}). \tag{B.548}$$

*If* $i \notin A$*, then*

$$g_{i,j} = \xi_{k-1} \prod_{l \in A \cup \{j\}} (\mathbf{w}_{i,l}^{(0)}). \tag{B.549}$$

**Lemma B.64** (Uniform Parity Functions: Existence of Good Networks (Alternative))**.** *Assume the same condition as in Theorem B.63. Define*

$$D = \frac{\sum_{l \in A} M_l}{\| \sum_{l \in A} M_l \|_2} \tag{B.550}$$

*and*

$$g^*(\mathbf{x}) = \sum_{i=0}^{k} (-1)^i \sqrt{k} \tag{B.551}$$

$$\cdot \left[ \sigma\left( \langle D, \mathbf{x} \rangle - \frac{2i-k-1}{\sqrt{k}} \right) - 2\sigma\left( \langle D, \mathbf{x} \rangle - \frac{2i-k}{\sqrt{k}} \right) + \sigma\left( \langle D, \mathbf{x} \rangle - \frac{2i-k+1}{\sqrt{k}} \right) \right]. \tag{B.552}$$

*For* $\mathcal{D}_{\mathrm{parity-uniform}}$ *setting, we have* $g^* \in \mathcal{F}_{d,3(k+1),B_F,S^{\infty}_{p,\gamma_\infty,B_G,B_{G1}}}$ *where* $B_F = (B_{a1}, B_{a2}, B_b) = \left(2\sqrt{k}, 2\sqrt{(k(k+1))}, \frac{k+1}{\sqrt{k}}\right)$*,* $p = \Theta\left(\frac{1}{2^{k-1}}\right)$*,* $\gamma_\infty = O\left(\frac{\sqrt{k}}{d-k}\right)$*,* $B_G = \Theta(B_{G1}) = \Theta(d^{-k})$ *and* $B_{x1} = \sqrt{d}$*,* $B_{x_2} = d$*. We also have* $\mathrm{OPT}_{d,3(k+1),B_F,S^{\infty}_{p,\gamma_\infty,B_G,B_{G1}}} = 0$*.*

*Proof of Theorem B.64.* Fix index $i$, with probability $p_1 = \Theta(2^{-k})$, we will have $\mathbf{w}_{i,j}^{(0)} = \text{sign}(a_i^{(0)}) \cdot \text{sign}(\xi_{k-1})$, for $\forall j$. For $\mathbf{w}_i^{(0)}$ that satisfy these conditions, we will have:

$$\text{sign}(a_i^{(0)})g_{i,j} = |\xi_{k-1}|, \ \ \forall j \in A \tag{B.553}$$

$$\text{sign}(a_i^{(0)})g_{i,j} = |\xi_{k+1}|, \ \ \forall j \notin A. \tag{B.554}$$

Then by Lemma 4 in Barak et al. (2022), we have

$$\left\|\frac{\text{sign}(a_i^{(0)})G(\mathbf{w}_i^{(0)}, \tilde{b})}{\|G(\mathbf{w}_i^{(0)}, \tilde{b})\|} - D\right\|_\infty \leqslant \max\left\{\left|\frac{1}{k\sqrt{\frac{1}{k}+\frac{1}{d-k}}} - \frac{1}{\sqrt{k}}\right|, \left|\frac{1}{(d-k)\sqrt{\frac{1}{k}+\frac{1}{d-k}}}\right|\right\} \tag{B.555}$$

$$\leqslant \frac{\sqrt{k}}{d-k} \tag{B.556}$$

and

$$\|\text{sign}(a_i^{(0)})G(\mathbf{w}_i^{(0)}, \tilde{b})\|_2 = \sqrt{k|\xi_{k-1}|^2 + (d-k)|\xi_{k+1}|^2} = \Theta(d^{\Theta(k)}). \tag{B.557}$$

From here, we can see that if we set $\gamma_\infty = \frac{\sqrt{k}}{d-k}$, $B_G = B_{G1} = \sqrt{k|\xi_{k-1}|^2 + (d-k)|\xi_{k+1}|^2}$, $p = p_1$, we will have $(D, +1), (D, -1) \in S^\infty_{p,\gamma_\infty,B_G,B_{G1}}$ by our symmetric initialization. As a result, we have $f^* \in \mathcal{F}_{d,3(k+1),B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}}$. Finally, it is easy to verify that $f^*(\mathbf{x}) = \text{XOR}(\mathbf{x}_A)$, thus $\text{OPT}_{d,3(k+1),B_F,S^\infty_{p,\gamma_\infty,B_G,B_{G1}}} = 0$. □

**Theorem B.65** (Uniform Parity Functions: Main Result (Alternative))**.** *For* $\mathcal{D}_{parity-uniform}$ *setting, for any* $\delta \in (0,1)$ *satisfying* $\delta \leqslant O(\frac{1}{d^2})$ *and for any* $\epsilon \in (0,1)$ *when*

$$m = \text{poly}\left(\log\left(\frac{1}{\delta}\right), \frac{1}{\epsilon}, 2^{\Theta(k)}, d\right), T = \Theta\left(d^{\Theta(k)}\right), n = \Theta\left(d^{\Theta(k)}\right) \tag{B.558}$$

*trained by Algorithm 1 with hinge loss, with probability at least* $1-\delta$ *over the initialization,* *with proper hyper-parameters, there exists* $t \in [T]$ *such that*

$$\Pr[\operatorname{sign}(g_{\Xi^{(t)}}(\mathbf{x})) \neq y] \leqslant \frac{k^2\sqrt{d\log(d)}}{d-k} + \epsilon. \tag{B.559}$$

*Proof of Theorem B.65.* Plug the values of parameters into Theorem B.62 and directly get the result. □

### B.5.7 Multiple Index Model with Low Degree Polynomial

**Problem Setup**

The multiple-index data problem has been used for studying network learning Bietti et al. (2022); Damian et al. (2022). We consider proving guarantees for the setting in Damian et al. (2022), following our framework. We use the properties of the problem to prove the key lemma (i.e., the existence of good networks) in our framework and then derive the final guarantee from our theorem of the simple setting (Theorem 3.4).

**Data Distributions.** We draw input from the distribution $\mathcal{D}_{\mathcal{X}} = \mathcal{N}(0, I_{d\times d})$, and we assume the target function is $g^*(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$, where $g^*$ is a degree $\tau$ polynomial normalized so that $\mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{X}}}[g^*(\mathbf{x})^2] = 1$.

**Assumption B.66.** *There exists linearly independent vectors* $u_1, \dots, u_r$ *such that* $g^*(\mathbf{x}) = g(\langle \mathbf{x}, u_1\rangle, \dots, \langle \mathbf{x}, u_r\rangle)$. $H := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{X}}}[\nabla^2 g^*(\mathbf{x})]$ *has rank* $r$*, where* $H$ *is a Hessian matrix.*

**Definition B.67.** *Denote the normalized condition number of* $H$ *by*

$$\kappa := \frac{\|H^\dagger\|}{\sqrt{r}}. \tag{B.560}$$

**Initialization and Loss.** For $\forall i \in [m]$, we use the following initialization:

$$a_i^{(0)} \sim \{-1, 1\}, \ \ \mathbf{w}_i^{(0)} \sim \mathcal{N}\left(0, \frac{1}{d} I_{d\times d}\right) \ \text{ and } \ b_i = 0. \tag{B.561}$$

For this regression problem, we use mean square loss:

$$\mathcal{L}_{\mathcal{D}_{\mathcal{X}}}(g_\Xi) = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{X}}}\left[(g_\Xi(\mathbf{x}) - g^*(\mathbf{x}))^2\right]. \tag{B.562}$$

**Training Process.** We use the following one-step training algorithm for this specific data distribution.

**Algorithm 7** Network Training via Gradient Descent Damian et al. (2022). Special case of Algorithm 4

```
Initialize $(\mathbf{a}^{(0)}, \mathbf{W}^{(0)}, \mathbf{b})$ as in Equation (3.9) and Equation (B.561); Sample $\mathcal{Z} \sim \mathcal{D}_{\mathcal{X}}^n$
$\rho = \frac{1}{n}\sum_{\mathbf{x}\in\mathcal{Z}} g^*(\mathbf{x}),\ \ \beta = \frac{1}{n}\sum_{\mathbf{x}\in\mathcal{Z}} g^*(\mathbf{x})\mathbf{x}$
$y = g^*(\mathbf{x}) - \rho - \beta \cdot \mathbf{x}$
$\mathbf{W}^{(1)} = \mathbf{W}^{(0)} - \eta^{(1)}(\nabla_{\mathbf{W}}\widetilde{\mathcal{L}}_{\mathcal{Z}}(f_{\Xi^{(0)}}) + \lambda^{(1)}\mathbf{W}^{(0)})$
Re-initialize $b_i \sim \mathcal{N}(0,1)$
for $t = 2$ to $T$ do
    $\mathbf{a}^{(t)} = \mathbf{a}^{(t-1)} - \eta^{(t)}\nabla_{\mathbf{a}}\widetilde{\mathcal{L}}_{\mathcal{Z}}(g_{\Xi^{(t-1)}})$
end for
```

**Lemma B.68** (Multiple Index Model with Low Degree Polynomial: Existence of Good Networks. Rephrase of Lemma 25 in Damian et al. (2022))**.** *Assume* $n \geqslant d^2 r\kappa^2(C_l \log(nmd))^{\tau+1}$, $d \geqslant C_d \kappa r^{3/2}$, *and* $m \geqslant r^\tau \kappa^{2\tau}(C_l \log(nmd))^{6\tau+1}$ *for sufficiently large constants* $C_d, C_l$, *and let* $\eta^{(1)} = \sqrt{\frac{d}{(C_l \log(nmd))^3}}$ *and* $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$. *Then with probability* $1 - \frac{1}{\text{poly}(m,d)}$, *there exists* $\tilde{\mathbf{a}} \in \mathbb{R}^m$ *such that* $f_{(\tilde{\mathbf{a}},\mathbf{W}^{(1)},\mathbf{b})}$ *satisfies*

$$\mathcal{L}_{\mathcal{D}_{\mathcal{X}}}\left(f_{(\tilde{\mathbf{a}},\mathbf{W}^{(1)},\mathbf{b})}\right) \leqslant O\left(\frac{1}{n} + \frac{r^\tau \kappa^{2\tau}(C_l \log(nmd))^{6\tau+1}}{m}\right) \tag{B.563}$$

*and*

$$\|\tilde{\mathbf{a}}\|_2^2 \leqslant O\left(\frac{r^\tau \kappa^{2\tau}(C_l \log(nmd))^{6\tau}}{m}\right). \tag{B.564}$$

### Multiple Index Model: Final Guarantee

Considering training by Algorithm 7, we have the following results.

**Theorem B.69** (Multiple Index Model with Low Degree Polynomial: Main Result). *Assume* $n \geqslant \Omega\left(d^2 r\kappa^2 (C_l \log(nmd))^{\tau+1} + m\right)$, $d \geqslant C_d \kappa r^{3/2}$, *and*

$$m \geqslant \Omega\left(\frac{1}{\epsilon} r^{\tau}\kappa^{2\tau}(C_l \log(nmd))^{6\tau+1}\right)$$

*for sufficiently large constants* $C_d, C_l$*. Let* $\eta^{(1)} = \sqrt{\frac{d}{(C_l \log(nmd))^3}}$ *and* $\lambda^{(1)} = \frac{1}{\eta^{(1)}}$*, and* $\eta = \eta^{(t)} = \Theta(m^{-1})$*, for all* $t \in \{2, 3, \ldots, T\}$*. For any* $\epsilon \in (0,1)$*, if* $T \geqslant \Omega\left(\frac{m^2}{\epsilon}\right)$*, then with properly set parameters and Algorithm 7, with high probability that there exists* $t \in [T]$ *such that*

$$\mathcal{L}_{\mathcal{D}_{\mathcal{X}}} g_{(a^{(t)}, \mathbf{W}^{(1)}, b)} \leqslant \epsilon. \tag{B.565}$$

*Proof of Theorem B.69.* By Theorem B.68, we have for properly chosen hyper-parameters,

$$\begin{aligned} \mathrm{OPT}_{\mathbf{W}^{(1)}, b, B_{a2}} \leqslant \mathcal{L}_{\mathcal{D}_{\mathcal{X}}}\left(f_{(\tilde{a}, \mathbf{W}^{(1)}, b)}\right) \leqslant & O\left(\frac{1}{n} + \frac{r^{\tau}\kappa^{2\tau}(C_l \log(nmd))^{6\tau+1})}{m}\right) && \text{(B.566)} \\ \leqslant & \frac{\epsilon}{3}. && \text{(B.567)} \end{aligned}$$

We compute the L-smooth constant of $\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(f_{(a, \mathbf{W}^{(1)}, b)}\right)$ to $a$.

$$\begin{aligned} & \left\| \nabla_a \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a_1, \mathbf{W}^{(1)}, b)}\right) - \nabla_a \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a_2, \mathbf{W}^{(1)}, b)}\right) \right\|_2 && \text{(B.568)} \\ = & \left\| \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left[ 2\left( g_{(a_1, \mathbf{W}^{(1)}, b)}(\mathbf{x}) - g^* - g_{(a_2, \mathbf{W}^{(1)}, b)}(\mathbf{x}) + g^* \right) \sigma(\mathbf{W}^{(1)\top}\mathbf{x} - b) \right] \right\|_2 && \text{(B.569)} \\ \leqslant & \left\| \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left[ 2\left( a_1^\top \sigma(\mathbf{W}^{(1)\top}\mathbf{x} - b) - a_2^\top \sigma(\mathbf{W}^{(1)\top}\mathbf{x} - b) \right) \sigma(\mathbf{W}^{(1)\top}\mathbf{x} - b) \right] \right\|_2 && \text{(B.570)} \\ \leqslant & \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left[ 2 \left\| a_1 - a_2 \right\|_2 \left\| \sigma(\mathbf{W}^{(1)\top}\mathbf{x} - b) \right\|_2^2 \right]. && \text{(B.571)} \end{aligned}$$

By the proof of Lemma 25 in Damian et al. (2022), we have for $\forall i \in [4m]$, with probability at least $1 - \frac{1}{\text{poly}(m,d)}$, $|\langle \mathbf{w}_i, \mathbf{x} \rangle| \leqslant 1$, with some large polynomial $\text{poly}(m, d)$. As a result, we have

$$\frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left\| \mathbf{W}^{(1)\top} \mathbf{x} \right\|_2^2 \leqslant m + \frac{1}{\text{poly}(m, d)} \leqslant O(m). \tag{B.572}$$

Thus, we have,

$$\begin{aligned} L &= O\left( \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left\| \sigma(\mathbf{W}^{(1)\top} \mathbf{x} - b) \right\|_2^2 \right) && \text{(B.573)} \\ &\leqslant O\left( \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left\| \mathbf{W}^{(1)\top} \mathbf{x} - b \right\|_2^2 \right) && \text{(B.574)} \\ &\leqslant O\left( \frac{1}{n} \sum_{\mathbf{x} \in \mathcal{Z}} \left\| \mathbf{W}^{(1)\top} \mathbf{x} \right\|_2^2 + \|b\|_2^2 \right) && \text{(B.575)} \\ &\leqslant O(m). && \text{(B.576)} \end{aligned}$$

This means that we can let $\eta = \Theta\left(m^{-1}\right)$ and we will get our convergence result. We can bound $\|a^{(1)}\|_2$ and $\|\tilde{a}\|_2$ by $\|a^{(1)}\|_2 = O\left(\sqrt{m}\right)$ and $\|\tilde{a}\|_2 = O\left(\frac{r^\tau \kappa^{2\tau} (C_l \log(nmd))^{6\tau}}{m}\right) = O(\epsilon)$. So, if we choose $T \geqslant \Omega\left(\frac{m}{\epsilon \eta}\right)$, there exists $t \in [T]$ such that $\widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(a^{(t)}, \mathbf{W}^{(1)}, b)}\right) - \widetilde{\mathcal{L}}_{\mathcal{Z}}\left(g_{(\tilde{a}, \mathbf{W}^{(1)}, b)}\right) \leqslant O\left(\frac{L\|a^{(1)} - \tilde{a}\|_2^2}{T}\right) \leqslant \epsilon/3$.

We also have $\sqrt{\frac{\|\tilde{a}\|_2^2 (\|\mathbf{W}^{(1)}\|_F^2 B_x^2 + \|b\|_2^2)}{n}} \leqslant \frac{\epsilon}{3}$. Then our theorem gets proved by Theorem 3.4. □

## B.6 Auxiliary Lemmas

In this section, we present some Lemmas used frequently.

**Lemma B.70** (Lemmas on Gradients)**.**

$$\nabla_{\mathbf{W}} \mathcal{L}_{(\mathbf{x},y)}(g_\Xi) = \left[ \frac{\partial \mathcal{L}_{(\mathbf{x},y)}(g_\Xi)}{\partial \mathbf{w}_1}, \dots, \frac{\partial \mathcal{L}_{(\mathbf{x},y)}(g_\Xi)}{\partial \mathbf{w}_i}, \dots, \frac{\partial \mathcal{L}_{(\mathbf{x},y)}(g_\Xi)}{\partial \mathbf{w}_{4m}} \right], \tag{B.577}$$

$$\frac{\partial \mathcal{L}_{(\mathbf{x},y)}(g_\Xi)}{\partial \mathbf{w}_i} = a_i \ell'(y g_\Xi(\mathbf{x})) y \left[ \sigma' \left( \langle \mathbf{w}_i, \mathbf{x} \rangle - b_i \right) \right] \mathbf{x}, \tag{B.578}$$

$$\nabla_{\mathbf{W}} \mathcal{L}_{\mathcal{D}}(g_\Xi) = \left[ \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_1}, \dots, \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_i}, \dots, \frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_{4m}} \right], \tag{B.579}$$

$$\frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial \mathbf{w}_i} = a_i \mathbb{E}_{(\mathbf{x},y)} \left[ \ell'(y g_\Xi(\mathbf{x})) y \left[ \sigma' \left( \langle \mathbf{w}_i, \mathbf{x} \rangle - b_i \right) \right] \mathbf{x} \right], \tag{B.580}$$

$$\frac{\partial \mathcal{L}_{\mathcal{D}}(g_\Xi)}{\partial a_i} = \mathbb{E}_{(\mathbf{x},y)} \left[ \ell'(y g_\Xi(\mathbf{x})) y \left[ \sigma \left( \langle \mathbf{w}_i, \mathbf{x} \rangle - b_i \right) \right] \right]. \tag{B.581}$$

*Proof.* These can be verified by direct calculation. □

**Lemma B.71** (Property of Symmetric Initialization)**.** *For any* $\mathbf{x} \in \mathbb{R}^d$*, we have* $g_{\Xi^{(0)}}(\mathbf{x}) = 0$ *. For all* $i \in [2m]$*, we have* $\mathbf{w}_i^{(1)} = -\mathbf{w}_{i+2m}^{(1)}$*. When input data is symmetric, i.e,* $\mathbb{E}_{(\mathbf{x},y)}[y\mathbf{x}] = \mathbf{0}$*, for all* $i \in [m]$*, we have* $\mathbf{w}_i^{(1)} = \mathbf{w}_{i+m}^{(1)}$*.*

*Proof of Theorem B.71.* By symmetric initialization, we have $g_{\Xi^{(0)}}(\mathbf{x}) = 0$. For all $i \in [2m]$, we have

$$\mathbf{w}_i^{(1)} = -\eta^{(1)} \ell'(0) a_i^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ y \sigma' \left[ \langle \mathbf{w}_i^{(0)}, \mathbf{x} \rangle - b_i \right] \mathbf{x} \right] \tag{B.582}$$

$$= \eta^{(1)} \ell'(0) a_{i+2m}^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ y \sigma' \left[ \langle \mathbf{w}_{i+2m}^{(0)}, \mathbf{x} \rangle - b_{i+2m} \right] \mathbf{x} \right] \tag{B.583}$$

$$= -\mathbf{w}_{i+2m}^{(1)}. \tag{B.584}$$

When $\mathbb{E}_{(\mathbf{x},y)}[y\mathbf{x}] = \mathbf{0}$, for all $i \in [m]$, we have

$$\mathbf{w}_i^{(1)} = -\eta^{(1)} \ell'(0) a_i^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ y \sigma' \left[ \langle \mathbf{w}_i^{(0)}, \mathbf{x} \rangle - b_i \right] \mathbf{x} \right] \tag{B.585}$$

$$= \eta^{(1)} \ell'(0) a_{i+m}^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ y \sigma' \left[ \langle -\mathbf{w}_{i+m}^{(0)}, \mathbf{x} \rangle + b_{i+m} \right] \mathbf{x} \right] \tag{B.586}$$

$$= \eta^{(1)} \ell'(0) a_{i+m}^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ y \sigma' \left[ \langle -\mathbf{w}_{i+m}^{(0)}, \mathbf{x} \rangle + b_{i+m} \right] \mathbf{x} - y\mathbf{x} \right] \tag{B.587}$$

$$= \eta^{(1)} \ell'(0) a_{i+m}^{(0)} \mathbb{E}_{(\mathbf{x},y)} \left[ -y \sigma' \left[ \langle \mathbf{w}_{i+m}^{(0)}, \mathbf{x} \rangle - b_{i+m} \right] \mathbf{x} \right] \tag{B.588}$$

$$=\mathbf{w}_{i+m}^{(1)}. \tag{B.589}$$

□

**Lemma B.72** (Property of Direction Neighborhood). *If* $\mathbf{w} \in \mathcal{C}_{D,\gamma}$*, we have* $\rho\mathbf{w} \in \mathcal{C}_{D,\gamma}$ *for any* $\rho \neq 0$*. We also have* $\mathbf{0} \notin \mathcal{C}_{D,\gamma}$*. Also, if* $(D, s) \in S_{p,\gamma,B_G}$*, we have* $(-D, s) \in S_{p,\gamma,B_G}$*.*

*Proof.* These can be verified by direct calculation. □

**Lemma B.73** (Maximum Gaussian Tail Bound). $M_n$ *is the maximum of* $n$ *i.i.d. standard normal Gaussian. Then*

$$\Pr\left(M_n \geqslant \sqrt{2\log n} + \frac{z}{\sqrt{2\log n}}\right) \leqslant e^{-z}. \tag{B.590}$$

*Proof.* These can be verified by direct calculation. □

**Lemma B.74** (Chi-squared Tail Bound). *If* $X$ *is a* $\chi^2(k)$ *random variable. Then,* $\forall z \in \mathbb{R}$*, we have*

$$\Pr(X \geqslant k + 2\sqrt{kz} + 2z) \leqslant e^{-z}. \tag{B.591}$$

*Proof.* These can be verified by direct calculation. □

**Lemma B.75** (Gaussian Tail Bound). *If* $g$ *is standard Gaussian and* $z > 0$*, we have*

$$\frac{1}{\sqrt{2\pi}}\frac{z}{z^2+1}e^{-z^2/2} < \Pr_{g\sim\mathcal{N}(0,1)}[g > z] < \frac{1}{\sqrt{2\pi}}\frac{1}{z}e^{-z^2/2}. \tag{B.592}$$

*Proof.* These can be verified by direct calculation. □

**Lemma B.76** (Gaussian Tail Expectation Bound). *If* $g$ *is standard Gaussian and* $z \in \mathbb{R}$*, we have*

$$|\mathbb{E}_{g\sim\mathcal{N}(0,1)}[\mathbb{I}[g > z]g]| < 2 \Pr_{g\sim\mathcal{N}(0,1)}[g > z]^{0.9}. \tag{B.593}$$

*Proof of Theorem B.76.* For any $p \in (0,1)$, we have

$$\left|\int_{-\infty}^{\sqrt{2}\mathrm{erf}^{-1}(2p-1)} \frac{e^{-\frac{x^2}{2}}x}{\sqrt{2\pi}}dx\right| < 2p^{0.9}, \tag{B.594}$$

where $\sqrt{2}\mathrm{erf}^{-1}(2p-1)$ is the quantile function of the standard Gaussian. We finish the proof by replacing $p$ to be $\Pr_{g\sim\mathcal{N}(0,1)}[g > z]$. □

**Lemma B.77.** *If a function* $g$ *satisfy* $h(n+2) = 2h(n+1) - (1-\rho^2)h(n) + \beta$ *for* $n \in \mathbb{N}_+$ *where* $\rho, \beta > 0$*, then* $h(n) = -\frac{\beta}{\rho^2} + c_1(1-\rho)^n + c_2(1+\rho)^n$*, where* $c_1, c_2$ *only depends on* $h(1)$ *and* $h(2)$.

*Proof.* These can be verified by direct calculation. □

**Lemma B.78** (Rademacher Complexity Bounds. Rephrase of Lemma 48 in Damian et al. (2022))**.** *For fixed* $\mathbf{W}, b$*, let* $\mathcal{F} = \{g_{(a,\mathbf{W},b)} : \|a\| \leqslant B_{a2}\}$*. Then,*

$$\mathfrak{R}(\mathcal{F}) \leqslant \sqrt{\frac{B_{a2}^2(\|\mathbf{W}\|_F^2 B_x^2 + \|b\|_2^2)}{n}}. \tag{B.595}$$

# C APPENDIX FOR CHAPTER 4

**Roadmap.** In Section C.1, we provide the potential limitations of this work. In Section C.2, we discuss the societal impacts of our work. In Section C.3, we provide more related works. In Section C.4, we introduce some definitions that will be used in the proof. In Section C.5, we introduce some auxiliary lemma from previous work that we need. In Section C.6, Section C.7, Section C.8, Section C.9, we provide the proof of our Lemmas and our main results. In particular, we provide two versions of proof (1) $k = 3$ and (2) general $k \geqslant 3$. We use $k = 3$ version to illustrate our proof intuition and then extend our proof to the general $k$ version. Finally, in Section C.10, we provide more experimental results and implementation details.

## C.1 Limitations

Our work has made progress in exploring how neural networks and Transformers can solve complex mathematical problems such as modular addition operation, but the practical application scope of their conclusions is limited.

On the other hand, we admit that our theorem can provide intuition but cannot fully explain the phenomena shown in Figure 4.4. Thus, we would like to introduce this more general data setting to the community so that we can study and understand grokking in a more broad way. Studying the relationship between the number of neurons and the grokking strength is interesting and important, and we will leave it as our future work.

## C.2 Societal Impact

Our work aims to understand the potential of large language models in mathematical reasoning and modular arithmetic. Our paper is purely theoretical and empirical in nature (mathematics problem) and thus we foresee no immediate negative ethical impact.

We propose that neural networks and transformers prefer to learn Fourier circuits when training on modular addition involving $k$ inputs under SGD, which may have a positive impact on the machine learning community. We hope our work will inspire effective algorithm design and promote a better understanding of large language models learning mechanisms.

## C.3 More Related Work

**Theoretical Work About Fourier Transform.** To calculate Fourier transform there are two main methodologies: one uses carefully chosen samples through hashing functions (referenced in works like Indyk et al. (2014); Indyk and Kapralov (2014); Kapralov (2016, 2017)) to achieve sublinear sample complexity and running time, while the other uses random samples (as discussed in Bourgain (2014); Haviv and Regev (2017); Nakos et al. (2019)) with sublinear sample complexity but nearly linear running time. There are many other works studying Fourier transform (Song, 2019; Jin et al., 2023; Gao et al., 2022; Lee et al., 2019b; Chen et al., 2020b; Song et al., 2022; Chen et al., 2016; Song et al., 2023a; Chen et al., 2023; Song et al., 2023b).

## C.4 More Notations and Definitions

We use $\mathbf{i}$ to denote $\sqrt{-1}$. Let $z = a + \mathbf{i}b$ denote a complex number where $a$ and $b$ are real numbers. Then we have $\overline{z} = a - \mathbf{i}b$ and $|z| := \sqrt{a^2 + b^2}$.

For any positive integer $n$, we use $[n]$ to denote set $\{1, 2, \cdots, n\}$. We use $\mathbb{E}[]$ to denote expectation. We use $\Pr[]$ to denote probability. We use $z^\top$ to denote the transpose of a vector $z$.

Considering a vector $z$, we denote the $\ell_2$ norm as $\|z\|_2 := (\sum_{i=1}^n z_i^2)^{1/2}$. We denote the $\ell_1$ norm as $\|z\|_1 := \sum_{i=1}^n |z_i|$. The number of non-zero entries in vector $z$ is defined as $\|z\|_0$. $\|z\|_\infty$ is defined as $\max_{i \in [n]} |z_i|$.

We define the vector norm and matrix norm as the following.

**Definition C.1** ($L_b$ (vector) norm). *Given a vector $v \in \mathbb{R}^n$ and $b \geqslant 1$, we have $\|v\|_b := (\sum_{i=1}^{n} |v_i|^b)^{1/b}$.*

**Definition C.2** ($L_{a,b}$ (matrix) norm). *The $L_{a,b}$ norm of a network with parameters $\theta = \{\theta_i\}_{i=1}^{m}$ is $\|\theta\|_{a,b} := (\sum_{i=1}^{m} \|\theta_i\|_a^b)^{1/b}$, where $\theta_i$ denotes the vector of concatenated parameters for a single neuron.*

We define our regularized training objective function.

**Definition C.3.** *Let $l$ be the cross-entropy loss. Our regularized training objective function is*

$$\mathcal{L}_\lambda(\theta) := \frac{1}{|D_p|} \sum_{(x,y) \in D_p} l(f(\theta, x), y) + \lambda \|\theta\|_{2,k+1}.$$

**Definition C.4.** *We define $\Theta^* := \arg\max_{\theta \in \Theta} h(\theta)$.*

Finally, let $\Omega := \mathbb{R}^{p \times (k+1)}$ denote the domain of each $\theta_i$, and let $\Omega'$ be a subset of $\Omega$. We say the parameter set $\theta = \{\theta_1, \dots, \theta_m\}$ has directional support on $\Omega'$, if for every $i \in [m]$, either $\theta_i = 0$ or there exists $\alpha_i > 0$ such that $\alpha_i \theta_i \in \Omega'$.

## C.5 Tools from Previous Work

Section C.5.1 states that we can use the single neuron level optimization to get the maximum-margin network. Section C.5.2 introduces the maximum-margin for multi-class.

### C.5.1 Tools from Previous Work: Implying Single/Combined Neurons

**Lemma C.5** (Lemma 5 in page 8 in Morwani et al. (2024)). *If the following conditions hold*

- *Given $\Theta := \{\theta : \|\theta\|_{a,b} \leqslant 1\}$.*

- *Given* $\Theta'^*_q := \arg\max_{\theta\in\Theta} \mathbb{E}_{(x,y)\sim q}[g'(\theta, x, y)]$.
- *Given* $\Omega := \{\theta_i : \|\theta_i\|_a \leqslant 1\}$.
- *Given* $\Omega'^*_q := \arg\max_{\theta_i\in\Omega} \mathbb{E}_{(x,y)\sim q}[\psi'(\theta, x, y)]$.

*Then:*

- *Let* $\theta \in \Theta'^*_q$. *We have* $\theta$ *only has directional support on* $\Omega'^*_q$.
- *Given* $\theta^*_1, \dots, \theta^*_m \in \Omega'^*_q$, *we have for any set of neuron scalars where* $\sum_{i=1}^m \alpha_i^\nu = 1, \alpha_i \geqslant 0$, *the weights* $\theta = \{\alpha_i\theta^*_i\}_{i=1}^m$ *is in* $\Theta'^*_q$.

Given $q^*$, then we can get the $\theta^*$ satisfying

$$\theta^* \in \arg\min_{\theta\in\Theta} \mathop{\mathbb{E}}_{(x,y)\sim q^*}[g'(\theta, x, y)]. \tag{C.1}$$

### C.5.2 Tools from Previous Work: Maximum Margin for Multi-Class

**Lemma C.6** (Lemma 6 in page 8 in Morwani et al. (2024))**.** *If the following conditions hold*

- *Given* $\Theta = \{\theta : \|\theta\|_{a,b} \leqslant 1\}$ *and* $\Theta'^*_q = \arg\max_{\theta\in\Theta} \mathbb{E}_{(x,y)\sim q}[g'(\theta, x, y)]$.
- *Given* $\Omega = \{\theta_i : \|\theta_i\|_a \leqslant 1\}$ *and* $\Omega'^*_q = \arg\max_{\theta_i\in\Omega} \mathbb{E}_{(x,y)\sim q}[\psi'(\theta, x, y)]$.
- *Suppose that* $\exists\{\theta^*, q^*\}$ *such that Equations equation 4.4 and equation C.1, and 4.8 holds.*

*Then, we can show:*

- $\theta^* \in \arg\max_{\theta\in\Theta} g(\theta, x, y)$
- $\forall\hat{\theta} \in \arg\max_{\theta\in\Theta} \min_{(x,y)\in D} g(\theta, x, y)$ *the below properties hold:*
  - $\hat{\theta}$ *only has directional support on* $\Omega'^*_{q^*}$.
  - $\forall(x, y) \in \mathrm{spt}(q^*), f(\hat{\theta}, x, y) - \max_{y'\in\mathcal{Y}\setminus\{y\}} f(\hat{\theta}, x, y') = \gamma^*$.

**Condition C.7** (Condition C.1 in page 8 in Morwani et al. (2024)). *We have* $g'(\theta^*, x, y) = g(\theta^*, x, y)$ *for all* $(x, y) \in \mathrm{spt}(q^*)$*, where* spt *is the support. It means:* $\{y' \in \mathcal{Y}\backslash\{y\} : \tau(x, y)[y'] > 0\} \subseteq \arg\max_{y' \in \mathcal{Y}\backslash\{y\}} f(\theta^*, x)[y']$.

# C.6 Class-weighted Max-margin Solution of Single Neuron

Section C.6.1 introduces some definitions. Section C.6.2 shows how we transfer the problem to discrete Fourier space. Section C.6.3 proposes the weighted margin of the single neuron. Section C.6.4 shows how we transfer the problem to discrete Fourier space for general $k$ version. Section C.6.5 provides the solution set for general $k$ version and the maximum weighted margin for a single neuron.

## C.6.1 Definitions

**Definition C.8.** *When* $k = 3$*, let*

$$\eta_{u_1,u_2,u_3,w}(\delta) := \mathbb{E}_{a_1,a_2,a_3}[(u_1(a_1) + u_2(a_2) + u_3(a_3))^3 w(a_1 + a_2 + a_3 - \delta)].$$

**Definition C.9.** *Let* $\eta$ *be defined in Definition C.8. When* $k = 3$*, provided the following conditions are met*

- *We denote* $\mathcal{B}$ *as the ball that* $\|u_1\|^2 + \|u_2\|^2 + \|u_3\|^2 + \|w\|^2 \leqslant 1$.

*We define*

$$\Omega_q'^* = \arg\max_{u_1,u_2,u_3,w \in \mathcal{B}} (\eta_{u_1,u_2,u_3,w}(0) - \mathbb{E}_{\delta \neq 0}[\eta_{u_1,u_2,u_3,w}(\delta)]).$$

## C.6.2 Transfer to Discrete Fourier Space

The goal of this section is to prove the following Lemma,

**Lemma C.10.** *When* $k = 3$*, provided the following conditions are met*

- *We denote $\mathcal{B}$ as the ball that $\|u_1\|^2 + \|u_2\|^2 + \|u_3\|^2 + \|w\|^2 \leqslant 1$.*
- *We define $\Omega_q'^*$ in Definition C.9.*
- *We adopt the uniform class weighting: $\forall c' \neq a_1 + a_2 + a_3,\ \ \tau(a_1, a_2, a_3)[c'] := 1/(p-1)$.*

*We have the following*

$$\Omega_q'^* = \arg\max_{u_1,u_2,u_3,,w\in\mathcal{B}} \frac{6}{(p-1)p^3} \sum_{j\neq 0} \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j).$$

*Proof.* We have

$$\begin{aligned}\eta_{u_1,u_2,u_3,w}(\delta) =& \mathbb{E}_{a_1,a_2,a_3}[(u_1(a_1) + u_2(a_2) + u_3(a_3))^3 w(a_1 + a_2 + a_3 - \delta)]\\ =& \mathbb{E}_{a_1,a_2,a_3}[(u_1(a_1)^3 + 3u_1(a_1)^2u_2(a_2) + 3u_1(a_1)^2u_3(a_3) + 3u_1(a_1)u_2(a_2)^2\\ &+ 6u_1(a_1)u_2(a_2)u_3(a_3) + 3u_1(a_1)u_3(a_3)^2 + u_2(a_2)^3 + 3u_2(a_2)^2u_3(a_3)\\ &+ 3u_2(a_2)u_3(a_3)^2 + u_3(a_3)^3)w(a_1 + a_2 + a_3 - \delta)].\end{aligned}$$

Recall $\mathcal{B}$ is defined as Lemma Statement.

The goal is to solve the following mean margin maximization problem:

$$\begin{aligned}&\arg\max_{u_1,u_2,u_3,w\in\mathcal{B}} (\eta_{u_1,u_2,u_3,w}(0) - \mathbb{E}_{\delta\neq 0}[\eta_{u_1,u_2,u_3,w}(\delta)])\\ =& \frac{p}{p-1}(\eta_{u_1,u_2,u_3,w}(0) - \mathbb{E}_{\delta}[\eta_{u_1,u_2,u_3,w}(\delta)]),\end{aligned} \tag{C.2}$$

where the equation follows $\tau(a_1, a_2, a_3)[c'] := 1/(p-1)\ \ \forall c' \neq a_1 + a_2 + a_3$ and $1 - \frac{1}{p-1} = \frac{p}{p-1}$.

First, note that

$$\begin{aligned}&\mathbb{E}_{a_1,a_2,a_3}[u_1(a_1)^3 w(a_1 + a_2 + a_3 - \delta)]\\ =& \mathbb{E}_{a_1}[u_1(a_1)^3 \mathbb{E}_{a_2,a_3}[w(a_1 + a_2 + a_3 - \delta)]]\\ =& 0,\end{aligned}$$

where the first step follows from taking out the $u_1(a_1)$ from the expectation for $a_2, a_3$, and the last step is from the definition of $w$.

Similarly for the $u_2(a_2)^3$,$u_3(a_3)^3$ components of $\eta$, they equal to 0.

Note that

$$\begin{aligned}
&\mathbb{E}_{a_1,a_2,a_3}[u_1(a_1)^2 u_2(a_2) w(a_1+a_2+a_3-\delta)]\\
=&\mathbb{E}_{a_1}[u_1(a_1)^2 \mathbb{E}_{a_2}[u_2(a_2)\mathbb{E}_{a_3}[w(a_1+a_2+a_3-\delta)]]]\\
=&0,
\end{aligned}$$

where the first step follows from simple algebra and the last step comes from the definition of $w$.

Similarly for the $u_1(a_1)^2u_3(a_3)$, $u_2(a_2)^2u_1(a_1)$, $u_2(a_2)^2u_3(a_3)$, $u_3(a_3)^2u_1(a_1)$, $u_3(a_3)^2u_2(a_2)$ components of $\eta$, they equal to 0.

Hence, we can rewrite Eq. equation C.2 as

$$\arg\max_{u_1,u_2,u_3,w\in\mathcal{B}} \frac{6p}{p-1}(\tilde{\eta}_{u_1,u_2,u_3,w}(0) - \mathbb{E}_\delta[\tilde{\eta}_{u_1,u_2,u_3,w}(\delta)]),$$

where

$$\tilde{\eta}_{u_1,u_2,u_3,w}(\delta) := \mathbb{E}_{a_1,a_2,a_3}[u_1(a_1)u_2(a_2)u_3(a_3)w(a_1+a_2+a_3-\delta)].$$

Let $\rho := e^{2\pi i/p}$, and let $\hat{u}_1, \hat{u}_2, \hat{u}_3, \hat{w}$ be the DFT of $u_1, u_2, u_3$, and $w$ respectively:

$$\begin{aligned}
&\tilde{\eta}_{u_1,u_2,u_3,w}(\delta)\\
=&\mathbb{E}_{a_1,a_2,a_3}[(\frac{1}{p}\sum_{j_1=0}^{p-1}\hat{u}_1(j_1)\rho^{j_1 a_1})(\frac{1}{p}\sum_{j_2=0}^{p-1}\hat{u}_2(j_2)\rho^{j_2 a_2})(\frac{1}{p}\sum_{j_3=0}^{p-1}\hat{u}_3(j_3)\rho^{j_3 a_3})(\frac{1}{p}\sum_{j_4=0}^{p-1}\hat{w}(j_4)\rho^{j_4(a_1+a_2+a_3-\delta)})]\\
=&\frac{1}{p^4}\sum_{j_1,j_2,j_3,j_4}\hat{u}_1(j_1)\hat{u}_2(j_2)\hat{u}_3(j_3)\hat{w}(j_4)\rho^{-j_4\delta}(\mathbb{E}_{a_1}[\rho^{(j_1+j_4)a_1}])(\mathbb{E}_{a_2}[\rho^{(j_2+j_4)a_2}])(\mathbb{E}_{a_3}[\rho^{(j_3+j_4)a_3}])
\end{aligned}$$

$$= \frac{1}{p^4} \sum_j \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j)\rho^{j\delta}$$

where the first step follows from $\rho := e^{2\pi i/p}$ and $\hat{u}_1, \hat{u}_2, \hat{u}_3, \hat{w}$ are the discrete Fourier transforms of $u_1, u_2, u_3, w$, the second step comes from simple algebra, the last step is from that only terms where $j_1 + j_4 = j_2 + j_4 = j_3 + j_4 = 0$ survive.

Hence, we need to maximize

$$\begin{aligned} & \frac{6p}{p-1}(\tilde{\eta}_{u_1,u_2,u_3,w}(0) - \mathbb{E}_\delta[\tilde{\eta}_{u_1,u_2,u_3,w}(\delta)]) \\ = & \frac{6p}{p-1}(\frac{1}{p^4}\sum_j \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j) - \frac{1}{p^4}\sum_j \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j)(\mathbb{E}_\delta \rho^{j\delta})) \\ = & \frac{6}{(p-1)p^3}\sum_{j\neq 0} \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j). \\ = & \frac{6}{(p-1)p^3}\sum_{j\in[-(p-1)/2,+(p-1)/2]\backslash 0} \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j). \end{aligned} \tag{C.3}$$

where the first step is from $\tilde{\eta}_{u_1,u_2,u_3,w}(\delta)$ definition, the second step is from $\mathbb{E}_\delta \rho^{j\delta} = 0$ when $j \neq 0$, and the last step follows from simple algebra.

□

### C.6.3 Get Solution Set

**Lemma C.11.** *When $k = 3$, provided the following conditions are met*

- *We denote $\mathcal{B}$ as the ball that $\|u_1\|^2 + \|u_2\|^2 + \|u_3\|^2 + \|w\|^2 \leqslant 1$.*
- *We define $\Omega_q^{'*}$ in Definition C.9.*
- *We adopt the uniform class weighting: $\forall c' \neq a_1 + a_2 + a_3,\ \ \tau(a_1, a_2, a_3)[c'] := 1/(p-1)$.*

- *For any $\zeta \in \{1, \dots, \frac{p-1}{2}\}$, there exists a scaling constant $\beta \in \mathbb{R}$ and*

$$\begin{aligned} u_1(a_1) &= \beta \cdot \cos(\theta^*_{u_1} + 2\pi\zeta a_1/p) \\ u_2(a_2) &= \beta \cdot \cos(\theta^*_{u_2} + 2\pi\zeta a_2/p) \\ u_3(a_3) &= \beta \cdot \cos(\theta^*_{u_3} + 2\pi\zeta a_3/p) \\ w(c) &= \beta \cdot \cos(\theta^*_{w} + 2\pi\zeta c/p) \end{aligned}$$

*where $\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_{w} \in \mathbb{R}$ are some phase offsets satisfying $\theta^*_{u_1} + \theta^*_{u_2} + \theta^*_{u_3} = \theta^*_{w}$.*

*Then, we have the following*

$$\Omega'^*_q = \{(u_1, u_2, u_3, w)\},$$

*and*

$$\max_{u_1,u_2,u_3,w \in \mathcal{B}} (\eta_{u_1,u_2,u_3,w}(0) - \mathbb{E}_{\delta \neq 0}[\eta_{u_1,u_2,u_3,w}(\delta)]) = \frac{3}{16} \cdot \frac{1}{p(p-1)}.$$

*Proof.* By Lemma C.10, we only need to maximize Equation equation C.3.

Thus, the mass of $\hat{u}_1, \hat{u}_2, \hat{u}_3$, and $\hat{w}$ must be concentrated on the same frequencies. For all $j \in \mathbb{Z}_p$, we have

$$\hat{u}_1(-j) = \overline{\hat{u}_1(j)}, \hat{u}_2(-j) = \overline{\hat{u}_2(j)}, \hat{u}_3(-j) = \overline{\hat{u}_3(j)}, \hat{w}(-j) = \overline{\hat{w}(j)} \tag{C.4}$$

as $u_1, u_2, u_3, w$ are real-valued.

For all $j \in \mathbb{Z}_p$ and for $u_1, u_2, u_3, w$, we denote $\theta_{u_1}, \theta_{u_2}, \theta_{u_3}, \theta_w \in [0, 2\pi)^p$ as their phase, e.g.:

$$\hat{u}_1(j) = |\hat{u}_1(j)| \exp(\mathbf{i}\theta_{u_1}(j)).$$

Consider the odd $p$, Equation equation C.3 becomes:

$\mathrm{equation\ C.3}$

$$
\begin{aligned}
&= \frac{6}{(p-1)p^3} \sum_{j\in[-(p-1)/2,+(p-1)/2]\backslash 0} \hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\hat{w}(-j) \\
&= \frac{6}{(p-1)p^3} \sum_{j=1}^{(p-1)/2} (\hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)\overline{\hat{w}(j)} + \overline{\hat{u}_1(j)\hat{u}_2(j)\hat{u}_3(j)}\hat{w}(j)) \\
&= \frac{6}{(p-1)p^3} \sum_{j=1}^{(p-1)/2} |\hat{u}_1(j)||\hat{u}_2(j)||\hat{u}_3(j)||\hat{w}(j)| \cdot \\
&\quad \big( \exp(\mathbf{i}(\theta_{u_1}(j) + \theta_{u_2}(j) + \theta_{u_3}(j) - \theta_w(j)) + \exp(\mathbf{i}(-\theta_{u_1}(j) - \theta_{u_2}(j) - \theta_{u_3}(j) + \theta_w(j)) \big) \\
&= \frac{12}{(p-1)p^3} \sum_{j=1}^{(p-1)/2} |\hat{u}_1(j)||\hat{u}_2(j)||\hat{u}_3(j)||\hat{w}(j)| \cos(\theta_{u_1}(j) + \theta_{u_2}(j) + \theta_{u_3}(j) - \theta_w(j)).
\end{aligned}
$$

where the first step comes from definition equation C.3, the second step follows from Eq. equation C.4, the third step comes from $\hat{u}_1(-j) = \overline{\hat{u}_1(j)}$ and $\hat{u}_1(j) = |\hat{u}_1(j)| \exp(\mathbf{i}\theta_{u_1}(j))$, the last step follow from Euler's formula.

Thus, we need to optimize:

$$
\max_{u_1,u_2,u_3,w\in\mathcal{B}} \frac{12}{(p-1)p^3} \sum_{j=1}^{(p-1)/2} |\hat{u}_1(j)||\hat{u}_2(j)||\hat{u}_3(j)||\hat{w}(j)| \cos(\theta_{u_1}(j) + \theta_{u_2}(j) + \theta_{u_3}(j) - \theta_w(j)). \tag{C.5}
$$

The norm constraint $\|u_1\|^2 + \|u_2\|^2 + \|u_3\|^2 + \|w\|^2 \leqslant 1$ is equivalent to

$$
\|\hat{u}_1\|^2 + \|\hat{u}_2\|^2 + \|\hat{u}_3\|^2 + \|\hat{w}\|^2 \leqslant p
$$

by using Plancherel's theorem. Thus, we need to select them in such a way that

$$
\theta_{u_1}(j) + \theta_{u_2}(j) + \theta_{u_3}(j) = \theta_w(j),
$$

ensuring that, for each j, the expression $\cos(\theta_{u_1}(j) + \theta_{u_2}(j) + \theta_{u_3}(j) - \theta_w(j)) = 1$ is maximized, except in cases where the scalar of the j-th term is 0.

This further simplifies the problem to:

$$\max_{|\hat{u}_1|,|\hat{u}_2|,|\hat{u}_3|,|\hat{w}|:\|\hat{u}_1\|^2+\|\hat{u}_2\|^2+\|\hat{u}_3\|^2+\|\hat{w}\|^2\leqslant p} \frac{12}{(p-1)p^3} \sum_{j=1}^{(p-1)/2} |\hat{u}_1(j)||\hat{u}_2(j)||\hat{u}_3(j)||\hat{w}(j)|. \tag{C.6}$$

Then, we have

$$|\hat{u}_1(j)||\hat{u}_2(j)||\hat{u}_3(j)||\hat{w}(j)| \leqslant (\frac{1}{4} \cdot (|\hat{u}_1(j)|^2 + |\hat{u}_2(j)|^2 + |\hat{u}_3(j)|^2 + |\hat{w}(j)|^2))^2. \tag{C.7}$$

where the first step is from inequality of quadratic and geometric means.

We define $z : \{1, \dots, \frac{p-1}{2}\} \to \mathbb{R}$ as

$$z(j) := |\hat{u}_1(j)|^2 + |\hat{u}_2(j)|^2 + |\hat{u}_3(j)|^2 + |\hat{w}(j)|^2.$$

We need to have $\hat{u}_1(0) = \hat{u}_2(0) = \hat{u}_3(0) = \hat{w}(0) = 0$. Then, the upper-bound of Eq. equation C.6 is given by

$$\begin{aligned}
&\frac{12}{(p-1)p^3} \cdot \max_{\|z\|_1 \leqslant \frac{p}{2}} \sum_{j=1}^{(p-1)/2} (\frac{z(j)}{4})^2 \\
=& \frac{3}{4(p-1)p^3} \cdot \max_{\|z\|_1 \leqslant \frac{p}{2}} \sum_{j=1}^{(p-1)/2} z(j)^2 \\
=& \frac{3}{4(p-1)p^3} \cdot \max_{\|z\|_1 \leqslant \frac{p}{2}} \|z\|_2^2 \\
\leqslant& \frac{3}{4(p-1)p^3} \cdot \frac{p^2}{4} \\
=& \frac{3}{16} \cdot \frac{1}{p(p-1)},
\end{aligned}$$

where the first step follows from simple algebra, the second step comes from the definition of $L_2$ norm, the third step follows from $\|z\|_2 \leqslant \|z\|_1 \leqslant \frac{p}{2}$, the last step comes from simple algebra.

For the inequality of quadratic and geometric means, Eq. equation C.7 becomes equality when $|\hat{u}_1(j)| = |\hat{u}_2(j)| = |\hat{u}_3(j)| = |\hat{w}(j)|$. To achieve $\|z\|_2 = \frac{p}{2}$, all the mass must be placed on a single frequency. Hence, for some frequency $\zeta \in \{1, \dots, \frac{p-1}{2}\}$, to achieve the upper bound, we have:

$$|\hat{u}_1(j)| = |\hat{u}_2(j)| = |\hat{u}_3(j)| = |\hat{w}(j)| = \begin{cases} \sqrt{p/8} & \text{if } j = \pm\zeta \\ 0 & \text{otherwise} \end{cases}, \tag{C.8}$$

In this case, Eq. equation C.6 matches the upper bound.

$$\frac{12}{(p-1)p^3} \cdot (\frac{p}{8})^2 = \frac{3}{16} \cdot \frac{1}{p(p-1)},$$

where the first step is by simple algebra. Hence, the maximum-margin is $\frac{3}{16} \cdot \frac{1}{p(p-1)}$.

Let $\theta^*_{u_1} := \theta_{u_1}(\zeta)$. Combining all the results, up to scaling, it is established that all neurons which maximize the expected class-weighted margin conform to the form:

$$\begin{aligned} u_1(a_1) &= \frac{1}{p}\sum_{j=0}^{p-1} \hat{u}_1(j)\rho^{ja_1} \\ &= \frac{1}{p} \cdot (\hat{u}_1(\zeta)\rho^{\zeta a_1} + \hat{u}_1(-\zeta)\rho^{-\zeta a_1}) \\ &= \frac{1}{p} \cdot (\sqrt{\frac{p}{8}}\exp(\mathbf{i}\theta^*_{u_1})\rho^{\zeta a_1} + \sqrt{\frac{p}{8}}\exp(-\mathbf{i}\theta^*_{u_1})\rho^{-\zeta a_1}) \\ &= \sqrt{\frac{1}{2p}}\cos(\theta^*_{u_1} + 2\pi\zeta a_1/p), \end{aligned}$$

where the first step comes from the definition of $u_1(a)$, the second step and third step follow from Eq. equation C.8, the last step follows from Euler's formula.

Similarly,

$$u_2(a_2) = \sqrt{\frac{1}{2p}} \cos(\theta^*_{u_2} + 2\pi\zeta a_2/p)$$
$$u_3(a_3) = \sqrt{\frac{1}{2p}} \cos(\theta^*_{u_3} + 2\pi\zeta a_3/p)$$
$$w(c) = \sqrt{\frac{1}{2p}} \cos(\theta^*_{w} + 2\pi\zeta c/p),$$

for some phase offsets $\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_{w} \in \mathbb{R}$ satisfying $\theta^*_{u_1} + \theta^*_{u_2} + \theta^*_{u_3} = \theta^*_{w}$ and some $\zeta \in \mathbb{Z}_p \backslash \{0\}$, where $u_1, u_2, u_3$, and $w$ shares the same $\zeta$.

□

### C.6.4 Transfer to Discrete Fourier Space for General $k$ Version

**Definition C.12.** *Let*

$$\eta_{u_1,\dots,u_k,w}(\delta) := \mathbb{E}_{a_1,\dots,a_k}[(u_1(a_1) + \dots + u_k(a_k))^k w(a_1 + \dots + a_k - \delta)].$$

**Definition C.13.** *Let $\eta$ be defined in Definition C.12. Provided the following conditions are met*

- *We denote $\mathcal{B}$ as the ball that $\|u_1\|^2 + \dots + \|u_k\|^2 + \|w\|^2 \leqslant 1$.*

*We define*

$$\Omega'^*_q = \arg\max_{u_1,\dots,u_k,w \in \mathcal{B}} (\eta_{u_1,\dots,u_k,w}(0) - \mathbb{E}_{\delta \neq 0}[\eta_{u_1,\dots,u_k,w}(\delta)]).$$

The goal of this section is to prove the following Lemma,

**Lemma C.14.** *Provided the following conditions are met*

- *Let $\mathcal{B}$ denote the ball that $\|u_1\|^2 + \dots + \|u_k\|^2 + \|w\|^2 \leqslant 1$.*

- *We define $\Omega_q^{\prime *}$ in Definition C.13.*
- *We adopt the uniform class weighting: $\forall c' \neq a_1 + \cdots + a_k,\ \ \tau(a_1, \ldots, a_k)[c'] := 1/(p-1)$.*

*We have the following*

$$\Omega_q^{\prime *} = \operatorname*{arg\,max}_{u_1,\ldots,u_k,w\in\mathcal{B}} \frac{k!}{(p-1)p^k} \sum_{j\neq 0} \hat{w}(-j) \prod_{i=1}^{k} \hat{u}_i(j).$$

*Proof.* We have

$$\eta_{u_1,\ldots,u_k,w}(\delta) = \mathbb{E}_{a_1,\ldots,a_k}[(u_1(a_1) + \cdots + u_k(a_k))^k w(a_1 + \cdots + a_k - \delta)].$$

The goal is to solve the following mean margin maximization problem:

$$\begin{aligned} &\operatorname*{arg\,max}_{u_1,\ldots,u_k,w\in\mathcal{B}} (\eta_{u_1,\ldots,u_k,w}(0) - \mathbb{E}_{\delta\neq 0}[\eta_{u_1,\ldots,u_k,w}(\delta)]) \\ =& \frac{p}{p-1}(\eta_{u_1,\ldots,u_k,w}(0) - \mathbb{E}_{\delta}[\eta_{u_1,\ldots,u_k,w}(\delta)]), \end{aligned} \tag{C.9}$$

where the equation follows $\tau(a_1, \ldots, a_k)[c'] := 1/(p-1)\ \ \forall c' \neq a_1 + \cdots + a_k$ and $1 - \frac{1}{p-1} = \frac{p}{p-1}$.

We note that all terms are zero rather than $w(\cdot) \cdot \prod_{i=1}^{k} u_i(a_i)$.

Hence, we can rewrite Eq. equation C.9 as

$$\operatorname*{arg\,max}_{u_1,\ldots,u_k,w\in\mathcal{B}} \frac{k!p}{p-1}(\tilde{\eta}_{u_1,\ldots,u_k,w}(0) - \mathbb{E}_{\delta}[\tilde{\eta}_{u_1,\ldots,u_k,w}(\delta)]),$$

where

$$\tilde{\eta}_{u_1,\ldots,u_k,w}(\delta) := \mathbb{E}_{a_1,\ldots,a_k}[w(a_1 + \cdots + a_k - \delta) \prod_{i=1}^{k} u_i(a_i)].$$

Let $\rho := e^{2\pi i/p}$, and $\hat{u}_1, \ldots, \hat{u}_k, \hat{w}$ denote the discrete Fourier transforms of $u_1, \ldots, u_k$, and $w$ respectively. We have

$$\tilde{\eta}_{u_1,\ldots,u_k,w}(\delta) = \frac{1}{p^{k+1}} \sum_{j=0}^{p-1} \hat{w}(-j)\rho^{j\delta} \prod_{i=1}^{k} \hat{u}_i(j)$$

which comes from $\rho := e^{2\pi i/p}$ and $\hat{u}_1, \ldots, \hat{u}_k, \hat{w}$ are the discrete Fourier transforms of $u_1, \ldots, u_k, w$.

Hence, we need to maximize

$$\begin{aligned}
&\frac{k!p}{p-1}(\tilde{\eta}_{u_1,\ldots,u_k,w}(0) - \mathbb{E}_\delta[\tilde{\eta}_{u_1,\ldots,u_k,w}(\delta)]) \\
=& \frac{k!p}{p-1} \cdot \left( \frac{1}{p^{k+1}} \sum_{j=0}^{p-1} \hat{w}(-j) \prod_{i=1}^{k} \hat{u}_i(j) - \frac{1}{p^{k+1}} \sum_{j=0}^{p-1} \hat{w}(-j)(\mathbb{E}_\delta[\rho^{j\delta}]) \prod_{i=1}^{k} \hat{u}_i(j) \right) \\
=& \frac{k!}{(p-1)p^k} \sum_{j\neq 0} \hat{w}(-j) \prod_{i=1}^{k} \hat{u}_i(j). \\
=& \frac{k!}{(p-1)p^k} \sum_{j\in[-(p-1)/2,+(p-1)/2]\backslash 0} \hat{w}(-j) \prod_{i=1}^{k} \hat{u}_i(j).
\end{aligned} \tag{C.10}$$

where the first step follows from the definition of $\tilde{\eta}_{u_1,\ldots,u_k,w}(\delta)$, the second step follows from $\mathbb{E}_\delta[\rho^{j\delta}] = 0$ when $j \neq 0$, the last step is from simple algebra.

□

### C.6.5 Get Solution Set for General k Version

**Lemma C.15** (Formal version of Lemma 4.10)**.** *Provided the following conditions are met*

- *We denote $\mathcal{B}$ as the ball that $\|u_1\|^2 + \cdots + \|u_k\|^2 + \|w\|^2 \leqslant 1$.*
- *Let $\Omega_q^{'*}$ be defined as Definition C.13.*

- *We adopt the uniform class weighting:* $\forall c' \neq a_1 + \cdots + a_k,\ \ \tau(a_1, \ldots, a_k)[c'] := 1/(p-1)$.
- *For any* $\zeta \in \{1, \ldots, \frac{p-1}{2}\}$*, there exists a scaling constant* $\beta \in \mathbb{R}$ *and*

$$\begin{aligned} u_1(a_1) &= \beta \cdot \cos(\theta^*_{u_1} + 2\pi\zeta a_1/p) \\ u_2(a_2) &= \beta \cdot \cos(\theta^*_{u_2} + 2\pi\zeta a_2/p) \\ &\cdots \\ u_k(a_k) &= \beta \cdot \cos(\theta^*_{u_k} + 2\pi\zeta a_k/p) \\ w(c) &= \beta \cdot \cos(\theta^*_{w} + 2\pi\zeta c/p) \end{aligned}$$

*where* $\theta^*_{u_1}, \ldots, \theta^*_{u_k}, \theta^*_w \in \mathbb{R}$ *are some phase offsets satisfying* $\theta^*_{u_1} + \cdots + \theta^*_{u_k} = \theta^*_w$.

*Then, we have the following*

$$\Omega'^*_q = \{(u_1, \ldots, u_k, w)\},$$

*and*

$$\max_{u_1,\ldots,u_k,w \in \mathcal{B}} (\eta_{u_1,\ldots,u_k,w}(0) - \mathbb{E}_{\delta \neq 0}[\eta_{u_1,\ldots,u_k,w}(\delta)]) = \frac{2(k!)}{(2k+2)^{(k+1)/2}(p-1)p^{(k-1)/2}}.$$

*Proof.* By Lemma C.14, we only need to maximize Equation equation C.10. Thus, the mass of $\hat{u}_1, \ldots, \hat{u}_k$, and $\hat{w}$ must be concentrated on the same frequencies. For all $j \in \mathbb{Z}_p$, we have

$$\hat{u}_i(-j) = \overline{\hat{u}_i(j)}, \quad \hat{w}(-j) = \overline{\hat{w}(j)} \tag{C.11}$$

as $u_1, \ldots, u_k, w$ are real-valued. For all $j \in \mathbb{Z}_p$ and for $u_1, u_2, u_3, w$, we denote $\theta_{u_1}, \ldots, \theta_{u_k}, \theta_w \in [0, 2\pi)^p$ as their phase, e.g.:

$$\hat{u}_1(j) = |\hat{u}_1(j)| \exp(\mathbf{i}\theta_{u_1}(j)). \tag{C.12}$$

Considering odd $p$, Equation equation C.10 becomes:

$$\begin{aligned}
\mathit{equation}\ \mathrm{C.10} &= \frac{k!}{(p-1)p^k} \sum_{j \in [-(p-1)/2, +(p-1)/2] \setminus 0} \hat{w}(-j) \prod_{i=1}^{k} \hat{u}_i(j) \\
&= \frac{k!}{(p-1)p^k} \sum_{j=1}^{(p-1)/2} (\prod_{i=1}^{k} \hat{u}_i(j) \overline{\hat{w}(j)} + \hat{w}(j) \prod_{i=1}^{k} \overline{\hat{u}_i(j)}) \\
&= \frac{2(k!)}{(p-1)p^k} \sum_{j=1}^{(p-1)/2} |\hat{w}(j)| \cos(\sum_{i=1}^{k} \theta_{u_i}(j) - \theta_w(j)) \prod_{i=1}^{k} |\hat{u}_i(j)|.
\end{aligned}$$

where the first step follows from definition equation C.10, the second step comes from Eq. equation C.11, the last step follows from Eq. equation C.12, i.e., Euler's formula.

Thus, we need to optimize:

$$\max_{u_1, \ldots, u_k, w \in \mathcal{B}} \frac{2(k!)}{(p-1)p^k} \sum_{j=1}^{(p-1)/2} |\hat{w}(j)| \cos(\sum_{i=1}^{k} \theta_{u_i}(j) - \theta_w(j)) \prod_{i=1}^{k} |\hat{u}_i(j)|. \tag{C.13}$$

We can transfer the norm constraint to

$$\|\hat{u}_1\|^2 + \cdots + \|\hat{u}_k\|^2 + \|\hat{w}\|^2 \leqslant p$$

by using Plancherel's theorem.

Therefore, we need to select them in a such way that $\theta_{u_1}(j) + \cdots + \theta_{u_k}(j) = \theta_w(j)$, ensuring that, for each j, the expression $\cos(\theta_{u_1}(j) + \cdots + \theta_{u_k}(j) - \theta_w(j)) = 1$ is maximized, except in cases where the scalar of the j-th term is 0.

This further simplifies the problem to:

$$\max_{\|\hat{u}_1\|^2 + \cdots + \|\hat{u}_k\|^2 + \|\hat{w}\|^2 \leqslant p} \frac{2(k!)}{(p-1)p^k} \sum_{j=1}^{(p-1)/2} |\hat{w}(j)| \prod_{i=1}^{k} |\hat{u}_i(j)|. \tag{C.14}$$

Then, we have

$$|\hat{w}(j)|\prod_{i=1}^{k}|\hat{u}_i(j)| \leqslant (\frac{1}{k+1}\cdot(|\hat{u}_1(j)|^2+\cdots+|\hat{u}_k(j)|^2+|\hat{w}(j)|^2))^{(k+1)/2}. \qquad \text{(C.15)}$$

where the first step follows from inequality of quadratic and geometric means.

We define $z:\{1,\dots,\frac{p-1}{2}\}\to\mathbb{R}$, where

$$z(j) := |\hat{u}_1(j)|^2+\cdots+|\hat{u}_k(j)|^2+|\hat{w}(j)|^2.$$

We need to have $\hat{u}_1(0)=\cdots=\hat{u}_k(0)=\hat{w}(0)=0$. Then, the upper-bound of Equation equation C.14 is given by

$$\begin{aligned}
&\frac{2(k!)}{(p-1)p^k}\cdot\max_{\|z\|_1\leqslant\frac{p}{2}}\sum_{j=1}^{(p-1)/2}(\frac{z(j)}{k+1})^{(k+1)/2}\\
=&\frac{2(k!)}{(k+1)^{(k+1)/2}(p-1)p^k}\cdot\max_{\|z\|_1\leqslant\frac{p}{2}}\sum_{j=1}^{(p-1)/2}z(j)^{(k+1)/2}\\
\leqslant&\frac{2(k!)}{(k+1)^{(k+1)/2}(p-1)p^k}\cdot(p/2)^{(k+1)/2}\\
=&\frac{2(k!)}{(2k+2)^{(k+1)/2}(p-1)p^{(k-1)/2}},
\end{aligned}$$

where the first step follows from simple algebra, the second step comes from the definition of $L_2$ norm, the third step follows from $\|z\|_2\leqslant\|z\|_1\leqslant\frac{p}{2}$, the last step follows from simple algebra.

For the inequality of quadratic and geometric means, Eq. equation C.15 becomes equality when $|\hat{u}_1(j)|=\cdots=|\hat{u}_k(j)|=|\hat{w}(j)|$. To achieve $\|z\|_2=\frac{p}{2}$, all the mass must be placed on a single frequency. Hence, for some frequency $\zeta\in\{1,\dots,\frac{p-1}{2}\}$, to achieve the upper bound, we have:

$$|\hat{u}_1(j)| = \cdots = |\hat{u}_k(j)| = |\hat{w}(j)| = \begin{cases} \sqrt{\frac{p}{2(k+1)}}, & \text{if } j = \pm\zeta; \\ 0, & \text{otherwise.} \end{cases} \tag{C.16}$$

In this case, Equation equation C.14 matches the upper bound. Hence, this is the maximum-margin.

Let $\theta^*_{u_1} := \theta_{u_1}(\zeta)$. Combining all the results, up to scaling, it is established that all neurons which maximize the expected class-weighted margin conform to the form:

$$\begin{aligned} u_1(a_1) &= \frac{1}{p}\sum_{j=0}^{p-1} \hat{u}_1(j)\rho^{j a_1} \\ &= \frac{1}{p} \cdot (\hat{u}_1(\zeta)\rho^{\zeta a_1} + \hat{u}_1(-\zeta)\rho^{-\zeta a_1}) \\ &= \frac{1}{p} \cdot (\sqrt{\frac{p}{2(k+1)}} \exp(\mathbf{i}\theta^*_{u_1})\rho^{\zeta a_1} + \sqrt{\frac{p}{2(k+1)}} \exp(-\mathbf{i}\theta^*_{u_1})\rho^{-\zeta a_1}) \\ &= \sqrt{\frac{2}{(k+1)p}} \cos(\theta^*_{u_1} + 2\pi\zeta a_1/p), \end{aligned}$$

where the first step comes from the definition of $u_1(a)$, the second step and third step follow from Eq. equation C.16, the last step follows from Eq. equation C.12 i.e., Euler's formula.

We have similar results for other neurons where $\theta^*_{u_1}, \ldots, \theta^*_{u_k}, \theta^*_w \in \mathbb{R}$ satisfying $\theta^*_{u_1} + \cdots + \theta^*_{u_k} = \theta^*_w$ and some $\zeta \in \mathbb{Z}_p \backslash \{0\}$, where $u_1, \ldots, u_k$, and $w$ shares the same $\zeta$.

□

## C.7 Construct Max Margin Solution

Section C.7.1 proposed the sum-to-product identities for k inputs. Section C.7.2 shows how we construct $\theta^*$ when $k = 3$. Section C.7.3 gives the constructions for

$\theta^*$ for general $k$ version.

### C.7.1 Sum-to-product Identities

**Lemma C.16** (Sum-to-product Identities). *If the following conditions hold*

- *Let* $a_1, \ldots, a_k$ *denote any* $k$ *real numbers*

*We have*

- ***Part 1.***

$$2^2 \cdot 2! \cdot a_1 a_2 = (a_1 + a_2)^2 - (a_1 - a_2)^2 - (-a_1 + a_2)^2 + (-a_1 - a_2)^2$$

- ***Part 2.***

$$\begin{aligned}&2^3 \cdot 3! \cdot a_1 a_2 a_3\\ &= (a_1 + a_2 + a_3)^3 - (a_1 + a_2 - a_3)^3 - (a_1 - a_2 + a_3)^3 - (-a_1 + a_2 + a_3)^3\\ &+ (a_1 - a_2 - a_3)^3 + (-a_1 + a_2 - a_3)^3 + (-a_1 - a_2 + a_3)^3 - (-a_1 - a_2 - a_3)^3\end{aligned}$$

- ***Part 3.***

$$2^k \cdot k! \cdot \prod_{i=1}^k a_i = \sum_{c \in \{-1,+1\}^k} (-1)^{(k - \sum_{i=1}^k c_i)/2} (\sum_{j=1}^k c_j a_j)^k.$$

*Proof.* **Proof of Part 1.**

We define $A_1, A_2, A_3, A_4$ as follows

$$A_1 := \ (a_1 + a_2)^2, A_2 := \ (a_1 - a_2)^2, A_3 := \ (-a_1 + a_2)^2, A_4 := \ (-a_1 - a_2)^2,$$

For the first term, we have

$$A_1 = a_1^2 + a_2^2 + 2a_1 a_2.$$

For the second term, we have

$$A_2 = a_1^2 + a_2^2 - 2a_1a_2.$$

For the third term, we have

$$A_3 = a_1^2 + a_2^2 - 2a_1a_2.$$

For the fourth term, we have

$$A_4 = a_1^2 + a_2^2 + 2a_1a_2.$$

Putting things together, we have

$$\begin{aligned}(a_1 + a_2)^2 - (a_1 - a_2)^2 - (-a_1 + a_2)^2 + (-a_1 - a_2)^2 &= A_1 - A_2 - A_3 + A_4 \\ &= 8a_1a_2 \\ &= 2^3a_1a_2\end{aligned}$$

**Proof of Part 2.**

We define $B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8$ as follows

$$\begin{aligned}B_1 &:= (a_1 + a_2 + a_3)^3, \\ B_2 &:= (a_1 + a_2 - a_3)^3, \\ B_3 &:= (a_1 - a_2 + a_3)^3, \\ B_4 &:= (-a_1 + a_2 + a_3)^3, \\ B_5 &:= (a_1 - a_2 - a_3)^3, \\ B_6 &:= (-a_1 + a_2 - a_3)^3, \\ B_7 &:= (-a_1 - a_2 + a_3)^3, \\ B_8 &:= (-a_1 - a_2 - a_3)^3,\end{aligned}$$

For the first term, we have

$$B_1 = a_1^3 + a_2^3 + a_3^3 + 3a_2a_3^2 + 3a_2a_1^2 + 3a_1a_3^2 + 3a_1a_2^2 + 3a_3a_1^2 + 3a_3a_2^2 + 6a_1a_2a_3.$$

For the second term, we have

$$B_2 = a_1^3 + a_2^3 - a_3^3 + 3a_2a_3^2 + 3a_2a_1^2 + 3a_1a_3^2 + 3a_1a_2^2 - 3a_3a_1^2 - 3a_3a_2^2 - 6a_1a_2a_3.$$

For the third term, we have

$$B_3 = a_1^3 - a_2^3 + a_3^3 - 3a_2a_3^2 - 3a_2a_1^2 + 3a_1a_3^2 + 3a_1a_2^2 + 3a_3a_1^2 + 3a_3a_2^2 - 6a_1a_2a_3.$$

For the fourth term, we have

$$B_4 = -a_1^3 + a_2^3 + a_3^3 + 3a_2a_3^2 + 3a_2a_1^2 - 3a_1a_3^2 - 3a_1a_2^2 + 3a_3a_1^2 + 3a_3a_2^2 - 6a_1a_2a_3.$$

For the fifth term, we have

$$B_5 = a_1^3 - a_2^3 - a_3^3 - 3a_2a_3^2 - 3a_2a_1^2 + 3a_1a_3^2 + 3a_1a_2^2 - 3a_3a_1^2 - 3a_3a_2^2 + 6a_1a_2a_3.$$

For the sixth term, we have

$$B_6 = -a_1^3 + a_2^3 - a_3^3 + 3a_2a_3^2 + 3a_2a_1^2 - 3a_1a_3^2 - 3a_1a_2^2 - 3a_3a_1^2 - 3a_3a_2^2 + 6a_1a_2a_3.$$

For the seventh term, we have

$$B_7 = -a_1^3 - a_2^3 + a_3^3 - 3a_2a_3^2 - 3a_2a_1^2 - 3a_1a_3^2 - 3a_1a_2^2 + 3a_3a_1^2 + 3a_3a_2^2 + 6a_1a_2a_3.$$

For the eighth term, we have

$$B_8 = -a_1^3 - a_2^3 - a_3^3 - 3a_2a_3^2 - 3a_2a_1^2 - 3a_1a_3^2 - 3a_1a_2^2 - 3a_3a_1^2 - 3a_3a_2^2 - 6a_1a_2a_3.$$

Putting things together, we have

$$\begin{aligned}&(a_1+a_2+a_3)^3-(a_1+a_2-a_3)^3-(a_1-a_2+a_3)^3-(-a_1+a_2+a_3)^3\\&+(a_1-a_2-a_3)^3+(-a_1+a_2-a_3)^3+(-a_1-a_2+a_3)^3-(-a_1-a_2-a_3)^3\\=&B_1-B_2-B_3-B_4+B_5+B_6+B_7-B_8\\=&48a_1a_2a_3\\=&3\cdot 2^4a_1a_2a_3\end{aligned}$$

**Proof of Part 3.**

$$2^k\cdot k!\cdot\prod_{i=1}^k a_i=\sum_{c\in\{-1,+1\}^k}(-1)^{(k-\sum_{i=1}^k c_i)/2}(\sum_{j=1}^k c_ja_j)^k.$$

We first let $a_1=0$. Then each term on RHS can find a corresponding negative copy of this term. In detail, let $c_1$ change sign and we have, $(-1)^{(k-c_1-\sum_{i=2}^k c_i)/2}(c_1\cdot 0+\sum_{j=2}^k c_ja_j)^k=-(-1)^{(k+c_1-\sum_{i=2}^k c_i)/2}(-c_1\cdot 0+\sum_{j=2}^k c_ja_j)^k$. We can find this mapping is always one-to-one and onto mapping with each other. Thus, we have RHS is constant 0 regardless of $a_2,\ldots,a_k$. Thus, $a_1$ is a factor of RHS. By symmetry, $a_2,\ldots,a_k$ also are factors of RHS. Since RHS is $k$-th order, we have RHS$=\alpha\prod_{i=1}^k a_i$ where $\alpha$ is a constant. Take $a_1=\cdots=a_k=1$, we have $\alpha=2^k\cdot k!=$RHS. Thus, we finish the proof. □

## C.7.2 Constructions for $\theta^*$

**Lemma C.17.** *When $k=3$, provided the following conditions are met*

- *We denote $\mathcal{B}$ as the ball that $\|u_1\|^2+\|u_2\|^2+\|u_3\|^2+\|w\|^2\leqslant 1$.*
- *We define $\Omega_q^{'*}$ in Definition C.9.*
- *We adopt the uniform class weighting: $\forall c'\neq a_1+a_2+a_3,\ \ \tau(a_1,a_2,a_3)[c']:=1/(p-1)$.*

- *Let* $\cos_\zeta(x)$ *denote* $\cos(2\pi\zeta x/p)$
- *Let* $\sin_\zeta(x)$ *denote* $\sin(2\pi\zeta x/p)$

*Then, we have*

- *The maximum* $L_{2,4}$*-margin solution* $\theta^*$ *will consist of* $16(p-1)$ *neurons* $\theta_i^* \in \Omega_q^{'*}$ *to simulate* $\frac{p-1}{2}$ *type of cosine computation, each cosine computation is uniquely determined a* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$*. In particular, for each* $\zeta$ *the cosine computation is* $\cos_\zeta(a_1 + a_2 + a_3 - c), \forall a_1, a_2, a_3, c \in \mathbb{Z}_p$.

*Proof.* Referencing Lemma C.11, we can identify elements within $\Omega_q^{'}$. Our set $\theta^*$ will be composed of $16(p-1)$ neurons, including 32 neurons dedicated to each frequency in the range $1, \dots, \frac{p-1}{2}$. Focusing on a specific frequency $\zeta$, for the sake of simplicity, let us use $\cos_\zeta(x)$ to represent $\cos(2\pi\zeta x/p)$ and $\sin_\zeta(x)$ likewise. We note:

$$\begin{aligned}
&\cos_\zeta(a_1 + a_2 + a_3 - c) && \text{(C.17)}\\
=&\cos_\zeta(a_1 + a_2 + a_3)\cos_\zeta(c) + \sin_\zeta(a_1 + a_2 + a_3)\sin_\zeta(c)\\
=&\cos_\zeta(a_1 + a_2)\cos_\zeta(a_3)\cos_\zeta(c) - \sin_\zeta(a_1 + a_2)\sin_\zeta(a_3)\cos_\zeta(c)\\
&+ \sin_\zeta(a_1 + a_2)\cos_\zeta(a_3)\sin_\zeta(c) + \cos_\zeta(a_1 + a_2)\sin_\zeta(a_3)\sin_\zeta(c)\\
=&(\cos_\zeta(a_1)\cos_\zeta(a_2) - \sin_\zeta(a_1)\sin_\zeta(a_2))\cos_\zeta(a_3)\cos_\zeta(c)\\
&- (\sin_\zeta(a_1)\cos_\zeta(a_2) + \cos_\zeta(a_1)\sin_\zeta(a_2))\sin_\zeta(a_3)\cos_\zeta(c)\\
&+ (\sin_\zeta(a_1)\cos_\zeta(a_2) + \cos_\zeta(a_1)\sin_\zeta(a_2))\cos_\zeta(a_3)\sin_\zeta(c)\\
&+ ((\cos_\zeta(a_1)\cos_\zeta(a_2) - \sin_\zeta(a_1)\sin_\zeta(a_2)))\sin_\zeta(a_3)\sin_\zeta(c)\\
=&\cos_\zeta(a_1)\cos_\zeta(a_2)\cos_\zeta(a_3)\cos_\zeta(c) - \sin_\zeta(a_1)\sin_\zeta(a_2)\cos_\zeta(a_3)\cos_\zeta(c)\\
&- \sin_\zeta(a_1)\cos_\zeta(a_2)\sin_\zeta(a_3)\cos_\zeta(c) - \cos_\zeta(a_1)\sin_\zeta(a_2)\sin_\zeta(a_3)\cos_\zeta(c)\\
&+ \sin_\zeta(a_1)\cos_\zeta(a_2)\cos_\zeta(a_3)\sin_\zeta(c) + \cos_\zeta(a_1)\sin_\zeta(a_2)\cos_\zeta(a_3)\sin_\zeta(c)\\
&+ \cos_\zeta(a_1)\cos_\zeta(a_2)\sin_\zeta(a_3)\sin_\zeta(c) - \sin_\zeta(a_1)\sin_\zeta(a_2)\sin_\zeta(a_3)\sin_\zeta(c) && \text{(C.18)}
\end{aligned}$$

where all steps comes from trigonometric function.

Each of these 8 terms can be implemented by 4 neurons $\phi_1, \phi_2, \cdots, \phi_4$. Consider the first term, $\cos_\zeta(a_1)\cos_\zeta(a_2)\cos_\zeta(a_3)\cos_\zeta(c)$.

For the $i$-th neuron, we have

$$\phi_i = (u_{i,1}(a_1) + u_{i,2}(a_2) + u_{i,3}(a_3))^3 \cdot w_i(c).$$

By changing $(\theta_{i,j})^*$, we can change the constant factor of $\cos_\zeta(\cdot)$ to be $+\beta$ or $-\beta$. Hence, we can view $u_{i,j}(\cdot), w_i(\cdot)$ as the following:

$$\begin{aligned} u_{i,1}(\cdot) &:= p_{i,1} \cdot \cos_\zeta(\cdot), \\ u_{i,2}(\cdot) &:= p_{i,2} \cdot \cos_\zeta(\cdot), \\ u_{i,3}(\cdot) &:= p_{i,3} \cdot \cos_\zeta(\cdot), \\ w_i(\cdot) &:= p_{i,4} \cdot \cos_\zeta(\cdot) \end{aligned}$$

where $p_{i,j} \in \{-1, 1\}$.

For simplicity, let $d_i$ denote $\cos_\zeta(a_i)$.

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w) = (0, 0, 0, 0)$, then

$$p_{1,1}, p_{1,2}, p_{1,3}, p_{1,4} = 1,$$

then we have

$$\phi_1 = (d_1 + d_2 + d_3)^3 \cos_\zeta(c).$$

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w) = (0, 0, \pi, \pi)$, then $p_{2,1}, p_{2,2} = 1$ and $p_{2,3}, p_{2,4} = -1$, then we have

$$\phi_2 = -(d_1 + d_2 - d_3)^3 \cos_\zeta(c).$$

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w) = (0, \pi, 0, \pi)$, then $p_{3,1}, p_{3,3} = 1$ and $p_{3,2}, p_{3,4} = -1$, then we have

$$\phi_3 = -(d_1 - d_2 + d_3)^3 \cos_\zeta(c).$$

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_{w}) = (\pi, 0, 0, \pi)$, then $p_{4,1}, p_{4,4} = -1$ and $p_{2,2}, p_{2,3} = 1$, then we have

$$\phi_4 = -(-d_1 + d_2 + d_3)^3 \cos_\zeta(c).$$

Putting them together, we have

$$\begin{aligned}
&\sum_{i=1}^{4} \phi_i(a_1, a_2, a_3) \\
= &\sum_{i=1}^{4} (u_{i,1}(a_1) + u_{i,2}(a_2) + u_{i,3}(a_3))^3 w_i(c) \\
= &\sum_{i=1}^{4} (p_{i,1} \cos_\zeta(a_1) + p_{i,2} \cos_\zeta(a_2) + p_{i,3} \cos_\zeta(a_3))^3 w_i(c) \\
= &[(d_1 + d_2 + d_3)^3 - (d_1 + d_2 - d_3)^3 - (d_1 - d_2 + d_3)^3 - (-d_1 + d_2 + d_3)^3] \cos_\zeta(c) \\
= &24 d_1 d_2 d_3 \cos_\zeta(c) \\
= &24 \cos_\zeta(a_1) \cos_\zeta(a_2) \cos_\zeta(a_3) \cos_\zeta(c) \qquad \text{(C.19)}
\end{aligned}$$

where the first step comes from the definition of $\phi_i$, the second step comes from the definition of $u_{i,j}$, the third step comes from $d_i = \cos_\zeta(a_i)$, the fourth step comes from simple algebra, the last step comes from $d_i = \cos_\zeta(a_i)$.

Similarly, consider $-\sin_\zeta(a_1) \sin_\zeta(a_2) \cos_\zeta(a_3) \cos_\zeta(c)$.

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_{w}) = (\pi/2, \pi/2, 0, \pi)$, then we have

$$\phi_1 = -(\sin_\zeta(a_1) + \sin_\zeta(a_2) + \cos_\zeta(a_3))^3 \cos_\zeta(c).$$

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w) = (\pi/2, \pi/2, -\pi, 0)$, then we have

$$\phi_2 = (\sin_\zeta(a_1) + \sin_\zeta(a_2) - \cos_\zeta(a_3))^3 \cos_\zeta(c).$$

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w) = (\pi/2, -\pi/2, 0, 0)$, then we have

$$\phi_3 = (\sin_\zeta(a_1) - \sin_\zeta(a_2) + \cos_\zeta(a_3))^3 \cos_\zeta(c).$$

We set $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w) = (-\pi/2, \pi/2, 0, 0)$, then we have

$$\phi_4 = (-\sin_\zeta(a_1) + \sin_\zeta(a_2) + \cos_\zeta(a_3))^3 \cos_\zeta(c).$$

Putting them together, we have

$$\begin{aligned} &\sum_{i=1}^{4} \phi_i(a_1, a_2, a_3) \\ = & -24 \sin_\zeta(a_1) \sin_\zeta(a_2) \cos_\zeta(a_3) \cos_\zeta(c) \end{aligned} \tag{C.20}$$

Similarly, all other six terms in Eq. equation C.17 can be composed by four neurons with different $(\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_w)$.

When we include such 56 neurons for all frequencies $\zeta \in \{1, \ldots, \frac{p-1}{2}\}$, we have that the network will calculate the following function

$$\begin{aligned} f(a_1, a_2, a_3, c) &= \sum_{\zeta=1}^{(p-1)/2} \cos_\zeta(a_1 + a_2 + a_3 - c) \\ &= \sum_{\zeta=1}^{p-1} \frac{1}{2} \cdot \exp(2\pi \mathbf{i} \zeta(a_1 + a_2 + a_3 - c)/p) \\ &= \begin{cases} \frac{p-1}{2} & \text{if } a_1 + a_2 + a_3 = c \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where the first step comes from the definition of $f(a_1, a_2, a_3, c)$, the second step comes from Euler's formula, the last step comes from the properties of discrete Fourier transform.

The scaling factor $\beta$ for each neuron can be selected such that the entire network maintains an $L_{2,4}$-norm of 1. In this setup, every data point lies exactly on the margin, meaning $q = \mathrm{unif}(\mathbb{Z}_p)$ uniformly covers points on the margin, thus meeting the criteria for $q^*$ as outlined in Definition 4.5. Furthermore, for any input $(a_1, a_2, a_3)$, the function $f$ yields an identical result across all incorrect labels $c'$, adhering to Condition 4.8. □

### C.7.3 Constructions for $\theta^*$ for General $k$ Version

**Lemma C.18** (Formal version of Lemma 4.11)**.** *Provided the following conditions are met*

- *We denote $\mathcal{B}$ as the ball that $\|u_1\|^2 + \cdots + \|u_k\|^2 + \|w\|^2 \leqslant 1$.*
- *We define $\Omega_q^{'*}$ in Definition C.9.*
- *We adopt the uniform class weighting: $\forall c' \neq a_1 + \cdots + a_k,\ \ \tau(a_1, \ldots, a_k)[c'] := 1/(p-1)$.*
- *Let $\cos_\zeta(x)$ denote $\cos(2\pi\zeta x/p)$*
- *Let $\sin_\zeta(x)$ denote $\sin(2\pi\zeta x/p)$*

*Then, we have*

- *The maximum $L_{2,k+1}$-margin solution $\theta^*$ will consist of $2^{2k-1} \cdot \frac{p-1}{2}$ neurons $\theta_i^* \in \Omega_q^{'*}$ to simulate $\frac{p-1}{2}$ type of cosine computation, each cosine computation is uniquely determined a $\zeta \in \{1, \ldots, \frac{p-1}{2}\}$. In particular, for each $\zeta$ the cosine computation is $\cos_\zeta(a_1 + \cdots + a_k - c), \forall a_1, \ldots, a_k, c \in \mathbb{Z}_p$.*

*Proof.* By Lemma C.11, we can get elements of $\Omega_q^{'*}$. Our set $\theta^*$ will be composed of $2^{2k-1} \cdot \frac{p-1}{2}$ neurons, including $2^{2k-1}$ neurons dedicated to each frequency in the range $1, \dots, \frac{p-1}{2}$. Focusing on a specific frequency $\zeta$, for the sake of simplicity, let us use $\cos_\zeta(x)$ to represent $\cos(2\pi\zeta x/p)$ and $\sin_\zeta(x)$ likewise.

We define

$$a_{[k]} := \sum_{i=1}^{k} a_k$$

and we also define

$$a_{k+1} := -c.$$

For easy of writing, we will write $\cos_\zeta$ as cos and $\sin_\zeta$ as sin. We have the following.

$$\begin{aligned}
& \cos_\zeta(\sum_{i=1}^{k} a_i - c) \\
= & \cos(\sum_{i=1}^{k} a_i - c) \\
= & \cos(a_{[k+1]}) \\
= & \cos(a_{[k]})\cos(a_{k+1}) - \sin(a_{[k]})\sin(a_{k+1}) \\
= & \cos(a_{[k-1]} + a_k)\cos(a_{k+1}) - \sin(a_{[k-1]} + a_k)\sin(a_{k+1}) \\
= & \cos(a_{[k-1]})\cos(a_k)\cos(a_{k+1}) - \sin(a_{[k-1]})\sin(a_k)\cos(a_{k+1}) \\
& - \sin(a_{[k-1]})\cos(a_k)\sin(a_{k+1}) - \cos(a_{[k-1]})\sin(a_k)\sin(a_{k+1}) \\
= & \sum_{b\in\{0,1\}^{k+1}} \prod_{i=1}^{k+1} \cos^{1-b_i}(a_i)\cdot\sin^{b_i}(a_i)\cdot\mathbf{1}[\sum_{i=1}^{k+1} b_i\%2 = 0]\cdot(-1)^{\mathbf{1}[\sum_{i=1}^{k+1} b_i\%4=2]},
\end{aligned} \tag{C.21}$$

where the first step comes from the simplicity of writing, the second step comes from the definition of $a_{[k+1]}$ and $a_{k+1}$, the third step comes from the trigonometric function, the fourth step also follows trigonometric function, and the last step comes from the below two observations:

- First, we observe that $\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b)$ and $\sin(a+b) = \sin(a)\cos(b) + \cos(a)\sin(b)$. When we split cos once, we will remove one cos product and we may add zero or two sin products. When we split sin once, we may remove one sin product and we will add one sin product as well. Thus, we can observe that the number of sin products in each term is always even.
- Second, we observe only when we split cos and add two sin products will introduce a $-1$ is this term. Thus, when the number of sin products $\%4 = 2$, the sign of this term will be $-1$. Otherwise, it will be $+1$.

Note that we have $2^k$ non-zero term in Eq. equation C.21. Each of these $2^k$ terms can be implemented by $2^{k-1}$ neurons $\phi_1, \cdots, \phi_{2^{k-1}}$.

For the $i$-th neuron, we have

$$\phi_i = (\sum_{j=1}^{k} u_{i,j}(a_j))^k \cdot w_i(c).$$

By changing $(\theta_{i,j})^*$, we can change the $u_{i,j}(a_j)$ from $\cos_\zeta(\cdot)$ to be $-\cos_\zeta(\cdot)$ or $\sin_\zeta(\cdot)$ or $-\sin_\zeta(\cdot)$. Denote $\theta^*_{u_i}$ as $(\theta_{i,a_i})^*$.

For simplicity, let $d_i$ denote the $i$-th product in one term of Eq. equation C.21. By fact that

$$2^k \cdot k! \cdot \prod_{i=1}^{k} d_i = \sum_{c \in \{-1,+1\}^k} (-1)^{(k - \sum_{i=1}^{k} c_i)/2} (\sum_{j=1}^{k} c_j d_j)^k,$$

each term can be constructed by $2^{k-1}$ neurons (note that there is a symmetric effect so we only need half terms). Based on Eq. equation C.21 and the above fact with carefully check, we can see that $\theta^*_{u_1} + \cdots + \theta^*_{u_k} = \theta^*_w$. Thus, we need $2^k \cdot 2^{k-1} \cdot \frac{p-1}{2}$ neurons in total.

When we include such $2^k \cdot 2^{k-1}$ neurons for all frequencies $\zeta \in \{1, \ldots, \frac{p-1}{2}\}$, we have the network will calculate the following function

$$\begin{aligned} f(a_1, \dots, a_k, c) &= \sum_{\zeta=1}^{(p-1)/2} \cos_\zeta(\sum_{i=1}^k a_i - c) \\ &= \sum_{\zeta=1}^{p-1} \frac{1}{2} \cdot \exp(2\pi \mathbf{i} \zeta (\sum_{i=1}^k a_i - c)/p) \\ &= \begin{cases} \frac{p-1}{2} & \text{if } \sum_{i=1}^k a_i = c \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

where the first step comes from the definition of $f(a_1, \dots, a_k, c)$, the second step comes from Euler's formula, the last step comes from the properties of discrete Fourier transform.

The scaling parameter $\beta$ for each neuron can be adjusted to ensure that the network possesses an $L_{2,k+1}$-norm of 1. For this network, all data points are positioned on the margin, which implies that $q = \mathrm{unif}(\mathbb{Z}_p)$ naturally supports points along the margin, aligning with the requirements for $q^*$ presented in Definition 4.5. Additionally, for every input $(a_1, \dots, a_k)$, the function $f$ assigns the same outcome to all incorrect labels $c'$, thereby fulfilling Condition 4.8. □

## C.8 Check Fourier Frequencies

Section C.8.1 proves all frequencies are used. Section C.8.2 proves all frequencies are used for general $k$ version.

### C.8.1 All Frequencies are Used

Let $f : \mathbb{Z}_p^4 \to \mathbb{C}$. Its multi-dimensional discrete Fourier transform is defined as:

$$\hat{f}(j_1, j_2, j_3, j_4)$$

$$:= \sum_{a_1 \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_1 a_1 / p} ( \sum_{a_2 \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_2 a_2 / p} ( \sum_{a_3 \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_3 a_3 / p} ( \sum_{c \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_4 c / p} f(a_1, a_2, a_3, c))).$$

**Lemma C.19.** *When $k = 3$, if the following conditions hold*

- *We adopt the uniform class weighting:* $\forall c' \neq a_1 + a_2 + a_3, \;\; \tau(a_1, a_2, a_3)[c'] := 1/(p-1)$.
- $f$ *is the maximum* $L_{2,4}$*-margin solution.*

*Then, for any* $j_1 = j_2 = j_3 = -j_4 \neq 0$*, we have* $\hat{f}(j_1, j_2, j_3, j_4) > 0$.

*Proof.* In this proof, let $j_1, j_2, j_3, j_4 \in \mathbb{Z}$, and $\theta_u = \theta_u^* \cdot \frac{p}{2\pi}$ to simplify the notation. By Lemma C.11,

$$u_1(a_1) = \sqrt{\frac{1}{2p}} \cos_p(\theta_{u_1} + \zeta a_1). \tag{C.22}$$

Let

$$\begin{aligned}
& f(a_1, a_2, a_3, c) \\
= & \sum_{h=1}^{H} \phi_h(a_1, a_2, a_3, c) \\
= & \sum_{h=1}^{H} (u_{h,1}(a_1) + u_{h,2}(a_2) + u_{h,3}(a_3))^3 w_h(c) \\
= & (\frac{1}{2p})^2 \sum_{h=1}^{H} (\cos_p(\theta_{u_{h,1}} + \zeta_h a_1) + \cos_p(\theta_{u_{h,2}} + \zeta_h a_2) + \cos_p(\theta_{u_{h,3}} + \zeta_h a_3))^3 \cos_p(\theta_{w_h} + \zeta_h c)
\end{aligned}$$

where each neuron conforms to the previously established form, and the width H function is an arbitrary margin-maximizing network. The first step is from the definition of $f(a_1, a_2, a_3, c)$, the subsequent step on the definition of $\phi_h(a_1, a_2, a_3, c)$, and the final step is justified by Eq. equation C.22.

We can divide each $\phi$ into ten terms:

$$\begin{aligned}
&\phi(a_1, a_2, a_3, c) \\
=\ &\phi^{(1)}(a_1, a_2, a_3, c) + \cdots + \phi^{(10)}(a_1, a_2, a_3, c) \\
=\ &\big(u_1(a_1)^3 + u_2(a_2)^3 + u_3(a_3)^3 + 3u_1(a_1)^2 u_2(a_2) + 3u_1(a_1)^2 u_3(a_3) + 3u_2(a_2)^2 u_1(a_1) \\
&+ 3u_2(a_2)^2 u_3(a_3) + 3u_3(a_3)^2 u_1(a_1) + 3u_3(a_3)^2 u_2(a_2) + 6u_1(a_1)u_2(a_2)u_3(a_3)\big)w(c).
\end{aligned}$$

Note, $\rho = e^{2\pi i/p}$. $\widehat{\phi}_1(j_1, j_2, j_3, j_4)$ is nonzero only for $j_1 = 0$, and $\widehat{\phi}_4(j_1, j_2, j_3, j_4)$ is nonzero only for $j_1 = j_2 = 0$. Similar to other terms. For the tenth term, we have

$$\begin{aligned}
\widehat{\phi}_{10}(j_1, j_2, j_3, j_4) &= 6 \sum_{a_1, a_2, a_3, c \in \mathbb{Z}_p} u_1(a_1)u_2(a_2)u_3(a_3)w(c)\rho^{-(j_1 a_1 + j_2 a_2 + j_3 a_3 + j_4 c)} \\
&= 6\hat{u}_1(j_1)\hat{u}_2(j_2)\hat{u}_3(j_3)\hat{w}(j_4).
\end{aligned}$$

In particular,

$$\begin{aligned}
\hat{u}_1(j_1) &= \sum_{a_1 \in \mathbb{Z}_p} \sqrt{\frac{1}{2p}} \cos_p(\theta_{u_1} + \zeta a_1)\rho^{-j_1 a_1} \\
&= (8p)^{-1/2} \sum_{a_1 \in \mathbb{Z}_p} (\rho^{\theta_{u_1} + \zeta a_1} + \rho^{-(\theta_{u_1} + \zeta a_1)})\rho^{-j_1 a_1} \\
&= (8p)^{-1/2}(\rho^{\theta_{u_1}} \sum_{a_1 \in \mathbb{Z}_p} \rho^{(\zeta - j_1)a_1} + \rho^{-\theta_{u_1}} \sum_{a_1 \in \mathbb{Z}_p} \rho^{-(\zeta + j_1)a_1}) \\
&= \begin{cases} \sqrt{p/8} \cdot \rho^{\theta_{u_1}} & \text{if } j_1 = +\zeta \\ \sqrt{p/8} \cdot \rho^{-\theta_{u_1}} & \text{if } j_1 = -\zeta \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

where the first step comes from $\hat{u}_1(j_1)$ definition, the second step comes from Euler's formula, the third step comes from simple algebra, the last step comes from the properties of discrete Fourier transform. Similarly for $\hat{u}_2, \hat{u}_3$ and $\hat{w}$. As we consider

$\zeta$ to be nonzero, we ignore the $\zeta = 0$ case. Hence, $\widehat{\phi}_{10}(j_1, j_2, j_3, j_4)$ is nonzero only when $j_1, j_2, j_3, j_4$ are all $\pm\zeta$. We can summarize that $\hat{\phi}(j_1, j_2, j_3, j_4)$ can only be nonzero if one of the following satisfies:

- $j_1 \cdot j_2 \cdot j_3 = 0$
- $j_1, j_2, j_3, j_4 = \pm\zeta$.

Setting aside the previously discussed points, it's established in Lemma C.6 that the function $f$ maintains a consistent margin for various inputs as well as over different classes, i.e., $f$ can be broken down as

$$f(a_1, a_2, a_3, c) = f_1(a_1, a_2, a_3, c) + f_2(a_1, a_2, a_3, c)$$

where

$$f_1(a_1, a_2, a_3, c) = F(a_1, a_2, a_3)$$

for some $F : \mathbb{Z}_p \times \mathbb{Z}_p \times \mathbb{Z}_p \to \mathbb{R}$, and

$$f_2(a_1, a_2, a_3, c) = \lambda \cdot \mathbf{1}_{a_1+a_2+a_3=c}$$

where $\lambda > 0$ is the margin of $f$. Then, we have the DFT of $f_1$ and $f_2$ are

$$\hat{f}_1(j_1, j_2, j_3, j_4) = \begin{cases} \hat{F}(j_1, j_2, j_3) & \text{if } j_4 = 0 \\ 0 & \text{otherwise} \end{cases}$$

and

$$\hat{f}_2(j_1, j_2, j_3, j_4) = \begin{cases} \lambda p^3 & \text{if } j_1 = j_2 = j_3 = -j_4 \\ 0 & \text{otherwise} \end{cases}.$$

Hence, when $j_1 = j_2 = j_3 = -j_4 \neq 0$, we must have $\hat{f}(j_1, j_2, j_3, j_4) > 0$. □

### C.8.2 All Frequencies are Used for General $k$ Version

Let $f : \mathbb{Z}_p^{k+1} \to \mathbb{C}$. Its multi-dimensional discrete Fourier transform is defined as:

$$\begin{aligned} &\hat{f}(j_1, \ldots, j_{k+1}) \\ :=& \sum_{a_1 \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_1 a_1/p}(\ldots(\sum_{a_k \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_k a_k/p}(\sum_{c \in \mathbb{Z}_p} e^{-2\pi \mathbf{i} \cdot j_{k+1} c/p} f(a_1, \ldots, a_k, c))). \end{aligned}$$

**Lemma C.20.** *If the following conditions hold*

- *We adopt the uniform class weighting:* $\forall c' \neq a_1 + \cdots + a_k,\ \ \tau(a_1, \ldots, a_k)[c'] := 1/(p-1)$.
- $f$ *is the maximum* $L_{2,k+1}$*-margin solution.*

*Then, for any* $j_1 = \cdots = j_k = -j_{k+1} \neq 0$*, we have* $\hat{f}(j_1, \ldots, j_{k+1}) > 0$.

*Proof.* For this proof, for all $j_1, \ldots, j_{k+1} \in \mathbb{Z}$, to simplify the notation, let $\theta_u = \theta_u^* \cdot \frac{p}{2\pi}$, by Lemma C.15, so

$$u_1(a_1) = \sqrt{\frac{2}{(k+1)p}} \cos_p(\theta_{u_1} + \zeta a_1). \tag{C.23}$$

Let

$f(a_1, \ldots, a_k, c)$

$$
\begin{aligned}
&= \sum_{h=1}^{H} \phi_h(a_1, \dots, a_k, c) \\
&= \sum_{h=1}^{H} (u_{h,1}(a_1) + \cdots + u_{h,k}(a_k))^k w_h(c) \\
&= (\frac{2}{(k+1)p})^{(k+1)/2} \sum_{h=1}^{H} (\cos_p(\theta_{u_{h,1}} + \zeta_h a_1) + \cdots + \cos_p(\theta_{u_{h,k}} + \zeta_h a_k))^k \cos_p(\theta_{w_h} + \zeta_h c)
\end{aligned}
$$

where each neuron conforms to the previously established form, and the width H function is an arbitrary margin-maximizing network. The first step is based on the definition of $f(a_1, \dots, a_k, c)$, the subsequent step on the definition of $\phi_h(a_1, \dots, a_k, c)$, and the final step is justified by Eq. equation C.23.

Each neuron $\phi$ we have

$$
\begin{aligned}
\widehat{\phi}(j_1, \dots, j_k, j_{k+1}) &= k! \sum_{a_1, \dots, a_k, c \in \mathbb{Z}_p} w(c) \rho^{-(j_1 a_1 + \cdots + j_k a_k + j_{k+1} c)} \prod_{i=1}^{k} u_i(a_i) \\
&= k! \hat{w}(j_{k+1}) \prod_{i=1}^{k} \hat{u}_i(j_i).
\end{aligned}
$$

In particular,

$$
\begin{aligned}
\hat{u}_1(j_1) &= \sum_{a_1 \in \mathbb{Z}_p} \sqrt{\frac{2}{(k+1)p}} \cos_p(\theta_{u_1} + \zeta a_1) \rho^{-j_1 a_1} \\
&= \sqrt{\frac{1}{2(k+1)p}} \sum_{a_1 \in \mathbb{Z}_p} (\rho^{\theta_{u_1} + \zeta a_1} + \rho^{-(\theta_{u_1} + \zeta a_1)}) \rho^{-j_1 a_1} \\
&= \sqrt{\frac{1}{2(k+1)p}} (\rho^{\theta_{u_1}} \sum_{a_1 \in \mathbb{Z}_p} \rho^{(\zeta - j_1) a_1} + \rho^{-\theta_{u_1}} \sum_{a_1 \in \mathbb{Z}_p} \rho^{-(\zeta + j_1) a_1})
\end{aligned}
$$

$$= \begin{cases} \sqrt{\frac{p}{2(k+1)}} \cdot \rho^{\theta_{u_1}} & \text{if } j_1 = +\zeta \\ \sqrt{\frac{p}{2(k+1)}} \cdot \rho^{-\theta_{u_1}} & \text{if } j_1 = -\zeta \\ 0 & \text{otherwise,} \end{cases}$$

where the first step comes from $\hat{u}_1(j_1)$ definition, the second step comes from Euler's formula, the third step comes from simple algebra, the last step comes from the properties of discrete Fourier transform. Similarly for $\hat{u}_i$ and $\hat{w}$. We consider $\zeta$ to be nonzero, so we ignore the $\zeta = 0$ case. Hence, $\widehat{\phi}(j_1, \dots, j_k, j_{k+1})$ is nonzero only when $j_1, \dots, j_k, j_{k+1}$ are all $\pm\zeta$. We can summarize that $\hat{\phi}(j_1, \dots, j_k, j_{k+1})$ can only be nonzero if one of the below conditions satisfies:

- $\prod_{i=1}^{k} j_i = 0$
- $j_1, \dots, j_k, j_{k+1} = \pm\zeta$.

Setting aside the previously discussed points, it's established in Lemma C.6 that the function $f$ maintains a consistent margin for various inputs as well as over different classes, i.e., $f$ can be broken down as

$$f(a_1, \dots, a_k, c) = f_1(a_1, \dots, a_k, c) + f_2(a_1, \dots, a_k, c)$$

where

$$f_1(a_1, \dots, a_k, c) = F(a_1, \dots, a_k)$$

for some $F : \mathbb{Z}_p^k \to \mathbb{R}$, and

$$f_2(a_1, \dots, a_k, c) = \lambda \cdot \mathbf{1}_{a_1 + \cdots + a_k = c}$$

where $\lambda > 0$ is the margin of $f$. Then, we have the DFT of $f_1$ and $f_2$ are

$$\hat{f}_1(j_1,\ldots,j_k,j_{k+1}) = \begin{cases} \hat{F}(j_1,\ldots,j_k) & \text{if } j_{k+1} = 0 \\ 0 & \text{otherwise} \end{cases}$$

and

$$\hat{f}_2(j_1,\ldots,j_k,j_{k+1}) = \begin{cases} \lambda p^k & \text{if } j_1 = \cdots = j_k = -j_{k+1} \\ 0 & \text{otherwise} \end{cases}.$$

Hence, when $j_1 = \cdots = j_k = -j_{k+1} \neq 0$, we must have $\hat{f}(j_1,\ldots,j_k,j_{k+1}) > 0$. □

## C.9 Main Result

Section C.9.1 proves the main result for $k = 3$. Section C.9.2 proves the general $k$ version of our main result.

### C.9.1 Main result for $k = 3$

**Theorem C.21.** *When $k = 3$, let $f(\theta, x)$ be the one-hidden layer networks defined in Section 4.3. If the following conditions hold*

- *We adopt the uniform class weighting:* $\forall c' \neq a_1 + a_2 + a_3,\ \ \tau(a_1, a_2, a_3)[c'] := 1/(p-1)$.
- $m \geqslant 16(p-1)$ *neurons.*

*Then we have the maximum $L_{2,4}$-margin network satisfying:*

- *The maximum $L_{2,4}$-margin for a given dataset $D_p$ is:*

$$\gamma^* = \frac{3}{16} \cdot \frac{1}{p(p-1)}.$$

- *For each neuron $\phi(\{u_1, u_2, u_3, w\}; a_1, a_2, a_3)$, there is a constant scalar $\beta \in \mathbb{R}$ and a frequency $\zeta \in \{1, \dots, \frac{p-1}{2}\}$ satisfying*

$$\begin{aligned} u_1(a_1) &= \beta \cdot \cos(\theta^*_{u_1} + 2\pi\zeta a_1/p) \\ u_2(a_2) &= \beta \cdot \cos(\theta^*_{u_2} + 2\pi\zeta a_2/p) \\ u_3(a_3) &= \beta \cdot \cos(\theta^*_{u_3} + 2\pi\zeta a_3/p) \\ w(c) &= \beta \cdot \cos(\theta^*_{w} + 2\pi\zeta c/p) \end{aligned}$$

  *where $\theta^*_{u_1}, \theta^*_{u_2}, \theta^*_{u_3}, \theta^*_{w} \in \mathbb{R}$ are some phase offsets satisfying $\theta^*_{u_1} + \theta^*_{u_2} + \theta^*_{u_3} = \theta^*_{w}$.*

- *For each frequency $\zeta \in \{1, \dots, \frac{p-1}{2}\}$, there exists one neuron using this frequency only.*

*Proof.* By Lemma C.11, we get the single neuron class-weighted margin solution set $\Omega'^*_q$ satisfying Condition 4.8 and $\gamma^*$.

By Lemma C.17 and Lemma C.5, we can construct network $\theta^*$ which uses neurons in $\Omega'^*_q$ and satisfies Condition 4.8 and Definition 4.5 with respect to $q = \mathrm{unif}(\mathbb{Z}_p)$. By Lemma C.6, we know it is the maximum-margin solution.

By Lemma C.19, when $j_1 = j_2 = j_3 = -j_4 \neq 0$, we must have $\hat{f}(j_1, j_2, j_3, j_4) > 0$. However, as discrete Fourier transform $\hat{\phi}$ of each neuron is nonzero, for each frequency, we must have that there exists one neuron using it.

□

### C.9.2 Main Result for General $k$ Version

**Theorem C.22** (Formal version of Theorem 4.9)**.** *Let $f(\theta, x)$ be the one-hidden layer networks defined in Section 4.3. If the following conditions hold*

- *We adopt the uniform class weighting: $\forall c' \neq a_1 + \cdots + a_k,\ \ \tau(a_1, \dots, a_k)[c'] := 1/(p-1)$.*
- *$m \geqslant 2^{2k-1} \cdot \frac{p-1}{2}$ neurons.*

*Then we have the maximum $L_{2,k+1}$-margin network satisfying:*

- *The maximum* $L_{2,k+1}$*-margin for a given dataset* $D_p$ *is:*

$$\gamma^* = \frac{2(k!)}{(2k+2)^{(k+1)/2}(p-1)p^{(k-1)/2}}.$$

- *For each neuron* $\phi(\{u_1, \dots, u_k, w\}; a_1, \dots, a_k)$ *there is a constant scalar* $\beta \in \mathbb{R}$ *and a frequency* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$ *satisfying*

$$\begin{aligned} u_1(a_1) &= \beta \cdot \cos(\theta^*_{u_1} + 2\pi\zeta a_1/p) \\ &\dots \\ u_k(a_k) &= \beta \cdot \cos(\theta^*_{u_k} + 2\pi\zeta a_k/p) \\ w(c) &= \beta \cdot \cos(\theta^*_w + 2\pi\zeta c/p) \end{aligned}$$

  *where* $\theta^*_{u_1}, \dots, \theta^*_{u_k}, \theta^*_w \in \mathbb{R}$ *are some phase offsets satisfying* $\theta^*_{u_1} + \dots + \theta^*_{u_k} = \theta^*_w$*.*

- *For every frequency* $\zeta \in \{1, \dots, \frac{p-1}{2}\}$*, there exists one neuron using this frequency only.*

*Proof.* Follow the same proof sketch as Theorem C.21 by Lemma C.15, Condition 4.8, Lemma C.18, Lemma C.5, Definition 4.5, Lemma C.6, Lemma C.20.

□

## C.10 More Empirical Details and Results

### C.10.1 Implement Details

**Licenses for Existing Assets & Open Access to Data and Code.** Our code is based on a brilliant open source repository, https://github.com/Sea-Snell/grokking, which requires MIT License. We provide all of our codes in the supplemental material, including dataset generation code. We do not require open data access as we run experiments on synthetic datasets, i.e., modular addition.

**Experimental Result Reproducibility.** We provide all of our codes in the supplemental material with a clear README file and clear configuration files for our experiments reproducibility.

**Experimental Setting/Details & Experiment Statistical Significance.** The detailed configuration can be found in supplemental material. We make a copy version here for convenience.

For two-layer neural network training, we have the following details:

- number of data loader workers: 4
- batch size: 1024
- learning rate: $5 \times 10^{-3}$
- regularization strength $\lambda$: 0.005
- AdamW hyper-parameter $(\beta_1, \beta_2)$: $(0.9, 0.98)$
- warm-up steps: 10

For one-layer Transformer training, we have the following details:

- number of data loader workers: 4
- batch size: 1024
- learning rate: $1 \times 10^{-3}$
- regularization strength $\lambda$: 0.001
- AdamW hyper-parameter $(\beta_1, \beta_2)$: $(0.9, 0.98)$
- warm-up steps: 10

All results we ran 3 times with different random seeds. In Figure 4.4, we reported the mean and variance range.

**Experiments Compute Resources.** All experiments is conducted on single A100 40G NVIDIA GPU. All experiments can be finished in at most three days.

### C.10.2 One-hidden Layer Neural Network

In Figure C.1 and Figure C.2, we use SGD to train a two-layer network with $m = 1536 = 2^{2k-2} \cdot (p-1)$ neurons, i.e., Eq. equation 4.2, on $k = 3$-sum mod-$p = 97$ addition dataset, i.e., Eq. equation 4.1. In Figure C.3 and Figure C.4, we use SGD to train a two-layer network with $m = 5632 = 2^{2k-2} \cdot (p-1)$ neurons, i.e., Eq. equation 4.2, on $k = 5$-sum mod-$p = 23$ addition dataset, i.e., Eq. equation 4.1.

Figure C.1 and Figure C.3 show that the networks trained with stochastic gradient descent have single-frequency hidden neurons, which support our analysis in Lemma 4.10. Furthermore, Figure C.2 and Figure C.4 demonstrate that the network will learn all frequencies in the Fourier spectrum which is consistent with our analysis in Lemma 4.11. Together, they verify our main results in Theorem 4.9 and show that the network trained by SGD prefers to learn Fourier-based circuits.

### C.10.3 One-layer Transformer

In Figure C.5 , we train a one-layer transformer with $m = 160$ heads attention, on $k = 3$-sum mod-$p = 61$ addition dataset, i.e., Eq. equation 4.1. In Figure C.6 , we train a one-layer transformer with $m = 160$ heads attention, on $k = 5$-sum mod-$p = 17$ addition dataset, i.e., Eq. equation 4.1.

Figure C.5 and Figure C.6 show that the one-layer transformer trained with stochastic gradient descent learns 2-dim cosine shape attention matrices, which is similar to one-hidden layer neural networks in Figure C.1 and Figure C.3. This means that the attention layer has a similar learning mechanism to neural networks in the modular arithmetic task, where it prefers to learn Fourier-based circuits when trained by SGD.

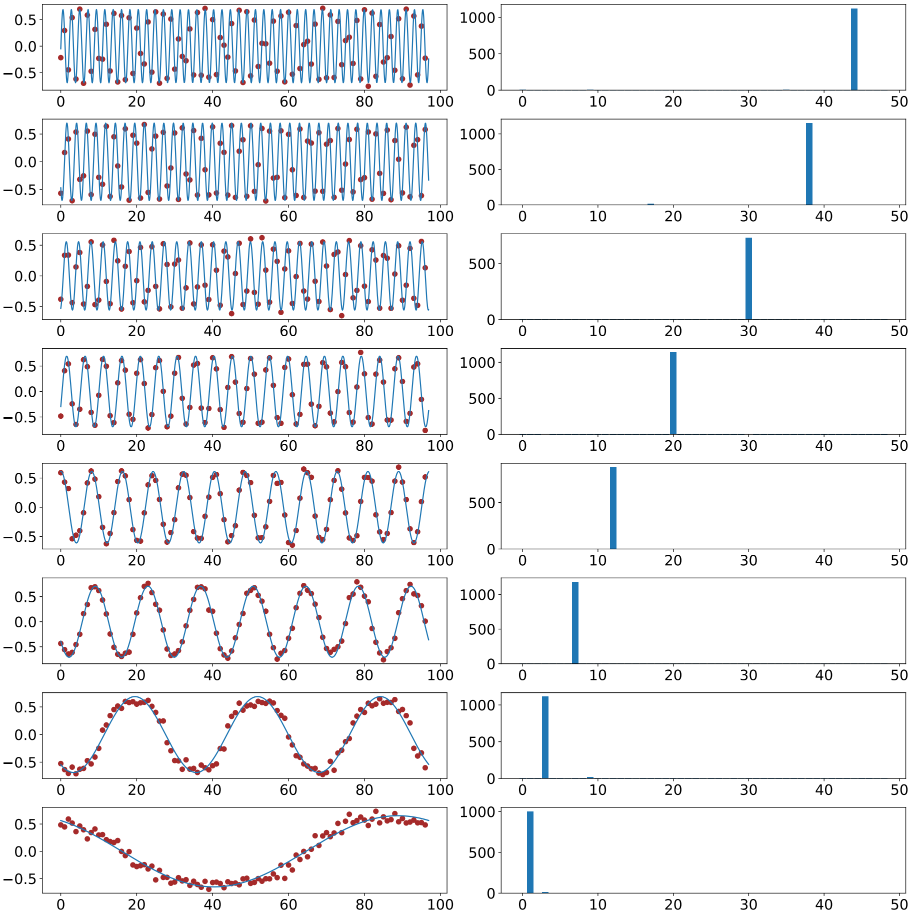


Figure C.1: Cosine shape of the trained embeddings (hidden layer weights) and corresponding power of Fourier spectrum. The two-layer network with $m = 1536$ neurons is trained on $k = 3$-sum mod-$p = 97$ addition dataset. We even split the whole datasets ($p^k = 97^3$ data points) into the training and test datasets. Every row represents a random neuron from the network. The left figure shows the final trained embeddings, with red dots indicating the true weight values, and the pale blue interpolation is achieved by identifying the function that shares the same Fourier spectrum. The right figure shows their Fourier power spectrum. The results in these figures are consistent with our analysis statements in Lemma 4.10.

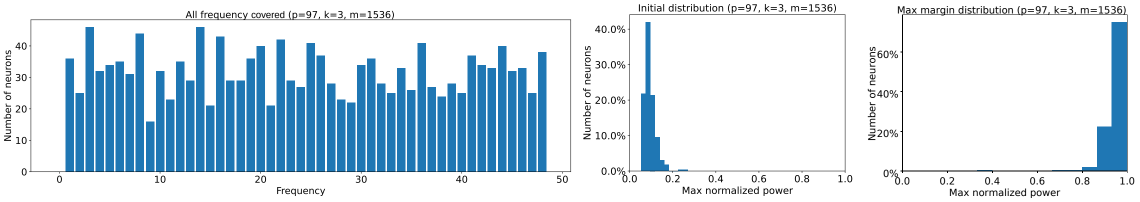


Figure C.2: All Fourier spectrum frequencies being covered and the maximum normalized power of the embeddings (hidden layer weights). The one-hidden layer network with $m = 1536$ neurons is trained on $k = 3$-sum mod-$p = 97$ addition dataset. We denote $\hat{u}[i]$ as the Fourier transform of $u[i]$. Let $\max_i |\hat{u}[i]|^2/(\sum |\hat{u}[j]|^2)$ be the maximum normalized power. Mapping each neuron to its maximum normalized power frequency, (a) shows the final frequency distribution of the embeddings. Similar to our construction analysis in Lemma 4.11, we have an almost uniform distribution over all frequencies. (b) shows the maximum normalized power of the neural network with random initialization. (c) shows, in frequency space, the embeddings of the final trained network are one-sparse, i.e., maximum normalized power being almost 1 for all neurons. This is consistent with our maximum-margin analysis results in Lemma 4.11.

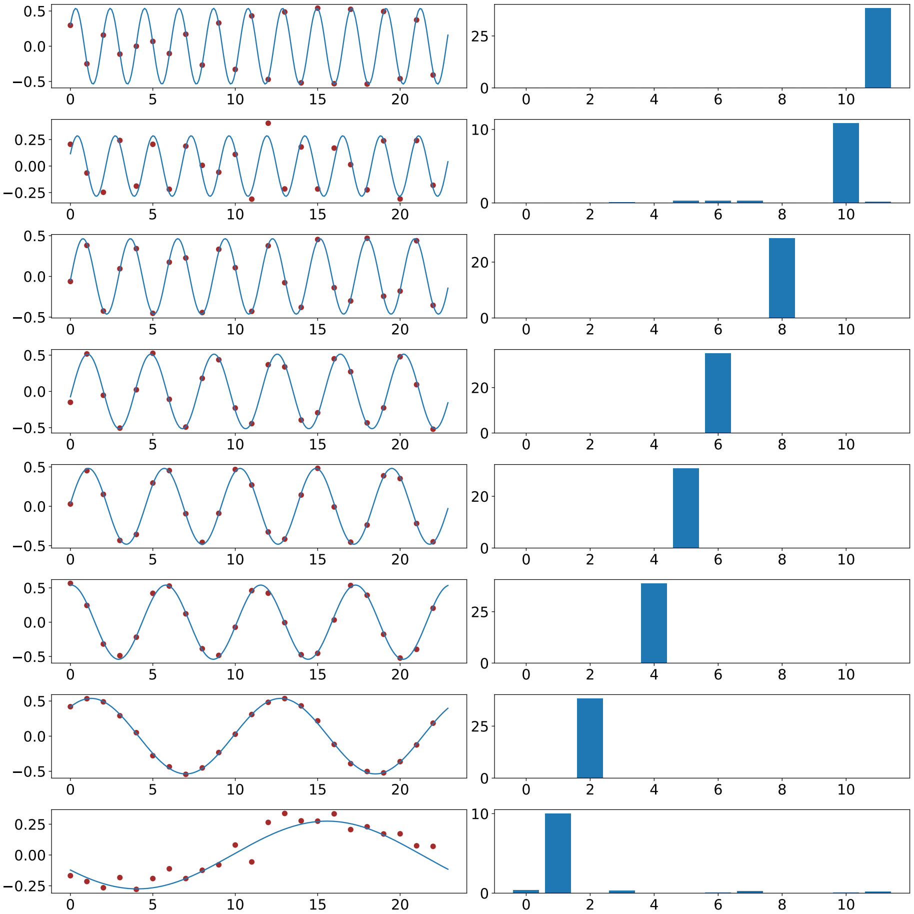


Figure C.3: Cosine shape of the trained embeddings (hidden layer weights) and corresponding power of Fourier spectrum. The two-layer network with $m = 5632$ neurons is trained on $k = 5$-sum mod-$p = 23$ addition dataset. We even split the whole datasets ($p^k = 23^5$ data points) into the training and test datasets. Every row represents a random neuron from the network. The left figure shows the final trained embeddings, with red dots indicating the true weight values, and the pale blue interpolation is achieved by identifying the function that shares the same Fourier spectrum. The right figure shows their Fourier power spectrum. The results in these figures are consistent with our analysis statements in Lemma 4.10.

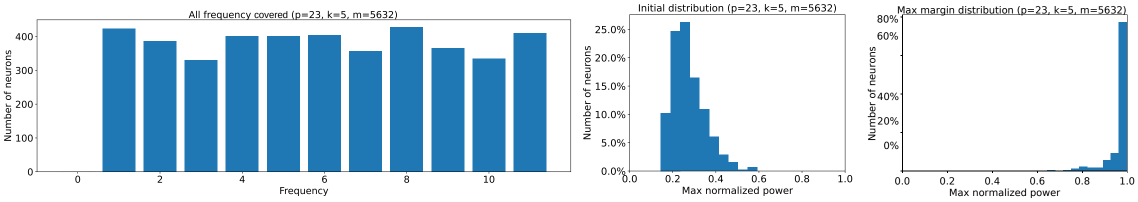


Figure C.4: All Fourier spectrum frequencies being covered and the maximum normalized power of the embeddings (hidden layer weights). The one-hidden layer network with $m = 5632$ neurons is trained on $k = 5$-sum mod-$p = 23$ addition dataset. We denote $\hat{u}[i]$ as the Fourier transform of $u[i]$. Let $\max_i |\hat{u}[i]|^2 / (\sum |\hat{u}[j]|^2)$ be the maximum normalized power. Mapping each neuron to its maximum normalized power frequency, (a) shows the final frequency distribution of the embeddings. Similar to our construction analysis in Lemma 4.11, we have an almost uniform distribution over all frequencies. (b) shows the maximum normalized power of the neural network with random initialization. (c) shows, in frequency space, the embeddings of the final trained network are one-sparse, i.e., maximum normalized power being almost 1 for all neurons. This is consistent with our maximum-margin analysis results in Lemma 4.11.

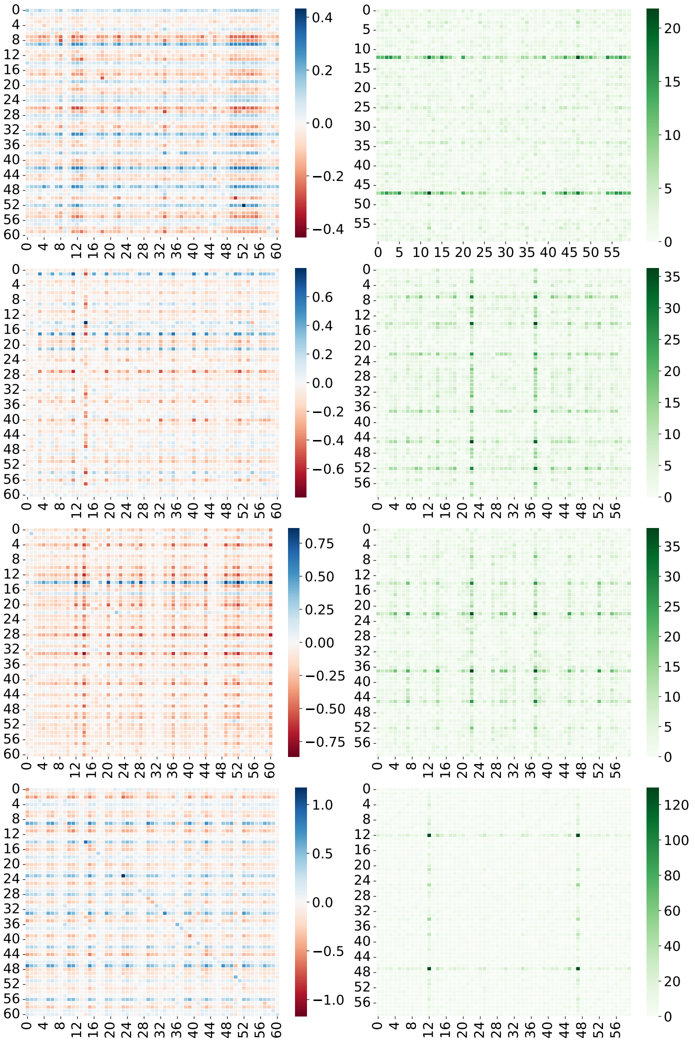


Figure C.5: 2-dimension cosine shape of the trained $W^{KQ}$ (attention weights) and their Fourier power spectrum. The one-layer transformer with attention heads $m = 160$ is trained on $k = 3$-sum mod-$p = 61$ addition dataset. We even split the whole datasets ($p^k = 61^3$ data points) into training and test datasets. Every row represents a random attention head from the transformer. The left figure shows the final trained attention weights being an apparent 2-dim cosine shape. The right figure shows their 2-dim Fourier power spectrum. The results in these figures are consistent with Figure C.1.

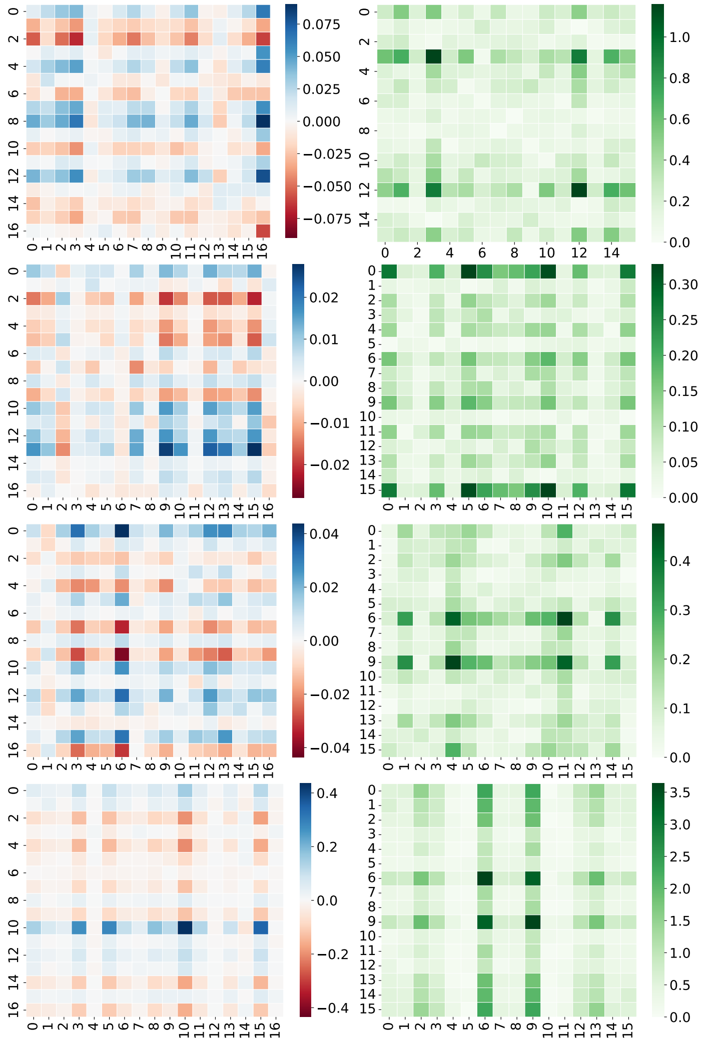


Figure C.6: 2-dimension cosine shape of the trained $W^{KQ}$ (attention weights) and their Fourier power spectrum. The one-layer transformer with attention heads $m = 160$ is trained on $k = 5$-sum mod-$p = 17$ addition dataset. We even split the whole datasets ($p^k = 17^5$ data points) into training and test datasets. Every row represents a random attention head from the transformer. The left figure shows the final trained attention weights being an apparent 2-dim cosine shape. The right figure shows their 2-dim Fourier power spectrum. The results in these figures are consistent with Figure C.3.

# D APPENDIX FOR CHAPTER 5

## D.1 Limitations

We study and understand an interesting phenomenon of in-context learning: smaller models are more robust to noise, while larger ones are more easily distracted, leading to different ICL behaviors. Although we study two stylized settings and give the closed-form solution, our analysis cannot extend to real Transformers easily due to the high non-convex function and complicated design of multiple-layer Transformers. Also, our work does not study optimization trajectory, which we leave as future work. On the other hand, we use simple binary classification real-world datasets to verify our analysis, which still has a gap for the practical user using the LLM scenario.

## D.2 Deferred Proof for Linear Regression

### D.2.1 Proof of Theorem 5.4.1

Here, we provide the proof of Theorem 5.4.1.

**Theorem 5.4.1** (Optimal rank-r solution for regression)**.** *Recall the loss function $\tilde{\ell}$ in Theorem 5.1. Let*

$$\mathbf{U}^*, u^* = \underset{\mathbf{U}\in\mathbb{R}^{d\times d}, \mathrm{rank}(\mathbf{U})\leqslant r, u\in\mathbb{R}}{\arg\min} \tilde{\ell}(\mathbf{U}, u).$$

*Then* $\mathbf{U}^* = c\mathbf{Q}\mathbf{V}^*\mathbf{Q}^\top, u = \frac{1}{c}$*, where* $c$ *is any nonzero constant, and* $\mathbf{V}^* = \mathrm{diag}([v_1^*, \dots, v_d^*])$ *satisfies for any* $i \leqslant r, v_i^* = \frac{N}{(N+1)\lambda_i + \mathrm{tr}(\mathbf{D})}$ *and for any* $i > r, v_i^* = 0$.

*Proof of Theorem 5.4.1.* Note that,

$$\begin{aligned} \underset{\mathbf{U}\in\mathbb{R}^{d\times d}, \mathrm{rank}(\mathbf{U})\leqslant r, u\in\mathbb{R}}{\arg\min} \tilde{\ell}(\mathbf{U}, u) &= \underset{\mathbf{U}\in\mathbb{R}^{d\times d}, \mathrm{rank}(\mathbf{U})\leqslant r, u\in\mathbb{R}}{\arg\min} \tilde{\ell}(\mathbf{U}, u) - \min_{\mathbf{U}\in\mathbb{R}^{d\times d}, u\in\mathbb{R}} \tilde{\ell}(\mathbf{U}, u) \\ &= \underset{\mathbf{U}\in\mathbb{R}^{d\times d}, \mathrm{rank}(\mathbf{U})\leqslant r, u\in\mathbb{R}}{\arg\min} (\tilde{\ell}(\mathbf{U}, u) - \min_{\mathbf{U}\in\mathbb{R}^{d\times d}, u\in\mathbb{R}} \tilde{\ell}(\mathbf{U}, u)). \end{aligned}$$

Thus, we may consider Equation (D.4) in Theorem D.1 only. On the other hand, we have

$$\begin{aligned}\Gamma =&(1+\frac{1}{N})\Lambda+\frac{1}{N}\operatorname{tr}(\Lambda)I_{d\times d}\\ =&(1+\frac{1}{N})\mathbf{Q}\mathbf{D}\mathbf{Q}^\top+\frac{1}{N}\operatorname{tr}(\mathbf{D})\mathbf{Q}I_{d\times d}\mathbf{Q}^\top\\ =&\mathbf{Q}((1+\frac{1}{N})\mathbf{D}+\frac{1}{N}\operatorname{tr}(\mathbf{D})I_{d\times d})\mathbf{Q}^\top.\end{aligned}$$

We denote $\mathbf{D}'=(1+\frac{1}{N})\mathbf{D}+\frac{1}{N}\operatorname{tr}(\mathbf{D})I_{d\times d}$. We can see $\Lambda^{\frac{1}{2}}=\mathbf{Q}\mathbf{D}^{\frac{1}{2}}\mathbf{Q}^\top$, $\Gamma^{\frac{1}{2}}=\mathbf{Q}\mathbf{D}'^{\frac{1}{2}}\mathbf{Q}^\top$, and $\Gamma^{-1}=\mathbf{Q}\mathbf{D}'^{-1}\mathbf{Q}^\top$. We denote $\mathbf{V}=u\mathbf{Q}^\top\mathbf{U}\mathbf{Q}$. Since $\Gamma$ and $\Lambda$ are commutable and the Frobenius norm (F-norm) of a matrix does not change after multiplying it by an orthonormal matrix, we have Equation (D.4) as

$$\begin{aligned}\tilde{\ell}(\mathbf{U},u)-\min_{\mathbf{U}\in\mathbb{R}^{d\times d},u\in\mathbb{R}}\tilde{\ell}(\mathbf{U},u)=&\frac{1}{2}\|\Gamma^{\frac{1}{2}}(u\Lambda^{\frac{1}{2}}\mathbf{U}\Lambda^{\frac{1}{2}}-\Lambda\Gamma^{-1})\|_F^2\\ =&\frac{1}{2}\|\Gamma^{\frac{1}{2}}\Lambda^{\frac{1}{2}}(u\mathbf{U}-\Gamma^{-1})\Lambda^{\frac{1}{2}}\|_F^2\\ =&\frac{1}{2}\|\mathbf{D}'^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}(\mathbf{V}-\mathbf{D}'^{-1})\mathbf{D}^{\frac{1}{2}}\|_F^2.\end{aligned}$$

As $\mathbf{W}^{KQ}$ is a matrix whose rank is at most r, we have $\mathbf{V}$ is also at most rank r. Then, we denote $\mathbf{V}^*=\arg\min_{\mathbf{V}\in\mathbb{R}^{d\times d},\operatorname{rank}(\mathbf{V})\leqslant r}\|\mathbf{D}'^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}(\mathbf{V}-\mathbf{D}'^{-1})\mathbf{D}^{\frac{1}{2}}\|_F^2$. We can see that $\mathbf{V}^*$ is a diagonal matrix. Denote $\mathbf{D}'=\operatorname{diag}([\lambda_1',\ldots,\lambda_d'])$ and $\mathbf{V}^*=\operatorname{diag}([v_1^*,\ldots,v_d^*])$. Then, we have

$$\|\mathbf{D}'^{\frac{1}{2}}\mathbf{D}^{\frac{1}{2}}(\mathbf{V}-\mathbf{D}'^{-1})\mathbf{D}^{\frac{1}{2}}\|_F^2 \tag{D.1}$$

$$=\sum_{i=1}^{d}(\lambda_i'^{\frac{1}{2}}\lambda_i(v_i^*-\frac{1}{\lambda_i'}))^2 \tag{D.2}$$

$$=\sum_{i=1}^{d}((1+\frac{1}{N})\lambda_i+\frac{\operatorname{tr}(\mathbf{D})}{N})\lambda_i^2(v_i^*-\frac{1}{(1+\frac{1}{N})\lambda_i+\frac{\operatorname{tr}(\mathbf{D})}{N}})^2. \tag{D.3}$$

As $\mathbf{V}^*$ is the minimum rank r solution, we have that $v_i^*\geqslant 0$ for any $i\in[d]$ and if $v_i^*>$

0, we have $v_i^* = \frac{1}{(1+\frac{1}{N})\lambda_i + \frac{\mathrm{tr}(\mathbf{D})}{N}}$. Denote $g(x) = ((1+\frac{1}{N})x + \frac{\mathrm{tr}(\mathbf{D})}{N})x^2(\frac{1}{(1+\frac{1}{N})x+\frac{\mathrm{tr}(\mathbf{D})}{N}})^2 = x^2(\frac{1}{(1+\frac{1}{N})x+\frac{\mathrm{tr}(\mathbf{D})}{N}})$. It is easy to see that $g(x)$ is an increasing function on $[0, \infty)$. Now, we use contradiction to show that $\mathbf{V}^*$ only has non-zero entries in the first r diagonal entries. Suppose $i > r$, such that $v_i^* > 0$, then we must have $j \leqslant r$ such that $v_j^* = 0$ as $\mathbf{V}^*$ is a rank r solution. We find that if we set $v_i^* = 0, v_j^* = \frac{1}{(1+\frac{1}{N})\lambda_j + \frac{\mathrm{tr}(\mathbf{D})}{N}}$ and all other values remain the same, Equation (D.3) will strictly decrease as $g(x)$ is an increasing function on $[0, \infty)$. Thus, here is a contradiction. We finish the proof by $\mathbf{V}^* = u\mathbf{Q}^\top \mathbf{U}^* \mathbf{Q}$. □

### D.2.2 Behavior Difference

**Theorem 5.4.2** (Behavior difference for regression)**.** *Let* $\mathbf{w} = \mathbf{Q}(\mathbf{s} + \xi) \in \mathbb{R}^d$ *where* $\mathbf{s}, \xi \in \mathbb{R}^d$ *are truncated and residual vectors defined above. The optimal rank-*r *solution* $f_{\mathrm{LSA},\theta}$ *in Theorem 5.4.1 satisfies:*

$$\begin{aligned}
&\mathcal{L}(f_{\mathrm{LSA},\theta}; \widehat{\mathbf{E}}) \\
:=&\mathbb{E}_{\mathbf{x}_1,\epsilon_1,\ldots,\mathbf{x}_M,\epsilon_M,\mathbf{x}_q}(f_{\mathrm{LSA},\theta}(\widehat{\mathbf{E}}) - \langle \mathbf{w}, \mathbf{x}_q \rangle)^2 \\
=&\frac{1}{M}\|\mathbf{s}\|^2_{(\mathbf{V}^*)^2\mathbf{D}^3} + \frac{1}{M}(\|\mathbf{s}+\xi\|^2_{\mathbf{D}} + \sigma^2)\,\mathrm{tr}((\mathbf{V}^*)^2\mathbf{D}^2) \\
&+ \|\xi\|^2_{\mathbf{D}} + \sum_{i\in[r]} \mathbf{s}_i^2 \lambda_i (\lambda_i v_i^* - 1)^2.
\end{aligned}$$

*Proof of Theorem 5.4.2.* By Theorem 5.4.1, w.l.o.g, letting $c = 1$, the optimal rank-r solution $f_{\mathrm{LSA},\theta}$ satisfies $\theta = (\mathbf{W}^{PV}, \mathbf{W}^{KQ})$, and

$$\mathbf{W}^{*PV} = \begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix}, \mathbf{W}^{*KQ} = \begin{pmatrix} \mathbf{U}^* & 0_d \\ 0_d^\top & 0 \end{pmatrix},$$

where $\mathbf{U}^* = \mathbf{Q}\mathbf{V}^*\mathbf{Q}^\top$.

We can see that $\mathbf{U}^*$ and $\Lambda$ commute. Denote $\widehat{\Lambda} := \frac{1}{M}\sum_{i=1}^M \mathbf{x}_i \mathbf{x}_i^\top$. Note that we have

$$\widehat{y}_q = f_{\mathrm{LSA},\theta}(\widehat{\mathbf{E}})$$

$$
\begin{aligned}
&=\begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix} \left(\frac{\widehat{\mathbf{E}}\widehat{\mathbf{E}}^\top}{M}\right) \begin{pmatrix} \mathbf{U}^* & 0_d \\ 0_d^\top & 0 \end{pmatrix} \mathbf{x}_q \\
&=\begin{pmatrix} 0_{d\times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix} \begin{pmatrix} \frac{1}{M}(\mathbf{x}_q\mathbf{x}_q^\top + \sum_{i=1}^M \mathbf{x}_i\mathbf{x}_i^\top) & \frac{1}{M}(\sum_{i=1}^M \mathbf{x}_i\mathbf{x}_i^\top \mathbf{w} + \sum_{i=1}^M \epsilon_i \mathbf{x}_i) \\ \frac{1}{M}(\sum_{i=1}^M \mathbf{w}^\top \mathbf{x}_i\mathbf{x}_i^\top + \sum_{i=1}^M \epsilon_i \mathbf{x}_i^\top) & \frac{1}{M}\sum_{i=1}^M (\mathbf{w}^\top \mathbf{x}_i + \epsilon_i)^2 \end{pmatrix} \\
&\quad \cdot \begin{pmatrix} \mathbf{U}^* & 0_d \\ 0_d^\top & 0 \end{pmatrix} \mathbf{x}_q \\
&=(\mathbf{w}^\top \widehat{\Lambda} + \frac{1}{M}\sum_{i=1}^M \epsilon_i \mathbf{x}_i^\top)\mathbf{U}^*\mathbf{x}_q.
\end{aligned}
$$

Then, we have

$$
\begin{aligned}
&\mathbb{E}_{\mathbf{x}_1,\epsilon_1,\dots,\mathbf{x}_M,\epsilon_M,\mathbf{x}_q}(\widehat{y}_q - \langle \mathbf{w}, \mathbf{x}_q\rangle)^2 \\
=&\mathbb{E}_{\mathbf{x}_1,\epsilon_1,\dots,\mathbf{x}_M,\epsilon_M,\mathbf{x}_q}(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q + \frac{1}{M}\sum_{i=1}^M \epsilon_i \mathbf{x}_i^\top \mathbf{U}^*\mathbf{x}_q - \mathbf{w}^\top \mathbf{x}_q)^2 \\
=&\underbrace{\mathbb{E}[(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q - \mathbf{w}^\top\mathbf{x}_q)^2]}_{\text{(I)}} + \underbrace{\mathbb{E}[(\frac{1}{M}\sum_{i=1}^M \epsilon_i \mathbf{x}_i^\top \mathbf{U}^*\mathbf{x}_q)^2]}_{\text{(II)}},
\end{aligned}
$$

where the last equality is due to i.i.d. of $\epsilon_i$. We see that the label noise can only have an effect in the second term. For the term (I) we have,

$$
\begin{aligned}
\text{(I)} =&\mathbb{E}[(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q - \mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q + \mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q - \mathbf{w}^\top\mathbf{x}_q)^2] \\
=&\underbrace{\mathbb{E}[(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q - \mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q)^2]}_{\text{(III)}} + \underbrace{\mathbb{E}[(\mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q - \mathbf{w}^\top\mathbf{x}_q)^2]}_{\text{(IV)}},
\end{aligned}
$$

where the last equality is due to $\mathbb{E}[\widehat{\Lambda}] = \Lambda$ and $\widehat{\Lambda}$ is independent with $\mathbf{x}_q$. Note the fact that $\mathbf{U}^*$ and $\Lambda$ commute. For the (III) term, we have

$$
\begin{aligned}
\text{(III)} =&\mathbb{E}[\mathbb{E}[(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q)^2 + (\mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q)^2 - 2(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q)(\mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q)]|\mathbf{x}_q] \\
=&\mathbb{E}[(\mathbf{w}^\top \widehat{\Lambda}\mathbf{U}^*\mathbf{x}_q)^2 - (\mathbf{w}^\top \Lambda\mathbf{U}^*\mathbf{x}_q)^2].
\end{aligned}
$$

By the property of trace, we have,

$$
\begin{aligned}
\text{(III)} =&\mathbb{E}[\operatorname{tr}(\widehat{\Lambda}\mathbf{w}\mathbf{w}^\top\widehat{\Lambda}(\mathbf{U}^*)^2\Lambda)] - \|\mathbf{w}\|^2_{(\mathbf{U}^*)^2\Lambda^3}\\
=&\mathbb{E}[\frac{1}{M^2}\operatorname{tr}((\sum_{i=1}^{M}\mathbf{x}_i\mathbf{x}_i^\top)\mathbf{w}\mathbf{w}^\top(\sum_{i=1}^{M}\mathbf{x}_i\mathbf{x}_i^\top)(\mathbf{U}^*)^2\Lambda)] - \|\mathbf{w}\|^2_{(\mathbf{U}^*)^2\Lambda^3}\\
=&\mathbb{E}[\frac{M-1}{M}\operatorname{tr}(\Lambda\mathbf{w}\mathbf{w}^\top\Lambda(\mathbf{U}^*)^2\Lambda) + \frac{1}{M}\operatorname{tr}(\mathbf{x}_1\mathbf{x}_1^\top\mathbf{w}\mathbf{w}^\top\mathbf{x}_1\mathbf{x}_1^\top(\mathbf{U}^*)^2\Lambda)] - \|\mathbf{w}\|^2_{(\mathbf{U}^*)^2\Lambda^3}\\
=&-\frac{1}{M}\|\mathbf{w}\|^2_{(\mathbf{U}^*)^2\Lambda^3} + \frac{1}{M}\mathbb{E}[\operatorname{tr}(\mathbf{x}_1\mathbf{x}_1^\top\mathbf{w}\mathbf{w}^\top\mathbf{x}_1\mathbf{x}_1^\top(\mathbf{U}^*)^2\Lambda)]\\
=&-\frac{1}{M}\|\mathbf{w}\|^2_{(\mathbf{U}^*)^2\Lambda^3} + \frac{1}{M}\mathbb{E}[\operatorname{tr}((\|\mathbf{w}\|^2_\Lambda\Lambda + 2\Lambda\mathbf{w}^\top\mathbf{w}\Lambda)(\mathbf{U}^*)^2\Lambda)]\\
=&\frac{1}{M}\|\mathbf{w}\|^2_{(\mathbf{U}^*)^2\Lambda^3} + \frac{1}{M}\|\mathbf{w}\|^2_\Lambda\operatorname{tr}((\mathbf{U}^*)^2\Lambda^2),
\end{aligned}
$$

where the third last equality is by Theorem D.2. Furthermore, injecting $\mathbf{w} = \mathbf{Q}(\mathbf{s}+\xi)$, as $\xi^\top\mathbf{V}^*$ is a zero vector, we have

$$
\begin{aligned}
\text{(III)} =& \frac{1}{M}\|\mathbf{s}+\xi\|^2_{(\mathbf{V}^*)^2\mathbf{D}^3} + \frac{1}{M}\|\mathbf{s}+\xi\|^2_{\mathbf{D}}\operatorname{tr}((\mathbf{V}^*)^2\mathbf{D}^2)\\
=& \frac{1}{M}\|\mathbf{s}\|^2_{(\mathbf{V}^*)^2\mathbf{D}^3} + \frac{1}{M}\|\mathbf{s}+\xi\|^2_{\mathbf{D}}\operatorname{tr}((\mathbf{V}^*)^2\mathbf{D}^2).
\end{aligned}
$$

Similarly, for the term (IV), we have

$$
\begin{aligned}
\text{(IV)} =&\mathbb{E}[((\mathbf{s}+\xi)^\top\mathbf{Q}^\top\Lambda\mathbf{U}^*\mathbf{x}_q - (\mathbf{s}+\xi)^\top\mathbf{Q}^\top\mathbf{x}_q)^2]\\
=&\mathbb{E}[(\mathbf{s}^\top\mathbf{D}\mathbf{V}^*\mathbf{Q}^\top\mathbf{x}_q - \mathbf{s}^\top\mathbf{Q}^\top\mathbf{x}_q - \xi^\top\mathbf{Q}^\top\mathbf{x}_q)^2]\\
=&\mathbf{s}^\top(\mathbf{V}^*)^2\mathbf{D}^3\mathbf{s} + \mathbf{s}^\top\mathbf{D}\mathbf{s} + \xi^\top\mathbf{D}\xi - 2\mathbf{s}^\top\mathbf{V}^*\mathbf{D}^2\mathbf{s}\\
=&\xi^\top\mathbf{D}\xi + \sum_{i\in[r]}\mathbf{s}_i^2\lambda_i(\lambda_i^2(v_i^*)^2 - 2\lambda_i v_i^* + 1)\\
=&\|\xi\|^2_{\mathbf{D}} + \sum_{i\in[r]}\mathbf{s}_i^2\lambda_i(\lambda_i v_i^* - 1)^2,
\end{aligned}
$$

where the third equality is due to $\mathbf{s}^\top\mathbf{A}\xi = 0$ for any diagonal matrix $\mathbf{A} \in \mathbb{R}^{d\times d}$.

Now, we analyze the label noise term. By $\mathbf{U}^*$ and $\Lambda$ being commutable, for the term (II), we have

$$
\begin{aligned}
(\mathrm{II}) =& \frac{\sigma^2}{M^2}\mathbb{E}[(\sum_{i=1}^{M} \mathbf{x}_i^\top \mathbf{U}^* \mathbf{x}_q)^2] \\
=& \frac{\sigma^2}{M^2}\mathbb{E}[\mathrm{tr}((\sum_{i=1}^{M} \mathbf{x}_i)^\top \mathbf{U}^* \Lambda \mathbf{U}^* (\sum_{i=1}^{M} \mathbf{x}_i))] \\
=& \frac{\sigma^2}{M}\mathbb{E}[\mathrm{tr}(\mathbf{x}_1^\top \mathbf{U}^* \Lambda \mathbf{U}^* \mathbf{x}_1)] \\
=& \frac{\sigma^2}{M}\,\mathrm{tr}((\mathbf{V}^*)^2 \mathbf{D}^2),
\end{aligned}
$$

where all cross terms vanish in the second equality. We conclude by combining four terms. □

**Theorem 5.4.3** (Behavior difference for regression, special case)**.** *Let* $0 \leqslant r \leqslant r' \leqslant d$ *and* $\mathbf{w} = \mathbf{Q}\mathbf{s}$ *where* $\mathbf{s}$ *is* $r$*-dim truncated vector. Denote the optimal rank-*$r$ *solution as* $f_1$ *and the optimal rank-*$r'$ *solution as* $f_2$*. Then,*

$$
\begin{aligned}
&\mathcal{L}(f_2; \widehat{\mathbf{E}}) - \mathcal{L}(f_1; \widehat{\mathbf{E}}) \\
=& \frac{1}{M}(\|\mathbf{s}\|_{\mathbf{D}}^2 + \sigma^2)(\sum_{i=r+1}^{r'} (\frac{N\lambda_i}{(N+1)\lambda_i + \mathrm{tr}(\mathbf{D})})^2).
\end{aligned}
$$

*Proof of Theorem 5.4.3.* Let $\mathbf{V}^* = \mathrm{diag}([v_1^*, \ldots, v_d^*])$ satisfying for any $i \leqslant r, v_i^* = \frac{N}{(N+1)\lambda_i + \mathrm{tr}(\mathbf{D})}$ and for any $i > r, v_i^* = 0$. Let $\mathbf{V}'^* = \mathrm{diag}([v'^*_1, \ldots, v'^*_d])$ be satisfied for any $i \leqslant r', v'^*_i = \frac{N}{(N+1)\lambda_i + \mathrm{tr}(\mathbf{D})}$ and for any $i > r', v'^*_i = 0$. Note that $\mathbf{V}^*$ is a truncated diagonal matrix of $\mathbf{V}'^*$. By Theorem 5.4.1 and Theorem 5.4.2, we have

$$
\begin{aligned}
\mathcal{L}(f_2; \widehat{\mathbf{E}}) - \mathcal{L}(f_1; \widehat{\mathbf{E}}) =& (\frac{1}{M}\|\mathbf{s}\|_{(\mathbf{V}'^*)^2\mathbf{D}^3}^2 + \frac{1}{M}(\|\mathbf{s}\|_{\mathbf{D}}^2 + \sigma^2)\,\mathrm{tr}((\mathbf{V}'^*)^2\mathbf{D}^2) + \sum_{i \in [r']} \mathbf{s}_i^2 \lambda_i (\lambda_i v'^*_i - 1)^2) \\
& - (\frac{1}{M}\|\mathbf{s}\|_{(\mathbf{V}^*)^2\mathbf{D}^3}^2 + \frac{1}{M}(\|\mathbf{s}\|_{\mathbf{D}}^2 + \sigma^2)\,\mathrm{tr}((\mathbf{V}^*)^2\mathbf{D}^2) + \sum_{i \in [r]} \mathbf{s}_i^2 \lambda_i (\lambda_i v_i^* - 1)^2) \\
=& \frac{1}{M}(\|\mathbf{s}\|_{\mathbf{D}}^2 + \sigma^2)(\mathrm{tr}((\mathbf{V}'^*)^2\mathbf{D}^2) - \mathrm{tr}((\mathbf{V}^*)^2\mathbf{D}^2))
\end{aligned}
$$

$$=\frac{1}{M}(\|\mathbf{s}\|_{\mathbf{D}}^2+\sigma^2)(\sum_{i=r+1}^{r'}(\frac{N\lambda_i}{(N+1)\lambda_i+\mathrm{tr}(\mathbf{D})})^2).$$

□

### D.2.3 Auxiliary Lemma

Theorem D.1 provides the structure of the quadratic form of our MSE loss.

**Lemma D.1** (Corollary A.2 in Zhang et al. (2023c))**.** *The loss function $\tilde{\ell}$ in Theorem 5.1 satisfies*

$$\min_{\mathbf{U}\in\mathbb{R}^{d\times d},u\in\mathbb{R}} \tilde{\ell}(\mathbf{U},u)=-\frac{1}{2}\,\mathrm{tr}[\Lambda^2\Gamma^{-1}],$$

*where $\mathbf{U}=c\Gamma^{-1}, u=\frac{1}{c}$ for any non-zero constant $c$ are minimum solution. We also have*

$$\tilde{\ell}(\mathbf{U},u)-\min_{\mathbf{U}\in\mathbb{R}^{d\times d},u\in\mathbb{R}} \tilde{\ell}(\mathbf{U},u)=\frac{1}{2}\|\Gamma^{\frac{1}{2}}(u\Lambda^{\frac{1}{2}}\mathbf{U}\Lambda^{\frac{1}{2}}-\Lambda\Gamma^{-1})\|_F^2. \tag{D.4}$$

**Lemma D.2.** *Let $\mathbf{x}\sim\mathcal{N}(0,\Lambda)$, $\epsilon\sim\mathcal{N}(0,\sigma^2)$ and $y=\langle\mathbf{w},\mathbf{x}\rangle+\epsilon$, where $\mathbf{w}\in\mathbb{R}^d$ is a fixed vector. Then we have*

$$\begin{aligned}\mathbb{E}[y^2\mathbf{x}\mathbf{x}^\top]=&\sigma^2\Lambda+\|\mathbf{w}\|_\Lambda^2\Lambda+2\Lambda\mathbf{w}^\top\mathbf{w}\Lambda,\\ \mathbb{E}(y\mathbf{x})\mathbb{E}(y\mathbf{x})^\top=&\Lambda^\top\mathbf{w}\mathbf{w}^\top\Lambda,\\ \mathbb{E}[(y\mathbf{x}-\mathbb{E}(y\mathbf{x}))(y\mathbf{x}-\mathbb{E}(y\mathbf{x}))^\top]=&\sigma^2\Lambda+\|\mathbf{w}\|_\Lambda^2\Lambda+\Lambda\mathbf{w}^\top\mathbf{w}\Lambda.\end{aligned}$$

*Proof of Theorem D.2.* As $y$ is a zero mean Gaussian, by Isserlis' theorem Wick (1950); Michalowicz et al. (2009), for any $i,j\in[d]$ we have

$$\begin{aligned}\mathbb{E}[y^2\mathbf{x}_i\mathbf{x}_j]=&\mathbb{E}[y^2]\mathbb{E}[\mathbf{x}_i\mathbf{x}_j]+2\mathbb{E}[y\mathbf{x}_i]\mathbb{E}[y\mathbf{x}_j]\\ =&(\sigma^2+\mathbf{w}^\top\Lambda\mathbf{w})\Lambda_{i,j}+2\Lambda_i^\top\mathbf{w}\mathbf{w}^\top\Lambda_j.\end{aligned}$$

Thus, we have $\mathbb{E}[y^2\mathbf{x}\mathbf{x}^\top]=(\sigma^2+\mathbf{w}^\top\Lambda\mathbf{w})\Lambda+2\Lambda\mathbf{w}^\top\mathbf{w}\Lambda$. Similarly, we also have

$\mathbb{E}(y\mathbf{x})\mathbb{E}(y\mathbf{x})^\top = \Lambda^\top \mathbf{w}\mathbf{w}^\top \Lambda$. Thus, we have

$$\begin{aligned}
&\mathbb{E}[(y\mathbf{x} - \mathbb{E}(y\mathbf{x}))(y\mathbf{x} - \mathbb{E}(y\mathbf{x}))^\top] \\
=&\mathbb{E}[y^2\mathbf{x}\mathbf{x}^\top - y\mathbf{x}\mathbb{E}(y\mathbf{x})^\top - \mathbb{E}(y\mathbf{x})y\mathbf{x}^\top + \mathbb{E}(y\mathbf{x})\mathbb{E}(y\mathbf{x})^\top] \\
=&\mathbb{E}[y^2\mathbf{x}\mathbf{x}^\top] - \mathbb{E}(y\mathbf{x})\mathbb{E}(y\mathbf{x})^\top \\
=&(\sigma^2 + \mathbf{w}^\top\Lambda\mathbf{w})\Lambda + \Lambda\mathbf{w}^\top\mathbf{w}\Lambda.
\end{aligned}$$

□

# D.3 Deferred Proof for Parity Classification

## D.3.1 Proof of Theorem 5.5.1

Here, we provide the proof of Theorem 5.5.1.

**Theorem 5.5.1** (Optimal solution for parity)**.** *Consider* $k = 2^{\nu_1}, d = 2^{\nu_2}$*, and let* $g_1^*$ *and* $g_2^*$ *denote the optimal solutions for* $m = 2(\nu_1 + 1)$ *and* $m = 2(\nu_2 + 1)$*, respectively.*

*When* $0 < p_{\mathcal{T}} < \frac{\frac{1}{4}-\gamma}{\frac{d(d-1)}{2}(\frac{1}{4}+\gamma)+\frac{1}{4}-\gamma}$, $g_1^*$ *neurons are a subset of* $g_2^*$ *neurons. Specifically, for any* $i \in [2(\nu_2 + 1)]$*, let* $\mathbf{V}^{*,(i)}$ *be diagonal matrix and*

- *For any* $i \in [\nu_2]$ *and* $i_\tau \in [d]$*, let* $\mathbf{a}_i^* = -1$ *and* $\mathbf{V}_{i_\tau,i_\tau}^{*,(i)} = (2\,\mathrm{digit}(\mathrm{bin}(i_\tau - 1), i) - 1)/(4\gamma)$*.*
- *For* $i = \nu_2 + 1$ *and any* $i_\tau \in [d]$*, let* $\mathbf{a}_i^* = +1$ *and* $\mathbf{V}_{i_\tau,i_\tau}^{*,(i)} = -\nu_j/(4\gamma)$ *for* $g_j^*$*.*
- *For* $i \in [2(\nu_2 + 1)] \setminus [\nu_2 + 1]$*, let* $\mathbf{a}_i^* = \mathbf{a}_{i-\nu_2-1}^*$ *and* $\mathbf{V}^{*,(i)} = -\mathbf{V}^{*,(i-\nu_2-1)}$*.*

*Let* $\mathbf{W}^{*,(i)} = \mathbf{G}\mathbf{V}^{*,(i)}\mathbf{G}^\top$*. Up to permutations,* $g_2^*$ *has neurons* $(\mathbf{a}^*, \mathbf{W}^{*,(1)}, \dots, \mathbf{W}^{*,(m)})$ *and* $g_1^*$ *has the* $\{1, \dots, \nu_1, \nu_2 + 1, \nu_2 + 2 \dots, \nu_2 + \nu_1 + 1, 2\nu_2 + 2\}$*-th neurons of* $g_2^*$*.*

*Proof of Theorem 5.5.1.* Recall $\mathbf{t}_\tau = (i_\tau, j_\tau)$. Let $\mathbf{z}_\tau \in \mathbb{R}^d$ satisfy $\mathbf{z}_{\tau,i_\tau} = \mathbf{z}_{\tau,j_\tau} = 2\gamma$ and all other entries are zero. Denote $\mathbf{V}^{(i)} = \mathbf{G}^\top\mathbf{W}^{(i)}\mathbf{G}$. Notice that $\|\mathbf{W}^{(i)}\|_F^2 = \|\mathbf{V}^{(i)}\|_F^2$.

Thus, we denote $\mathbf{V}^{*,(i)} = \mathbf{G}^\top \mathbf{W}^{*,(i)} \mathbf{G}$. Then, we have

$$
\begin{aligned}
&\mathbb{E}_\tau[\ell(y_{\tau,q} \cdot g(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}))]\\
=&\mathbb{E}_\tau[\ell(y_{\tau,q}(\sum_{i\in[m]} \mathbf{a}_i \sigma[\frac{\mathbf{y}_\tau^\top \mathbf{X}_\tau}{N} \mathbf{W}^{(i)} \mathbf{x}_{\tau,q}]))]\\
=&\mathbb{E}_\tau[\ell(y_{\tau,q}(\sum_{i\in[m]} \mathbf{a}_i \sigma[\mathbf{z}_\tau^\top \mathbf{V}^{(i)} \phi_{\tau,q}]))]\\
=&\mathbb{E}_\tau[\ell(y_{\tau,q}(\sum_{i\in[m]} \mathbf{a}_i \sigma[2\gamma(\mathbf{V}^{(i)}_{i_\tau,:} + \mathbf{V}^{(i)}_{j_\tau,:})\phi_{\tau,q}]))].
\end{aligned}
$$

We can see that for any $i \in [m]$, $|\mathbf{a}_i^*| = 1$ and $\mathbf{V}^{*,(i)}_{j,l} = 0$ when $j \neq l$. As ReLU is a homogeneous function, we have

$$
\begin{aligned}
&\mathbb{E}_\tau[\ell(y_{\tau,q} \cdot g^*(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}))]\\
=&\underbrace{(1-p_{\mathcal{T}})\mathbb{E}[\ell(2\gamma\phi_{\tau,q,i_\tau}\phi_{\tau,q,j_\tau}(\sum_{i\in[m]} \mathbf{a}_i^* \sigma[\mathbf{V}^{*,(i)}_{i_\tau,i_\tau}\phi_{\tau,q,i_\tau} + \mathbf{V}^{*,(i)}_{j_\tau,j_\tau}\phi_{\tau,q,j_\tau}]))|\mathbf{t}_\tau \in S_1]}_{\text{(I)}}\\
&+\underbrace{p_{\mathcal{T}}\mathbb{E}[\ell(2\gamma\phi_{\tau,q,i_\tau}\phi_{\tau,q,j_\tau}(\sum_{i\in[m]} \mathbf{a}_i^* \sigma[\mathbf{V}^{*,(i)}_{i_\tau,i_\tau}\phi_{\tau,q,i_\tau} + \mathbf{V}^{*,(i)}_{j_\tau,j_\tau}\phi_{\tau,q,j_\tau}]))|\mathbf{t}_\tau \in S_2]}_{\text{(II)}}.
\end{aligned}
$$

We have

$$
\begin{aligned}
\text{(I)} =&(1-p_{\mathcal{T}}) \cdot \Bigg\{(\frac{1}{4}+\gamma)\mathbb{E}[\ell(2\gamma(\sum_{i\in[m]} \mathbf{a}_i^* \sigma[\mathbf{V}^{*,(i)}_{i_\tau,i_\tau} + \mathbf{V}^{*,(i)}_{j_\tau,j_\tau}]))|\mathbf{t}_\tau \in S_1]\\
&+\frac{1}{4}\mathbb{E}[\ell(-2\gamma(\sum_{i\in[m]} \mathbf{a}_i^* \sigma[\mathbf{V}^{*,(i)}_{i_\tau,i_\tau} - \mathbf{V}^{*,(i)}_{j_\tau,j_\tau}]))|\mathbf{t}_\tau \in S_1]\\
&+(\frac{1}{4}-\gamma)\mathbb{E}[\ell(2\gamma(\sum_{i\in[m]} \mathbf{a}_i^* \sigma[-\mathbf{V}^{*,(i)}_{i_\tau,i_\tau} - \mathbf{V}^{*,(i)}_{j_\tau,j_\tau}]))|\mathbf{t}_\tau \in S_1]\\
&+\frac{1}{4}\mathbb{E}[\ell(-2\gamma(\sum_{i\in[m]} \mathbf{a}_i^* \sigma[-\mathbf{V}^{*,(i)}_{i_\tau,i_\tau} + \mathbf{V}^{*,(i)}_{j_\tau,j_\tau}]))|\mathbf{t}_\tau \in S_1]\Bigg\}.
\end{aligned}
$$

We can get a similar equation for (II).

**We make some definitions to be used.** We define a pattern as $(z_1, \{(i_\tau, z_2), (j_\tau, z_3)\})$, where $z_1, z_2, z_3 \in \{\pm 1\}$. We define a pattern is covered by a neuron means there exists $i \in [m]$, such that $\mathbf{a}^*_i = z_1$ and $\mathrm{sign}(\mathbf{V}^{*,(i)}_{i_\tau, i_\tau}) = z_2$ and $\mathrm{sign}(\mathbf{V}^{*,(i)}_{j_\tau, j_\tau}) = z_3$. We define a neuron as being positive when its $\mathbf{a}^*_i = +1$ and being negative when its $\mathbf{a}^*_i = -1$. We define a pattern as being positive if $z_1 = +1$ and being negative if $z_1 = -1$.

Then all terms in (I) and (II) can be written as:

$$\alpha \mathbb{E}[\ell(2\gamma z_1 (\sum_{i \in [m]} \mathbf{a}^*_i \sigma[z_2 \mathbf{V}^{*,(i)}_{i_\tau, i_\tau} + z_3 \mathbf{V}^{*,(i)}_{j_\tau, j_\tau}]))],$$

where $\alpha$ is the scalar term. Note that there are total $\frac{k(k-1)}{2} \times 4$ patterns in (I) and $(\frac{d(d-1)}{2} - \frac{k(k-1)}{2}) \times 4$ patterns in (II). The loss depends on the weighted sum of non-covered patterns. To have zero loss, we need all patterns to be covered by $m$ neurons, i.e., $(\mathbf{a}^*, \mathbf{V}^{*,(1)}, \ldots, \mathbf{V}^{*,(m)})$.

Note that one neuron at most cover $\frac{d(d-1)}{2}$ patterns. Also, by $0 < p_{\mathcal{T}} < \frac{\frac{1}{4} - \gamma}{\frac{d(d-1)}{2}(\frac{1}{4} + \gamma) + \frac{1}{4} - \gamma}$, we have

$$\frac{d(d-1)}{2} p_{\mathcal{T}} (\frac{1}{4} + \gamma) < (1 - p_{\mathcal{T}})(\frac{1}{4} - \gamma),$$

which means the model will only cover all patterns in (I) before covering a pattern in (II) in purpose.

Now, we show that the minimum number of neurons to cover all patterns in (I) and (II) is $2(\nu_2 + 1)$.

**First, we show that $2(\nu_2 + 1)$ neurons are enough to cover all patterns in (I) and (II).** For $i \in [\nu_2]$ and $i_\tau \in [d]$, $\mathbf{V}^{(i)}_{i_\tau, i_\tau} = (2\,\mathrm{digit}(\mathrm{bin}(i_\tau - 1), i) - 1)/(4\gamma)$ and all non-diagonal entries in $\mathbf{V}^{(i)}$ being zero and $\mathbf{a}_i = -1$. For $i = \nu_2 + 1$ and $i_\tau \in [d]$,

$\mathbf{V}^{(i)}_{i_\tau,i_\tau} = -\nu_2/(4\gamma)$ and all non-diagonal entries in $\mathbf{V}^{(i)}$ being zero and $\mathbf{a}_i = +1$. For $i \in [2(\nu_2+1)] \setminus [\nu_2+1]$, let $\mathbf{V}^{(i)} = -\mathbf{V}^{(i-\nu_2-1)}$ and $\mathbf{a}_i = \mathbf{a}_{i-\nu_2-1}$.

We can check that this construction can cover all patterns in (I) and (II) and only needs $2(\nu_2+1)$ neurons. $\mathbf{V}^{(\nu_2+1)}$ and $\mathbf{V}^{(2(\nu_2+1))}$ cover all positive patterns. All other neurons cover all negative patterns. This is because $\text{bin}(i_\tau)$ and $\text{bin}(j_\tau)$ have at least one digit difference. If $\text{bin}(i_\tau)$ and $\text{bin}(j_\tau)$ are different in the $i$-th digit, then $(-1, \{(i_\tau, -1), (j_\tau, +1)\})$ and $(-1, \{(i_\tau, +1), (j_\tau, -1)\})$ are covered by the $i$-th and $i+\nu_2+1$-th neuron.

We can also check that the scalar $\frac{1}{4\gamma}$ and $\frac{\nu_2}{4\gamma}$ is the optimal value. Note that

(1) For any negative patterns, the positive neurons will not have a cancellation effect on the negative neurons, i.e., when $y_q = -1$, the positive neurons will never activate.

(2) For each negative neuron, there exist some patterns that are uniquely covered by it.

(3) For any positive patterns, there are at most $\nu_2 - 1$ negative neurons that will have a cancellation effect on the positive neurons, i.e., when $y_q = +1$, these negative neurons will activate simultaneously. Also, we can check that there is a positive pattern such that there are $\nu_2 - 1$ negative neurons that will have a cancellation effect.

(4) For two positive neurons, there exist some patterns that are uniquely covered by one of them.

Due to hinge loss, we can see that $\frac{1}{4\gamma}$ is tight for negative neurons as (1) and (2). Similarly, we can also see that $\frac{\nu_2}{4\gamma}$ is tight for positive neurons as (3) and (4).

**Second, we prove that we need at least $2(\nu_2+1)$ neurons to cover all patterns in (I) and (II).** We can see that we need at least 2 positive neurons to cover all positive patterns. Then, we only need to show that $2\nu_2 - 1$ neurons are not enough to cover all negative patterns. We can prove that all negative patterns are covered equivalent to all numbers from $\{0, 1, \dots, 2^{\nu_2} - 1\}$ are encoded by $\{(\mathbf{V}_{i,i}^{(1)}, \dots, \mathbf{V}_{i,i}^{(\nu_2)}) \mid i \in [k]\}$. Then $2\nu_2 - 1$ is not enough to do so.

Therefore, the minimum number of neurons to cover all patterns in (I) and (II) is $2(\nu_2 + 1)$.

Thus, when $m = 2(\nu_1 + 1)$, the optimal solution will cover all patterns in (I) but not all in (II). When $m \geqslant 2(\nu_2 + 1)$, the optimal solution will cover all patterns in (I) and (II). We see that $g_1^*$ neurons as the subset of $g_2^*$ neurons, while the only difference is that the scalar of positive neurons is $\frac{\nu_1}{4\gamma}$ for $g_1^*$ and $\frac{\nu_2}{4\gamma}$ for $g_2^*$. Thus, we finished the proof. □

### D.3.2 Proof of Theorem 5.5.2

Here, we provide the proof of Theorem 5.5.2.

**Theorem 5.5.2** (Behavior difference for parity)**.** *Assume the same condition as Theorem 5.5.1. For $j \in \{1, 2\}$, Let $\theta_j$ denote the parameters of $g_j^*$. For $l \in [M]$, let $\xi_l$ be uniformly drawn from $\{\pm 1\}^d$, and $\Xi = \frac{\sum_{l \in [M]} \xi_l}{M}$. Then, for any $\delta \in (0, 1)$, with probability at least $1 - \delta$ over the randomness of test data, we have*

$$\begin{aligned} g_j^*(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}) = {} & h(\theta_j, 2\gamma\hat{\phi}_{\tau,q} + P_{\mathbf{D}_j}(\Xi)) + \epsilon_j \\ := \sum_{i \in [m]} & \mathbf{a}_i^* \sigma[\operatorname{diag}(\mathbf{V}^{*,(i)})^\top (2\gamma\hat{\phi}_{\tau,q} + P_{\mathbf{D}_j}(\Xi))] + \epsilon_j \end{aligned}$$

*where $\epsilon_j = O(\sqrt{\frac{\nu_j}{M} \log \frac{1}{\delta}})$ and we have*

- $2\gamma\hat{\phi}_{\tau,q}$ *is the signal useful for prediction:* $0 = \ell(y_q \cdot h(\theta_1, 2\gamma\hat{\phi}_{\tau,q})) = \ell(y_q \cdot h(\theta_2, 2\gamma\hat{\phi}_{\tau,q}))$.
- $P_{\mathbf{D}_1}(\Xi))$ *and* $P_{\mathbf{D}_2}(\Xi))$ *is noise not related to labels, and* $\frac{\mathbb{E}[\|P_{\mathbf{D}_1}(\Xi))\|_2^2]}{\mathbb{E}[\|P_{\mathbf{D}_2}(\Xi))\|_2^2]} = \frac{\nu_1 + 1}{\nu_2 + 1}$.

*Proof of Theorem 5.5.2.* Let $\Phi^\tau = [\phi_{\tau,1}, \dots, \phi_{\tau,M}]^\top \in \mathbb{R}^{M \times d}$. Recall $\mathbf{t}_\tau = (i_\tau, j_\tau)$. Let $\mathbf{z}_\tau \in \mathbb{R}^d$ satisfy $\mathbf{z}_{\tau,i_\tau} = \mathbf{z}_{\tau,j_\tau} = 2\gamma$ and all other entries are zero. We see $\mathbf{t}_\tau$ as an index set and let $\mathbf{r}_\tau = [d] \setminus \mathbf{t}_\tau$. Then, we have

$$\begin{aligned}
& g_2^*(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}) \\
= & \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\frac{\mathbf{y}_\tau^\top \mathbf{X}_\tau}{M} \mathbf{W}^{*,(i)} \mathbf{x}_{\tau,q}] \\
= & \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\frac{\mathbf{y}_\tau^\top \Phi^\tau}{M} \mathbf{V}^{*,(i)} \phi_{\tau,q}] \\
= & \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\frac{\mathbf{y}_\tau^\top \Phi^\tau_{:,\mathbf{t}_\tau}}{M} \mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,:} \phi_{\tau,q,\mathbf{t}_\tau} + \frac{\mathbf{y}_\tau^\top \Phi^\tau_{:,\mathbf{r}_\tau}}{M} \mathbf{V}^{*,(i)}_{\mathbf{r}_\tau,:} \phi_{\tau,q,\mathbf{r}_\tau}].
\end{aligned}$$

Note that we can absorb the randomness of $\mathbf{y}_\tau$, $\Phi^\tau_{:,\mathbf{r}_\tau}$, $\phi_{\tau,q,\mathbf{r}_\tau}$ together.

Let $z_i$ for $i \in [n]$ uniformly draw from $\{-1, +1\}$. By Chernoff bound for binomial distribution (Theorem D.3), for any $0 < \epsilon < 1$, we have

$$\Pr(|\frac{\sum_{i \in [n]} z_i}{n}| \geqslant \epsilon) \leqslant 2 \exp(-\frac{\epsilon^2 n}{6}).$$

Thus, for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the randomness of evaluation data, such that

$$|\Xi_{\mathbf{t}_\tau}^\top \operatorname{diag}(\mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,\mathbf{t}_\tau})| \leqslant O(\sqrt{\frac{1}{M} \log \frac{1}{\delta}}).$$

Then, for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the randomness of evaluation data, we have

$$\begin{aligned}
& g_2^*(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}) \\
= & \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\frac{\mathbf{y}_\tau^\top \Phi^\tau_{:,\mathbf{t}_\tau}}{M} \mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,:} \phi_{\tau,q,\mathbf{t}_\tau} + \Xi^\top \operatorname{diag}(\mathbf{V}^{*,(i)}) - \Xi_{\mathbf{t}_\tau}^\top \operatorname{diag}(\mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,\mathbf{t}_\tau})] \\
= & \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\mathbf{z}_\tau^\top \mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,:} \phi_{\tau,q,\mathbf{t}_\tau} + \Xi^\top \operatorname{diag}(\mathbf{V}^{*,(i)}) - \Xi_{\mathbf{t}_\tau}^\top \operatorname{diag}(\mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,\mathbf{t}_\tau})] \\
= & \sum_{i \in [m]} \mathbf{a}_i^* \sigma[2\gamma \operatorname{diag}(\mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,\mathbf{t}_\tau})^\top \phi_{\tau,q,\mathbf{t}_\tau} + \Xi^\top \operatorname{diag}(\mathbf{V}^{*,(i)}) - \Xi_{\mathbf{t}_\tau}^\top \operatorname{diag}(\mathbf{V}^{*,(i)}_{\mathbf{t}_\tau,\mathbf{t}_\tau})]
\end{aligned}$$

$$= \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\mathrm{diag}(\mathbf{V}^{*,(i)})^\top (2\gamma \hat{\phi}_{\tau,q} + \Xi) - \Xi_{\mathbf{t}_\tau}^\top \mathrm{diag}(\mathbf{V}_{\mathbf{t}_\tau,\mathbf{t}_\tau}^{*,(i)})]$$

$$= \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\mathrm{diag}(\mathbf{V}^{*,(i)})^\top (2\gamma \hat{\phi}_{\tau,q} + \Xi) + O(\sqrt{\frac{1}{M} \log \frac{1}{\delta}})]$$

$$= \sum_{i \in [m]} \mathbf{a}_i^* \sigma[\mathrm{diag}(\mathbf{V}^{*,(i)})^\top (2\gamma \hat{\phi}_{\tau,q} + P_{\mathbf{D}_2}(\Xi)) + O(\sqrt{\frac{1}{M} \log \frac{1}{\delta}})]$$

$$= h(\theta_2, 2\gamma \hat{\phi}_{\tau,q} + P_{\mathbf{D}_2}(\Xi)) + O(\sqrt{\frac{\nu_2}{M} \log \frac{1}{\delta}}).$$

Similarly, we have $g_1^*(\mathbf{X}_\tau, \mathbf{y}_\tau, \mathbf{x}_{\tau,q}) = h(\theta_1, 2\gamma \hat{\phi}_{\tau,q} + P_{\mathbf{D}_1}(\Xi)) + O(\sqrt{\frac{\nu_1}{M} \log \frac{1}{\delta}})$.

As $\mathbf{t}_\tau \in S_1$ and the number of $(\phi_{i_\tau}, \phi_{j_\tau})$ being balanced as training, by careful checking, we can see that $\ell(y_q \cdot h(\theta_1, 2\gamma \hat{\phi}_{\tau,q})) = \ell(y_q \cdot h(\theta_2, 2\gamma \hat{\phi}_{\tau,q})) = 0$ and we have $2\gamma \hat{\phi}_{\tau,q}$ is the signal part.

On the other hand, we know that all the first half columns in $\mathbf{D}_2$ are orthogonal with each other, and the second half columns in $\mathbf{D}_2$ are opposite to the first half columns. We have the same fact to $\mathbf{D}_1$. As $\Xi$ is a symmetric noise distribution, we have $\frac{\mathbb{E}[\|P_{\mathbf{D}_1}(\Xi))\|_2^2]}{\mathbb{E}[\|P_{\mathbf{D}_2}(\Xi))\|_2^2]} = \frac{\nu_1+1}{\nu_2+1}$ and we have $P_{\mathbf{D}_1}(\Xi))$ and $P_{\mathbf{D}_2}(\Xi))$ is the noise part. □

### D.3.3 Auxiliary Lemma

**Lemma D.3** (Chernoff bound for binomial distribution). *Let* $Z \sim \mathrm{Bin}(n, p)$ *and let* $\mu = \mathbb{E}[Z]$. *For any* $0 < \epsilon < 1$, *we have*

$$\Pr(|Z - \mu| \geqslant \epsilon\mu) \leqslant 2\exp(-\frac{\epsilon^2 \mu}{3}).$$

# E APPENDIX FOR CHAPTER 6

## E.1 Proof of Theoretical Analysis

### E.1.1 Auxiliary lemmas

We first present some Lemmas we will use later.

**Lemma E.1.** *For the logistic loss* $\ell(z) = \ln(1+\exp(-z))$*, we have the following statements (1)* $\ell(z)$ *is strictly decreasing and convex function on* $\mathbb{R}$ *and* $\ell(z) > 0$*; (2)* $\ell'(z) = \frac{-1}{1+\exp(z)}$*,* $\ell'(z) \in (-1, 0)$*; (3)* $\ell'(z)$ *is strictly concave on* $[0, +\infty)$*, (4) for any* $c > 0$*,* $\ell'(z + c) \leqslant \exp(-c)\ell'(z)$*.*

*Proof.* These can be verified by direct calculation. □

**Lemma E.2.**

$$\frac{\partial \mathcal{L}_{(x,y)}(\mathbf{w})}{\partial \mathbf{w}_j} = \ell'(y f_{\mathbf{w}}(\mathbf{x}))\mathbf{z}_j, \tag{E.1}$$

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}_j} = \mathbb{E}_{(x,y)}\left[\ell'(y f_{\mathbf{w}}(\mathbf{x}))\mathbf{z}_j\right] \tag{E.2}$$

$$\frac{\partial \mathcal{L}^{\lambda}(\mathbf{w})}{\partial \mathbf{w}_j} = \mathbb{E}_{(x,y)}\left[\ell'(y f_{\mathbf{w}}(\mathbf{x}))\mathbf{z}_j\right] + \lambda \mathbf{w}_j \tag{E.3}$$

*Proof.* These can be verified by direct calculation. □

**Lemma E.3.** *For any* $j \in R$*, we have probability density function of* $\mathbf{z}_j$ *with mean* $\frac{1}{2}$ *and variance* $\frac{1}{12}$ *following the form*

$$f_{\{\mathbf{z}_j\}}(z) = \begin{cases} 1, & \textit{if} \quad 0 \leqslant z \leqslant 1 \\ 0, & \textit{otherwise} \ . \end{cases}$$

*For any* $j \in U$, *we have probability density function of* $\mathbf{z}_j$ *with mean* $\gamma$ *and variance* $\frac{1}{3} - \gamma^2$ *following the form*

$$f_{\{\mathbf{z}_j\}}(z) = \begin{cases} \frac{1}{2} - \gamma, & \textit{if} \quad -1 \leqslant z < 0 \\ \frac{1}{2} + \gamma, & \textit{if} \quad 0 \leqslant z \leqslant 1 \\ 0, & \textit{otherwise} \ . \end{cases}$$

*Proof.* Then these can be verified by direct calculation from the definition. □

**Lemma E.4.** *We have* $\mathbb{P}\left[\sum_{j \in U} \mathbf{z}_j \leqslant 0\right] \leqslant \exp\left(-\frac{(d-r)\gamma^2}{2}\right)$, $\mathbb{P}\left[\sum_{j \in R} \mathbf{z}_j \leqslant \frac{r}{4}\right] \leqslant \exp\left(-\frac{r}{8}\right)$.

*Proof.* By Hoeffding's inequality,

$$\begin{aligned} \mathbb{P}\left[\sum_{j \in U} \mathbf{z}_j \leqslant 0\right] =& \mathbb{P}\left[\sum_{j \in U} (\mathbf{z}_j - \gamma) \leqslant -(d-r)\gamma\right] && \text{(E.4)} \\ \leqslant& \exp\left(-\frac{(d-r)\gamma^2}{2}\right). && \text{(E.5)} \end{aligned}$$

The others are proven in a similar way. □

### E.1.2 Optimal solution of ERM-$\ell_2$ on ID task

**Lemma E.5** (Restatement of Lemma 6.1 (1)(2))**.** *Consider the ID setting with the ERM-*$\ell_2$ *objective function. Then the optimal* $\mathbf{w}^*$ *for the ERM-*$\ell_2$ *objective function following conditions (1) for any* $j \in R$, $\mathbf{w}_j^* =: \alpha$ *and (2) for any* $j \in U$, $\mathbf{w}_j^* := \beta$.

*Proof of Lemma E.5.*

$$\begin{aligned} \mathcal{L}^\lambda(\mathbf{w}^*) =& \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}_{\mathrm{id}}} \mathcal{L}_{(\mathbf{x},y)}(\mathbf{w}^*) + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2 \\ =& \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}_{\mathrm{id}}} \ell(y f_{\mathbf{w}^*}(\mathbf{x})) + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2 \\ =& \mathbb{E}_{(\mathbf{x},y) \sim \mathcal{D}_{\mathrm{id}}} \ell\left(\sum_{j=1}^{d} \mathbf{w}_j^* \mathbf{z}_j\right) + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2 \end{aligned}$$

By Lemma E.1, we have $\mathcal{L}^{\lambda}(\mathbf{w})$ a is convex function. By symmetry of $\mathbf{z}_j$, for any $l, l' \in R, l \neq l'$,

$$\mathbb{E}\left[\ell\left(\sum_{j=1}^{d} \mathbf{w}_j^* \mathbf{z}_j\right)\right] + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2 \tag{E.6}$$

$$= \frac{1}{2}\left(\mathbb{E}\left[\ell\left(\sum_{j\in[d], j\neq l, j\neq l'} \mathbf{w}_j^* \mathbf{z}_j + \mathbf{w}_l^* \mathbf{z}_l(\mathbf{x}, y) + \mathbf{w}_{l'}^* \mathbf{z}_{l'}(\mathbf{x}, y)\right)\right] + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2\right) \tag{E.7}$$

$$+ \frac{1}{2}\left(\mathbb{E}\left[\ell\left(\sum_{j\in[d], j\neq l, j\neq l'} \mathbf{w}_j^* \mathbf{z}_j + \mathbf{w}_l^* \mathbf{z}_{l'}(\mathbf{x}, y) + \mathbf{w}_{l'}^* \mathbf{z}_l(\mathbf{x}, y)\right)\right] + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2\right) \tag{E.8}$$

$$\geqslant \mathbb{E}\left[\ell\left(\sum_{j\in[d], j\neq l, j\neq l'} \mathbf{w}_j^* \mathbf{z}_j + \frac{\mathbf{w}_l^* + \mathbf{w}_{l'}^*}{2} \mathbf{z}_{l'}(\mathbf{x}, y) + \frac{\mathbf{w}_l^* + \mathbf{w}_{l'}^*}{2} \mathbf{z}_l(\mathbf{x}, y)\right)\right] + \frac{\lambda}{2}\|\mathbf{w}^*\|_2^2, \tag{E.9}$$

where the last inequality follows Jensen's inequality. The minimum is achieved when $\mathbf{w}_l^* = \mathbf{w}_{l'}^*$.

A similar argument as above proves statement (2). □

Now, we will bound the $\alpha$ and $\beta$. Recall that for any $j \in R$, $\mathbf{w}_j^* =: \alpha$ and for any $j \in U$, $\mathbf{w}_j^* := \beta$.

**Lemma E.6** (Restatement of Lemma 6.1 (3))**.** *Let* $\alpha, \beta$ *be values defined in the Lemma E.5. Then, we have* $0 < \beta < \alpha < \frac{1}{\sqrt{r}}$. *Moreover,* $\frac{\alpha}{\beta} < \frac{3}{4\gamma}$.

*Proof of Lemma E.6.* By Lemma E.5

$$\mathcal{L}^{\lambda}(\mathbf{w}^*) = \mathbb{E}\left[\ell\left(\alpha \sum_{j\in R} \mathbf{z}_j + \beta \sum_{j\in U} \mathbf{z}_j\right)\right] + \frac{\lambda}{2}(r\alpha^2 + (d-r)\beta^2) \tag{E.10}$$

$$= \mathcal{L}^{\lambda}(\alpha, \beta). \tag{E.11}$$

By Lemma E.2, we have for any $j \in [d]$

$$\frac{\partial \mathcal{L}^{\lambda}(\mathbf{w}^*)}{\partial \mathbf{w}_j^*} = \mathbb{E}_{(x,y)\sim\mathcal{D}_{id}}\left[\ell'(y f_{\mathbf{w}}^*(\mathbf{x}))\mathbf{z}_j\right] + \lambda \mathbf{w}_j^* = 0. \tag{E.12}$$

We first prove $\beta < \alpha$. For any $j \in R, j' \in U$, we have

$$\begin{aligned}\lambda\alpha =&\lambda \mathbf{w}_j^* &\text{(E.13)}\\ =&-\mathbb{E}_{(x,y)\sim\mathcal{D}_{id}}\left[\ell'(y f_{\mathbf{w}}^*(\mathbf{x}))\mathbf{z}_j\right] &\text{(E.14)}\\ >&-\mathbb{E}_{(x,y)\sim\mathcal{D}_{id}}\left[\ell'(y f_{\mathbf{w}}^*(\mathbf{x}))\mathbf{z}_{j'}(\mathbf{x},y)\right] &\text{(E.15)}\\ =&\lambda \mathbf{w}_{j'}^* = \lambda\beta. &\text{(E.16)}\end{aligned}$$

Then, we prove $\beta \geqslant 0$ by contradiction. Suppose $\beta < 0$,

$$\begin{aligned}&\mathcal{L}^{\lambda}(\alpha,\beta) - \mathcal{L}^{\lambda}(\alpha,-\beta) &\text{(E.17)}\\ =&\mathbb{E}\left[\ell\left(\alpha\sum_{j\in R}\mathbf{z}_j + \beta\sum_{j\in U}\mathbf{z}_j\right)\right] - \mathbb{E}\left[\ell\left(\alpha\sum_{j\in R}\mathbf{z}_j - \beta\sum_{j\in U}\mathbf{z}_j\right)\right]. &\text{(E.18)}\end{aligned}$$

Note that for any $j, j' \in U, j \neq j'$, the norm of $\mathbf{z}_j$ is independent with its sign and $\mathbf{z}_j, \mathbf{z}_{j'}(\mathbf{x}, y)$ are independent. From $\gamma > 0$, we can get $\mathbb{P}[\mathbf{z}_j > 0] > \frac{1}{2}$. Thus, by $\ell$ strictly decreasing we have

$$\mathbb{P}\left[\ell\left(\alpha\sum_{j\in R}\mathbf{z}_j + \beta\sum_{j\in U}\mathbf{z}_j\right) \geqslant z\right] > \mathbb{P}\left[\ell\left(\alpha\sum_{j\in R}\mathbf{z}_j - \beta\sum_{j\in U}\mathbf{z}_j\right) \geqslant z\right], \tag{E.19}$$

where $\beta$ case is strictly stochastically dominate $-\beta$ case. Thus, $\mathcal{L}^{\lambda}(\alpha,\beta) - \mathcal{L}^{\lambda}(\alpha,-\beta) > 0$. This is contradicted by $\beta$ being the optimal value. Thus, we have $\beta \geqslant 0$.

Now, we prove $\alpha < \frac{1}{\sqrt{r}}$, for any $k \in R$,

$$\lambda\alpha = -\mathbb{E}_{(x,y)\sim\mathcal{D}_{id}}\left[\ell'\left(\alpha\sum_{j\in R}\mathbf{z}_j + \beta\sum_{j\in U}\mathbf{z}_j\right)\mathbf{z}_k\right] \tag{E.20}$$

$$\leqslant -\mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\mathrm{id}}}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j+\beta\sum_{j\in U}\mathbf{z}_j\right)\mathbf{z}_k\right] \tag{E.21}$$

$$= -\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j+\beta\sum_{j\in U}\mathbf{z}_j\right)\right]\mathbb{E}[\mathbf{z}_k] \tag{E.22}$$

$$= -\frac{1}{2}\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j+\beta\sum_{j\in U}\mathbf{z}_j\right)\middle|\sum_{j\in U}\mathbf{z}_j>0\right]\mathbb{P}\left[\sum_{j\in U}\mathbf{z}_j>0\right] \tag{E.23}$$

$$-\frac{1}{2}\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j+\beta\sum_{j\in U}\mathbf{z}_j\right)\middle|\sum_{j\in U}\mathbf{z}_j\leqslant 0\right]\mathbb{P}\left[\sum_{j\in U}\mathbf{z}_j\leqslant 0\right] \tag{E.24}$$

$$\leqslant -\frac{1}{2}\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j\right)\right]+\frac{1}{2}\exp\left(-\frac{(d-r)\gamma^2}{2}\right), \tag{E.25}$$

where the last inequality is from $\beta \geqslant 0$ and $\ell'(z)\in(-1,0)$. Using Lemma E.4 one more time, we have

$$-\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j\right)\right] \tag{E.26}$$

$$= -\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j\middle|\sum_{j\in R,j\neq k}\mathbf{z}_j>\frac{r-1}{4}\right)\right]\mathbb{P}\left[\sum_{j\in R,j\neq k}\mathbf{z}_j>\frac{r-1}{4}\right] \tag{E.27}$$

$$-\mathbb{E}\left[\ell'\left(\alpha\sum_{j\in R,j\neq k}\mathbf{z}_j\middle|\sum_{j\in R,j\neq k}\mathbf{z}_j\leqslant\frac{r-1}{4}\right)\right]\mathbb{P}\left[\sum_{j\in R,j\neq k}\mathbf{z}_j\leqslant\frac{r-1}{4}\right] \tag{E.28}$$

$$\leqslant -\ell'\left(\frac{\alpha(r-1)}{4}\right)+\frac{1}{2}\exp\left(-\frac{r-1}{8}\right) \tag{E.29}$$

$$= \frac{1}{1+\exp\left(\frac{\alpha(r-1)}{4}\right)}+\frac{1}{2}\exp\left(-\frac{r-1}{8}\right). \tag{E.30}$$

Thus, we have

$$\lambda\alpha \leqslant \frac{1}{2\left(1+\exp\left(\frac{\alpha(r-1)}{4}\right)\right)}+\frac{1}{4}\exp\left(-\frac{r-1}{8}\right)+\frac{1}{2}\exp\left(-\frac{(d-r)\gamma^2}{2}\right). \tag{E.31}$$

Suppose $\alpha \geqslant \frac{1}{\sqrt{r}}$, we have contradiction,

$$\text{RHS} < O\left(\exp\left(-\frac{\sqrt{r}}{5}\right)\right) < \text{LHS}. \tag{E.32}$$

Thus, we get $\alpha < \frac{1}{\sqrt{r}}$.

Now, we prove $\frac{\alpha}{\beta} \leqslant \frac{3}{4\gamma}$, for any $k \in R, l \in U$, denote $Z = \alpha \sum_{j \in R, j \neq k} \mathbf{z}_j + \beta \sum_{j \in U, j \neq l} \mathbf{z}_j$, by Lemma E.1, we have

$$\begin{aligned}
\frac{\alpha}{\beta} =& \frac{-\mathbb{E}\left[\ell'\left(\alpha \sum_{j \in R} \mathbf{z}_j + \beta \sum_{j \in U} \mathbf{z}_j\right) \mathbf{z}_k\right]}{-\mathbb{E}\left[\ell'\left(\alpha \sum_{j \in R} \mathbf{z}_j + \beta \sum_{j \in U} \mathbf{z}_j\right) \mathbf{z}_l\right]} && \text{(E.33)}\\
\leqslant& \frac{-\mathbb{E}\left[\ell'\left(Z\right) \mathbf{z}_k\right]}{-\mathbb{E}\left[\ell'\left(Z + 2\alpha\right) \mathbf{z}_l | \mathbf{z}_l \geqslant 0\right] \mathbb{P}[\mathbf{z}_l \geqslant 0] - \mathbb{E}\left[\ell'\left(Z\right) \mathbf{z}_l | \mathbf{z}_l < 0\right] \mathbb{P}[\mathbf{z}_l < 0]} && \text{(E.34)}\\
=& \frac{-\mathbb{E}\left[\ell'\left(Z\right)\right]}{-\mathbb{E}\left[\ell'\left(Z + 2\alpha\right)\right]\left(\frac{1}{2} + \gamma\right) + \mathbb{E}\left[\ell'\left(Z\right)\right]\left(\frac{1}{2} - \gamma\right)} && \text{(E.35)}\\
\leqslant& \frac{-\mathbb{E}\left[\ell'\left(Z\right)\right]}{-\exp(-2\alpha)\mathbb{E}\left[\ell'\left(Z\right)\right]\left(\frac{1}{2} + \gamma\right) + \mathbb{E}\left[\ell'\left(Z\right)\right]\left(\frac{1}{2} - \gamma\right)} && \text{(E.36)}\\
=& \frac{1}{\exp(-2\alpha)\left(\frac{1}{2} + \gamma\right) - \left(\frac{1}{2} - \gamma\right)} && \text{(E.37)}\\
\leqslant& \frac{1}{\exp\left(\frac{-2}{\sqrt{r}}\right)\left(\frac{1}{2} + \gamma\right) - \left(\frac{1}{2} - \gamma\right)} && \text{(E.38)}\\
\leqslant& \frac{1}{2\gamma - \left(1 - \exp\left(\frac{-2}{\sqrt{r}}\right)\right)} && \text{(E.39)}\\
\leqslant& \frac{1}{2\gamma - \frac{2}{\sqrt{r}}} && \text{(E.40)}\\
<& \frac{3}{4\gamma}, && \text{(E.41)}
\end{aligned}$$

where the second inequality follows Lemma E.1 and the second last inequality follows $1 + z \leqslant \exp(z)$ for $z \in \mathbb{R}$ and $\gamma > \frac{3}{\sqrt{r}}$. □

### E.1.3 Optimal solution of ERM-rank on ID task

**Lemma E.7.** *Consider ID setting with ERM-rank objective function. Denote* $\mathsf{R}_{\text{rank}}$ *is any subset of* $\mathsf{R}$ *with size* $|\mathsf{R}_{\text{rank}}| = \mathsf{B}_{\text{rank}}$*, we have an optimal* $\mathbf{w}^*$ *for the ERM-rank objective function following conditions* (1) *for any* $j \in \mathsf{R}_{\text{rank}}$, $\mathbf{w}^*_j > 0$ *and* (2) *for any* $j \notin \mathsf{R}_{\text{rank}}$, $\mathbf{w}^*_j = 0$.

*Proof of Lemma E.7.* For any $j \in \mathsf{U}$, if $\mathbf{w}^*_j = \theta \neq 0$, there exists $k \in \mathsf{R}$ s.t. $\mathbf{w}^*_k = 0$ by objective function condition. When we reassign $\mathbf{w}^*_j = 0, \mathbf{w}^*_k = |\theta|$, the objective function becomes smaller. This is a contradiction. Thus, we finish the proof. □

### E.1.4 OOD gap between two objective function

**Proposition E.8** (Restatement of Proposition 6.3)**.** *Assume* $1 \leqslant \mathsf{B}_{\text{rank}} \leqslant r, \lambda > \Omega\left(\frac{\sqrt{r}}{\exp\left(\frac{\sqrt{r}}{5}\right)}\right), d > \frac{r}{\gamma^2} + r, r > C$*, where* $C$ *is some constant* $< 20$*. The optimal solution for the ERM-rank objective function on the ID tasks has 100% OOD test accuracy, while the optimal solution for the ERM-*$\ell_2$ *objective function on the ID tasks has OOD test accuracy at most* $\exp\left(-\frac{r}{10}\right) \times 100\%$ (*much worse than random guessing*).

*Proof of Proposition E.8.* We denote $\mathbf{w}^*_{\mathrm{rank}}$ as the optimal solution for the ERM-rank objective function. By Lemma E.7, the test accuracy for the ERM-rank objective function is

$$\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}[y f_{\mathbf{w}^*_{\mathrm{rank}}}(\mathbf{x}) \geqslant 0] = \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}\left[\sum_{j\in\mathsf{R}} \mathbf{w}^*_{\mathrm{rank},j}\mathbf{z}_j + \sum_{j\in\mathsf{U}} \mathbf{z}_j \mathbf{w}^*_{\mathrm{rank},j} \geqslant 0\right] \tag{E.42}$$

$$= \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}\left[\sum_{j\in\mathsf{R}} \mathbf{w}^*_{\mathrm{rank},j}\mathbf{z}_j \geqslant 0\right] \tag{E.43}$$

$$= 1. \tag{E.44}$$

We denote $\mathbf{w}^*_{\ell_2}$ as the optimal solution for the ERM-rank objective function. We have $\alpha, \beta$ defined in Lemma E.6. By Lemma E.6, the test accuracy for the ERM-$\ell_2$

objective function is

$$\begin{aligned}
\mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}[y f_{\mathbf{w}^*_{\ell_2}}(\mathbf{x}) \geqslant 0] =& \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}\left[\alpha\sum_{j\in R}\mathbf{z}_j + \beta\sum_{j\in U}\mathbf{z}_j \geqslant 0\right] && \text{(E.45)}\\
\leqslant& \mathbb{P}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{ood}}}\left[\frac{3}{4\gamma}\sum_{j\in R}\mathbf{z}_j + \sum_{j\in U}\mathbf{z}_j \geqslant 0\right] && \text{(E.46)}\\
=& \mathbb{P}\left[\frac{3}{4\gamma}\sum_{j\in R}\left(\mathbf{z}_j - \frac{1}{2}\right) + \sum_{j\in U}(\mathbf{z}_j + \gamma) \geqslant -\frac{3r}{8\gamma} + (d-r)\gamma\right] && \text{(E.47)}
\end{aligned}$$

By Hoeffding's inequality and $d > \frac{r}{\gamma^2} + r > 5r$, we have

$$\begin{aligned}
&\mathbb{P}\left[\frac{3}{4\gamma}\sum_{j\in R}\left(\mathbf{z}_j - \frac{1}{2}\right) + \sum_{j\in U}(\mathbf{z}_j + \gamma) \geqslant -\frac{3r}{8\gamma} + (d-r)\gamma\right] && \text{(E.48)}\\
\leqslant& \exp\left(-\frac{2\left(-\frac{3r}{8\gamma} + (d-r)\gamma\right)^2}{4d}\right) && \text{(E.49)}\\
=& \exp\left(-\frac{\frac{9r^2}{32\gamma^2} + 2(d-r)^2\gamma^2 - \frac{3r}{2}(d-r)}{4d}\right) && \text{(E.50)}\\
\leqslant& \exp\left(-\frac{2(d-r)^2\gamma^2 - \frac{3r}{2}(d-r)}{5(d-r)}\right) && \text{(E.51)}\\
=& \exp\left(-\frac{4(d-r)\gamma^2 - 3r}{10}\right) && \text{(E.52)}\\
\leqslant& \exp\left(-\frac{r}{10}\right). && \text{(E.53)}
\end{aligned}$$

□

## E.2 More Experiments Details and Results

| Algorithm | C | L | S | V | Average |
|---|---|---|---|---|---|
| IRM | 98.6 ± 0.1 | 64.9 ± 0.9 | 73.4 ± 0.6 | 77.3 ± 0.9 | 78.5 |
| GroupDRO | 97.3 ± 0.3 | 63.4 ± 0.9 | 69.5 ± 0.8 | 76.7 ± 0.7 | 76.7 |
| MLDG | 97.4 ± 0.2 | 65.2 ± 0.7 | 71.0 ± 1.4 | 75.3 ± 1.0 | 77.2 |
| CORAL | 98.3 ± 0.1 | <u>66.1</u> ± 1.2 | 73.4 ± 0.3 | 77.5 ± 1.2 | 78.8 |
| MMD | 97.7 ± 0.1 | 64.0 ± 1.1 | 72.8 ± 0.2 | 75.3 ± 3.3 | 77.5 |
| DANN | <u>99.0</u> ± 0.3 | 65.1 ± 1.4 | 73.1 ± 0.3 | 77.2 ± 0.6 | 78.6 |
| CDANN | 97.1 ± 0.3 | 65.1 ± 1.2 | 70.7 ± 0.8 | 77.1 ± 1.5 | 77.5 |
| MTL | 97.8 ± 0.4 | 64.3 ± 0.3 | 71.5 ± 0.7 | 75.3 ± 1.7 | 77.2 |
| SagNet | 97.9 ± 0.4 | 64.5 ± 0.5 | 71.4 ± 1.3 | 77.5 ± 0.5 | 77.8 |
| ARM | 98.7 ± 0.2 | 63.6 ± 0.7 | 71.3 ± 1.2 | 76.7 ± 0.6 | 77.6 |
| VREx | 98.4 ± 0.3 | 64.4 ± 1.4 | 74.1 ± 0.4 | 76.2 ± 1.3 | 78.3 |
| RSC | 97.9 ± 0.1 | 62.5 ± 0.7 | 72.3 ± 1.2 | 75.6 ± 0.8 | 77.1 |
| AND-mask | 97.8 ± 0.4 | 64.3 ± 1.2 | 73.5 ± 0.7 | 76.8 ± 2.6 | 78.1 |
| SelfReg | 96.7 ± 0.4 | 65.2 ± 1.2 | 73.1 ± 1.3 | 76.2 ± 0.7 | 77.8 |
| mDSDI | 97.6 ± 0.1 | **66.4** ± 0.4 | 74.0 ± 0.6 | 77.8 ± 0.7 | 79.0 |
| Fishr | 98.9 ± 0.3 | 64.0 ± 0.5 | 71.5 ± 0.2 | 76.8 ± 0.7 | 77.8 |
| ERM | 97.7 ± 0.4 | 64.3 ± 0.9 | 73.4 ± 0.5 | 74.6 ± 1.3 | 77.5 |
| ERM-NU (ours) | 97.9 ± 0.4 | 65.1 ± 0.3 | 73.2 ± 0.9 | 76.9 ± 0.5 | 78.3 |
| Mixup | 98.3 ± 0.6 | 64.8 ± 1.0 | 72.1 ± 0.5 | 74.3 ± 0.8 | 77.4 |
| Mixup-NU (ours) | 97.9 ± 0.2 | 64.1 ± 1.4 | 73.1 ± 0.9 | 74.8 ± 0.5 | 77.5 |
| SWAD | 98.8 ± 0.1 | 63.3 ± 0.3 | <u>75.3</u> ± 0.5 | <u>79.2</u> ± 0.6 | <u>79.1</u> |
| SWAD-NU (ours) | **99.1** ± 0.4 | 63.6 ± 0.4 | **75.9** ± 0.4 | **80.5** ± 1.0 | **79.8** |

Table E.1: Results on VLCS. For each column, bold indicates the best performance, and underline indicates the second-best performance.

| Algorithm | A | C | P | S | Average |
|---|---|---|---|---|---|
| IRM | 84.8 ± 1.3 | 76.4 ± 1.1 | 96.7 ± 0.6 | 76.1 ± 1.0 | 83.5 |
| GroupDRO | 83.5 ± 0.9 | 79.1 ± 0.6 | 96.7 ± 0.3 | 78.3 ± 2.0 | 84.4 |
| MLDG | 85.5 ± 1.4 | 80.1 ± 1.7 | 97.4 ± 0.3 | 76.6 ± 1.1 | 84.9 |
| CORAL | 88.3 ± 0.2 | 80.0 ± 0.5 | 97.5 ± 0.3 | 78.8 ± 1.3 | 86.2 |
| MMD | 86.1 ± 1.4 | 79.4 ± 0.9 | 96.6 ± 0.2 | 76.5 ± 0.5 | 84.6 |
| DANN | 86.4 ± 0.8 | 77.4 ± 0.8 | 97.3 ± 0.4 | 73.5 ± 2.3 | 83.6 |
| CDANN | 84.6 ± 1.8 | 75.5 ± 0.9 | 96.8 ± 0.3 | 73.5 ± 0.6 | 82.6 |
| MTL | 87.5 ± 0.8 | 77.1 ± 0.5 | 96.4 ± 0.8 | 77.3 ± 1.8 | 84.6 |
| SagNet | 87.4 ± 1.0 | 80.7 ± 0.6 | 97.1 ± 0.1 | 80.0 ± 0.4 | 86.3 |
| ARM | 86.8 ± 0.6 | 76.8 ± 0.5 | 97.4 ± 0.3 | 79.3 ± 1.2 | 85.1 |
| VREx | 86.0 ± 1.6 | 79.1 ± 0.6 | 96.9 ± 0.5 | 77.7 ± 1.7 | 84.9 |
| RSC | 85.4 ± 0.8 | 79.7 ± 1.8 | 97.6 ± 0.3 | 78.2 ± 1.2 | 85.2 |
| AND-mask | 85.3 ± 1.4 | 79.2 ± 2.0 | 96.9 ± 0.4 | 76.2 ± 1.4 | 84.4 |
| SelfReg | 87.9 ± 1.0 | 79.4 ± 1.4 | 96.8 ± 0.7 | 78.3 ± 1.2 | 85.6 |
| mDSDI | 87.7 ± 0.4 | 80.4 ± 0.7 | **98.1** ± 0.3 | 78.4 ± 1.2 | 86.2 |
| Fishr | 88.4 ± 0.2 | 78.7 ± 0.7 | 97.0 ± 0.1 | 77.8 ± 2.0 | 85.5 |
| ERM | 84.7 ± 0.4 | 80.8 ± 0.6 | 97.2 ± 0.3 | 79.3 ± 1.0 | 85.5 |
| ERM-NU (ours) | 87.4 ± 0.5 | 79.6 ± 0.9 | 96.3 ± 0.7 | 79.0 ± 0.5 | 85.6 |
| Mixup | 86.1 ± 0.5 | 78.9 ± 0.8 | 97.6 ± 0.1 | 75.8 ± 1.8 | 84.6 |
| Mixup-NU (ours) | 86.7 ± 0.3 | 78.0 ± 1.3 | 97.3 ± 0.3 | 77.3 ± 2.0 | 84.8 |
| SWAD | 89.3 ± 0.2 | **83.4** ± 0.6 | 97.3 ± 0.3 | 82.5 ± 0.5 | 88.1 |
| SWAD-NU (ours) | **89.8** ± 1.1 | 82.8 ± 1.0 | 97.7 ± 0.3 | **83.7** ± 1.1 | **88.5** |

Table E.2: Results on PACS.

| Algorithm | A | C | P | R | Average |
|---|---|---|---|---|---|
| IRM | 58.9 ± 2.3 | 52.2 ± 1.6 | 72.1 ± 2.9 | 74.0 ± 2.5 | 64.3 |
| GroupDRO | 60.4 ± 0.7 | 52.7 ± 1.0 | 75.0 ± 0.7 | 76.0 ± 0.7 | 66.0 |
| MLDG | 61.5 ± 0.9 | 53.2 ± 0.6 | 75.0 ± 1.2 | 77.5 ± 0.4 | 66.8 |
| CORAL | 65.3 ± 0.4 | 54.4 ± 0.5 | 76.5 ± 0.1 | 78.4 ± 0.5 | 68.7 |
| MMD | 60.4 ± 0.2 | 53.3 ± 0.3 | 74.3 ± 0.1 | 77.4 ± 0.6 | 66.3 |
| DANN | 59.9 ± 1.3 | 53.0 ± 0.3 | 73.6 ± 0.7 | 76.9 ± 0.5 | 65.9 |
| CDANN | 61.5 ± 1.4 | 50.4 ± 2.4 | 74.4 ± 0.9 | 76.6 ± 0.8 | 65.8 |
| MTL | 61.5 ± 0.7 | 52.4 ± 0.6 | 74.9 ± 0.4 | 76.8 ± 0.4 | 66.4 |
| SagNet | 63.4 ± 0.2 | 54.8 ± 0.4 | 75.8 ± 0.4 | 78.3 ± 0.3 | 68.1 |
| ARM | 58.9 ± 0.8 | 51.0 ± 0.5 | 74.1 ± 0.1 | 75.2 ± 0.3 | 64.8 |
| VREx | 60.7 ± 0.9 | 53.0 ± 0.9 | 75.3 ± 0.1 | 76.6 ± 0.5 | 66.4 |
| RSC | 60.7 ± 1.4 | 51.4 ± 0.3 | 74.8 ± 1.1 | 75.1 ± 1.3 | 65.5 |
| AND-mask | 59.5 ± 1.1 | 51.7 ± 0.2 | 73.9 ± 0.4 | 77.1 ± 0.2 | 65.6 |
| SelfReg | 63.6 ± 1.4 | 53.1 ± 1.0 | 76.9 ± 0.4 | 78.1 ± 0.4 | 67.9 |
| mDSDI | 62.4 ± 0.5 | 54.4 ± 0.4 | 76.2 ± 0.5 | 78.3 ± 0.1 | 67.8 |
| Fishr | **68.1** ± 0.3 | 52.1 ± 0.4 | 76.0 ± 0.2 | 80.4 ± 0.2 | 69.2 |
| ERM | 61.3 ± 0.7 | 52.4 ± 0.3 | 75.8 ± 0.1 | 76.6 ± 0.3 | 66.5 |
| ERM-NU (ours) | 63.3 ± 0.2 | 54.2 ± 0.3 | 76.7 ± 0.2 | 78.2 ± 0.3 | 68.1 |
| Mixup | 62.4 ± 0.8 | 54.8 ± 0.6 | 76.9 ± 0.3 | 78.3 ± 0.2 | 68.1 |
| Mixup-NU (ours) | 64.3 ± 0.5 | 55.9 ± 0.6 | 76.9 ± 0.4 | 78.0 ± 0.6 | 68.8 |
| SWAD | 66.1 ± 0.4 | 57.7 ± 0.4 | 78.4 ± 0.1 | 80.2 ± 0.2 | 70.6 |
| SWAD-NU (ours) | 67.5 ± 0.3 | **58.4** ± 0.6 | **78.6** ± 0.9 | **80.7** ± 0.1 | **71.3** |

Table E.3: Results on OfficeHome.

| Algorithm | L100 | L38 | L43 | L46 | Average |
|---|---|---|---|---|---|
| IRM | 54.6 ± 1.3 | 39.8 ± 1.9 | 56.2 ± 1.8 | 39.6 ± 0.8 | 47.6 |
| GroupDRO | 41.2 ± 0.7 | 38.6 ± 2.1 | 56.7 ± 0.9 | 36.4 ± 2.1 | 43.2 |
| MLDG | 54.2 ± 3.0 | 44.3 ± 1.1 | 55.6 ± 0.3 | 36.9 ± 2.2 | 47.7 |
| CORAL | 51.6 ± 2.4 | 42.2 ± 1.0 | 57.0 ± 1.0 | 39.8 ± 2.9 | 47.6 |
| MMD | 41.9 ± 3.0 | 34.8 ± 1.0 | 57.0 ± 1.9 | 35.2 ± 1.8 | 42.2 |
| DANN | 51.1 ± 3.5 | 40.6 ± 0.6 | 57.4 ± 0.5 | 37.7 ± 1.8 | 46.7 |
| CDANN | 47.0 ± 1.9 | 41.3 ± 4.8 | 54.9 ± 1.7 | 39.8 ± 2.3 | 45.8 |
| MTL | 49.3 ± 1.2 | 39.6 ± 6.3 | 55.6 ± 1.1 | 37.8 ± 0.8 | 45.6 |
| SagNet | 53.0 ± 2.9 | 43.0 ± 2.5 | 57.9 ± 0.6 | 40.4 ± 1.3 | 48.6 |
| ARM | 49.3 ± 0.7 | 38.3 ± 2.4 | 55.8 ± 0.8 | 38.7 ± 1.3 | 45.5 |
| VREx | 48.2 ± 4.3 | 41.7 ± 1.3 | 56.8 ± 0.8 | 38.7 ± 3.1 | 46.4 |
| RSC | 50.2 ± 2.2 | 39.2 ± 1.4 | 56.3 ± 1.4 | 40.8 ± 0.6 | 46.6 |
| AND-mask | 50.0 ± 2.9 | 40.2 ± 0.8 | 53.3 ± 0.7 | 34.8 ± 1.9 | 44.6 |
| SelfReg | 48.8 ± 0.9 | 41.3 ± 1.8 | 57.3 ± 0.7 | 40.6 ± 0.9 | 47.0 |
| mDSDI | 53.2 ± 3.0 | 43.3 ± 1.0 | 56.7 ± 0.5 | 39.2 ± 1.3 | 48.1 |
| Fishr | 50.2 ± 3.9 | 43.9 ± 0.8 | 55.7 ± 2.2 | 39.8 ± 1.0 | 47.4 |
| ERM | 49.8 ± 4.4 | 42.1 ± 1.4 | 56.9 ± 1.8 | 35.7 ± 3.9 | 46.1 |
| ERM-NU (ours) | 52.5 ± 1.2 | 45.0 ± 0.5 | 60.2 ± 0.2 | 40.7 ± 1.0 | 49.6 |
| Mixup | **59.6** ± 2.0 | 42.2 ± 1.4 | 55.9 ± 0.8 | 33.9 ± 1.4 | 47.9 |
| Mixup-NU (ours) | 55.1 ± 3.1 | 45.8 ± 0.7 | 56.4 ± 1.2 | 41.1 ± 0.6 | 49.6 |
| SWAD | 55.4 ± 0.0 | 44.9 ± 1.1 | 59.7 ± 0.4 | 39.9 ± 0.2 | 50.0 |
| SWAD-NU (ours) | 58.1 ± 3.3 | **47.7** ± 1.6 | **60.5** ± 0.8 | **42.3** ± 0.9 | **52.2** |

Table E.4: Results on Terra Incognita.

| Algorithm | clip | info | paint | quick | real | sketch | Average |
|---|---|---|---|---|---|---|---|
| IRM | 48.5 ± 2.8 | 15.0 ± 1.5 | 38.3 ± 4.3 | 10.9 ± 0.5 | 48.2 ± 5.2 | 42.3 ± 3.1 | 33.9 |
| GroupDRO | 47.2 ± 0.5 | 17.5 ± 0.4 | 33.8 ± 0.5 | 9.3 ± 0.3 | 51.6 ± 0.4 | 40.1 ± 0.6 | 33.3 |
| MLDG | 59.1 ± 0.2 | 19.1 ± 0.3 | 45.8 ± 0.7 | 13.4 ± 0.3 | 59.6 ± 0.2 | 50.2 ± 0.4 | 41.2 |
| CORAL | 59.2 ± 0.1 | 19.7 ± 0.2 | 46.6 ± 0.3 | 13.4 ± 0.4 | 59.8 ± 0.2 | 50.1 ± 0.6 | 41.5 |
| MMD | 32.1 ± 13.3 | 11.0 ± 4.6 | 26.8 ± 11.3 | 8.7 ± 2.1 | 32.7 ± 13.8 | 28.9 ± 11.9 | 23.4 |
| DANN | 53.1 ± 0.2 | 18.3 ± 0.1 | 44.2 ± 0.7 | 11.8 ± 0.1 | 55.5 ± 0.4 | 46.8 ± 0.6 | 38.3 |
| CDANN | 54.6 ± 0.4 | 17.3 ± 0.1 | 43.7 ± 0.9 | 12.1 ± 0.7 | 56.2 ± 0.4 | 45.9 ± 0.5 | 38.3 |
| MTL | 57.9 ± 0.5 | 18.5 ± 0.4 | 46.0 ± 0.1 | 12.5 ± 0.1 | 59.5 ± 0.3 | 49.2 ± 0.1 | 40.6 |
| SagNet | 57.7 ± 0.3 | 19.0 ± 0.2 | 45.3 ± 0.3 | 12.7 ± 0.5 | 58.1 ± 0.5 | 48.8 ± 0.2 | 40.3 |
| ARM | 49.7 ± 0.3 | 16.3 ± 0.5 | 40.9 ± 1.1 | 9.4 ± 0.1 | 53.4 ± 0.4 | 43.5 ± 0.4 | 35.5 |
| VREx | 47.3 ± 3.5 | 16.0 ± 1.5 | 35.8 ± 4.6 | 10.9 ± 0.3 | 49.6 ± 4.9 | 42.0 ± 3.0 | 33.6 |
| RSC | 55.0 ± 1.2 | 18.3 ± 0.5 | 44.4 ± 0.6 | 12.2 ± 0.2 | 55.7 ± 0.7 | 47.8 ± 0.9 | 38.9 |
| AND-mask | 52.3 ± 0.8 | 16.6 ± 0.3 | 41.6 ± 1.1 | 11.3 ± 0.1 | 55.8 ± 0.4 | 45.4 ± 0.9 | 37.2 |
| SelfReg | 58.5 ± 0.1 | 20.7 ± 0.1 | 47.3 ± 0.3 | 13.1 ± 0.3 | 58.2 ± 0.2 | 51.1 ± 0.3 | 41.5 |
| mDSDI | 62.1 ± 0.3 | 19.1 ± 0.4 | 49.4 ± 0.4 | 12.8 ± 0.7 | 62.9 ± 0.3 | 50.4 ± 0.4 | 42.8 |
| Fishr | 58.2 ± 0.5 | 20.2 ± 0.2 | 47.7 ± 0.3 | 12.7 ± 0.2 | 60.3 ± 0.2 | 50.8 ± 0.1 | 41.7 |
| ERM | 58.1 ± 0.3 | 18.8 ± 0.3 | 46.7 ± 0.3 | 12.2 ± 0.4 | 59.6 ± 0.1 | 49.8 ± 0.4 | 40.9 |
| ERM-NU (ours) | 60.9 ± 0.0 | 21.1 ± 0.2 | 49.9 ± 0.3 | 13.7 ± 0.2 | 62.5 ± 0.2 | 52.5 ± 0.4 | 43.4 |
| Mixup | 55.7 ± 0.3 | 18.5 ± 0.5 | 44.3 ± 0.5 | 12.5 ± 0.4 | 55.8 ± 0.3 | 48.2 ± 0.5 | 39.2 |
| Mixup-NU (ours) | 59.5 ± 0.3 | 20.5 ± 0.1 | 49.3 ± 0.4 | 13.3 ± 0.5 | 59.6 ± 0.3 | 51.5 ± 0.2 | 42.3 |
| SWAD | 66.0 ± 0.1 | 22.4 ± 0.3 | 53.5 ± 0.1 | 16.1 ± 0.2 | 65.8 ± 0.4 | 55.5 ± 0.3 | 46.5 |
| SWAD-NU (ours) | **66.6** ± 0.2 | **23.2** ± 0.2 | **54.3** ± 0.2 | **16.2** ± 0.2 | **66.1** ± 0.6 | **56.2** ± 0.2 | **47.1** |

Table E.5: Results on DomainNet.

| Algorithm | VLCS | PACS | OfficeHome | TerraInc | DomainNet | Average |
|---|---|---|---|---|---|---|
| SWAD | 79.1 ± 0.1 | 88.1 ± 0.1 | 70.6 ± 0.2 | 50.0 ± 0.3 | 46.5 ± 0.1 | 66.9 |
| SWAD-CORAL | 78.9 ± 0.1 | 88.3 ± 0.1 | 71.3 ± 0.1 | 51.0 ± 0.1 | 46.8 ± 0.0 | 67.3 |
| SWAD-MIRO | 79.6 ± 0.2 | 88.4 ± 0.1 | **72.4** ± 0.1 | **52.9** ± 0.2 | 47.0 ± 0.0 | **68.1** |
| SWAD-NU (ours) | **79.8** ± 0.2 | **88.5** ± 0.2 | 71.3 ± 0.3 | 52.2 ± 0.3 | **47.1** ± 0.1 | 67.8 |

Table E.6: Methods combined with SWAD full results on DomainBed benchmark.

# F APPENDIX FOR CHAPTER 7

## F.1 Proofs for Section 7.2.1

**Theorem F.1** (Restatement of Theorem 7.1). *If $\ell(t) = -t$, then the contrastive loss is equivalent to the PCA objective on $\phi_{z_R}$:*

$$\mathbb{E}\left[\ell\left(\phi(x)^\top[\phi(x^+) - \phi(x^-)]\right)\right] = -\mathbb{E}\left[\|\phi_{z_R} - \phi_0\|^2\right]. \tag{F.1}$$

*If additionally $\phi(x)$ is linear in $x$, then the contrastive loss is equivalent to the linear PCA objective on data from the distribution $p_{\bar{x}}$ of $\bar{x} = \mathbb{E}_{z_U}[x]$:*

$$\mathbb{E}\left[\ell\left(\phi(x)^\top[\phi(x^+) - \phi(x^-)]\right)\right] = -\mathbb{E}\left[\|\phi(\bar{x}) - \phi_0\|^2\right]. \tag{F.2}$$

*Proof.* We first present some preliminaries for the proof. Recall that in our hidden representation data model $x = g(z)$. The learned representation is $\phi(x) = \phi(g(z)) = \phi \circ g(z)$. For brevity, let us define $\phi(x) = \phi \circ g(z) := h(z)$. Also, the hidden representations corresponding to $(x, x^+, x^-)$ are given by $(z, z^+, z^-)$, where

$$z = [z_R\,;\, z_U],\ \ z^+ = [z_R\,;\, z_U^+],\ \ z^- = [z_R^-\,;\, z_U^-],$$

where $z_R$ and $z_R^-$ are sampled independently from the distribution $\mathcal{D}_R$; and $z_U, z_U^+$, and $z_U^-$ are sampled independently from the distribution $\mathcal{D}_U$. The expectation of an arbitrary function $f(z, z^+, z^-)$ can be simplified as follows:

$$\begin{aligned}\mathbb{E}_{(z,z^+,z^-)}\left[f(z,z^+,z^-)\right] &= \mathbb{E}_{(z_R,z_R^-,z_U,z_U^+,z_U^-)}\left[f(z,z^+,z^-)\right]\\ &= \mathbb{E}_{(z_R,z_R^-)}\left[\mathbb{E}_{(z_U,z_U^+,z_U^-)}\left[f(z,z^+,z^-) \mid z_R, z_R^-\right]\right].\end{aligned}$$

The second step follows the law of iterated expectations.

The negative expected contrastive loss is

$$-\,\mathbb{E}_{(x,x^+,x^-)}\left[\,\ell\left(\phi(x)^\top[\phi(x^+) - \phi(x^-)]\right)\,\right] \tag{F.3}$$

$$= -\mathbb{E}_{(z,z^+,z^-)}\left[\,\ell\left(\phi(g(z))^\top[\phi(g(z^+)) - \phi(g(z^-))]\right)\,\right] \tag{F.4}$$

$$= \mathbb{E}_{(z,z^+,z^-)}\left[\,h(z)^\top[h(z^+) - h(z^-)]\,\right] \tag{F.5}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[\mathbb{E}\left[\,h(z)^\top[h(z^+) - h(z^-)] \mid z_R, z_R^-\,\right]\right] \tag{F.6}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[\mathbb{E}\left[h(z) \mid z_R\right]^\top \left(\mathbb{E}\left[h(z^+) \mid z_R\right] - \mathbb{E}\left[h(z^-) \mid z_R^-\right]\right)\right] \tag{F.7}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[\mathbb{E}\left[\phi(x) \mid z_R\right]^\top \left(\mathbb{E}\left[\phi(x^+) \mid z_R\right] - \mathbb{E}\left[\phi(x^-) \mid z_R^-\right]\right)\right] \tag{F.8}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[\phi_{z_R}^\top\left(\phi_{z_R} - \phi_{z_R^-}\right)\right]. \tag{F.9}$$

The second equality follows from the choice of loss $\ell(t) = -t$, and the fourth equality follows from the fact that $z_U, z_U^+$, and $z_U^-$ are sampled independently from the distribution $\mathcal{D}_U$. Also, we have defined $\phi_{z_R} := \mathbb{E}\left[\phi(x) \mid z_R\right]$.

Denote the centered representation as $\bar{\phi}_{z_R} = \phi_{z_R} - \phi_0$. Then we have

$$-\mathbb{E}_{(x,x^+,x^-)}\left[\ell\left(\phi(x)^\top[\phi(x^+) - \phi(x^-)]\right)\right] \tag{F.10}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[\phi_{z_R}^\top\left(\phi_{z_R} - \phi_{z_R^-}\right)\right] \tag{F.11}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[(\bar{\phi}_{z_R} + \phi_0)^\top\left(\bar{\phi}_{z_R} + \phi_0 - \bar{\phi}_{z_R^-} - \phi_0\right)\right] \tag{F.12}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[(\bar{\phi}_{z_R} + \phi_0)^\top\left(\bar{\phi}_{z_R} - \bar{\phi}_{z_R^-}\right)\right] \tag{F.13}$$

$$= \mathbb{E}_{(z_R,z_R^-)}\left[\bar{\phi}_{z_R}^\top\bar{\phi}_{z_R} - \bar{\phi}_{z_R}^\top\bar{\phi}_{z_R^-}\right] + \mathbb{E}_{(z_R,z_R^-)}\left[\phi_0^\top\left(\bar{\phi}_{z_R} - \bar{\phi}_{z_R^-}\right)\right]. \tag{F.14}$$

Since $\bar{\phi}_{z_R}$ and $\bar{\phi}_{z_R^-}$ are independent with mean 0, we have $\mathbb{E}_{(z_R,z_R^-)}[\bar{\phi}_{z_R}^\top\bar{\phi}_{z_R^-}] = 0$, $\mathbb{E}_{(z_R,z_R^-)}[\phi_0^\top\bar{\phi}_{z_R}] = 0$, and $\mathbb{E}_{(z_R,z_R^-)}[\phi_0^\top\bar{\phi}_{z_R^-}] = 0$. Therefore,

$$-\mathbb{E}_{(x,x^+,x^-)}\left[\ell\left(\phi(x)^\top[\phi(x^+) - \phi(x^-)]\right)\right] \tag{F.15}$$

$$= \mathbb{E}_{z_R}\left[\bar{\phi}_{z_R}^\top\bar{\phi}_{z_R}\right] \tag{F.16}$$

$$= \mathbb{E}_{z_R}\left[\|\bar{\phi}_{z_R}\|^2\right] \tag{F.17}$$

$$= \mathbb{E}_{z_R}\left[\|\phi_{z_R} - \phi_0\|^2\right], \tag{F.18}$$

which is the PCA objective on the mean representation $\phi_{z_R}$.

If additionally $\phi(x)$ is linear in $x$, then

$$-\mathbb{E}_{(x,x^+,x^-)}\left[\ell\left(\phi(x)^\top[\phi(x^+)-\phi(x^-)]\right)\right] \tag{F.19}$$
$$=\mathbb{E}_{z_R}\left[\|\phi_{z_R}-\phi_0\|^2\right] \tag{F.20}$$
$$=\mathbb{E}_{\bar{x}}\left[\|\phi(\bar{x})-\phi(x_0)\|^2\right] \tag{F.21}$$

which is the linear PCA objective on the data from the distribution of $\bar{x}=\mathbb{E}[x|z_R]$. □

**Theorem F.2** (Restatement of Theorem 7.2)**.** *Under Assumptions* ***(A1)(A2)(A3)****:*

(1) *The optimal representation $\phi^*$ does not encode $z_U$: $\phi^*\circ g(z)$ is independent of $z_U$.*

(2) *For any invariant feature $i\in R$, there exists $B_i>0$ such that as long as the representations' norm $B_r \geqslant B_i$, the optimal representation encodes $z_i$. Furthermore, if $z_R$ is discrete, then $B_i$ is monotonically decreasing in $\Pr[z_{R\backslash\{i\}}=z^-_{R\backslash\{i\}}, z_i\neq z^-_i]$, the probability that in $z_R$ and $z^-_R$, the $i$-th feature varies while the others remain the same.*

*Proof.* (1) Recall that

$$\phi_{z_R}=\mathbb{E}[\phi\circ g(z)\mid z_R],\quad \phi_0=\mathbb{E}_z[\phi\circ g(z)]=\mathbb{E}_{z_R}[\phi_{z_R}]. \tag{F.22}$$

Then the contrastive loss at pre-training is:

$$\mathbb{E}_{(x,x^+,x^-)}\left[\ell\left(\phi(x)^\top[\phi(x^+)-\phi(x^-)]\right)\right] \tag{F.23}$$
$$=\mathbb{E}_{(z,z^+,z^-)}\left[\ell\left((\phi\circ g(z))^\top(\phi\circ g(z^+)-\phi\circ g(z^-))\right)\right] \tag{F.24}$$
$$=\mathbb{E}_{(z_R,z_R^-)}\left[\mathbb{E}\left[\ell\left((\phi\circ g(z))^\top(\phi\circ g(z^+)-\phi\circ g(z^-))\right)\mid z_R,z_R^-\right]\right] \tag{F.25}$$
$$\geqslant\mathbb{E}_{(z_R,z_R^-)}\left[\ell\left(\mathbb{E}\left[(\phi\circ g(z))^\top(\phi\circ g(z^+)-\phi\circ g(z^-))\mid z_R,z_R^-\right]\right)\right] \tag{F.26}$$
$$=\mathbb{E}_{(z_R,z_R^-)}\left[\ell\left(\mathbb{E}[\phi\circ g(z)\mid z_R]^\top\left(\mathbb{E}[\phi\circ g(z^+)\mid z_R]-\mathbb{E}[\phi\circ g(z^-)\mid z_R^-]\right)\right)\right] \tag{F.27}$$
$$=\mathbb{E}_{(z_R,z_R^-)}\left[\ell\left(\phi_{z_R}^\top\phi_{z_R}-\phi_{z_R}^\top\phi_{z_R^-}\right)\right], \tag{F.28}$$

where the inequality comes from the convexity of $\ell(z)$ and Jensen's inequality applied to the inner expectation. The inequality becomes equality when the representation function $\phi$ is invariant to the spurious features $z_U$, i.e., with probability 1 over the distribution, $\phi \circ g(z) = \phi_{z_R}$. Therefore, the spurious features $z_U$ are not encoded in the optimal representation, proving the first part.

(2) First consider the case when $z$ has discrete values from a finite set. When the generative function $g(z)$ is not independent of $z_i$, we assume for contradiction that the optimal representation $\phi$ is independent of $z_i$. From (1), we know that it is independent of $z_U$. So there exists an f such that $\phi \circ g(z) = f(z_{R\setminus\{i\}})$. Without loss of generality, suppose $U = \emptyset$, then $\phi \circ g(z) = f(z_{-i})$.

Since the generative function $g(z)$ is not independent of $z_i$, there exist $z$ and $z^-$, such that $z_{-i} = z^-_{-i}$, $z_i \neq z^-_i$, $g(z) \neq g(z^-)$, and $z, z^-$ have non-zero probabilities. So $\Pr[z_{-i} = z^-_{-i}, z_i \neq z^-_i] > 0$.

Now construct a new representation function $\bar{\phi} \in \mathbb{R}^{k+n}, n = |\mathcal{Z}|$ such that $\bar{\phi} \circ g(z) = h(z)$ as follows :

$$h(z) = \left[\sqrt{1-\alpha^2} f(z_{-i}), \quad \alpha \|f(z_{-i})\| \mathbf{I}_z\right] \tag{F.29}$$

where $\mathbf{I}_z$ is the one-hot encoding of the value $z$. Note that $\bar{\phi}$ still satisfies that norm bound since $\|\bar{\phi}(x)\| = \|h(z)\| = \|f(z_{-i})\|$. We next show that the contrastive loss of $\bar{\phi}$ can be smaller than that of $\phi$, leading to a contradiction and finishing the proof.

The contrastive loss of $\bar{\phi}$ (using the fact that $z^+ = z$ when $U = \emptyset$) is

$$\mathbb{E}_{(z,z^-)} \left[\ell\left(h(z)^\top h(z) - h(z)^\top h(z^-)\right)\right] \tag{F.30}$$

$$= \mathbb{E}_{(z,z^-)} \left[\ell\left(h(z)^\top h(z) - h(z)^\top h(z^-)\right) | z \neq z^-\right] \Pr[z \neq z^-] \; + \; \mathbb{E}_{z,z^-} [\ell(0)] \Pr[z = z^-]. \tag{F.31}$$

We only need to consider the first term.

$$\mathbb{E}_{(z,z^-)} \left[\ell\left(h(z)^\top h(z) - h(z)^\top h(z^-)\right) | z \neq z^-\right] \Pr[z \neq z^-] \tag{F.32}$$

$$= \mathbb{E}_{(z,z^-)} \Big[ \underbrace{\ell\left(\|f(z_{-i})\|^2 - (1-\alpha^2) f(z_{-i})^\top f(z^-_{-i})\right)}_{T_1} | z_{-i} \neq z^-_{-i} \Big] \Pr[z_{-i} \neq z^-_{-i}] \tag{F.33}$$

$$+\ \mathbb{E}_{(z,z^-)}\left[\ \underbrace{\ell\left(\alpha^2\|f(z_{-i})\|^2\right)}_{T_2}\ |\, z_{-i}=z^-_{-i},\ z_i\neq z^-_i\ \right]\Pr[z_{-i}=z^-_{-i},\ z_i\neq z^-_i]. \quad \text{(F.34)}$$

When $\alpha = 0$, the above reduces to the corresponding terms for $\phi$, so we would like to show that there exists non-zero $\alpha$ that leads to smaller loss values.

Recall that $\ell(\cdot)$ is decreasing by property (A3). Let $\alpha = \sqrt{1/2}/B_r$, where $B_r = \|f(z_{-i})\|$. Then when switching from $\phi$ to $\bar{\phi}$, $T_2$ goes from $\ell(0)$ to $\ell(1/2)$, a constant reduction. For $T_1$, if $f(z_{-i})^\top f(z^-_{-i})$ is positive, then $T_1$ decreases; if $f(z_{-i})^\top f(z^-_{-i})$ is negative, then $T_1$ increases from $\ell(B_r^2 - f(z_{-i})^\top f(z^-_{-i}))$ to $\ell(B_r^2 - f(z_{-i})^\top f(z^-_{-i}) + \alpha^2 f(z_{-i})^\top f(z^-_{-i}))$. Note that $|\alpha^2 f(z_{-i})^\top f(z^-_{-i})| \leqslant 1$ (by the Cauchy-Schwarz inequality); so the increase in $T_1$ diminishes when $B_r$ grows, by the property (A3) of $\ell$. Then when $B_r$ is large enough, the increase in $T_1$ is smaller than the decrease in $T_2$. So from $\phi$ to $\bar{\phi}$, the contrastive loss decreases, contradicting that $\phi$ is optimal. Finally, since the reduction in (F.34) is smaller when $\Pr[z_{-i} = z^-_{-i}, z_i \neq z^-_i]$ is smaller, then $B_i$ needs to be larger. So $B_i$ is monotonically decreasing in $\Pr[z_{-i} = z^-_{-i}, z_i \neq z^-_i]$.

Now consider the general case when $z$ may not be from a finite set. For any $\epsilon_0 > 0$, there exists a $\ell_2$ ball $\mathcal{B}$ of bounded radius such that the probability of $z$ outside the ball is at most $\epsilon_0$. Since $\phi \circ g$'s are regular by assumption, there exists a partition $\mathcal{Z} \cap \mathcal{B}$ into finitely many subsets such that in each subset and for each $\phi \circ g$, the function value varies by at most $\epsilon_0$. Construct a new distribution $\mathcal{D}'_z$ for $z$: select a representative point in each subset, and put a probability mass to it equal to that of the original distribution $\mathcal{D}_z$ in this subset, and normalize the probabilities over the subsets. The new distribution is over a finite set so the above argument holds. Furthermore, the difference in the $T_1$ term for $\mathcal{D}'_z$ and $\mathcal{D}_z$ can be made arbitrarily small by choosing sufficiently small $\epsilon_0$; similarly for $T_2$. Then the argument also holds for $\mathcal{D}_z$, which completes the proof for the general case. □

### F.1.1 Inductive Biases are Needed for Analyzing Prediction Success

We have analyzed what features are encoded in the representation. However, encoding the information does not equate to good prediction performance, in particular, with linear predictors. Recently, Saunshi et al. (2022) demonstrated that existing analyses that ignore the inductive biases of the model and algorithm cannot adequately explain the prediction success, and provided examples where such analysis can lead to vacuous bounds. One may wonder if our hidden representation data model can provide inductive biases that avoid such vacuous bounds. Unfortunately, similar issues as in Saunshi et al. (2022) remain.

To illustrate that inductive biases are still needed in our data model, consider the following simple example. Suppose $z_R \in \{-1, 1\}^2$ and can be recovered from $x$; the label $y$ is simply the first coordinate in $z_R$. Suppose the representation satisfies $\phi(x) \in \mathbb{R}^2, \|\phi(x)\| = 1$, and contrastive learning uses the logistic loss $\ell(z)$. Let $\phi(x)$ be such that $\phi \circ g(z) = h(z_R)$, and $h((-1,-1)) = (-1,0), h((-1,1)) = (1,0), h((1,-1)) = (0,-1), h((1,1)) = (0,1)$. It can be verified that this $\phi$ is optimal for the contrastive loss. However, on the representation $\phi$, the classification is an XOR-problem (Fig. F.1), for which there is no non-trivial error bound for linear predictors. This contradicts the success of linear probing in practice.

Furthermore, some restrictions on the data distributions are also needed. Suppose *all* optimal representations are linearly separable with certain inductive biases on the representation function class. Suppose the label $y$ depends on $z_R$. Without restrictions on the labeling function, one can consider a random $y \in \{-1, +1\}$ over any $z_R$. Then for any linear predictor on any optimal representation, in expectation the error is $1/2$, so there is always a labeling function for which no non-trivial error can be achieved. Our analysis thus requires restrictions on the dependence of the label on $z_R$ (in particular, we will assume linear dependence).

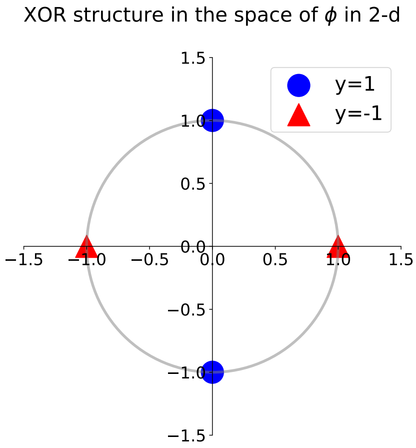


Figure F.1: A two-dim example of XOR structure in the space of $\phi$.

## F.2 Proofs and More Analysis for Section 7.2.2

### F.2.1 Lemmas for a more general setting

We will prove the results in a more general setting, where the mixture can be uneven and the variances of different types of features can be different. The results in Section 7.2.2 then follow from these lemmas.

In the more general setting, the diverse pre-training data is a mixture of data from $T$ different tasks $\mathcal{D}_t$'s, while the target task is one of the tasks. In the mixture, the task $\mathcal{D}_t$ has weight $w_t > 0$ and $\sum_{t=1}^{T} w_t = 1$. All tasks share a public feature set $S$ of size $s$, and each task $\mathcal{D}_t$ additionally owns a private disjoint feature set $P_t$ of size $r - s$, i.e., $P_t \cap S = \emptyset$ for $t \in [T]$ and $P_{t_1} \cap P_{t_2} = \emptyset$ for $t_1 \neq t_2$. The invariant features for $\mathcal{D}_t$ are then $R_t = S \cup P_t$. All invariant features are $\cup_{t=1}^{T} R_t \subseteq R$, $k := |R|$, and spurious features are $U = [d] \setminus R$. In task $\mathcal{D}_t$, the positive pairs $(x, x^+)$ are generated as follows:

$$z_S \sim \mathcal{N}(0, \sigma_{S,t}^2 I), z_{P_t} \sim \mathcal{N}(0, \sigma_{R,t}^2 I), z_{R \setminus R_t} = 0, \tag{F.35}$$

$$z_U \sim \mathcal{N}(0, \sigma_{U,t}^2 I), z = [z_R; z_U], \quad x = g(z), \tag{F.36}$$

$$z_U^+ \sim \mathcal{N}(0, \sigma_{U,t}^2 I), z^+ = [z_R; z_U^+], \quad x^+ = g(z^+), \tag{F.37}$$

and $x^{-}$ is simply an i.i.d. copy from the same distribution as $x$. In practice, multiple independent negative examples are used, and thus we consider the following contrastive loss

$$\min_{\phi \in \Phi} \mathbb{E}_{(x,x^{+})} \left[ \ell \left( \phi(x)^{\top} (\phi(x^{+}) - \mathbb{E}_{x^{-}} \phi(x^{-})) \right) \right] \tag{F.38}$$

to pre-train a representation $\phi$. Then, when using $\phi$ for prediction in the target task $\mathcal{D}_t$, the predictor class should contain a predictor matching the ground-truth label, so consider the class:

$$\mathcal{F}_{\phi,t} = \{f(z) = u_t^{\top} z : u_t \in \mathbb{R}^k, \|u_t\| \leqslant B_{\phi,t}\} \tag{F.39}$$

where $B_{\phi,t}$ is the minimum value such that there exists $u_t \in \mathcal{F}_{\phi,t}$ with $y = u_t^{\top} \phi(x)$ on $\mathcal{D}_t$.

Recall that we assume a linear data model and linear representation functions $\phi$:

- $x$ is linear in $z$: $x = g(z) = Mz$ where $M \in \mathbb{R}^{d \times d}$ is an orthonormal dictionary. The label in task $\mathcal{D}_t$ is linear in its invariant features $y = (u_t^{*})^{\top} z_{R_t}$ for some $u_t^{*} \in \mathbb{R}^r$.
- The representations are linear functions with weights of bounded spectral/Frobenius norms:

  $$\Phi = \{\phi(x) = Wx : W \in \mathbb{R}^{k \times d}, \|W\| \leqslant 1, \|W\|_F \leqslant \sqrt{r}\}.$$

  Here the norm bounds are chosen to be the minimum values to allow recovering the invariant features in the target task, i.e., there exists $\phi \in \Phi$ such that $\phi(x) = [z_{R_t}; \mathbf{0}]$.

**Lemma F.3.** *Consider the above setting. Let* $\alpha, \alpha_t (t \in [T])$ *be the optimizer for*

$$\min_{\tilde{\alpha}, \tilde{\alpha}_1, \ldots, \tilde{\alpha}_T} \sum_{t=1}^{T} w_t \mathbb{E} \left[ \ell \left( \tilde{\alpha} \sigma_{S,t}^2 Z + \tilde{\alpha}_t \sigma_{R,t}^2 Z_t \right) \right], \tag{F.40}$$

$$\textit{subject to} \quad \tilde{\alpha}s + \sum_{t=1}^{T} \tilde{\alpha}_t (r - s) \leqslant r, \tag{F.41}$$

$$\tilde{\alpha}, \tilde{\alpha}_t \in [0, 1], \tag{F.42}$$

*where* $Z \sim \chi_s^2$ *and* $Z_t \sim \chi_{r-s}^2$.

*Then the optimal representation* $\phi^*(x)$ *the loss* (*F.38*) *in contrastive learning satisfies* $\phi^*(x) = W^* x$ *with any* $W^*$ *of the form:*

$$W^* = [QA^*, \boldsymbol{0}]M^{-1} \tag{F.43}$$

*where* $Q \in \mathbb{R}^{k \times k}$ *is any orthonormal matrices,* $A^*$ *is a* $k \times k$ *diagonal matrix with*

$$A_{jj}^* = \begin{cases} \sqrt{\alpha} & \text{if } j \in S, \\ \sqrt{\alpha_t} & \text{if } j \in P_t, \\ 0 & \text{otherwise,} \end{cases} \tag{F.44}$$

*and the matrix of zeros has size* $k \times (d - k)$.

*Proof.* For each $\mathcal{D}_t$,

$$\mathbb{E}_{(x,x^+)} \left[ \ell \left( \phi(x)^\top [\phi(x^+) - \mathbb{E}_{x^-} \phi(x^-)] \right) \right] \tag{F.45}$$

$$= \mathbb{E}_{(z,z^+)} \left[ \ell \left( (WMz)^\top (WMz^+ - \mathbb{E}_{z^-}[WMz^-]) \right) \right] \tag{F.46}$$

$$= \mathbb{E}_{(z,z^+)} \left[ \ell \left( z^\top (M^\top W^\top WM)(z^+ - \mathbb{E}_{z^-}[z^-]) \right) \right] \tag{F.47}$$

$$\geqslant \mathbb{E}_{z_R} \left[ \ell \left( (\mathbb{E}_{z_U}[z])^\top M^\top W^\top WM (\mathbb{E}_{z_U^+}[z^+] - \mathbb{E}_{z^-}[z^-]) \right) \right] \tag{F.48}$$

$$= \mathbb{E}_{z_R} \left[ \ell \left( [z_R; \mathbf{0}]^\top M^\top W^\top WM ([z_R; \mathbf{0}] - 0) \right) \right] \tag{F.49}$$

$$= \mathbb{E}_{z_R} \left[ \ell \left( \|WM[z_R; \mathbf{0}]\|^2 \right) \right] \tag{F.50}$$

where the inequality comes from the convexity of $\ell(t)$ and Jensen's inequality. Similar to Theorem 7.2, the equality holds if and only if $WMz$ does not depend on $z_U$ and $WMz^+$ does not depend on $z_U^+$, so the optimal solution should satisfy this condition.

Let $WM = [A_R, A_U]$ where $A_R \in \mathbb{R}^{k\times k}, A_U \in \mathbb{R}^{k\times(d-k)}$. By rotational invariance of $z_S$, and $z_{P_t}$, without loss of generality, we can assume $A_R = QA$ where $A$ is a diagonal matrix with diagonal entries $a_{jj}$'s and Q is any orthonormal matrix. Furthermore, $A_U = 0$ in the optimal solution since it does not affect the loss but only decreases the norm bound on $A_R$. So on data from the task $\mathcal{D}_t$,

$$\mathbb{E}_{\mathcal{D}_t}\left[\ell\left(\|WM[z_R;\mathbf{0}]\|^2\right)\right] = \mathbb{E}_{z_{R_t}}\left[\ell\left(\sum_{j\in R_t} a_{jj}^2 z_j^2\right)\right]. \tag{F.51}$$

Then on the mixture,

$$\begin{aligned}
&\mathbb{E}_{(x,x^+)}\left[\ell\left(\phi(x)^\top[\phi(x^+) - \mathbb{E}_{x^-}\phi(x^-)]\right)\right] && \text{(F.52)}\\
\geqslant &\sum_{t=1}^{T} w_t \mathbb{E}_{\{z_j\}}\left[\ell\left(\sum_{j\in R_t} a_{jj}^2 z_j^2\right)\right] && \text{(F.53)}\\
= &\sum_{t=1}^{T} w_t \mathbb{E}_{\{\tilde{z}_j\sim\mathcal{N}(0,1)\}}\left[\ell\left(\sum_{j\in S} a_{jj}^2\sigma_{S,t}^2\tilde{z}_j^2 + \sum_{j\in P_t} a_{jj}^2\sigma_{R,t}^2\tilde{z}_j^2\right)\right] && \text{(F.54)}\\
:= &g(\{a_{jj}\}), && \text{(F.55)}
\end{aligned}$$

where each $\tilde{z}_j$ is a random variable drawn from standard Gaussian.

Now consider the minimum of the function $g(\{a_{jj}\})$ on the right hand side, under the constraints that $|a_{jj}| \leqslant 1$ and $\sum_j a_{jj}^2 \leqslant r$. Before finishing the proof of Lemma F.3, we have the following claim for this optimization.

**Claim F.4.** *There exist $\alpha, \alpha_t$ satisfying $0 \leqslant \alpha, \alpha_t \leqslant 1$ and $\alpha s + \sum_{t=1}^{T}\alpha_t(r-s) = \sum_j a_{jj}^2 \leqslant r$, such that the minimum of the above optimization (F.55) is achieved when $a_{jj}^2 = \alpha$ for any $j \in S$, and $a_{jj}^2 = \alpha_t$ for any $j \in P_t$ and $t \in [T]$.*

*Proof.* We need to prove that to achieve the minimum,

(1) $a_{\ell\ell}^2 = a_{\ell'\ell'}^2$ for any $\ell \neq \ell' \in S$;

(2) $a_{\ell\ell}^2 = a_{\ell'\ell'}^2$ for any $\ell \neq \ell' \in P_t$ and any $t \in [T]$;

For (1): By symmetry of $z_j$'s and the convexity of $\ell(\cdot)$, for any $t \in [T]$,

$$\begin{aligned}
&\mathbb{E}\left[\ell\left(\sum_{j\in R_t} a_{jj}^2 z_j^2\right)\right] && \text{(F.56)}\\
&= \frac{1}{2}\mathbb{E}\left[\ell\left(\sum_{j\in S, j\neq \ell, j\neq \ell'} a_{jj}^2 z_j^2 + a_{\ell\ell}^2 z_\ell^2 + a_{\ell'\ell'}^2 z_{\ell'}^2 + \sum_{j\in P_t} a_{jj}^2 z_j^2\right)\right] && \text{(F.57)}\\
&+ \frac{1}{2}\mathbb{E}\left[\ell\left(\sum_{j\in S, j\neq \ell, j\neq \ell'} a_{jj}^2 z_j^2 + a_{\ell\ell}^2 z_{\ell'}^2 + a_{\ell'\ell'}^2 z_\ell^2 + \sum_{j\in P_t} a_{jj}^2 z_j^2\right)\right] && \text{(F.58)}\\
&\geqslant \mathbb{E}\left[\ell\left(\sum_{j\in S, j\neq \ell, j\neq \ell'} a_{jj}^2 z_j^2 + \frac{a_{\ell\ell}^2 + a_{\ell'\ell'}^2}{2} z_{\ell'}^2 + \frac{a_{\ell\ell}^2 + a_{\ell'\ell'}^2}{2} z_\ell^2 + \sum_{j\in P_t} a_{jj}^2 z_j^2\right)\right]. && \text{(F.59)}
\end{aligned}$$

Then

$$\begin{aligned}
&g(\{a_{jj}\}) && \text{(F.60)}\\
&\geqslant \sum_{t=1}^{T} w_t \mathbb{E}\left[\ell\left(\sum_{j\in S, j\neq \ell, j\neq \ell'} a_{jj}^2 z_j^2 + \frac{a_{\ell\ell}^2 + a_{\ell'\ell'}^2}{2} z_{\ell'}^2 + \frac{a_{\ell\ell}^2 + a_{\ell'\ell'}^2}{2} z_\ell^2 + \sum_{j\in P_t} a_{jj}^2 z_j^2\right)\right]. && \text{(F.61)}
\end{aligned}$$

Therefore, the minimum is achieved when $a_{\ell\ell}^2 = a_{\ell'\ell'}^2$.

A similar argument as above proves statement (2). □

These statements mean that, for any $t \in [T]$, the minimum is achieved when $a_{jj}^2 = \alpha$ for $j \in S$, and $a_{jj}^2 = \alpha_t$ for $j \in P_t$, for some values $\alpha, \alpha_t \geqslant 0$. Let $Z = \sum_{j\in S} \tilde{z}_j^2, Z_t = \sum_{j\in P_t} \tilde{z}_j^2$. Then $Z \sim \chi_s^2$ and $Z_t \sim \chi_{r-s}^2$, and we have:

$$\begin{aligned}
g(\{a_{jj}\}) &= \sum_{t=1}^{T} w_t \mathbb{E}\left[\ell\left(\sum_{j\in S} \alpha\sigma_{S,t}^2 \tilde{z}_j^2 + \sum_{j\in P_t} \alpha_t \sigma_{R,t}^2 \tilde{z}_j^2\right)\right] && \text{(F.62)}\\
&= \sum_{t=1}^{T} w_t \mathbb{E}\left[\ell\left(\alpha\sigma_{S,t}^2 Z + \alpha_t \sigma_{R,t}^2 Z_t\right)\right]. && \text{(F.63)}
\end{aligned}$$

Given the constraint $\alpha s + \sum_{t=1}^{T} \alpha_t (r-s) = \sum_j a_{jj}^2 \leqslant r, 0 \leqslant \alpha, \alpha_t \leqslant 1$, we complete the proof of Lemma F.3. □

Given this result we can now analyze the generalization error when predicting on the target task $\mathcal{D}_t$.

**Lemma F.5.** *Consider any* $t \in [T]$. *Let* $\nu_{t,1} = \sum_{j \in S} (u_t^*)_j^2$ *and* $\nu_{t,2} = \sum_{j \in P_t} (u_t^*)_j^2$. *Suppose in* $\varphi^*$ (*calculated in Lemma F.3*), $\alpha, \alpha_t > 0$. *Suppose the prediction loss* $\ell_c$ *is* $L$-*Lipschitz.*

*Then the Empirical Risk Minimizer* $\hat{u}_t \in \mathcal{F}_{\varphi^*,t}$ *on* $\varphi^*$ *using* $m$ *labeled data points from* $\mathcal{D}_t$ *has risk*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}_t{}^\top \varphi^*(x), y)] \leqslant 8\sqrt{\frac{2\ln(4/\delta)}{m}}$$
$$+ 4L\sqrt{\frac{1}{m}\left(\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\alpha_t}\right)} \left(\sqrt{s\alpha\sigma_{S,t}^2 + (r-s)\alpha_t\sigma_{R,t}^2} + O\left(\sqrt{\frac{\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}^2 r}{s\alpha\sigma_{S,t}^2 + (r-s)\alpha_t\sigma_{R,t}^2}}\right)\right).$$

*Proof.* For any $t \in [T]$, we only need to bound the Rademacher complexity $\mathcal{R}_m(\mathcal{F}_{\varphi^*,t})$ of $\mathcal{F}_{\varphi^*,t}$; the statement then follows from standard generalization bounds,

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}_t{}^\top \varphi^*(x), y)] \leqslant 4L\mathcal{R}_m(\mathcal{F}_{\varphi^*,t}) + 8\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

Given the representation $\varphi^*$ in Lemma F.3, to ensure there exists a predictor in $\mathcal{F}_{\varphi^*,t}$ matching the ground-truth label, $f(\varphi^*(x)) = u_t^\top \varphi^*(x) = y = (u_t^*)^\top z_{R_t}$, predictor $u_t$ should satisfy

$$\begin{aligned}
\mathbb{E}_{\mathcal{D}_t}[(\hat{y}-y)^2] = 0 &\Leftrightarrow \forall z_{R_t}, \quad u_t{}^\top [QA^*, \mathbf{0}] M^{-1} M [z_{R_t}; \mathbf{0}; z_U] = u_t^{*\top} z_{R_t} && \text{(F.64)}\\
&\Leftrightarrow \forall z_{R_t}, \quad u_t{}^\top QA^* [z_{R_t}; \mathbf{0}] = u_t^{*\top} z_{R_t} && \text{(F.65)}\\
&\overset{(*)}{\Leftrightarrow} A^*_{1:r,1:r} (Q^\top)_{1:r,1:k} u_t = u_t^* && \text{(F.66)}\\
&\Leftrightarrow \forall v \in \mathbb{R}^r, \quad u_t = Q_{1:k,1:r} (A^*_{1:r,1:r})^{-1} u_t^* + Q_{1:k,r+1:k} v. && \text{(F.67)}
\end{aligned}$$

The $(*)$ is from non-zero variance for $z_{R_t}$. $u_t = Q_{1:k,1:r}(A^*_{1:r,1:r})^{-1}u^*_t$ is the least-norm optimal solution, so we have $B_{\phi^*,t} = \|Q_{1:k,1:r}(A^*_{1:r,1:r})^{-1}u^*_t\| = \sqrt{\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\alpha_t}}$. So the predictor class should be

$$\mathcal{F}_{\phi^*,t} = \left\{ f(\phi^*) = u_t^\top \phi^* : u_t \in \mathbb{R}^k, \|u_t\| \leqslant B_{\phi^*,t} = \sqrt{\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\alpha_t}} \right\}. \tag{F.68}$$

The empirical Rademacher complexity and Rademacher complexity of $\mathcal{F}_{\phi^*,t}$ with $m$ samples are

$$\begin{aligned}
\hat{\mathcal{R}}_m(\mathcal{F}_{\phi^*,t}) =& \frac{1}{m}\mathbb{E}_\sigma \left[ \sup_{f_{u,\phi} \in \mathcal{F}_{\phi^*,t}} \sum_{i=1}^m \sigma_i f_{u,\phi}(x^{(i)}) \right] && \text{(F.69)}\\
=& \frac{1}{m}\mathbb{E}_\sigma \left[ \sup_{\|u\| \leqslant B_{\phi^*,t}} \sum_{i=1}^m \sigma_i {u_t}^\top Q A^* [z^{(i)}_{R_t}; \mathbf{0}] \right] && \text{(F.70)}\\
=& \frac{1}{m}\mathbb{E}_\sigma \left[ \sup_{\|u\| \leqslant B_{\phi^*,t}} {u_t}^\top \sum_{i=1}^m \sigma_i Q_{1:k,1:r} A^*_{1:r,1:r} z^{(i)}_{R_t} \right] && \text{(F.71)}\\
=& \frac{B_{\phi^*,t}}{m}\mathbb{E}_\sigma \left[ \left\| \sum_{i=1}^m \sigma_i Q_{1:k,1:r} A^*_{1:r,1:r} z^{(i)}_{R_t} \right\| \right], && \text{(F.72)}
\end{aligned}$$

$$\begin{aligned}
\mathcal{R}_m(\mathcal{F}_{\phi^*,t}) =& \mathbb{E}_{z_R, z_U} \left[ \hat{\mathcal{R}}_m(\mathcal{F}_{\phi^*,t}) \right] && \text{(F.73)}\\
=& \frac{B_{\phi^*,t}}{m}\mathbb{E}_{z^{(i)}_{R_t}} \left[ \mathbb{E}_\sigma \left[ \left\| \sum_{i=1}^m \sigma_i Q_{1:k,1:r} A^*_{1:r,1:r} z^{(i)}_{R_t} \right\| \right] \right] && \text{(F.74)}\\
=& \frac{B_{\phi^*,t}}{m}\mathbb{E}_{z^{(i)}_{R_t}} \left[ \left\| A^*_{1:r,1:r} \sum_{i=1}^m z^{(i)}_{R_t} \right\| \right]. && \text{(F.75)}
\end{aligned}$$

For any $t \in [T]$, define $X_t := A^*_{1:r,1:r} \sum_{i=1}^m z^{(i)}_{R_t}$. Note that for $j \in S$, $X_{t,j} = \alpha \sum_{i=1}^m z^{(i)}_j$ is a Gaussian of mean zero and variance

$$\mathbb{E}[X^2_{t,j}] = \alpha \mathbb{E}\left[ \left( \sum_{i=1}^m z^{(i)}_j \right)^2 \right] = \alpha \mathbb{E}\left[ \sum_{i=1}^m \left( z^{(i)}_j \right)^2 \right] = m\alpha\sigma^2_{S,t}.$$

Similarly, for $j \in P_t$, $X_{t,j} = \alpha_t \sum_{i=1}^{m} z_j^{(i)}$ is a Gaussian of mean zero and variance $\mathbb{E}[X_{t,j}^2] = m\alpha_t \sigma_{R,t}^2$. Since $X_{t,j}$ is sub-gaussian, $X_{t,j}^2 - m\alpha\sigma_{S,t}^2$ for $j \in S$ and $X_{t,j}^2 - m\alpha_t\sigma_{R,t}^2$ for $j \in P_t$ are sub-exponential and more precisely

$$\|X_{t,j}^2 - m\alpha\sigma_{S,t}^2\|_{\psi_1} \leqslant C_1\|X_{t,j}^2\|_{\psi_1} = C_1\|X_{t,j}\|_{\psi_2}^2 \leqslant C_2 m\alpha\sigma_{S,t}^2, \quad j \in S, \tag{F.76}$$

$$\|X_{t,j}^2 - m\alpha_t\sigma_{R,t}^2\|_{\psi_1} \leqslant C_1\|X_{t,j}^2\|_{\psi_1} = C_1\|X_{t,j}\|_{\psi_2}^2 \leqslant C_2 m\alpha_t\sigma_{R,t}^2, \quad j \in P_t, \tag{F.77}$$

where $C_1, C_2$ are absolute constants and $C_2 > 1$. Let $K = \max(C_2 m\alpha\sigma_{S,t}^2, C_2 m\alpha_t\sigma_{R,t}^2) = C_2 m \max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}$ and $\mu := m(s\alpha\sigma_{S,t}^2 + (r-s)\alpha_t\sigma_{R,t}^2)$. By Bernstein's inequality, we have for every $\gamma \geqslant 0$ that

$$\mathbb{P}\left\{\left|\frac{1}{r}(\|X_t\|^2 - \mu)\right| \geqslant \gamma\right\} \leqslant 2\exp\left[-c\min\left(\frac{\gamma^2}{K^2}, \frac{\gamma}{K}\right) r\right] \tag{F.78}$$

$$\Rightarrow \mathbb{P}\left\{\left|\frac{\|X_t\|^2}{\mu} - 1\right| \geqslant \frac{r\gamma}{\mu}\right\} \tag{F.79}$$

$$\leqslant 2\exp\left[-\frac{c}{C_2^2}\min\left(\frac{\gamma^2}{m^2\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}^2}, \frac{\gamma}{m\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}}\right) r\right], \tag{F.80}$$

where $c$ is an absolute constant. For all numbers $z \geqslant 0$, we have $|z - 1| \geqslant \delta \Rightarrow |z^2 - 1| \geqslant \max(\delta, \delta^2)$. Thus, for any $\delta \geqslant 0$, we have

$$\mathbb{P}\left\{\left|\frac{\|X_t\|}{\sqrt{\mu}} - 1\right| \geqslant \delta\right\} \tag{F.81}$$

$$\leqslant \mathbb{P}\left\{\left|\frac{\|X_t\|_2^2}{\mu} - 1\right| \geqslant \max(\delta, \delta^2)\right\} \tag{F.82}$$

$$\leqslant 2\exp\left[-\frac{c}{C_2^2}\min\left(\left(\frac{\mu\max(\delta,\delta^2)}{m\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}r}\right)^2, \frac{\mu\max(\delta,\delta^2)}{m\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}r}\right) r\right] \tag{F.83}$$

$$\leqslant 2\exp\left[-\frac{c}{C_2^2}\left(\frac{\mu}{m\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}r}\right)^2 \min\left(\left(\max(\delta,\delta^2)\right)^2, \max(\delta,\delta^2)\right) r\right] \tag{F.84}$$

$$= 2\exp\left[-\frac{c}{C_2^2}\frac{\mu^2}{m^2\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}^2 r}\delta^2\right], \tag{F.85}$$

where the last inequality is from $\mu = m(s\alpha\sigma_{S,t}^2 + (r-s)\alpha_t\sigma_{R,t}^2) \leqslant m\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}r$. Changing variables to $\theta = \delta\sqrt{\mu}$, we obtain the desired sub-gaussian tail

$$\mathbb{P}\{|\|X_t\| - \sqrt{\mu}| \geqslant \theta\} \leqslant 2\exp\left[-\frac{c}{C_2^2}\frac{\mu}{m^2\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}^2 r}\theta^2\right]. \tag{F.86}$$

By generalization of integral identity, we have

$$\begin{aligned}|\mathbb{E}\,[\|X_t\| - \sqrt{\mu}]| =& \left|\int_0^\infty \mathbb{P}\{\|X_t\| - \sqrt{\mu} > \theta\}d\theta - \int_{-\infty}^0 \mathbb{P}\{\|X_t\| - \sqrt{\mu} < \theta\}d\theta\right| &\text{(F.87)}\\ \leqslant& 2\int_0^\infty \mathbb{P}\{|\|X_t\| - \sqrt{\mu}| > \theta\}d\theta &\text{(F.88)}\\ \leqslant& 4\int_0^\infty \exp\left[-\frac{c}{C_2^2}\frac{\mu}{m^2\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}^2 r}\theta^2\right]d\theta &\text{(F.89)}\\ \leqslant& C_3\frac{m\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}\sqrt{r}}{\sqrt{\mu}}, &\text{(F.90)}\end{aligned}$$

where $C_3$ is an absolute constant. Thus, we have

$$\begin{aligned}&\left|\mathcal{R}_m(\mathcal{F}_{\phi^*,t}) - \sqrt{\frac{1}{m}\left(\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\alpha_t}\right)(s\alpha\sigma_{S,t}^2 + (r-s)\alpha_t\sigma_{R,t}^2)}\right| &\text{(F.91)}\\ =& \frac{B_{\phi^*,t}}{m}|\mathbb{E}\,[\|X_t\| - \sqrt{\mu}]| &\text{(F.92)}\\ \leqslant& O\left(\sqrt{\frac{1}{m}\left(\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\alpha_t}\right)\frac{\max\{\alpha\sigma_{S,t}^2, \alpha_t\sigma_{R,t}^2\}^2 r}{s\alpha\sigma_{S,t}^2 + (r-s)\alpha_t\sigma_{R,t}^2}}\right). &\text{(F.93)}\end{aligned}$$

□

### F.2.2 Proofs of Proposition 7.3 and Proposition 7.4

Given the lemmas for the general case, we are now ready to prove the results in Proposition 7.3 and Proposition 7.4.

**Proposition F.6** (Restatement of Proposition 7.3). *Suppose* $\sigma_{S,t} = \sigma_{R,t} = \sigma_{U,t} = 1$ *for any* $t \in [T]$. *The representation* $\phi^*$ *obtained on an even mixture of data from all the tasks*

$\{\mathcal{D}_t : 1 \leqslant t \leqslant T\}$ *satisfies* $\phi^* \circ g(z) = Q\left(\sum_{j \in S} \sqrt{\alpha} z_j e_j + \sum_{j \in R \setminus S} \sqrt{\beta} z_j e_j\right)$ *for some* $\alpha \in [0,1],\ \beta = \min\left(1, \frac{r-\alpha s}{T(r-s)}\right)$*, where* $e_j$*'s are the basis vectors and* $Q$ *is any orthonormal matrix.*

*The Empirical Risk Minimizer* $\hat{u} \in \mathcal{F}_{\phi^*,t}$ *on* $\phi^*$ *using* $m$ *labeled data points from* $\mathcal{D}_t$ *has risk*

$$\begin{aligned}&\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}^\top \phi^*(x), y)]\\ \leqslant& 4L\sqrt{\frac{1}{m}\left(\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\beta}\right)}\left(\sqrt{s\alpha + (r-s)\beta} + O\left(\sqrt{\frac{r}{s\alpha + (r-s)\beta}}\right)\right) + 8\sqrt{\frac{2\ln(4/\delta)}{m}}.\end{aligned}$$

*Proof.* This follows from Lemma F.3, and considering the optimal $\alpha, \alpha_t$ for the following:

$$\begin{aligned} g(\{\alpha, \alpha_t\}) &= \sum_{t=1}^{T} w_t \mathbb{E}\left[\ell\left(\alpha\sigma_{S,t}^2 Z + \alpha_t \sigma_{R,t}^2 Z_t\right)\right] && \text{(F.94)}\\ &= \frac{1}{T}\sum_{t=1}^{T} \mathbb{E}\left[\ell\left(\alpha Z + \alpha_t Z_1\right)\right] && \text{(F.95)}\\ &\geqslant \mathbb{E}\left[\ell\left(\alpha Z + Z_1 \sum_{t=1}^{T} \frac{1}{T}\alpha_t\right)\right]. && \text{(F.96)}\end{aligned}$$

The second equation is from that $Z_t$'s follow the same distribution by the symmetry of $\tilde{z}_j$'s. The inequality comes from the convexity of $\ell(t)$ and Jensen's inequality. So the minimum is achieved when $\alpha_t := \beta$ for any $t \in [T]$, leading to

$$g(\{\alpha, \alpha_t\}) = \mathbb{E}\left[\ell\left(\alpha Z + \beta Z_1\right)\right] \tag{F.97}$$

subject to the constraints $\alpha s + T\beta(r-s) \leqslant r$, $0 \leqslant \alpha, \beta \leqslant 1$. Then we get $\phi^* \circ g(z) = W^* M z = Q\left(\sum_{j \in S} \sqrt{\alpha} z_j e_j + \sum_{j \in R \setminus S} \sqrt{\beta} z_j e_j\right)$ for some $\alpha \in [0,1],\ \beta = \min\left(1, \frac{r-\alpha s}{T(r-s)}\right)$, where $e_j$'s are the basis vectors and $Q$ is any orthonormal matrix.

Finally, the generalization bound follows from Lemma F.5, and that

$$O\left(\sqrt{\frac{\max\{\alpha,\beta\}^2 r}{s\alpha+(r-s)\beta}}\right) = O\left(\sqrt{\frac{r}{s\alpha+(r-s)\beta}}\right). \tag{F.98}$$

This completes the proof. □

**Proposition F.7** (Restatement of Proposition 7.4). *Suppose* $\sigma_{S,t}=\sigma_{R,t}=\sigma_{U,t}=1$. *The representation* $\phi_t^*$ *obtained on data from* $\mathcal{D}_t$ *satisfies* $\phi_t^*\circ g(z)=Q\left(\sum_{j\in R_t} z_j e_j\right)$ *where* $e_j$*'s are the basis vectors and* $Q$ *is any orthonormal matrix.*
*The Empirical Risk Minimizer* $\hat{u}\in\mathcal{F}_{\phi_t^*,t}$ *on* $\phi_t^*$ *using* $m$ *labeled data points from* $\mathcal{D}_t$ *has risk*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}^\top\phi_t^*(x),y)] \leqslant 4L\sqrt{\frac{r}{m}}\|u_t^*\| + 8\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

*While on task* $\mathcal{D}_i(i\neq t)$*, any linear predictor on* $\phi_t^*$ *has error at least* $\min_u \mathbb{E}_{\mathcal{D}_i}[\ell_c(u^\top z_S,y)]$.

*Proof.* Following Lemma F.3 (with $r=s$), we get $\phi_t^*\circ g(z)=Q\left(\sum_{j\in R_t} z_j e_j\right)$, where $e_j$'s are the basis vectors and Q is any orthonormal matrix. Following the same argument as in the proof of Lemma F.5, we get

$$\mathcal{R}_m(\mathcal{F}_{\phi^*,t}) = \frac{\|u_t^*\|}{m}\mathbb{E}_{z_{R_t}^{(i)}}\left[\left\|\sum_{i=1}^m z_{R_t}^{(i)}\right\|\right] \tag{F.99}$$

$$\leqslant \sqrt{\frac{r}{m}}\|u_t^*\|, \tag{F.100}$$

where the last inequality comes from the property of chi-squared distribution expectation. □

### F.2.3 Implication for the trade-off

The propositions then imply the trade-off between universality and label efficiency. Below we formalize the example discussed in Section 7.2.2.

**Proposition F.8** (A specific version of Proposition 7.3). *Suppose* $\sigma_{S,t} = \sigma_{R,t} = \sigma_{U,t} = 1$ *for any* $t \in [T]$ *and* $r = 2s$. *The representation* $\phi^*$ *obtained on an even mixture of data from all the tasks* $\{\mathcal{D}_t : 1 \leqslant t \leqslant T\}$ *satisfies* $\phi^* \circ g(z) = Q\left(\sum_{j \in S} z_j e_j + \sum_{j \in R \setminus S} \sqrt{\frac{1}{T}} z_j e_j\right)$, *where* $e_j$*'s are the basis vectors and* $Q$ *is any orthonormal matrix.*
*The Empirical Risk Minimizer* $\hat{u} \in \mathcal{F}_{\phi^*,t}$ *on* $\phi^*$ *using* $m$ *labeled data points from* $\mathcal{D}_t$ *has risk*

$$\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}^\top \phi^*(x), y)] \leqslant 4L\sqrt{\frac{1}{m}\left(\nu_{t,1} + T\nu_{t,2}\right)}\left(\sqrt{r\left(\frac{T+1}{2T}\right)} + O\left(1\right)\right) + 8\sqrt{\frac{2\ln(4/\delta)}{m}}.$$

*Proof.* This follows from Proposition 7.3, and noting that when $r = 2s$, $\alpha = 1$ and $\beta = 1/T$ are the optimal for:

$$g(\{\alpha, \beta\}) = \mathbb{E}\left[\ell\left(\alpha Z + \beta Z_1\right)\right] \tag{F.101}$$

$$= \mathbb{E}\left[\ell\left(\alpha Z + \frac{r - \alpha s}{T(r - s)} Z_1\right)\right] \tag{F.102}$$

$$= \mathbb{E}\left[\ell\left(\alpha Z + \frac{2 - \alpha}{T} Z_1\right)\right] \tag{F.103}$$

subject to the constraints $\alpha s + T\beta(r - s) \leqslant r$, $0 \leqslant \alpha, \beta \leqslant 1$. To see this, note that $Z \sim \chi^2_s$ and $Z_1 \sim \chi^2_{r-s} = \chi^2_s$ follow the same distribution, so $\alpha Z + \frac{2-\alpha}{T} Z_1$ for $\alpha = 1$ will stochastically dominate its value for other $\alpha \in [0, 1)$. The optimal is then achieved when $\alpha = 1$ and $\beta = \frac{2-\alpha}{T} = \frac{1}{T}$. □

### F.2.4 Improving the Trade-off by Contrastive Regularization

The above analysis shows that contrastive learning a representation on unlabeled data from the target task can help in prediction on this target task. This suggests that given a representation $\phi^*$ pre-trained on diverse data, one can fine-tune it by contrastive learning on some unlabeled data from the target task to get a representation that can lead to better prediction on the target task. In the following, we will formally show that this is indeed the case for the illustrative example in Section 7.2.2.

Recall that in this example, $\sigma_{S,t} = \sigma_{R,t} = \sigma_{U,t} = 1$, $r = 2s$, and $\nu_{t,1} = \nu_{t,2}$. The representation $\phi^*$ obtained on an even mixture of data from all the tasks $\{\mathcal{D}_t : 1 \leqslant t \leqslant T\}$ satisfies $\phi^* \circ g(z) = Q\left(\sum_{j \in S} \sqrt{\alpha} z_j e_j + \sum_{j \in R \setminus S} \sqrt{\beta} z_j e_j\right)$, where $e_j$'s are the basis vectors and $Q$ is any orthonormal matrix, and $\alpha = 1, \beta = \frac{1}{T}$.

Now, suppose we are given unlabeled data from $\mathcal{D}_t$, and we use them to finetune $\phi^*(x) = W^* x$ by contrastive learning on these unlabeled data. That is, we find $W$ near $W^*$ to minimize the contrastive loss on the unlabeled data from $\mathcal{D}_t$:

$$\min_{\phi(x) = Wx} \mathbb{E}\left[\ell\left(\phi(x)^\top (\phi(x^+) - \mathbb{E}_{x^-} \phi(x^-))\right)\right] \tag{F.104}$$

$$\text{subject to} \quad \|W - W^*\|_F \leqslant \gamma, \|W\| \leqslant 1. \tag{F.105}$$

for some small $\gamma > 0$.

**Proposition F.9.** *For (F.104), $\phi^*_{CR,t}$ satisfying the following on $x$ from $\mathcal{D}_t$ is an optimal representation:*

$$\phi^*_{CR,t} \circ g(z) = Q\left(\sum_{j \in S} \sqrt{\alpha} z_j e_j + \sum_{j \in P_t} \sqrt{\beta} z_j e_j\right)$$

*where $\sqrt{\alpha} = 1, \sqrt{\beta} = \min\left(1, \sqrt{\frac{1}{T} + \frac{\gamma}{\sqrt{s}}}\right)$.*

*Proof.* Following the argument in Lemma F.3, we still have that $\phi^*_{CR,t}(x) = Wx$ where $W = Q_2[A_2; \mathbf{0}]M^{-1}$ for any orthonormal matrix $Q_2$ and some diagonal matrix $A_2 = \text{diagonal}(a_{jj})$, with $a_{jj} = \sqrt{\alpha}$ for $j \in S$ and $a_{jj} = \sqrt{\beta}$ for $j \in P_t$ for some $\alpha, \beta \in [0, 1]$. And the contrastive loss is:

$$\mathbb{E}\left[\ell\left(\phi(x)^\top (\phi(x^+) - \mathbb{E}_{x^-} \phi(x^-))\right)\right] = \mathbb{E}\left[\ell\left(\sum_{j \in R_t} a_{jj}^2 z_j^2\right)\right] \tag{F.106}$$

$$= \mathbb{E}\left[\ell\left(\alpha \sum_{j \in S} z_j^2 + \beta \sum_{j \in P_t} z_j^2\right)\right]. \tag{F.107}$$

Recall that $\phi^*(x) = W^* x$ with $W = Q[A; \mathbf{0}]M^{-1}$ for any orthonormal matrix $Q$ and some diagonal matrix $A$, with $A_{jj} = 1$ for $j \in S$ and $A_{jj} = \sqrt{1/T}$ for $j \in R_i \setminus S$ for any $i \in [T]$. Then

$$\begin{aligned} \|W - W^*\|_F &= \|Q_2[A_2; \mathbf{0}]M^{-1} - Q[A; \mathbf{0}]M^{-1}\|_F && \text{(F.108)} \\ &= \|Q_2 A_2 - QA\|_F && \text{(F.109)} \\ &= \|A_2 - Q_2^{-1} Q A\|_F. && \text{(F.110)} \end{aligned}$$

Since $Q_2^{-1}Q$ is a rotation and $A, A_2$ are diagonal, we can always set $Q_2 = Q$ without increasing $\|W - W^*\|_F$. Then

$$\begin{aligned} \|W - W^*\|_F^2 &= \|A_2 - A\|_F^2 && \text{(F.111)} \\ &= s(\sqrt{\alpha} - 1)^2 + s(\sqrt{\beta} - \sqrt{1/T})^2 + \sum_{j \in P_i, i \neq t} ((A_2)_{jj} - \sqrt{1/T})^2. && \text{(F.112)} \end{aligned}$$

To minimize the contrastive loss, we need $\alpha, \beta$ to be as large as possible, subject to $\|W - W^*\|_F^2 \leqslant \gamma^2$, and $\alpha, \beta, (A_2)_{jj}^2 \in [0, 1]$. The optimal is then achieved when $\alpha = 1, \sqrt{\beta} = \min\left(1, \sqrt{\frac{1}{T}} + \frac{\gamma}{\sqrt{s}}\right)$, and $(A_2)_{jj} = \sqrt{1/T}$ for $j \in P_i, i \neq t$. □

Now, recall that by Proposition 7.3, the ERM has risk:

$$\begin{aligned} &\mathbb{E}_{(x,y)\sim\mathcal{D}_t}[\ell_c(\hat{u}^\top \phi^*(x), y)] \\ &\leqslant 4L\sqrt{\frac{1}{m}\left(\frac{\nu_{t,1}}{\alpha} + \frac{\nu_{t,2}}{\beta}\right)}\left(\sqrt{s\alpha + (r-s)\beta} + O\left(\sqrt{\frac{r}{s\alpha + (r-s)\beta}}\right)\right) + 8\sqrt{\frac{2\ln(4/\delta)}{m}} \\ &= 4L\sqrt{\frac{1}{m}\left(\frac{\|u_t^2\|^2}{2\alpha} + \frac{\|u_t^2\|^2}{2\beta}\right)}\left(\sqrt{s\alpha + s\beta} + O\left(1\right)\right) + 8\sqrt{\frac{2\ln(4/\delta)}{m}} \\ &= O\left(L\sqrt{\frac{r\|u_t^*\|^2}{m}\left(2 + \frac{\alpha}{\beta} + \frac{\beta}{\alpha}\right)}\right) \end{aligned}$$

With the fine-tuning using contrastive learning, in the representation learned, $\alpha$ remains to be 1, while $\beta$ increases from $1/T$ to $(\sqrt{1/T} + \gamma/\sqrt{s})^2$. Then the error bound decreases. This shows that fine-tuning with contrastive learning on unlabeled data from the target task can emphasize the task-specific features $z_{P_t}$, which then leads to better prediction performance.

## F.3 More Experimental Details and Results

### F.3.1 Datasets

**CIFAR-10.** CIFAR-10 (Krizhevsky et al., 2009) dataset consists of 60,000 $32 \times 32$ color images in 10 classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck. Each class has 6,000 images. There are 50,000 training images and 10,000 test images.

**CINIC-10.** CINIC-10 (Darlow et al., 2018) consists of $32 \times 32$ color images from both CIFAR and ImageNet and has 90,000 training images with ten classes identical to CIFAR-10.

**ImageNet.** ImageNet (Deng et al., 2009) is a huge visual dataset which is composed of 1,281,167 training data and 50,000 test data with 1,000 classes. We crop each color image to $224 \times 224$ as the standard setting.

**ImageNet32.** ImageNet32 (Deng et al., 2009) is a huge dataset made up of small color images called the down-sampled version of ImageNet. ImageNet32 comprises 1,281,167 training data and 50,000 test data with 1,000 classes. Each color image is down-sampled to $32 \times 32$.

**ImageNet22k.** ImageNet22k (Deng et al., 2009; Ridnik et al., 2021) is a superset of ImageNet which contains 14.2M training data and 522,500 test data with 21,841 classes. We crop each color image to $224 \times 224$ as the standard setting.

**ImageNet-Bird.** The ImageNet-Bird is a subset of ImageNet and contains all bird-related images from ImageNet, which have 59 classes and 76k training images.

**ImageNet-Vehicle.** The ImageNet-Vehicle is a subset of ImageNet and contains all vehicle-related images from ImageNet, which have 43 classes and 55k training images.

**ImageNet-Cat/Ball/Shop/Clothing/Fruit.** The ImageNet-Cat/Ball/Shop/Clothing /Fruit is a subset of ImageNet and contains all cat/ball/shop/clothing/fruit-related images from ImageNet, which have 76 classes and 100k training images.

**GCC-15M.** GCC-15M denotes the merged version of GCC-3M (Sharma et al., 2018) and GCC-12M (Changpinyo et al., 2021). It is a diverse dataset of visual concepts with image-text pairs meant to be used for vision-and-language pre-training. GCC-15M contains 15M training data and more than 600k concepts.

**SVHN.** The Street View House Numbers (Netzer et al., 2011) contains 10 digits color images of size 32 × 32 in the natural scene. It has 73,257 digits for training and 26,032 digits for testing.

**MNIST.** The Modified National Institute of Standards and Technology (LeCun et al., 1998) is a database of handwritten gray-scale digits of size 28 × 28. It contains 60,000 training images and 10,000 testing images.

**EMNIST.** Extended MNIST (Cohen et al., 2017) includes gray images of handwritten letters and digits. The images in EMNIST were converted into the same size 28 × 28 by the same process as MNIST. EMNIST-Letters has 145,600 lowercase characters with 26 balanced classes, and EMNIST-Digits has 280,000 characters with ten balanced classes.

**Fashion-MNIST.** Fashion-MNIST (Xiao et al., 2017) is a dataset of 28 × 28 gray-scale images with ten classes: T-shirt/top, trouser, pullover, dress, coat, sandal, shirt, sneaker, bag, ankle boot. The training set size is 60,000, and the test set size is 10,000.

**Fer2013.** Fer2013 is a dataset (Goodfellow et al., 2013) of 48 × 48 gray-scale images with 7 face expression classes: angry, disgust, fear, happy, sad, surprise, neutral. The training set size is 28,709, and the test set size is 3,589.

**FaceScrub.** FaceScrub (Ng and Winkler, 2014) is a dataset with 141,130 color face images of 695 public figures.

**GTSRB.** The German Traffic Sign Recognition Benchmark (Stallkamp et al., 2012) is a dataset of color images depicting 43 different traffic signs. The images are not of fixed dimensions and have a rich background and varying light conditions as expected of photographed images of traffic signs. The original training set contains 34,799 images, and the original test set contains 12,630 images. We resize each image to 32×32. The dataset has a significant imbalance in the number of sample occurrences across classes. We use data augmentation techniques to enlarge the training data and balance the number of samples in each class. We construct a class-preserving data augmentation pipeline consisting of rotation, translation, and projection transforms and apply this pipeline to the training images until each class contains 2,500 examples. So we construct a new training set containing 107,500 images in total. We also construct a new test set by randomly selecting 10,000 images from the original test set for evaluation.

**IMDB.** IMDB (Maas et al., 2011) is a large movie review text dataset. The dataset is for binary sentiment classification, positive review or negative review. The dataset contains 25,000 movie reviews for training and 25,000 for testing.

**AGNews.** AGNews (Zhang et al., 2015) is a sub-dataset of AG's corpus of news articles for text topic classification. It covers the 4 largest classes: world, sports, business, sci/tech. The AG News contains 30,000 training and 1,900 test samples per class.

### F.3.2 Verifying the Existence of the Trade-off

**Model.** We evaluate three popular contrastive learning frameworks, MoCo v2 (He et al., 2020b), NNCLR (Dwibedi et al., 2021) and SimSiam (Chen and He, 2021) . MoCo v2 can be regarded as SimCLR (Chen et al., 2020d) equipped with a memory bank, while NNCLR uses nearest-neighbor as the positive pairs. SimSiam can be regarded as a modification from BYOL (Grill et al., 2020) similar to Barlow Twins (Zbontar et al., 2021), which does not need negative pairs. We follow the same data augmentation methods as SimSiam (Chen and He, 2021) for all datasets.

**Datasets.** We consider three sets of data. In the first set, our downstream task is CIFAR-10, and the pre-training datasets may include CINIC-10, SVHN, GTSRB, and ImageNet32. CINIC-10 has classes identical to CIFAR-10 and is the most target-relevant, while the others are less similar to CIFAR-10. This then provides more and more diverse pre-training data w.r.t. the target task. In the second set, our downstream task is MNIST, and the pre-training datasets may include EMNIST-Digits&Letters, Fashion-MNIST, GTSRB, and ImageNet32. Here, the handwritten dataset EMNIST-Digits&Letters is the most target-relevant. In the last set, our downstream task is Fer2013, a face expression classification dataset. The pre-training datasets may include FaceScrub, CIFAR-10, SVHN, and ImageNet32, where the face dataset Facescrub is the most target-relevant.

**Evaluation & Methods.** We pre-train a ResNet18 network (He et al., 2016) as a feature extractor under different contrastive learning methods using SGD for 800 epochs with a cosine learning-rate scheduler, the base learning rate of 0.06, weight decay 5e-4, momentum 0.9 and batch size 512. Then we fix the pre-trained feature extractor and train a linear classifier (Linear Probing, LP) on 1%, 5%, 10%, 20%, 100% of the labeled data from the downstream task. For LP we use SGD for 200 epochs with a cosine learning-rate scheduler, the base learning rate of 5.0, no weight decay, momentum 0.9, and batch size 256. We finally report the test accuracy on a specific target task and the weighted average test accuracy on all pre-training datasets (i.e., using them as the downstream tasks). We use the class number of each pre-training dataset as the weight, which is consistent with random guessing as a baseline. We have three types of models pre-trained on three sets of datasets. Thus, we have nine tasks in total. For each task, we have two pre-trained models initialized by different random seeds. We evaluated each model three times and we report the average test accuracy with standard deviation based on multiple runs (six evaluations).

In Figs. F.2(a)(b)(c), we report results for MoCo v2, NNCLR, SimSiam (respectively) on CIFAR-10 as the downstream task. The size and diversity of unlabeled data for pre-training are increased on the $x$-axis by incrementally adding datasets in the following order: CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (only use a 500k subset)(I). Then, we do LP on CIFAR-10 using different proportions of labeled samples. Similarly, in Figs. F.2(d)(e)(f), we report results for three models on MNIST. We incrementally add pretraining datasets in the following order: EMNIST-Digits&Letters (E), Fashion-MNIST (F), GTSRB (G), ImageNet32 (I). In Figs. F.2(g)(h)(i), the downstream task is Fer2013 and we incrementally add datasets in the following order: FaceScrub (I), CIFAR-10 (C), SVHN (S), ImageNet32 (I).

**Results.** In Figs. F.2, when the pre-training data becomes more diverse, the average test accuracy on all pre-training datasets increases (i.e., universality improves), while the test accuracy on the specific target task decreases (i.e., label efficiency drops). This shows a clear trade-off between universality and label efficiency. Moreover, with fewer labeled data (from the green line to the red line), the trade-off phenomenon will be more significant. It supports our claim that diverse pre-training data allow learning diverse features for better universality, but can down-weight the features for a specific task resulting in worse prediction. As more diverse unlabeled data are included, more labeled data from the target task is needed to achieve comparably-good prediction accuracy. This validates our analysis of the trade-off in Section 7.2.2.

In Figs. F.2(a)(b)(d)(e)(f)(h), the average test accuracy ($x$-axis) decreases in the end because when we add one pre-training dataset, it may hurt the test accuracy of all other pre-training datasets. In Figs. F.2(g)(h), the downstream task test accuracy ($y$-axis) increases in the beginning because when the pre-training unlabeled data from relevant tasks is not sufficiently large, introducing other pre-training datasets will help the model to learn features relevant for the downstream task. However, the downstream task test accuracy will drop later as in other figures.

Please refer to Appendix F.3.5 for more figures.

### Larger Scale Experiments

The datasets involved are ImageNet (1.2M data points, 1k classes), ImageNet22k (14M, 22k classes), and GCC-15M (15M). We compare two UniCL representations (Yang et al., 2022): the more specific representation pre-trained on ImageNet, and the one pre-trained on the more diverse dataset ImageNet+GCC-15M. We compare their performance in two tasks: classification on ImageNet (using 2k labeled data) and classification on ImageNet22k (using 44k labeled data).

The results are reported in Table F.1. From the specific representation to the diverse one, we observe that the test accuracy on ImageNet decreases (i.e., efficiency drops), while the test accuracy over ImageNet22k increases (i.e., universality improves). This again confirms the trade-off.

| LP Accuracy | **Pre-training** dataset | |
|---|---|---|
| **Target** dataset | ImageNet | ImageNet+GCC-15M |
| ImageNet (2k label) | 79.05 | 77.66 |
| ImageNet22k (44k label) | 9.02 | 9.86 |

Table F.1: LP test accuracy on ImageNet and ImageNet22k with UniCL (Swin-T) pre-trained 500 epochs on ImageNet and ImageNet+GCC-15M.

### F.3.3 Inspecting the Trade-off

#### Feature Similarity

For each set of pre-training data, we extract a set of features for the target task, CIFAR-10, MNIST, and Fer2013 respectively, using the pre-trained representation function. In Fig. F.3 (rows/columns are pre-training data; numbers/colors show the similarity) first row, the features from different pre-training datasets have low similarities. This is consistent with our setup in Section 7.2.2 that different tasks only share some features and each owns many private ones. In the second row, we can see a decreasing trend of similarity in each row of each sub-figure. This indicates when gradually adding more diverse pre-training data, the obtained representation will encode more downstream task-irrelevant information and become less similar to that before adding more pre-training data. It will hurt the downstream task performance. This result is consistent with our Proposition 7.3 and 7.4.

Finally, we would like to verify in Theorem 7.2 via CKA similarities. The theorem says that when we increase the norm bound, the representation can encode more and more features. To verify this, our experiments will vary the weight decay regularization coefficient (larger weight decay corresponds to a smaller norm bound and fewer features learned in the representation). Fig. F.4 shows that the linear CKA similarity decreases with the increase of the weight decay, then it provides some support for the theorem.

#### The Effect of Pre-training and Labeled Data Sizes

We have also conducted finer-grained investigations into the trade-off by varying the pre-training dataset size and the downstream labeled data on the specific task. Table F.2 shows the results for different pre-training datasets (ImageNet-Bird, ImageNet, and 10% of ImageNet data) and different labeled datasets (500 to 8k samples from ImageNet-Bird, and the whole ImageNet). Table F.3 shows the results for a similar setting with ImageNet-Vehicle. Table F.4 shows the results of UniCL with Swin-T backbone using different pre-training datasets (ImageNet, and ImageNet+GCC-15M) and different labeled datasets (2k samples from ImageNet, 44k samples from ImageNet22k, the whole ImageNet, and the whole ImageNet22k).

First, we find that the trade-off is hidden when a small amount of data from the specific task is used for pre-training. The results show that when the specific pre-training data is small, the representation learned is noisy and the downstream prediction performance is poor. This is not surprising: in the extreme case with only 1 pre-training image from the bird task or vehicle task, there is essentially no information for pre-training. In this case, the results are well inside the Pareto front of the trade-off curve and thus cannot demonstrate the trade-off.

Second, we find that the trade-off is hidden when a large amount of labeled data are available for learning predictors on the representation in the specific task. If a large amount of labeled data is available for training the predictor, the prediction performance is similar when using the specific or universal representations. This is consistent with the insights from our analysis: when pre-training on diverse data, the features specific to the target task are down-weighted but can still be in the representation, then with a large amount of labeled data from the specific task, the sample complexity issue is not significant, and thus the trade-off is hidden.

The above experimental studies show that the trade-off is revealed when we have *large-scale pre-training data* and *a limited amount of labeled data from the target task*, which is indeed the typical interesting case for using pre-trained representations (especially the large foundation models). The trade-off is significant in this case and thus crucial for further development of pre-training representations.

| LP Accuracy | **Pre-training** dataset | | |
|---|---|---|---|
| **Target** dataset | ImageNet-Bird | ImageNet | 10% ImageNet |
| ImageNet-Bird (500 label) | 76.00 | 58.78 | 30.06 |
| ImageNet-Bird (1k label) | 81.42 | 69.39 | 35.96 |
| ImageNet-Bird (2k label) | 82.88 | 79.66 | 39.49 |
| ImageNet-Bird (4k label) | 83.83 | 83.59 | 44.13 |
| ImageNet-Bird (8k label) | 84.71 | 85.93 | 48.50 |
| ImageNet (all label) | 41.38 | 73.20 | 54.08 |

Table F.2: LP test accuracy on ImageNet-Bird and ImageNet with MoCo v3 (ViT-S) pre-trained on ImageNet-Bird and ImageNet.

| LP Accuracy | **Pre-training** dataset | | |
|---|---|---|---|
| **Target** dataset | ImageNet-Vehicle | ImageNet | 10% ImageNet |
| ImageNet-Vehicle (500 label) | 61.81 | 57.86 | 31.48 |
| ImageNet-Vehicle (1k label) | 70.93 | 67.90 | 37.76 |
| ImageNet-Vehicle (2k label) | 72.09 | 69.81 | 39.07 |
| ImageNet-Vehicle (4k label) | 74.14 | 72.93 | 39.62 |
| ImageNet-Vehicle (8k label) | 75.53 | 74.55 | 43.53 |
| ImageNet (all label) | 39.84 | 73.20 | 54.08 |

Table F.3: LP test accuracy on ImageNet-Vehicle and ImageNet with MoCo v3 (ViT-S) pre-trained on ImageNet-Vehicle and ImageNet.

| LP Accuracy | **Pre-training** dataset | |
|---|---|---|
| **Target** dataset | ImageNet | ImageNet+GCC-15M |
| ImageNet (2k label) | 79.05 | 77.66 |
| ImageNet22k (44k label) | 9.02 | 9.86 |
| ImageNet (all label) | 79.61 | 81.37 |
| ImageNet22k (all label) | 30.90 | 36.69 |

Table F.4: LP test accuracy on ImageNet and ImageNet22k with UniCL (Swin-T) pre-trained 500 epochs on ImageNet and ImageNet+GCC-15M.

### More Ablation Studies

We report results from three ablation studies: (1) varying the class number of ImageNet32, (2) varying the percentage of target-relevant pre-training data, and (3) replacing CINIC-10 with CIFAR-10 in the pre-training dataset. The results consistently show the trade-off.

**Varying the Class Number of ImageNet32.** To further support **A1**, we show that the trade-off between universality and label efficiency also exists under a fixed dataset size. In Fig. F.5, we pre-train MoCo v2 and SimSiam on CIFAR-10 +

ImageNet32(200k) and keep the same setting as Fig. F.2 except that we vary the class number of ImageNet32(200k). In previous experiments, we randomly pick 500,000 images from ImageNet32 without considering labels. Here, we fix the number of classes to 50, 100, 200, 500, 1000. Then we randomly sample 200,000 images from the subset of classes. The downstream task is CIFAR-10. In Fig. F.5, we observe that with a fixed pre-training datasets size, e.g., 250,000, when the data is more diverse, the pre-training will learn more irrelevant features, and the performance will drop on the downstream task. This supports our analysis as well.

**Varying Target-Relevant pre-training Data Percentage.** In Fig. F.6, we use (a)(d) 100% (b)(e) 50% (c)(f) 20% CINIC-10 to train MoCo v2 and SimSiam, and keep the same setting as Fig. F.2. For Fig. F.6 (b) with 50% CINIC-10, test accuracy drops, e.g., the test accuracy of 1% CIFAR-10 in Fig. F.6 (a) 80.63% vs. (b) 76.45%. We can also see the decreasing curve in Fig. F.6 (b). On the other hand, we also have test accuracy drops in Fig. F.6 (c)(e)(f). However, we can see a U-curve rather than a strictly decreasing curve in Fig. F.6 (c)(e)(f). ImageNet32 is more relevant with CIFAR-10 than SVHN and GTSRB, consistent with human intuition. When we have a small partition of CINIC-10 that does not cover all target-relevant features, the feature extractor can learn these missing features from ImageNet32. Although there are many irrelevant features in ImageNet32, the positive effect is larger than the negative effect, and so it plots a U-curve. It is consistent with our statement that we need a large and target-relevant pre-training dataset rather than a diverse irrelevant one.

**Replacing CINIC-10 With CIFAR-10.** In Fig. F.7, we keep the same setting as Fig. F.2 except we replace CINIC-10 with CIFAR-10. Note that our downstream task is still CIFAR-10. In Fig. F.7, we can see the same phenomena and similar performance as Fig. F.2. Thus, if we have a good choice of a task-relevant pre-training dataset, we can get a similar performance as pre-training on the downstream task domain directly.

The pre-training unlabeled data from diverse tasks may have a positive effect when the pre-training unlabeled data from similar (relevant) tasks is not sufficiently large. Moreover, if we choose a good task-relevant pre-training dataset, we can directly get a similar performance as pre-training on the downstream task. However, when the task-relevant dataset is sufficient, the performance will drop if we introduce task-irrelevant data in the pre-training dataset (Fig. F.6 (a)(d)).

### F.3.4 Improving the Trade-off: Finetune with Contrastive Regularization

| **Pretrain** data | Method | 1% label data | 5% label data | 10% label data | 20% label data | 100% label data |
|---|---|---|---|---|---|---|
| CINIC10 | LP | 80.63±0.01 | **84.42±0.01** | 85.88±0.05 | 86.75±0.01 | 88.41±0.01 |
| | FT | 68.74±0.46 | 81.46±0.34 | 85.20±0.14 | 88.69±0.29 | 93.58±0.14 |
| | Ours | **83.66±0.43** | 83.04±0.08 | **86.18±0.24** | **89.61±0.12** | **94.51±0.02** |
| +SVHN | LP | 77.83±0.01 | 81.43±0.04 | 82.95±0.01 | 83.53±0.01 | 85.18±0.01 |
| | FT | 66.01±0.52 | 80.20±0.07 | 83.96±0.46 | 88.01±0.07 | 93.35±0.10 |
| | Ours | **79.95±0.36** | **81.77±0.11** | **84.98±0.02** | **88.97±0.14** | **94.26±0.01** |
| +GTSRB | LP | 75.64±0.01 | 78.18±0.01 | 79.80±0.02 | 79.81±0.01 | 82.07±0.01 |
| | FT | 62.79±0.57 | 78.90±0.41 | 83.89±0.03 | 87.85±0.03 | 93.42±0.13 |
| | Ours | **76.20±0.36** | **80.26±0.05** | **84.11±0.06** | **88.74±0.01** | **94.32±0.13** |
| +ImageNet32 | LP | 68.26±0.01 | 71.04±0.01 | 73.29±0.02 | 73.73±0.02 | 75.64±0.03 |
| | FT | 65.40±0.16 | **78.99±0.21** | 82.96±0.17 | 86.75±0.10 | 92.92±0.06 |
| | Ours | **71.70±1.20** | 78.94±0.06 | **83.48±0.02** | **87.77±0.13** | **93.66±0.12** |

Table F.5: Test accuracy on CIFAR-10 with different evaluation methods on MoCo v2 under different percentages of labeled data. From top to bottom: incrementally add datasets for pre-training.

**Evaluation & Methods.** We use the feature extractor (ResNet18) pre-trained as in Section 7.3.1 by MoCo v2. Then, we report three evaluation methods, Linear Probing (LP), Finetune (FT), and Finetune with Contrastive Regularization (Ours) on CIFAR-10 under different percentages of labeled data as in Appendix F.3.2. LP follows the training protocol in Section 7.3.1. FT and Ours learn a linear predictor and update the representation, and use the same data augmentation for a fair comparison. FT follows the training in MAE (He et al., 2022) mostly, using SGD for 200 epochs with a cosine learning-rate scheduler, the base learning rate of 0.06, weight decay 5e-4, momentum 0.9 and batch size 256. Moreover, Ours uses the contrastive loss from MoCo v2 and regularization coefficient $\lambda = 0.1$.

**Results.** In Table F.5, the trade-off phenomenon also exists for FT evaluation, where the FT test accuracy drops when the pre-training dataset contains more diverse data points. Table F.5 shows that Ours can achieve better performance than the other baselines. In particular, it outperforms the typical fine-tuning method consistently, even when the latter also uses the same amount of data augmentation. This confirms the benefit of contrastive regularization. Fig. F.8 visualizes the features of different methods by t-SNE. It shows that contrastive regularization can downweight the downstream task-irrelevant invariant feature, so it can improve the model generalization ability, which is consistent with the discussion in Section 7.2.2.

| **Target** data | Method | Aug | 1% label data | 5% label data | 10% label data | 20% label data | 100% label data |
|---|---|---|---|---|---|---|---|
| ImageNet | LP | 1 | 66.04±0.26 | 76.28±0.05 | 78.06±0.02 | 78.73±0.03 | 77.84±0.02 |
| | FT | 1 | 71.09±0.05 | 75.32±0.11 | 76.38±0.01 | 76.92±0.10 | 82.97±0.02 |
| | FT | 10 | 73.28±0.01 | 76.43±0.14 | 78.69±0.04 | 81.56±0.10 | 83.65±0.01 |
| | FT | 100 | 71.83±0.05 | 77.55±0.07 | 80.16±0.09 | 82.23±0.01 | 83.58±0.05 |
| | Ours | 10 | **74.36±0.09** | **78.60±0.03** | 80.02±0.08 | 81.28±0.07 | 84.94±0.09 |
| | Ours | 100 | 73.03±0.04 | 78.41±0.04 | **80.48±0.05** | **82.42±0.03** | **85.34±0.07** |
| SVHN | LP | 1 | 59.24±0.02 | 57.25±0.08 | 56.32±0.17 | 58.35±0.08 | 63.44±0.01 |
| | FT | 1 | 55.20±0.16 | 61.73±0.32 | 64.91±0.12 | 64.70±0.64 | 65.76±0.10 |
| | FT | 10 | 61.24±0.08 | 65.89±0.54 | 68.61±0.52 | 70.89±0.07 | 78.22±0.18 |
| | FT | 100 | 65.23±0.03 | 72.34±0.07 | 75.13±0.11 | 77.20±0.03 | 80.13±0.14 |
| | Ours | 10 | 62.87±0.42 | 67.56±0.23 | 70.16±0.02 | 72.23±0.09 | 78.72±0.37 |
| | Ours | 100 | **65.30±0.43** | **72.61±0.07** | **75.34±0.12** | **77.64±0.05** | **82.50±0.01** |
| GTSRB | LP | 1 | 71.20±0.61 | 83.97±0.08 | 85.31±0.02 | 86.34±0.06 | 86.56±0.01 |
| | FT | 1 | 77.18±0.40 | 87.65±0.18 | 88.56±0.75 | 88.76±0.01 | 89.83±0.39 |
| | FT | 10 | 84.55±0.30 | 88.74±0.32 | 89.11±0.11 | 89.52±0.12 | 90.74±0.06 |
| | FT | 100 | 86.28±0.26 | 90.50±0.29 | 90.83±0.08 | 91.15±0.01 | 91.89±0.12 |
| | Ours | 10 | 84.84±0.33 | 89.53±0.13 | 89.44±0.01 | 91.35±0.12 | 92.01±0.28 |
| | Ours | 100 | **87.24±0.51** | **90.86±0.05** | **91.29±0.26** | **92.11±0.15** | **92.65±0.10** |

Table F.6: Test accuracy for different evaluation methods on different datasets using foundation model CLIP (backbone ViT-L). We do not use data augmentation for LP. We evaluate FT without data augmentation, with 10 augmentation and with 100 augmentation to each training images. For Ours, we use 10, 100 augmentation.

### Larger Foundation Models

We verify our method's effectiveness in real-world scenarios. When users use foundation models, they cannot access the model and can only call the official API, (e.g. GPT-3, DALL·E (Ramesh et al., 2022)). The pricing for GPT-3 to get feature embedding is $0.20 / 1k tokens. If users would like to use a foundation model on their downstream task, the most efficient way is to get feature embedding of their

| **Target** data | Method | Aug | 1% label data | 5% label data | 10% label data | 20% label data | 100% label data |
|---|---|---|---|---|---|---|---|
| CIFAR-10 | LP | 1 | **91.44±0.01** | 93.51±0.01 | 94.11±0.01 | 94.22±0.01 | 95.82±0.01 |
| | FT | 1 | 89.93±0.38 | 94.21±0.35 | 95.15±0.06 | 95.80±0.06 | 96.12±0.11 |
| | FT | 10 | 90.82±0.19 | 93.24±0.08 | 94.59±0.08 | 95.48±0.01 | 96.17±0.12 |
| | FT | 100 | 90.84±0.03 | 93.05±0.02 | 94.23±0.08 | 95.37±0.01 | **97.01±0.07** |
| | Ours | 10 | 90.73±0.18 | **94.28±0.03** | **95.43±0.15** | **95.83±0.14** | 96.71±0.10 |
| | Ours | 100 | 91.04±0.06 | 94.00±0.06 | 95.29±0.01 | 95.72±0.05 | 96.86±0.06 |
| SVHN | LP | 1 | 39.40±0.02 | 50.50±0.02 | 54.61±0.02 | 57.66±0.01 | 61.92±0.01 |
| | FT | 1 | 38.82±0.10 | 48.15±0.37 | 51.03±0.14 | 52.07±0.09 | 56.19±0.33 |
| | FT | 10 | 45.00±0.34 | 52.61±0.24 | 54.45±0.12 | 57.05±0.11 | 65.36±0.33 |
| | FT | 100 | 45.70±0.13 | 54.51±0.23 | 57.76±0.01 | 61.13±0.39 | 69.10±0.39 |
| | Ours | 10 | **46.53±0.23** | 55.59±0.03 | 57.32±0.05 | 59.20±0.09 | 66.29±0.20 |
| | Ours | 100 | 46.27±0.09 | **55.70±0.07** | **59.45±0.04** | **61.97±0.13** | **70.22±0.01** |
| GTSRB | LP | 1 | 48.52±0.03 | 66.68±0.02 | 71.69±0.01 | 72.27±0.02 | 75.37±0.01 |
| | FT | 1 | 52.54±0.67 | 72.19±0.33 | 73.75±0.71 | 73.30±0.46 | 75.16±0.01 |
| | FT | 10 | 58.68±0.01 | 75.21±0.23 | 75.87±0.06 | 76.75±0.28 | 76.45±0.29 |
| | FT | 100 | 61.65±0.16 | 75.22±0.77 | 76.49±0.31 | 77.67±0.63 | 78.05±0.30 |
| | Ours | 10 | 59.28±0.04 | 75.35±0.31 | 77.38±0.09 | **80.52±0.12** | **81.28±0.10** |
| | Ours | 100 | **63.61±0.92** | **76.70±0.17** | **77.85±0.23** | 80.13±0.01 | 80.92±0.19 |

Table F.7: Test accuracy for different evaluation methods on different datasets using foundation model MoCo v3 (backbone ViT-B). We do not use data augmentation for LP. We evaluate FT without data augmentation, with 10 augmentations and with 100 augmentations to each training image. For Ours, we use 10, 100 augmentation.

| **Target** data | Method | 1% label data | 5% label data | 10% label data | 20% label data | 100% label data |
|---|---|---|---|---|---|---|
| IMDB | LP | 79.72±0.06 | 83.08±0.20 | 81.48±0.89 | 84.87±0.03 | 86.49±0.16 |
| | FT | 82.54±2.88 | 87.78±0.42 | 87.96±1.27 | 89.49±1.02 | 92.31±0.26 |
| | Ours | **84.48±1.62** | **90.12±0.41** | **90.41±0.49** | **90.79±1.58** | **92.85±0.03** |
| AGNews | LP | 85.52±0.31 | 86.78±0.62 | 86.75±0.66 | 87.62±0.24 | 87.76±0.66 |
| | FT | 88.74±0.34 | 90.76±0.70 | 91.22±0.07 | 92.36±0.14 | 93.57±0.23 |
| | Ours | **89.20±0.72** | **91.22±0.06** | **91.33±1.33** | **92.45±0.01** | **93.94±0.02** |

Table F.8: Test accuracy for different evaluation methods on different datasets using foundation model SimCSE (backbone BERT). We do not use data augmentation for LP. We evaluate FT and Ours with the same data augmentation as SimCSE.

data from the API and train a small model, called adapter (Hu et al., 2022; Sung et al., 2022), on these embedding rather than on raw input data.

We evaluate CLIP (with ViT-L as the representation backbone), MoCo v3 (ViT-B backbone), and SimCSE (Gao et al., 2021b) (BERT backbone). They are trained on (image, text), (image, image), and (text, text) contrastive pairs, respectively, so cover a good spectrum of methods.

For CLIP and MoCo v3, we fix the backbone for all evaluation methods. For

LP, we add a linear FC layer on top of the backbone. For FT and Ours, we insert a two-layer ReLU neural network (1024 dimensions for the hidden layer) as an adapter between the backbone and the linear classification layer. For Ours, we apply SimCLR contrastive loss on the adapted feature (output of adapter) and set $\lambda = 1.0$. We use the same training strategy as Section 7.3.1 for LP and as Table F.5 for FT and Ours. In line with the actual situation, we control the number of augmentation number used in FT and Ours (more data augmentation leads to higher prices in practice).

We conduct experiments on NLP tasks as well. SimCSE proposes a contrastive learning method for sentence embeddings, which uses the dropout feature from BERT as data augmentation. The max sequence embedding length is 512. For SimCSE, all three evaluation methods use a linear classifier. For all evaluation methods, we use AdamW (Loshchilov and Hutter, 2018) with weight decay = 0.01 and train 3 epochs with batch size 16. For LP, we fix the backbone and set the learning rate as 5e-3. For FT and Ours, we train the backbone and linear classification layer simultaneously using unique data augmentation in each training step and set the learning rate as 5e-5. For Ours, we apply SimCSE contrastive loss on the feature (output of the backbone) and set $\lambda = 1.0$.

As in Appendix F.3.2, we report LP, FT, and Ours on $1\%, 5\%, 10\%, 20\%, 100\%$ of the labeled data from the downstream task in Table F.6, Table F.7 and Table F.8 for CLIP, MoCo v3, and SimCSE respectively. For CLIP and MoCo v3 in Table F.6 and Table F.7, we consider different data augmentation numbers, e.g. 10, 100. For SimCSE, we use the standard data augmentation, i.e. generating unique augmentation for each gradient step.

These tables again show our method can consistently improve the downstream prediction performance in all three real-world scenarios, and quite significantly in some cases (e.g., MoCo v3 on GTSRB). This shows that the method is also useful for large foundation models, including the case when the foundation models cannot be fine-tuned and only the extracted embeddings can be adapted.

### The Effect of Contrastive Regularization on Linear Probing

We show that contrastive regularization can also improve over linear probing. Note that the contrastive regularization loss term is only relevant to the backbone; see the definition in Equation (9). While linear probing fixes the backbone. Thus, we cannot do contrastive regularization and linear probing at the same time. To show its effect, we first apply contrastive regularization to update the backbone, and after that use linear probing.

The results in Table F.9 show that contrastive regularization leads to better prediction accuracy. Furthermore, the improvement is more significant on diverse pre-training data, consistent with our analysis.

| | **Pre-training** dataset | | | |
|---|---|---|---|---|
| Method | CINIC-10 | +SVHN | +GTSRB | +ImageNet32 |
| LP | 88.41 | 85.18 | 82.07 | 75.64 |
| Contrastive regularization then Linear probing | 88.38 | 86.91 | 85.95 | 82.43 |

Table F.9: Test accuracy on CIFAR-10 with different evaluation methods on MoCo v2 with ResNet18 backbone. From left to right: incrementally add datasets for pre-training.

### The Effect of Contrastive Regularization on Closing the Gap

We show that contrastive regularization can reduce the target task performance gap between pre-training on the specific dataset (the same or similar as the target task) and that on diverse datasets.

We pre-train with MoCo v3 (backbone ViT-S) on ImageNet-Bird or the whole ImageNet and then perform linear probing (LP) on the target task of ImageNet-Bird with 1k labeled samples. The results in Table F.10 show that pre-training on the diverse data leads to worse performance. Then we pre-train on ImageNet followed by contrastive regularization on ImageNet-Bird and then perform linear probing (LP) on the target task. This leads to improved performance than without contrastive regularization, closing the gap between pre-training on diverse and specific data. Similar results are observed in Table F.11 when ImageNet-Vehicles are used. This confirms the benefits of contrastive regularization for highlighting task-specific features.

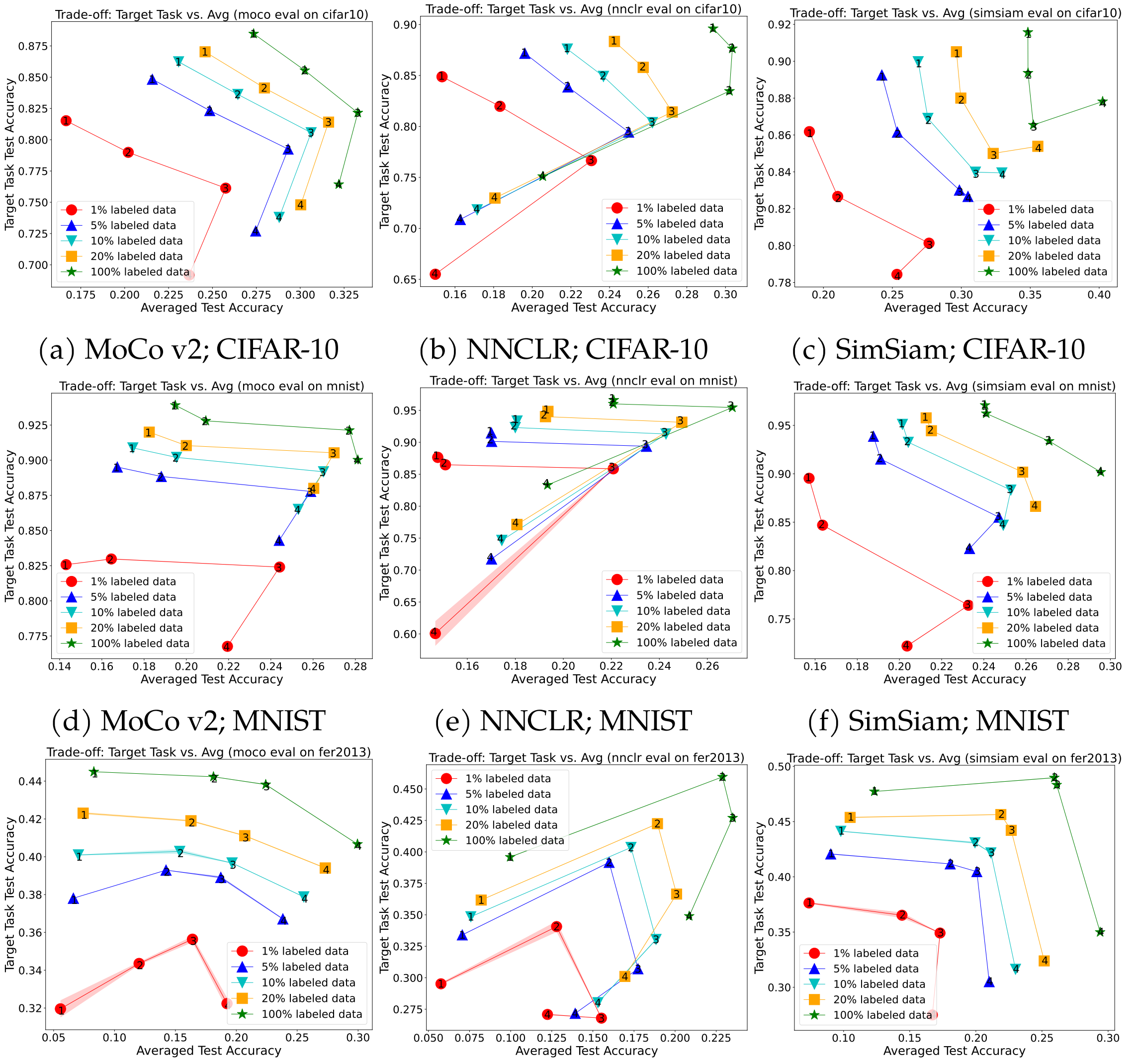
| LP Accuracy | **Pre-training** dataset | | |
|---|---|---|---|
| **Target** dataset | ImageNet-Bird | ImageNet | ImageNet then Contrastive Reguarlization on ImageNet-Bird |
| ImageNet-Bird (1k label) | 81.42 | 69.39 | 73.25 |

Table F.10: LP test accuracy on ImageNet-Bird and ImageNet with MoCo v3 (ViT-S) pre-trained on ImageNet-Bird and ImageNet.

| LP Accuracy | **Pre-training** dataset | | |
|---|---|---|---|
| **Target** dataset | ImageNet-Vehicle | ImageNet | ImageNet then Contrastive Reguarlization on ImageNet-Vehicle |
| ImageNet-Vehicle (1k label) | 70.93 | 67.90 | 71.34 |

Table F.11: LP test accuracy on ImageNet-Vehicle and ImageNet with MoCo v3 (ViT-S) pre-trained on ImageNet-Vehicle and ImageNet.

### F.3.5 Additional Results Verifying Existence of the Trade-off

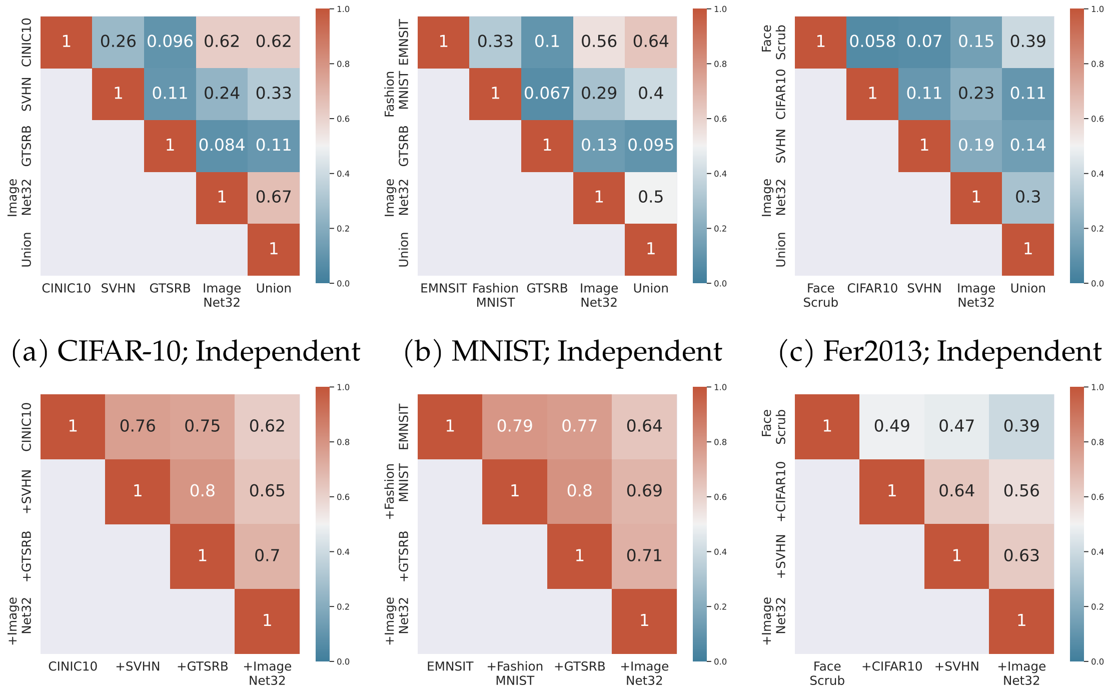


(a) MoCo v2; CIFAR-10 (b) NNCLR; CIFAR-10 (c) SimSiam; CIFAR-10

(d) MoCo v2; MNIST (e) NNCLR; MNIST (f) SimSiam; MNIST

(g) MoCo v2; Fer2013 (h) NNCLR; Fer2013 (i) SimSiam; Fer2013

Figure F.2: Trade-off of universality and label efficiency for MoCo v2, NNCLR, SimSiam on downstream tasks CIFAR-10, MNIST, Fer2013. “1, 2, 3, 4" means incrementally adding datasets for pre-training. The $x$-axis is the average test accuracy of Linear Probing on all 4 datasets. The $y$-axis is test accuracy on the target task. Pre-training data: (a)(b)(c) CINIC-10, SVHN, GTSRB, and ImageNet32. Target task: CIFAR-10. (d)(e)(f) EMNIST-Digits&Letters, Fashon-MNIST, GTSRB, ImageNet32. Target: MNIST. (g)(h)(i) FaceScrub, CIFAR-10, SVHN, ImageNet32. Target: Fer2013.



(a) CIFAR-10; Independent (b) MNIST; Independent (c) Fer2013; Independent

(d) CIFAR-10; Incremental (e) MNIST; Incremental (f) Fer2013; Incremental

Figure F.3: Linear CKA similarity score among downstream task features from MoCo v2 pretrained on three sets of datasets. For “Independent", each representation model in the first four columns/rows is pre-trained on a single dataset. “Union" indicates the model pre-trained on the union among four disjoint datasets. “Incremental" means from left column to right, from top row to bottom, we incrementally add datasets for pre-training.

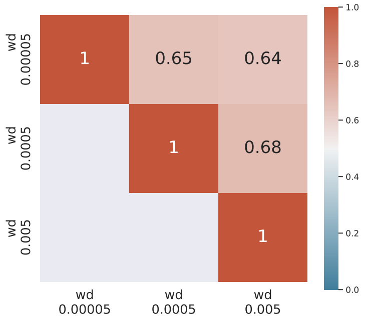
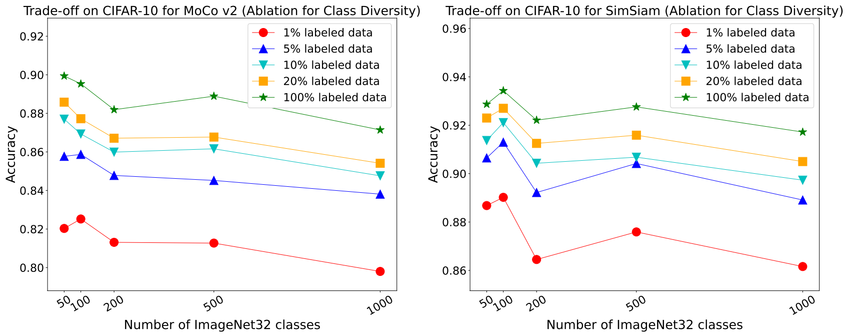


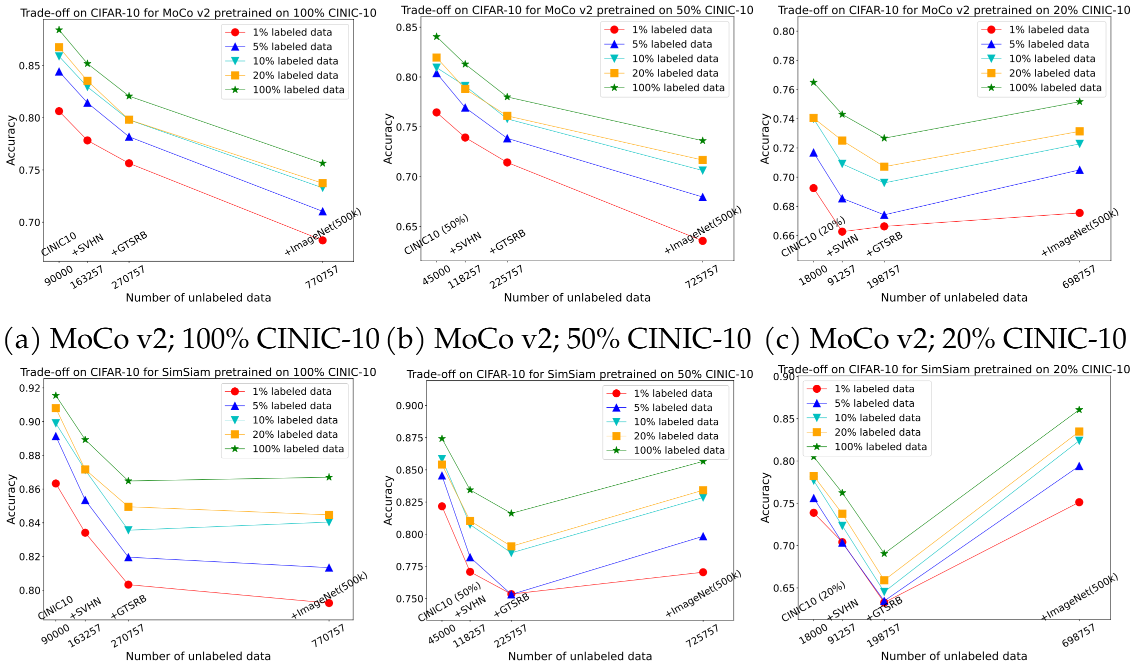
Figure F.4: Linear CKA similarity among CIFAR10 features from MoCo v2 pre-trained on CINIC10. Each representation in the first three columns/rows is pre-trained with a different weight decay value.



(a) MoCo v2

(b) SimSiam

Figure F.5: Pre-train MoCo v2 and SimSiam on CIFAR-10 + ImageNet32(200k) with varying number of classes of ImageNet32 from 50 to 1000 ($x$-axis) under a fixed size of pre-training data. The $y$-axis is LP test accuracy on CIFAR-10.


Trade-off on CIFAR-10 for MoCo v2 pretrained on 100% CINIC-10
Trade-off on CIFAR-10 for MoCo v2 pretrained on 50% CINIC-10
Trade-off on CIFAR-10 for MoCo v2 pretrained on 20% CINIC-10
Trade-off on CIFAR-10 for SimSiam pretrained on 100% CINIC-10
Trade-off on CIFAR-10 for SimSiam pretrained on 50% CINIC-10
Trade-off on CIFAR-10 for SimSiam pretrained on 20% CINIC-10

(a) MoCo v2; 100% CINIC-10 (b) MoCo v2; 50% CINIC-10 (c) MoCo v2; 20% CINIC-10

(d) SimSiam; 100% CINIC-10 (e) SimSiam; 50% CINIC-10 (f) SimSiam; 20% CINIC-10

Figure F.6: Trade-off on CIFAR-10 LP test accuracy (y-axis) for MoCo v2 and SimSiam with varying target relevant (CINIC-10) pre-training data percentage (100%, 50%, 20%).

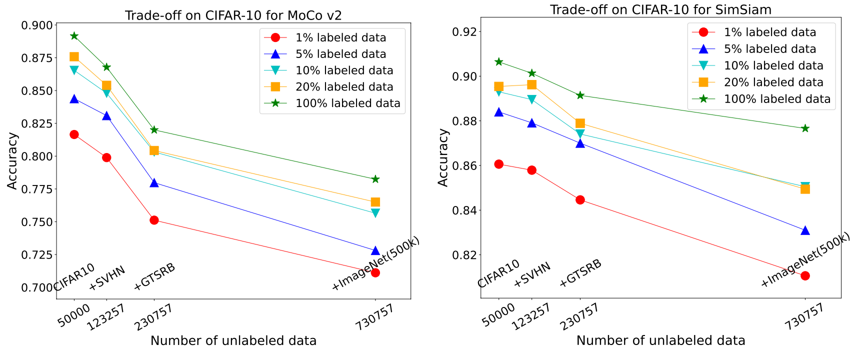
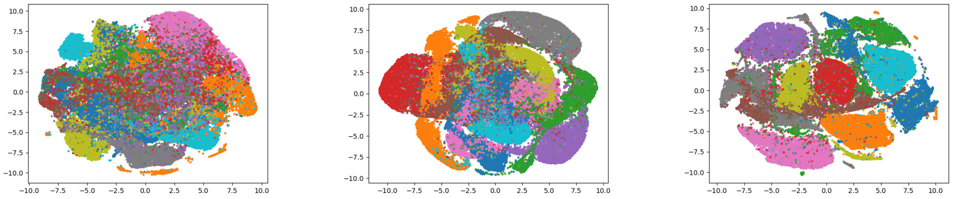

Trade-off on CIFAR-10 for MoCo v2
Trade-off on CIFAR-10 for SimSiam

Figure F.7: Trade-off on CIFAR-10 LP test accuracy (y-axis) for MoCo v2 and SimSiam pre-trianed on datasets including CIFAR-10.

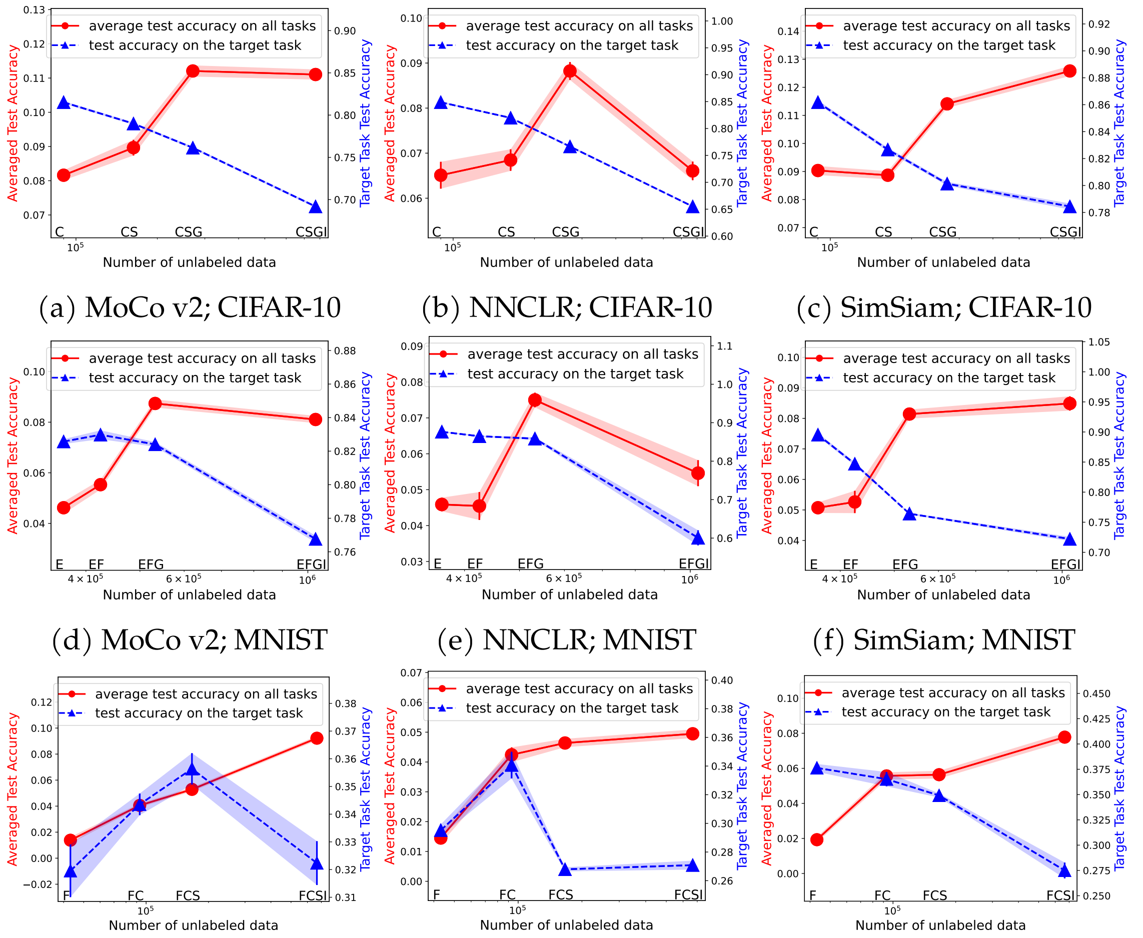


(a) Linear Probing

(b) Finetune

(c) Ours

Figure F.8: The t-SNE visualization (Van der Maaten and Hinton, 2008) for CIFAR-10 training data normalized features from different evaluation methods, where the model is pre-trained on (CSGI) defined in Fig. 7.3. FT and Ours are trained on the 20% CIFAR-10 training dataset. Different colors correspond to different classes.



(a) MoCo v2; CIFAR-10

(b) NNCLR; CIFAR-10

(c) SimSiam; CIFAR-10

(d) MoCo v2; MNIST

(e) NNCLR; MNIST

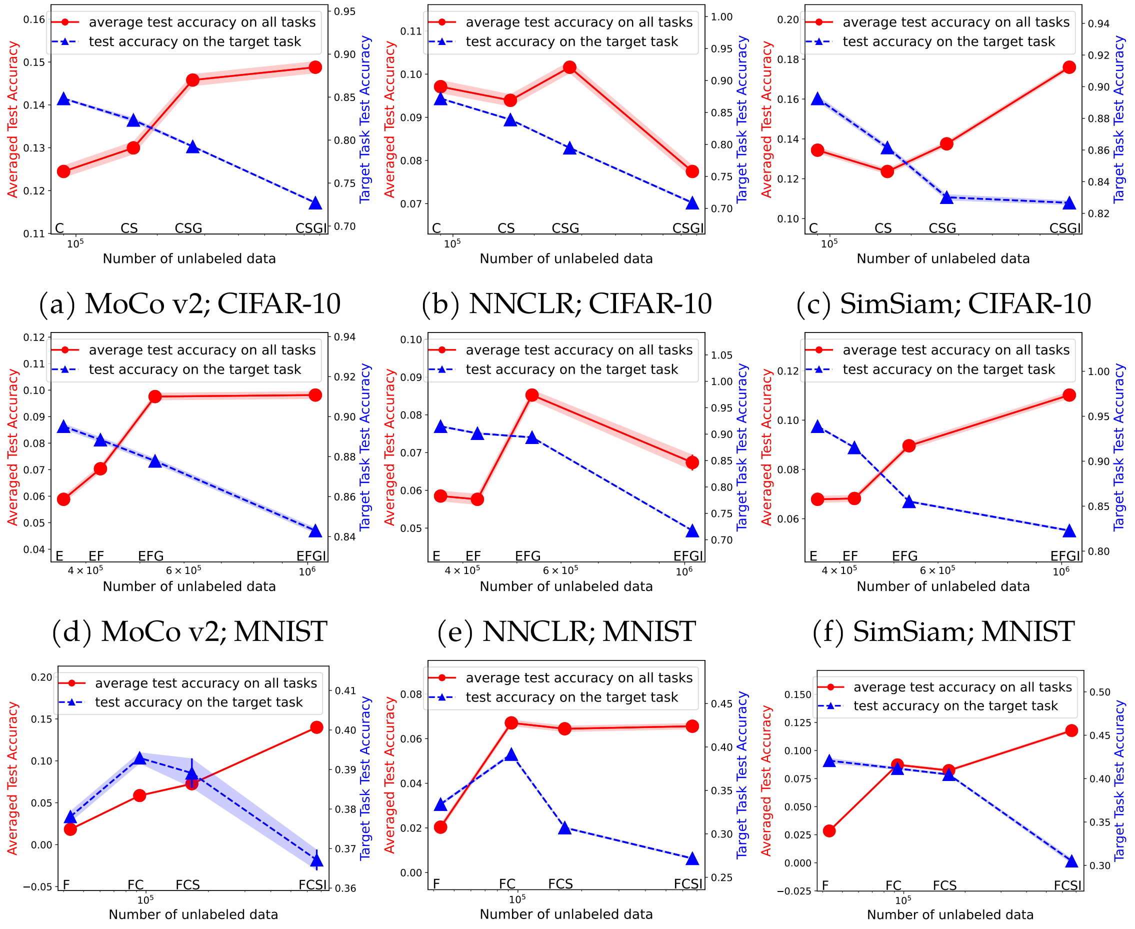
(f) SimSiam; MNIST

(g) MoCo v2; Fer2013

(h) NNCLR; Fer2013

(i) SimSiam; Fer2013

Figure F.9: Trade-off of universality and label efficiency for MoCo v2, NNCLR, SimSiam on downstream tasks CIFAR-10, MNIST, Fer2013. $x$-axis: incrementally add datasets for pre-training. Pre-training data: (a)(b)(c) CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I). For example, "CS" on the $x$-axis means CINIC-10+SVHN. Target task: CIFAR-10. Red line: average test accuracy of Linear Probing on all 4 datasets. Blue line: test accuracy on the target task. (d)(e)(f) EMNIST-Digits&Letters (E), Fashon-MNIST (F), GTSRB (G), ImageNet32 (I). Target: MNIST. (g)(h)(i) FaceScrub (F), CIFAR-10 (C), SVHN (S), ImageNet32 (I). Target: Fer2013. All evaluations are trained with 1% labeled data.



(a) MoCo v2; CIFAR-10

(b) NNCLR; CIFAR-10

(c) SimSiam; CIFAR-10

(d) MoCo v2; MNIST

(e) NNCLR; MNIST

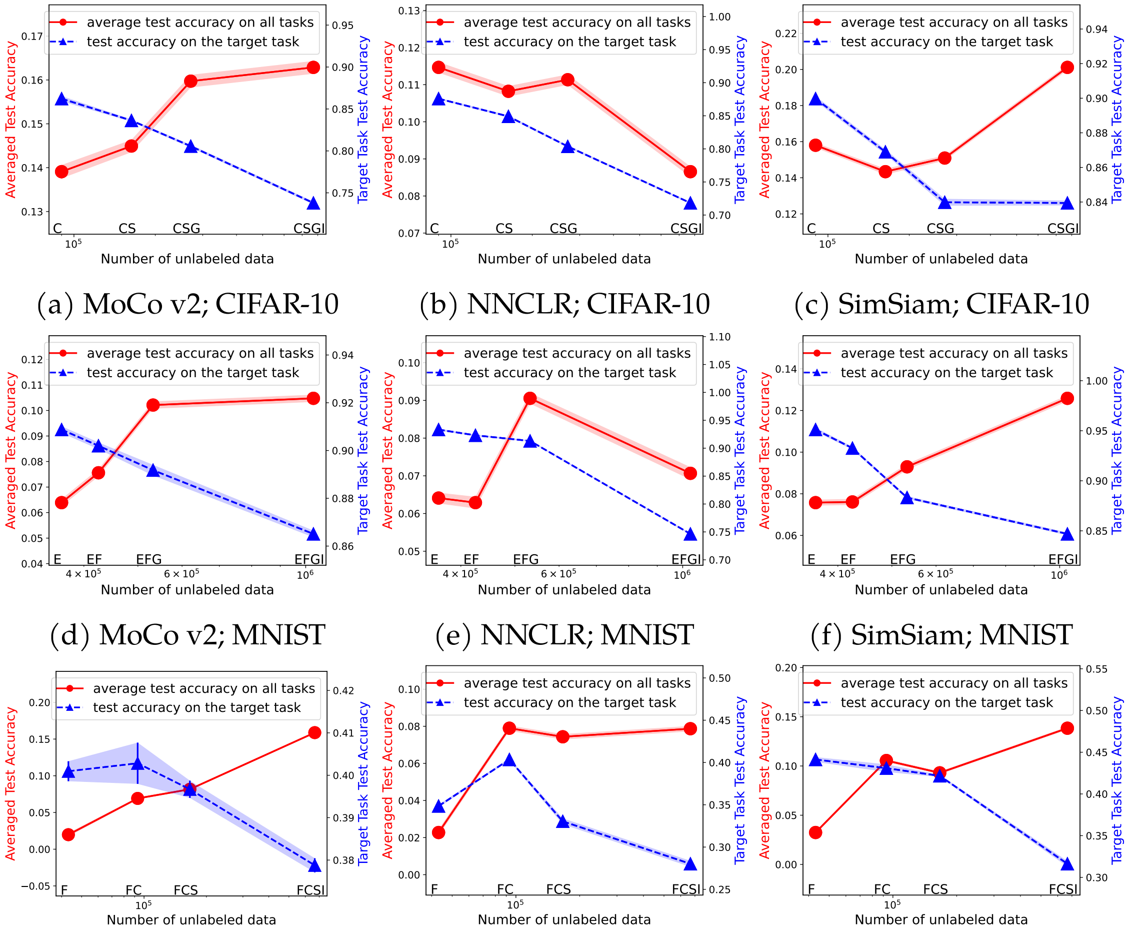
(f) SimSiam; MNIST

(g) MoCo v2; Fer2013

(h) NNCLR; Fer2013

(i) SimSiam; Fer2013

Figure F.10: Trade-off of universality and label efficiency for MoCo v2, NNCLR, SimSiam on downstream tasks CIFAR-10, MNIST, Fer2013. $x$-axis: incrementally add datasets for pre-training. Pre-training data: (a)(b)(c) CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I). For example, "CS" on the $x$-axis means CINIC-10+SVHN. Target task: CIFAR-10. Red line: average test accuracy of Linear Probing on all 4 datasets. Blue line: test accuracy on the target task. (d)(e)(f) EMNIST-Digits&Letters (E), Fashon-MNIST (F), GTSRB (G), ImageNet32 (I). Target: MNIST. (g)(h)(i) FaceScrub (F), CIFAR-10 (C), SVHN (S), ImageNet32 (I). Target: Fer2013. All evaluations are trained with 5% labeled data.

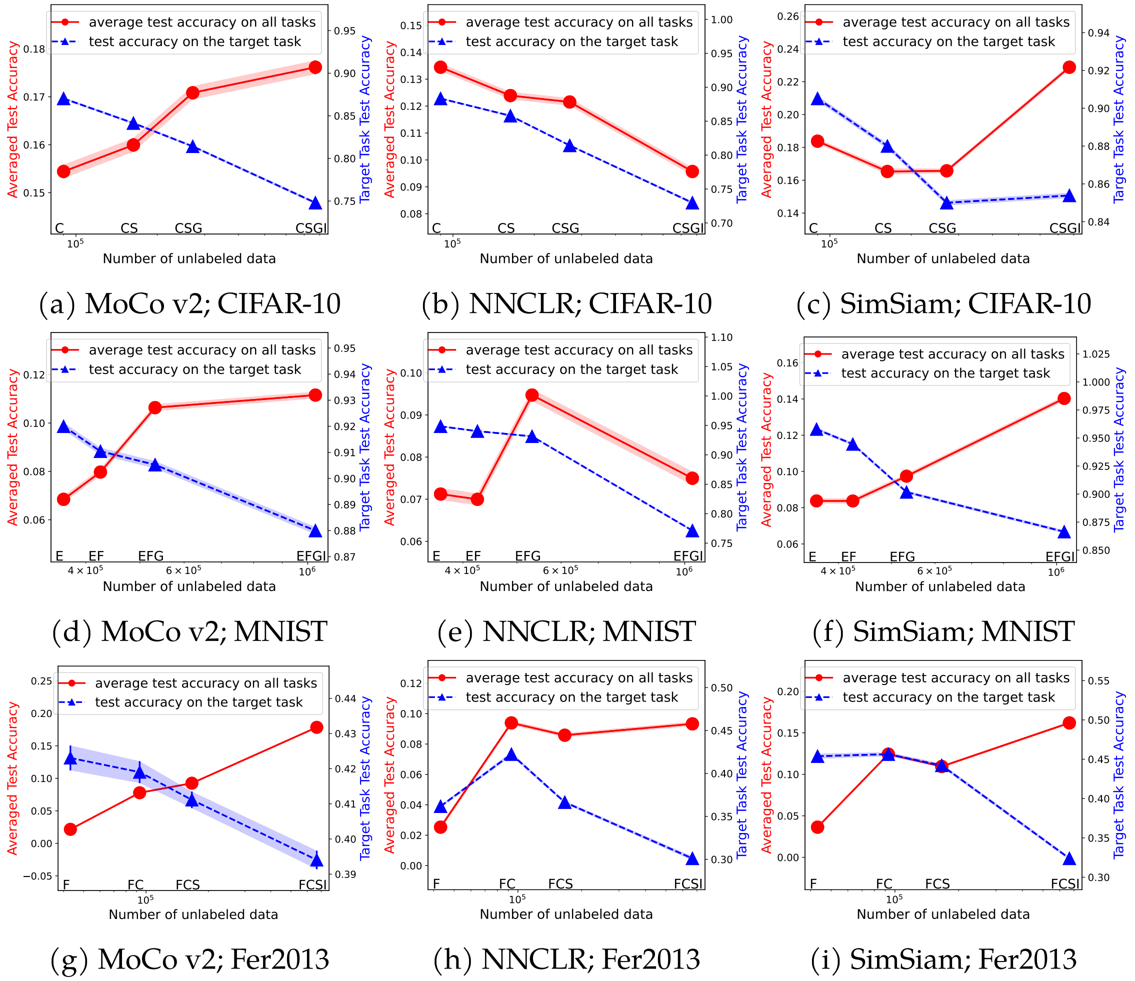


(a) MoCo v2; CIFAR-10 (b) NNCLR; CIFAR-10 (c) SimSiam; CIFAR-10

(d) MoCo v2; MNIST (e) NNCLR; MNIST (f) SimSiam; MNIST

(g) MoCo v2; Fer2013 (h) NNCLR; Fer2013 (i) SimSiam; Fer2013

Figure F.11: Trade-off of universality and label efficiency for MoCo v2, NNCLR, SimSiam on downstream tasks CIFAR-10, MNIST, Fer2013. $x$-axis: incrementally add datasets for pre-training. Pre-training data: (a)(b)(c) CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I). For example, "CS" on the $x$-axis means CINIC-10+SVHN. Target task: CIFAR-10. Red line: average test accuracy of Linear Probing on all 4 datasets. Blue line: test accuracy on the target task. (d)(e)(f) EMNIST-Digits&Letters (E), Fashon-MNIST (F), GTSRB (G), ImageNet32 (I). Target: MNIST. (g)(h)(i) FaceScrub (F), CIFAR-10 (C), SVHN (S), ImageNet32 (I). Target: Fer2013. All evaluations are trained with 10% labeled data.



(a) MoCo v2; CIFAR-10

(b) NNCLR; CIFAR-10

(c) SimSiam; CIFAR-10

(d) MoCo v2; MNIST

(e) NNCLR; MNIST

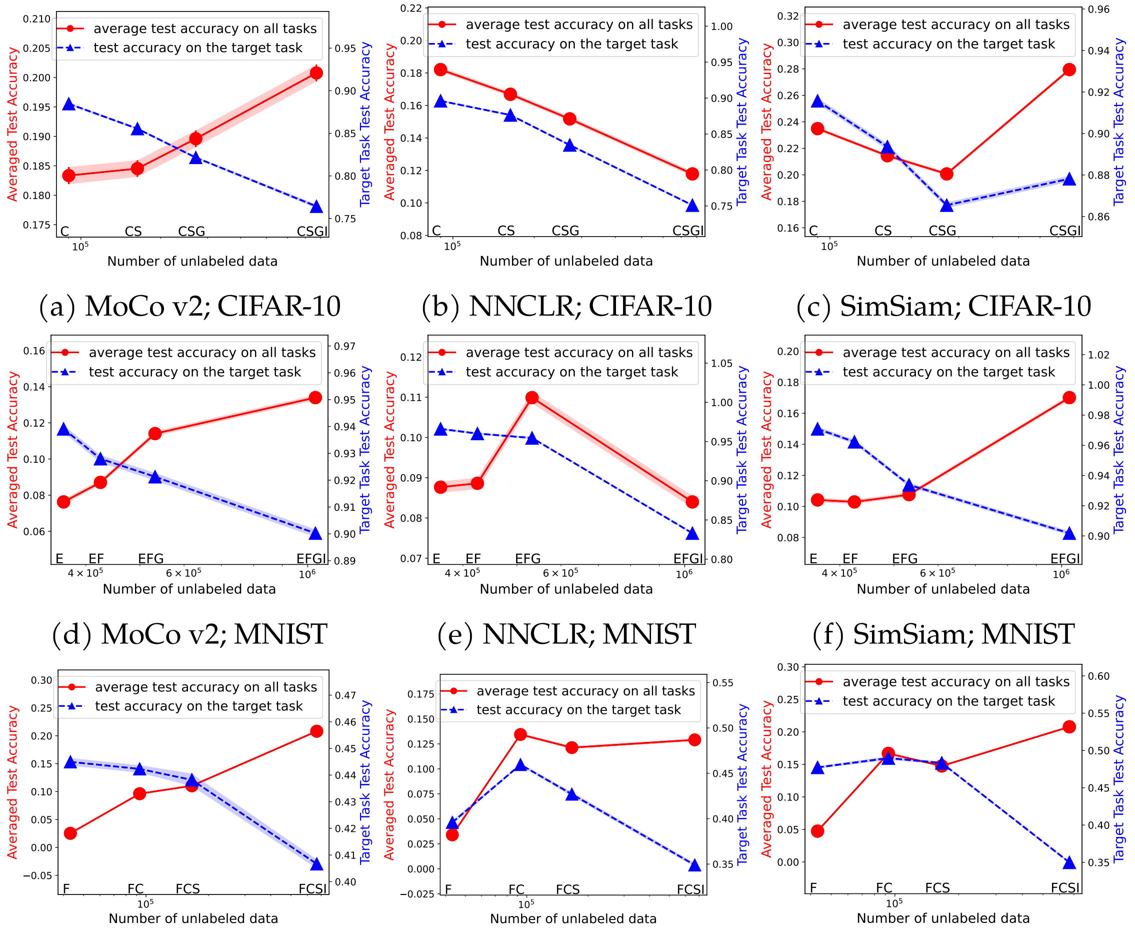
(f) SimSiam; MNIST

(g) MoCo v2; Fer2013

(h) NNCLR; Fer2013

(i) SimSiam; Fer2013

Figure F.12: Trade-off of universality and label efficiency for MoCo v2, NNCLR, SimSiam on downstream tasks CIFAR-10, MNIST, Fer2013. $x$-axis: incrementally add datasets for pre-training. Pre-training data: (a)(b)(c) CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I). For example, "CS" on the $x$-axis means CINIC-10+SVHN. Target task: CIFAR-10. Red line: average test accuracy of Linear Probing on all 4 datasets. Blue line: test accuracy on the target task. (d)(e)(f) EMNIST-Digits&Letters (E), Fashon-MNIST (F), GTSRB (G), ImageNet32 (I). Target: MNIST. (g)(h)(i) FaceScrub (F), CIFAR-10 (C), SVHN (S), ImageNet32 (I). Target: Fer2013. All evaluations are trained with 20% labeled data.



(a) MoCo v2; CIFAR-10 (b) NNCLR; CIFAR-10 (c) SimSiam; CIFAR-10

(d) MoCo v2; MNIST (e) NNCLR; MNIST (f) SimSiam; MNIST

(g) MoCo v2; Fer2013 (h) NNCLR; Fer2013 (i) SimSiam; Fer2013

Figure F.13: Trade-off of universality and label efficiency for MoCo v2, NNCLR, SimSiam on downstream tasks CIFAR-10, MNIST, Fer2013. x-axis: incrementally add datasets for pre-training. Pre-training data: (a)(b)(c) CINIC-10 (C), SVHN (S), GTSRB (G), and ImageNet32 (I). For example, "CS" on the x-axis means CINIC-10+SVHN. Target task: CIFAR-10. Red line: average test accuracy of Linear Probing on all 4 datasets. Blue line: test accuracy on the target task. (d)(e)(f) EMNIST-Digits&Letters (E), Fashon-MNIST (F), GTSRB (G), ImageNet32 (I). Target: MNIST. (g)(h)(i) FaceScrub (F), CIFAR-10 (C), SVHN (S), ImageNet32 (I). Target: Fer2013. All evaluations are trained with 100% labeled data.

## G.1 More Preliminary

In this section, we introduce some key definitions of language modeling modules. We begin with the input embedding function and the output embedding function. They are functions that bridge between the input token space and the real vector space.

**Definition G.1** (Input embedding function and input tokens)**.** *The input embedding function* $\mathcal{E} : \mathcal{V}^n \to \mathbb{R}^{n \times d}$ *maps the input tokens to hidden features using the vocabulary dictionary* $D^{\mathrm{voc}} \in \mathbb{R}^{|\mathcal{V}| \times d}$. *Let* $T \in \mathcal{V}^n$ *be input tokens. Then, we have* $\mathcal{E}(T) \in \mathbb{R}^{n \times d}$ *and* $\mathcal{E}(T)_i = D^{\mathrm{voc}}_{T_i} \in \mathbb{R}^d$ *for any* $i \in [n]$.

**Definition G.2** (Output embedding function)**.** *The output embedding function* $\mathcal{G} : \mathbb{R}^d \to \mathbb{R}^{|\mathcal{V}|}$ *maps hidden features to the probability logits of the vocabulary dictionary.*

We introduce Softmax, which allows self-attention to learn the probability distribution rather than function anymore.

**Definition G.3** (Softmax)**.** *Let* $z \in \mathbb{R}^n$. *We define* $\mathsf{Softmax} : \mathbb{R}^n \to \mathbb{R}^n$ *satisfying*

$$\mathsf{Softmax}(z) := \exp(z)/\langle \exp(z), \mathbf{1}_n \rangle.$$

## G.2 Detailed Comparison with Other Methods

GemFilter reduces both running time and GPU memory usage in both the prompt computation and iterative generation phases, whereas SnapKV (Li et al., 2024g) and H2O (Zhang et al., 2023e) focus only on the iterative generation phase. During the prompt computation phase, standard attention computes and stores the entire KV cache for all layers in GPU memory, which is used during the generation phase. SnapKV and H2O, on the other hand, compute the entire KV cache for all layers but only store a portion of it in GPU memory (e.g., $k = 1024$). They use the selected

KV cache for memory-efficient generation. SnapKV selects important clustered positions of the KV cache from an 'observation' window located at the end of the prompt, while H2O greedily drops tokens based on cumulative attention scores to retain only a small portion of the KV cache. In contrast, GemFilter avoids computing the KV cache for all layers during the prompt computation phase.

Compared to SnapKV and H2O, there are two additional differences. First, SnapKV and H2O maintain separate index sets for each layer and attention head, resulting in $m \cdot h$ index sets in total. This leads to different behaviors across attention heads, making their intermediate mechanisms more difficult to interpret. On the other hand, GemFilter uses a single index set, $J$, allowing for easier interpretability by enabling the printing of the selected sequence for human review before the second run (see a real example in Figure 9.1). Another distinction lies in how positional embeddings are handled. In SnapKV and H2O, the maximum positional embedding distance is $n + t$, as the same positional embedding is used in both the prompt computation and iterative generation phases. However, in GemFilter's second run, the maximum positional embedding distance is reduced to $k + t$ because the input token length is reduced from $n$ to $k$, and the $\mathsf{RoPE}$ function[1] is re-computed. This reduction makes GemFilter more efficient, as the model can better handle shorter input sequences, as demonstrated in Figure 9.4 (a).

## G.3 Proof of Time Complexity

**Theorem G.4** (Complexity analysis. Restatement of Theorem 9.3)**.** *Let $n$ be the input sequence (prompt) length and $d$ the hidden feature dimensions. In our Algorithm 2, GemFilter uses the $r$-th layer as a filter to select $k$ input tokens. Let SnapKV and H2O also use $k$ as their cache size. Assume the LLM has $m$ attention layers, each with $h$ attention heads, and each transformer layer's parameters consume $w$ GPU memory. Assuming that we generate $t$ tokens with the* Gen *function and $n \geqslant \max\{d, k, t\}$, the following table summarizes the complexity for standard attention, SnapKV and H2O, and GemFilter:*

[1] $\mathsf{RoPE}$ is the rotary positional embedding (Su et al., 2024), encoding the positional information of tokens.

| *Complexity* | | *Standard attention* | *SnapKV and H2O* | *GemFilter* |
|---|---|---|---|---|
| *Time* | *Prompt Comp.* | $\Theta(mhn^2d)$ | $\Theta(mhn^2d)$ | $\Theta(rhn^2d)$ |
| | *Iter. generation* | $\Theta(mh(nt+t^2)d)$ | $\Theta(mh(kt+t^2)d)$ | $\Theta(mh(k^2+t^2)d)$ |
| *GPU mem.* | *Prompt Comp.* | $mw+2mhnd$ | $mw+2hnd+2mhkd$ | $rw+2hnd$ |
| | *Iter. generation* | $mw+2mh(n+t)d$ | $mw+2mh(k+t)d$ | $mw+2mh(k+t)d$ |

*Proof of Theorem 9.3.* We prove each method separately.

**Proof of standard attention:**

During prompting computation, it takes $\Theta(mhn^2d)$ time complexity, as there are $m$ transformer layers, each layer has $h$ attention head, and each head takes $\Theta(n^2d)$ to calculate the attention ($\mathsf{Attn}_i$ in Definition 9.2) and $\Theta(nd)$ for other operations ($g_i$ in Definition 9.2).

During iterative generation, it takes $\Theta(mh(nt+t^2)d)$ time complexity.

During prompting computation, $mw$ GPU memory consumption is taken for the model weights and $2mhnd$ GPU memory consumption for the KV cache.

During iterative generation, it takes $mw$ GPU memory consumption for the model weights and $2mh(n+t)d$ GPU memory consumption for the KV cache.

**Proof of SnapKV and H2O:**

During prompting computation, it takes $\Theta(mhn^2d)$ time complexity, which is the same as standard attention.

During iterative generation, it takes $\Theta(mh(kt+t^2)d)$ time complexity, as it reduces the KV cache size from $n$ to $k$.

During prompting computation, $mw$ GPU memory is consumed for the model weights, $2hnd$ for the selection of the key-value matrix for each layer, and $2mhkd$ for the selected KV cache.

During iterative generation, $mw$ GPU memory is consumed for the model weights and $2mh(k+t)d$ GPU memory is consumed for the KV cache.

**Proof of our Algorithm 2 GemFilter:**

During prompting computation, GemFilter takes $\Theta(rhn^2d)$ time complexity, which is faster than other methods.

During iterative generation, it takes $\Theta(mh(k^2 + kt + t^2)d) = \Theta(mh(k^2 + t^2)d)$ time complexity, as it reduces the KV cache size from $n$ to $k$.

During prompting computation, $rw + 2hnd$ GPU memory is consumed for the model weights and the selection of the key value matrix for each layer.

During iterative generation, $mw + 2mh(k + t)d$ GPU memory is consumed for the KV cache and model weights.

Thus, we finish the proof. □

## G.4 More Details about Experiments

### G.4.1 PyTorch Code

We provide the PyTorch code of Algorithm 2 GemFilter below, where our method only needs a few lines of adaptation based on standard attention[2].

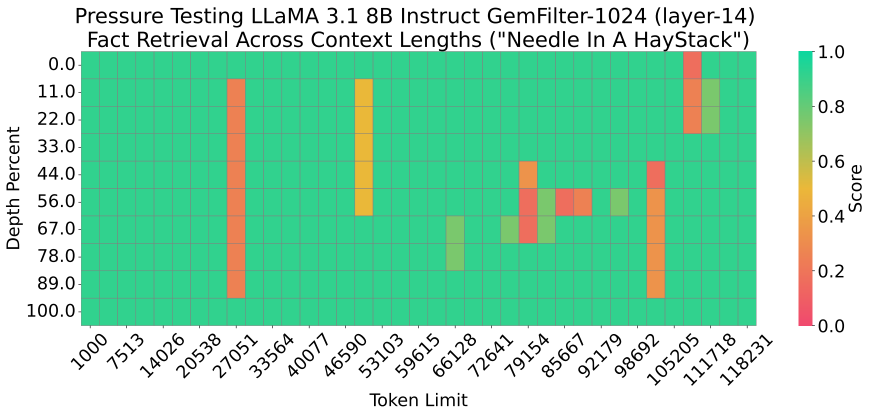
```
# find the selected input for the specific attention layer
def find_context(self, query_states, key_states, k):
    # repeat kv for group query attention
    key_states = repeat_kv(key_states, self.num_key_value_groups)
    # only use the last query token for the top k selection
    top_k_indices = top_index(key_states, query_states[:, :, -1:,
   :], k)
    # sort the index into the correct order
    return torch.sort(top_k_indices, dim=-1).indecies

def top_index(keys, queries, k, kernel=5):
    # calculate the inner product
    in_pro = torch.matmul(queries, keys.transpose(-1, -2))
    # cumulate the score over all attention heads in one attention
   layer
    in_pro = torch.sum(in_pro, dim=1, keepdim=True)
    # use 1D pooling for clustering, similar as SnapKV
    in_pro = F.avg_pool1d(in_pro, kernel=kernel, padding=kernel//2,
   stride=1)
```

[2] https://github.com/huggingface/transformers/blob/v4.43-release/src/transformers/models/mistral/modeling_mistral.py

```
17      return torch.topk(in_pro, k, dim=-1).indices
```

### G.4.2 Implementation Details

All the Needle in a Haystack and LongBench experiments run on A100-40GB GPUs. All the experiments of running time and memory complexity are evaluated on H100-80GB GPUs. We use HuggingFace v4.43 PyTorch implementation. There is no randomness or training in all baseline methods or our method. For the SnapKV/H2O, we use 32 recent size/observation window, which is the optimal choice suggested by Li et al. (2024g); Xu et al. (2024c). However, GemFilter does not have an observation window. We use a maximum pooling kernel size (line 16 of the PyTorch code below) of 5 for SnapKV and our method. For generation, we use standard generation (greedy generation)[3], where *num_beams=1, do_sample = False*.



(a) GemFilter-1024 (layer-14). LLaMA 3.1 average score: 0.870.

Figure G.1: Needle in a Haystack performance comparison of different filter layers with LLaMA 3.1 8B Instruct model. The $x$-axis represents the length of the input tokens, while the $y$-axis shows the position depth percentage of the 'needle' information (e.g., 0% indicates the beginning, and 100% indicates the end). A higher score reflects better performance, meaning more effective retrieval of the 'needle' information.

[3] https://huggingface.co/docs/transformers/v4.43.2/en/main_classes/text_generation

### G.4.3 More Needle in a Haystack

We provide more results of Section 9.4.1 here. In Figure G.2, GemFilter outperforms All KV (standard attention) and SnapKV by a large margin with Phi 3.5 Mini 3.8B Instruct. In Figure G.1, we use layer 14 of LLama 3.1 as the input filter layer, which is an empirical support of the ablation study in Section 9.4.3, as it can also obtain good performance on the Needle in a Haystack benchmark.

### G.4.4 Ablation Study on Row Selection

To understand the intuition behind selecting tokens with the most attention specifically from the last query, we study using different rows rather than the last row in the attention matrix for indices selection, as shown in Figure 9.2.

In Figure G.3, we introduce two methods: (a) selecting the middle rows of the attention matrix and (2) selecting rows with the largest $\ell_2$ norm. As we can see, both methods fail in the Needle in a Haystack task. It shows that selecting the last query token is essential in our method.

### G.4.5 Ablation Study on Runs

Note that the performance improvement of GemFilter may stem from two factors: (1) the selection of important tokens, and (2) the re-computation of these tokens, which might mitigate issues like "lost-in-the-middle". To understand whether both factors made contributions, we provide an ablation study to isolate the contribution of each factor.

In Figure G.4, we introduce GemFilter-One-Run, which does not have the second run as GemFilter. In detail, after getting the indices, which is exactly the same as GemFilter, it directly uses this index set to evict the KV cache for all attention heads and attention layers and continuously conducts the iterative generation phase.

### Difference from GemFilter and SnapKV

It is different from GemFilter as (1) it requires computing full attention matrices for all layers for the KV cache eviction, so it does not save prompt computation phase complexity; (2) it does not have the second run so that the RoPE positional distance is not updated as GemFilter, where its distance between 'needle' and query can be very large.

It is different from SnapKV as all attention heads and attention layers share the same index set, while SnapKV has different index sets for different attention heads and different attention layers.

### Results

As we can see in Figure G.4, the GemFilter-One-Run has a comparable performance with GemFilter, while it is worse when the distance between the query and the 'needle' is large. This is expected as the RoPE positional distance does not update in GemFilter-One-Run. On the other hand, the GemFilter-One-Run takes a larger running time complexity and a larger memory consumption than GemFilter as it requires computing full attention matrices for all layers, while GemFilter only needs to compute the first few layers.

## G.4.6 Index Selection

In Figure G.5, we visualize the top-$k$, $k = 100$, indices over length $n = 46,530$ of each attention layer in GemFilter and SnapKV when using the Mistral Nemo 12B Instruct model and evaluating on Needle in a Haystack. The GemFilter uses layer-19 as its filter layer. Recall that GemFilter selects the top-$k$ indices based on the summation of all attention heads, so each attention layer only has one index set. The SnapKV selects top-$k$ indices for each attention head, so each attention layer only has $h = 32$ index sets, where $h$ is the number of attention heads in each attention layer. Thus, for GemFilter and SnapKV, we plot 1 and 32 index sets for each attention layer, respectively.

In Figure G.5, the red dots mean the selected tokens for the corresponding layer and input tokens. The blue rectangle represents the position of the needle information. The output of GemFilter is *"The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day."* which is totally correct. The output of SnapKV is *"The best thing to do in San Francisco is eat a sandwich."* which is partially correct.

We can see that GemFilter is only focused on the needle information and recent information, while SnapKV focuses on a wide range of tokens, which may distract its attention. We can also conclude that GemFilter and SnapKV have very different selection mechanisms.

### G.4.7 LLaMA 3.1 Chat Template

In Table G.1, we report the performance of different methods on the LongBench QA task using LLaMA 3.1 8B Instruct and its official LLaMA Chat template[4]. In the following, we show the PyTorch code of the way we use the LLaMA Chat template.

```
messages = [
    {"role": "system", "content": ""},
    {"role": "user", "content": prompt}]

input = tokenizer.apply_chat_template(messages,
    add_generation_prompt=True, return_tensors="pt", return_dict=True
    ).to(device)
```

In Table G.1, we can see that, after applying the template, all methods gain a large improvement in performance compared to Table 9.1. Also, we can see that GemFilter has a performance comparable to that of other state-of-the-art methods. It is interesting to understand the difference between the attention mechanisms with and without using a chat template. We leave it as our future work.

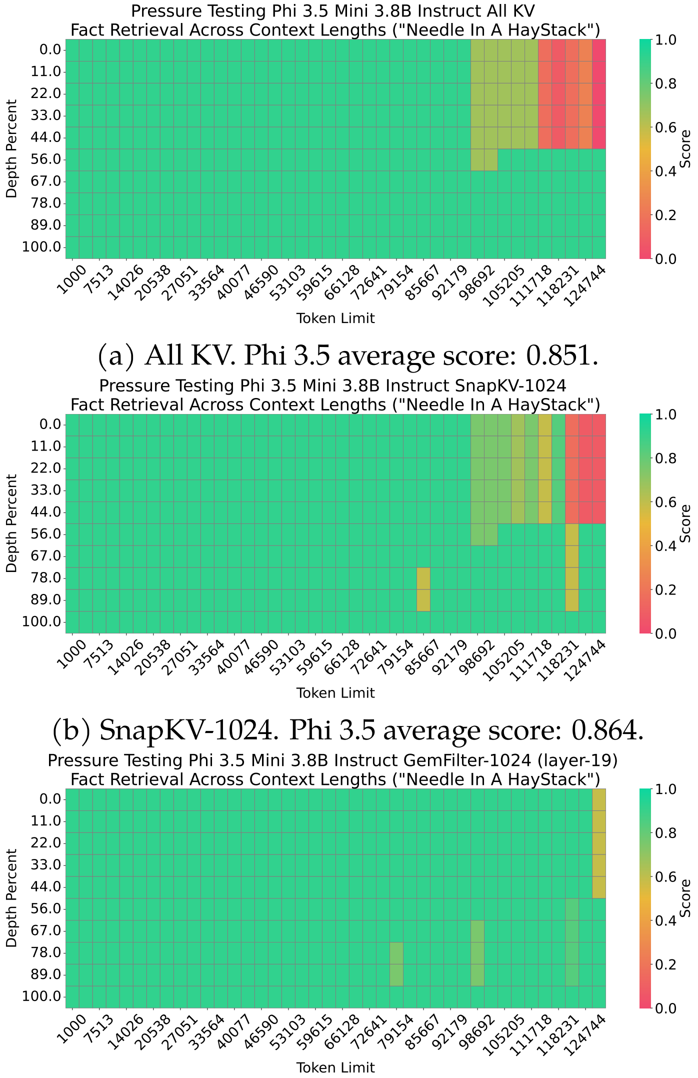
[4] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

Table G.1: Performance comparison on LongBench across various methods when using LLaMA 3.1 8B Instruct and its official LLaMA Chat template. A larger number means better performance. The best score is **boldfaced**.

| | Single-Document QA | | | Multi-Document QA | | | |
|---|---|---|---|---|---|---|---|
| Method | NrtvQA | Qasper | MF-en | HotpotQA | 2WikiMQA | Musique | Average |
| All KV | 25.08 | **44.06** | 55.08 | 47.86 | 49.19 | 27.46 | 41.46 |
| MInference | **29.61** | 43.89 | 54.76 | 51.72 | 49.55 | 28.17 | 42.95 |
| SnapKV-1024 | 29.01 | 41.67 | **56.22** | **56.81** | 49.32 | **31.56** | **44.10** |
| GemFilter-1024 | 22.8 | 40.78 | 48.05 | 54.33 | **50.03** | 30.03 | 41.00 |

### G.4.8 More Results of Index Selection

In this section, we provide more results of index selection on LLaMA 3.1 8B Instruct and Phi 3.5 Mini 3.8B Instruct, where the setting is similar as Figure G.5.

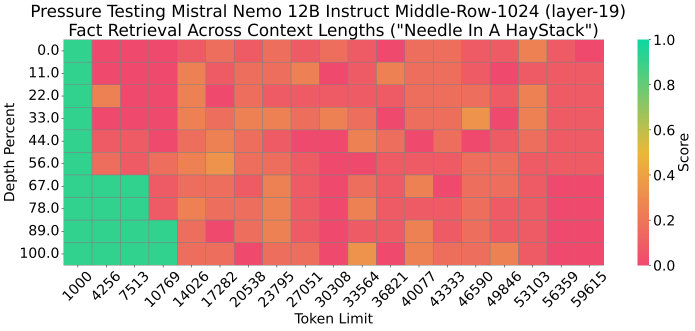


(a) All KV. Phi 3.5 average score: 0.851.

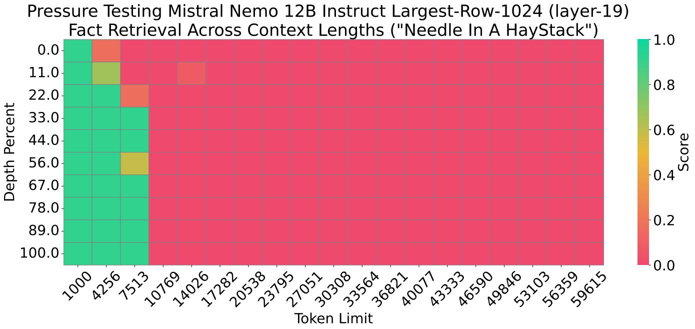
(b) SnapKV-1024. Phi 3.5 average score: 0.864.

(c) GemFilter-1024 (layer-19). Phi 3.5 average score: 0.910.

Figure G.2: Needle in a Haystack performance comparison of different methods using the Phi 3.5 Mini 3.8B Instruct model. The $x$-axis represents the length of the input tokens, while the $y$-axis shows the position depth percentage of the 'needle' information (e.g., 0% indicates the beginning, and 100% indicates the end). A higher score reflects better performance, meaning more effective retrieval of the 'needle' information. GemFilter significantly outperforms both standard attention (full KV cache) and SnapKV.

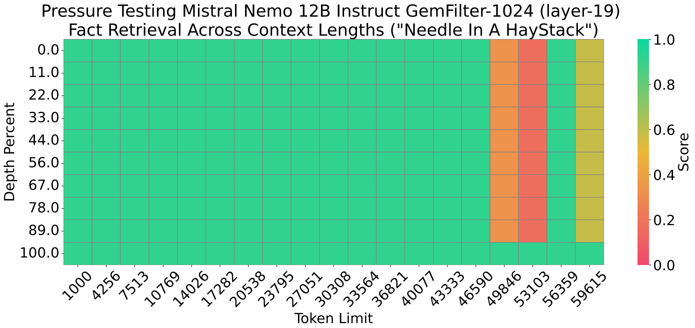


(a) Middle-Row-1024 (layer-19). Mistral Nemo average score: 0.198.

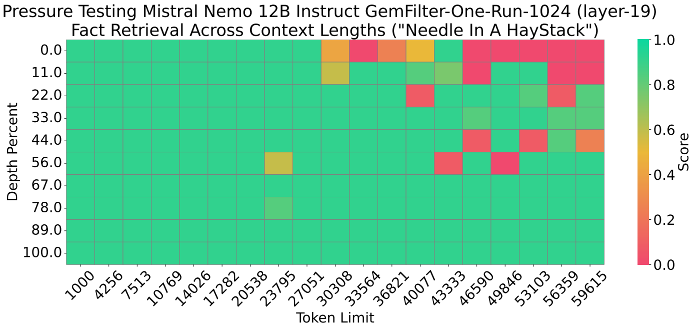


(b) Largest-Row-1024 (layer-19). Mistral Nemo average score: 0.125.

Figure G.3: Needle in a Haystack performance comparison of different methods using the Mistral Nemo 12B Instruct model. The $x$-axis represents the length of the input tokens, while the $y$-axis shows the position depth percentage of the 'needle' information (e.g., 0% indicates the beginning, and 100% indicates the end). A higher score reflects better performance, meaning more effective retrieval of the 'needle' information. (a) is using the middle row to select top $k$ indices and (b) is using the row with largest $\ell_2$ norm to select top $k$ indices.



(a) GemFilter-1024 (layer-19). Mistral Nemo average score: 0.838.



(b) GemFilter-One-Run-1024 (layer-19). Mistral Nemo average score: 0.827.

Figure G.4: Needle in a Haystack performance comparison of different methods using the Mistral Nemo 12B Instruct model. The $x$-axis represents the length of the input tokens, while the $y$-axis shows the position depth percentage of the 'needle' information (e.g., 0% indicates the beginning, and 100% indicates the end). A higher score reflects better performance, meaning more effective retrieval of the 'needle' information. (a) is our method GemFilter and (b) is the degenerate version GemFilter-One-Run for ablation study.

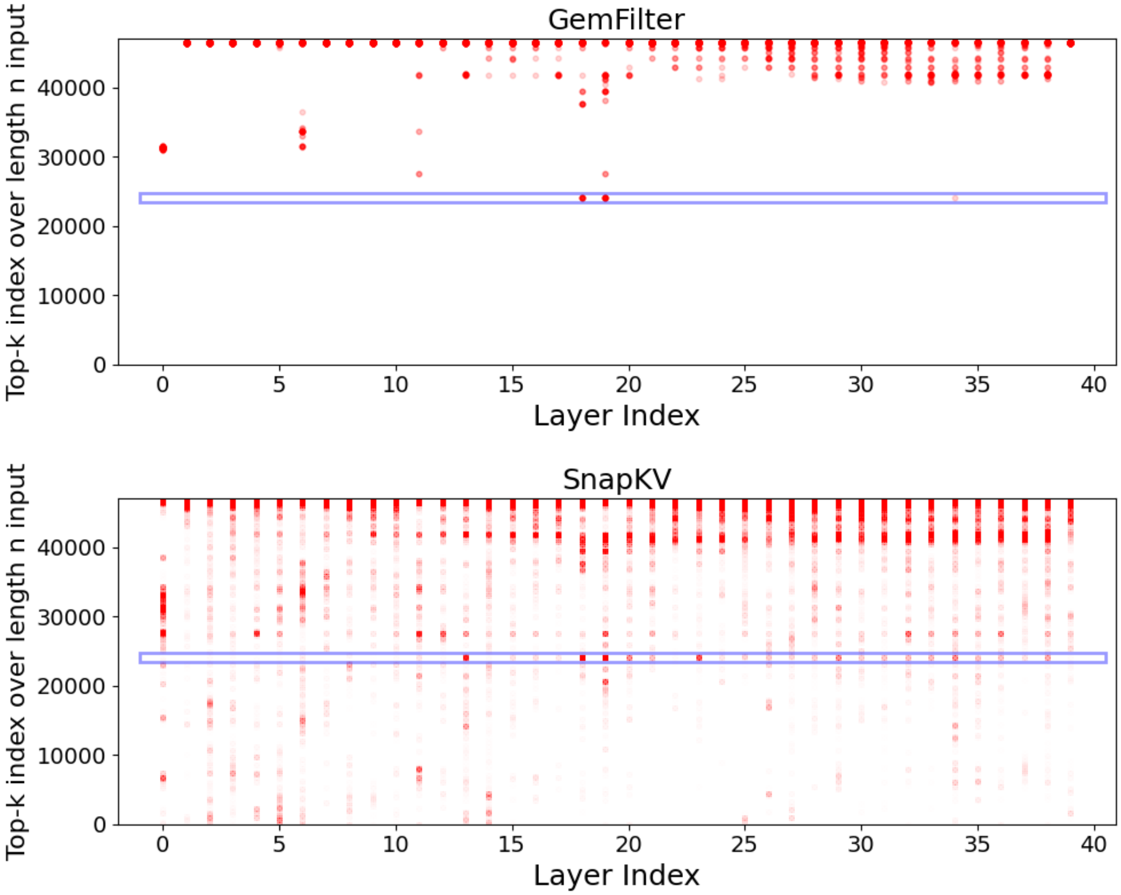


Figure G.5: Needle in a Haystack visualization of the top-k indices of each attention layer in GemFilter and SnapKV when using the **Mistral Nemo 12B Instruct** model. The GemFilter uses layer-19 (the same as other experiments) as its filter layer. Both GemFilter and SnapKV use $k = 100$, i.e., the number of selected tokens. The $x$-axis is the layer index, 40 layers in total. The $y$-axis is the input index, where the input token length is $n = 46,530$. We use 50% as the position depth percentage of the 'needle' information. The red dots mean the selected tokens for the corresponding layer and input tokens. The blue rectangle represents the position of the needle information. The output of GemFilter is *"The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day."* which is totally correct. The output of SnapKV is *"The best thing to do in San Francisco is eat a sandwich."* which is partially correct.

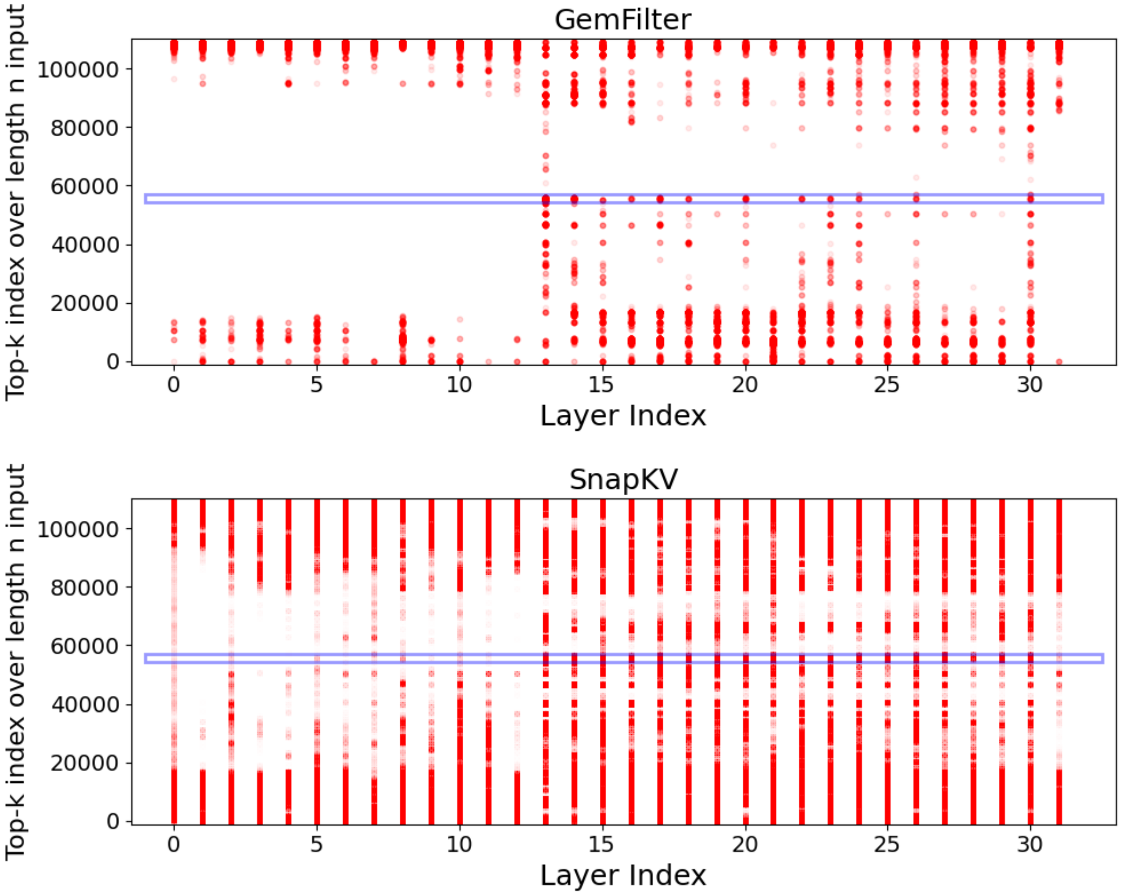


Figure G.6: Needle in a Haystack visualization of the top-k indices of each attention layer in GemFilter and SnapKV when using the **LLaMA 3.1 8B Instruct** model. The GemFilter uses layer-13 (the same as other experiments) as its filter layer. Both GemFilter and SnapKV use $k = 1024$, i.e., the number of selected tokens. The $x$-axis is the layer index, 32 layers in total. The $y$-axis is the input index, where the input token length is $n = 108,172$. We use 50% as the position depth percentage of the 'needle' information. The red dots mean the selected tokens for the corresponding layer and input tokens. The blue rectangle represents the position of the needle information. The output of GemFilter is *"Eat a sandwich and sit in Dolores Park on a sunny day."* which is totally correct. The output of SnapKV is *"Eat a sandwich at a deli in the Mission District."* which is partially correct.

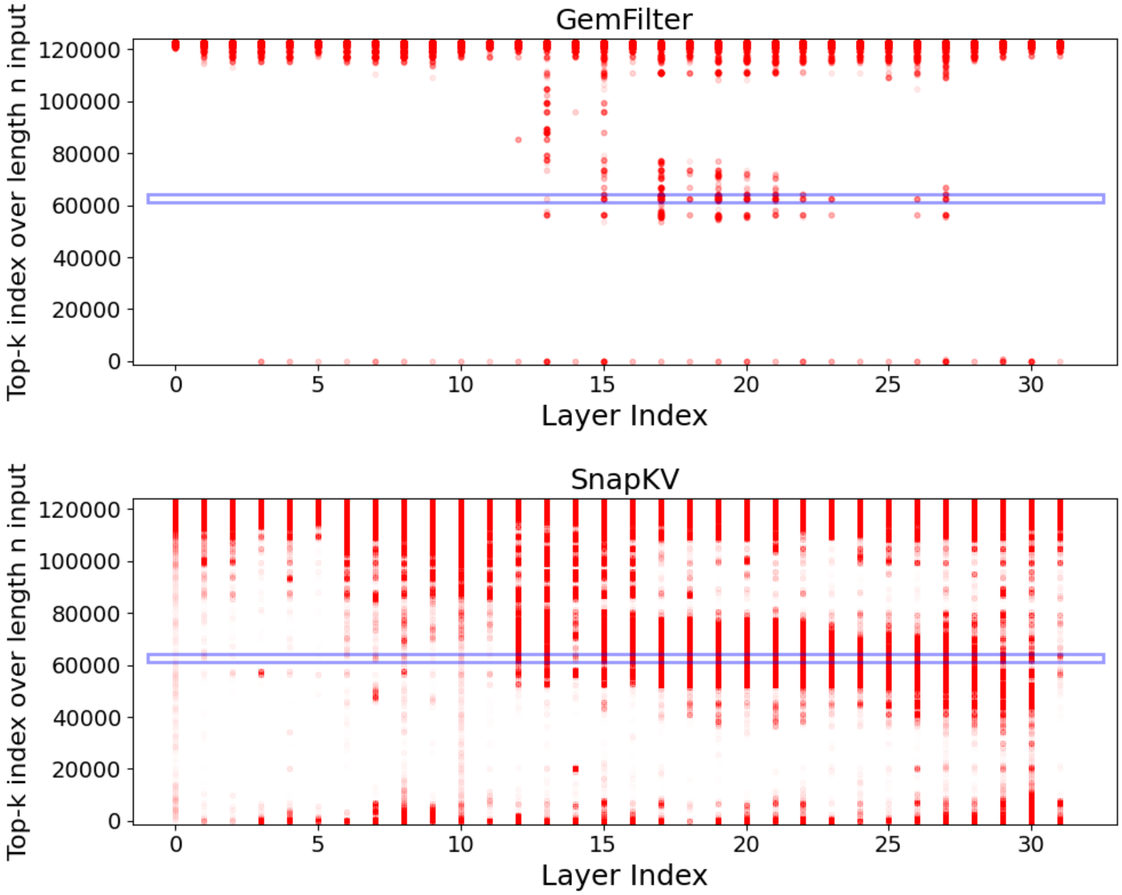


Figure G.7: Needle in a Haystack visualization of the top-$k$ indices of each attention layer in GemFilter and SnapKV when using the **Phi 3.5 Mini 3.8B Instruct** model. The GemFilter uses layer-19 (the same as other experiments) as its filter layer. Both GemFilter and SnapKV use $k = 1024$, i.e., the number of selected tokens. The $x$-axis is the layer index, 32 layers in total. The $y$-axis is the input index, where the input token length is $n = 122,647$. We use 50% as the position depth percentage of the 'needle' information. The red dots mean the selected tokens for the corresponding layer and input tokens. The blue rectangle represents the position of the needle information. The output of GemFilter is *"Sit in Dolores Park on a sunny day and eat a sandwich."* which is totally correct. The output of SnapKV is *"Eat a sandwich."* which is partially correct.

Abbe, Emmanuel, Enric Boix Adsera, and Theodor Misiakiewicz. 2022a. The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks. In *Conference on learning theory*, 4782–4887. PMLR.

Abbe, Emmanuel, Samy Bengio, Elisabetta Cornacchia, Jon Kleinberg, Aryo Lotfi, Maithra Raghu, and Chiyuan Zhang. 2022b. Learning to reason with neural networks: Generalization, unseen data and boolean measures. In *Advances in neural information processing systems*, ed. Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho.

Abdel-Hamid, Ossama, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. 2014. Convolutional neural networks for speech recognition. *IEEE ACM Trans. Audio Speech Lang. Process.* 22(10):1533–1545.

Abdin, Marah, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Achiam, Josh, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Agrawal, Sweta, Chunting Zhou, Mike Lewis, Luke Zettlemoyer, and Marjan Ghazvininejad. 2022. In-context examples selection for machine translation. *arXiv preprint arXiv:2212.02437*.

Ahn, Kwangjun, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. 2024a. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems* 36.

Ahn, Kwangjun, Xiang Cheng, Minhak Song, Chulhee Yun, Ali Jadbabaie, and Suvrit Sra. 2024b. Linear attention is (maybe) all you need (to understand transformer optimization). In *The twelfth international conference on learning representations*.

AI, Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3/.

Akiyama, Shunta, and Taiji Suzuki. 2021. On learnability via gradient method for two-layer relu neural networks in teacher-student setting. In *International conference on machine learning*, 152–162. PMLR.

———. 2023. Excess risk of two-layer reLU neural networks in teacher-student settings and its superiority to kernel methods. In *The eleventh international conference on learning representations*.

Akyürek, Ekin, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2022. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*.

Akyurek, Ekin, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2023. What learning algorithm is in-context learning? investigations with linear models. In *The eleventh international conference on learning representations*.

Alayrac, Jean-Baptiste, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*.

Allen-Zhu, Zeyuan, and Yuanzhi Li. 2019. What can resnet learn efficiently, going beyond kernels? In *Advances in neural information processing systems*.

———. 2020a. Backward feature correction: How deep learning performs deep learning. *arXiv preprint arXiv:2001.04413*.

———. 2020b. Towards understanding ensemble, knowledge distillation and self-distillation in deep learning. *arXiv preprint arXiv:2012.09816*.

———. 2022. Feature purification: How adversarial training performs robust deep learning. In *2021 ieee 62nd annual symposium on foundations of computer science (focs)*, 977–988. IEEE.

———. 2023. Physics of language models: Part 1, context-free grammar. *arXiv preprint arXiv:2305.13673*.

Allen-Zhu, Zeyuan, Yuanzhi Li, and Yingyu Liang. 2019a. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*.

Allen-Zhu, Zeyuan, Yuanzhi Li, and Zhao Song. 2019b. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*.

Alman, Josh, and Zhao Song. 2023. Fast attention requires bounded entries. *Advances in Neural Information Processing Systems* 36.

———. 2024. How to capture higher-order correlations? generalizing matrix softmax attention to kronecker computation. In *The twelfth international conference on learning representations*.

Amodei, Dario, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Jingdong Chen, Mike Chrzanowski, Adam Coates, Greg Diamos, Erich Elsen, Jesse H. Engel, Linxi Fan, Christopher Fougner, Awni Y. Hannun, Billy Jun, Tony Han, Patrick LeGresley, Xiangang Li, Libby Lin, Sharan Narang, Andrew Y. Ng, Sherjil Ozair, Ryan Prenger, Sheng Qian, Jonathan Raiman, Sanjeev Satheesh, David Seetapun, Shubho Sengupta, Chong Wang, Yi Wang, Zhiqian Wang, Bo Xiao, Yan Xie, Dani Yogatama, Jun Zhan, and Zhenyao Zhu. 2016. Deep speech 2 : End-to-end speech recognition in english and mandarin. In *Proceedings of the 33nd international conference on machine learning, ICML 2016, new york city, ny, usa, june 19-24, 2016*, ed. Maria-Florina Balcan and Kilian Q. Weinberger, vol. 48 of *JMLR Workshop and Conference Proceedings*, 173–182. JMLR.org.

An, Shengnan, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. 2023. How do in-context examples affect compositional generalization? *arXiv preprint arXiv:2305.04835*.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku.

Anthropic. 2024. Claude 3.5 sonnet.

Arjovsky, Martin, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. 2019. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*.

Arora, Sanjeev, Nadav Cohen, Noah Golowich, and Wei Hu. 2018. A convergence analysis of gradient descent for deep linear neural networks. In *International conference on learning representations*.

Arora, Sanjeev, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. 2019a. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International conference on machine learning*, 322–332. PMLR.

Arora, Sanjeev, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. 2019b. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*.

Arora, Sanjeev, and Anirudh Goyal. 2023. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*.

Artur Back, de Luca, and Kimon Fountoulakis. 2024. Simulation of graph algorithms with looped transformers. *arXiv preprint arXiv:2402.01107*.

Ba, Jimmy, Murat A Erdogdu, Taiji Suzuki, Zhichao Wang, Denny Wu, and Greg Yang. 2022. High-dimensional asymptotics of feature learning: How one gradient step improves the representation. *arXiv preprint arXiv:2205.01445*.

Bahng, Hyojin, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. 2020. Learning de-biased representations with biased representations. In *International conference on machine learning*, 528–539. PMLR.

Bai, Yu, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. 2024a. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Advances in neural information processing systems* 36.

Bai, Yu, and Jason D Lee. 2019. Beyond linearization: On quadratic and higher-order approximation of wide neural networks. In *International conference on learning representations*.

Bai, Yushi, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, et al. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Bai, Yushi, Jiajie Zhang, Xin Lv, Linzhi Zheng, Siqi Zhu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024b. Longwriter: Unleashing 10,000+ word generation from long context llms. *arXiv preprint arXiv:2408.07055*.

Baker, Nicholas, Hongjing Lu, Gennady Erlikhman, and Philip J. Kellman. 2018. Deep convolutional networks do not classify based on global object shape. *PLOS Computational Biology* 14(12):1–43.

Balestriero, Randall, and Yann LeCun. 2022. Contrastive and non-contrastive self-supervised learning recover global and local spectral embedding methods. *arXiv preprint arXiv:2205.11508*.

Barak, Boaz, Benjamin Edelman, Surbhi Goel, Sham Kakade, Eran Malach, and Cyril Zhang. 2022. Hidden progress in deep learning: Sgd learns parities near the computational limit. *Advances in Neural Information Processing Systems* 35: 21750–21764.

Barbu, Andrei, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. 2019. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in neural information processing systems* 32:9453–9463.

Bartlett, Peter L, Philip M Long, Gábor Lugosi, and Alexander Tsigler. 2020. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences* 117(48):30063–30070.

Beery, Sara, Grant Van Horn, and Pietro Perona. 2018. Recognition in terra incognita. In *Proceedings of the european conference on computer vision (eccv)*, 456–473.

Belinkov, Yonatan. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics* 48(1):207–219.

Belkin, Mikhail, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences* 116(32):15849–15854.

Bhojanapalli, Srinadh, Chulhee Yun, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. 2020. Low-rank bottleneck in multi-head attention models. In *International conference on machine learning*. PMLR.

Bietti, Alberto, Joan Bruna, Clayton Sanford, and Min Jae Song. 2022. Learning single-index models with shallow neural networks. *arXiv preprint arXiv:2210.15651*.

Bietti, Alberto, Vivien Cabannes, Diane Bouchacourt, Herve Jegou, and Leon Bottou. 2023. Birth of a transformer: A memory viewpoint. In *Thirty-seventh conference on neural information processing systems*.

Blanchard, Gilles, Aniket Anand Deshmukh, Ürun Dogan, Gyemin Lee, and Clayton Scott. 2021. Domain generalization by marginal transfer learning. *The Journal of Machine Learning Research* 22(1):46–100.

Blei, David M, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research* 3:993–1022.

Blum, Avrim, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. 1994. Weakly learning dnf and characterizing statistical query learning using fourier analysis. In *Proceedings of the twenty-sixth annual acm symposium on theory of computing*, 253–262.

Blum, Avrim, and Ronald L Rivest. 1989. Training a 3-node neural network is np-complete. In *Advances in neural information processing systems*, 494–501.

Bommasani, Rishi, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Bourgain, Jean. 2014. An improved estimate in the restricted isometry problem. In *Geometric aspects of functional analysis*, 65–70. Springer.

Boyd, Stephen P, and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

Bronstein, Ido, Alon Brutzkus, and Amir Globerson. 2022. On the inductive bias of neural networks for learning read-once dnfs. In *Uncertainty in artificial intelligence*, 255–265. PMLR.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*.

Bubeck, Sébastien, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Bui, Manh-Ha, Toan Tran, Anh Tran, and Dinh Phung. 2021. Exploiting domain-specific features to enhance domain generalization. *Advances in Neural Information Processing Systems* 34:21189–21201.

Cammarata, Nick, Gabriel Goh, Shan Carter, Ludwig Schubert, Michael Petrov, and Chris Olah. 2020. Curve detectors. *Distill* 5(6):e00024–003.

Candes, Emmanuel, and Benjamin Recht. 2012. Exact matrix completion via convex optimization. *Communications of the ACM* 55(6):111–119.

Candès, Emmanuel J, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust principal component analysis? *Journal of the ACM (JACM)* 58(3):1–37.

Candes, Emmanuel J, and Justin Romberg. 2006. Quantitative robust uncertainty principles and optimally sparse decompositions. *Foundations of Computational Mathematics* 6(2):227–254.

Cao, Yuan, Zixiang Chen, Misha Belkin, and Quanquan Gu. 2022. Benign overfitting in two-layer convolutional neural networks. In *Advances in neural information processing systems*, ed. Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho.

Cao, Yuan, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. 2020. Towards understanding the spectral bias of deep learning. 1912.01198.

Cao, Yuan, and Quanquan Gu. 2019. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in Neural Information Processing Systems* 32:10836–10846.

Caron, Mathilde, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *European conference on computer vision*.

Cha, Junbum, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. 2021. Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems* 34:22405–22418.

Cha, Junbum, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. 2022. Domain generalization by mutual-information regularization with pre-trained models. In *Computer vision–eccv 2022: 17th european conference, tel aviv, israel, october 23–27, 2022, proceedings, part xxiii*, 440–457. Springer.

Chandrasekaran, Venkat, Sujay Sanghavi, Pablo A Parrilo, and Alan S Willsky. 2011. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization* 21(2):572–596.

Changpinyo, Soravit, Piyush Sharma, Nan Ding, and Radu Soricut. 2021. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 3558–3568.

Chatterji, Niladri S, Philip M Long, and Peter L Bartlett. 2021. When does gradient descent with logistic loss find interpolating two-layer networks? *Journal of Machine Learning Research* 22(159):1–48.

Chen, Beidi, Tri Dao, Eric Winsor, Zhao Song, Atri Rudra, and Christopher Ré. 2021. Scatterbrain: Unifying sparse and low-rank attention. *Advances in Neural Information Processing Systems (NeurIPS)* 34:17413–17426.

Chen, Bo, Xiaoyu Li, Yingyu Liang, Jiangxuan Long, Zhenmei Shi, and Zhao Song. 2024a. Circuit complexity bounds for rope-based transformer architecture. *arXiv preprint arXiv:2411.07602*.

Chen, Bo, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. 2024b. Bypassing the exponential dependency: Looped transformers efficiently learn in-context by multi-step gradient descent. *arXiv preprint arXiv:2410.11268*.

Chen, Bo, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, and Zhao Song. 2024c. Hsr-enhanced sparse attention acceleration. *arXiv preprint arXiv:2410.10165*.

Chen, Liang-Chieh, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. 2018. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Trans. Pattern Anal. Mach. Intell.* 40(4):834–848.

Chen, Minshuo, Yu Bai, Jason D Lee, Tuo Zhao, Huan Wang, Caiming Xiong, and Richard Socher. 2020a. Towards understanding hierarchical learning: Benefits of neural representations. *arXiv preprint arXiv:2006.13436*.

Chen, Minshuo, Haoming Jiang, Wenjing Liao, and Tuo Zhao. 2019a. Efficient approximation of deep relu networks for functions on low dimensional manifolds. *Advances in neural information processing systems* 32:8174–8184.

———. 2019b. Nonparametric regression on low-dimensional manifolds using deep relu networks: Function approximation and statistical recovery. *arXiv preprint arXiv:1908.01842*.

Chen, Sitan, Jerry Li, and Zhao Song. 2020b. Learning mixtures of linear regressions in subexponential time via fourier moments. In *Proceedings of the 52nd annual acm sigact symposium on theory of computing*, 587–600.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020c. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020d. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th international conference on machine learning, ICML 2020, 13-18 july 2020, virtual event*, vol. 119 of *Proceedings of Machine Learning Research*, 1597–1607. PMLR.

Chen, Xiang, Zhao Song, Baocheng Sun, Junze Yin, and Danyang Zhuo. 2023. Query complexity of active learning for function family with nearly orthogonal basis. *arXiv preprint arXiv:2306.03356*.

Chen, Xinlei, and Kaiming He. 2021. Exploring simple siamese representation learning. In *IEEE conference on computer vision and pattern recognition, CVPR 2021, virtual, june 19-25, 2021*, 15750–15758. Computer Vision Foundation / IEEE.

Chen, Xue, Daniel M Kane, Eric Price, and Zhao Song. 2016. Fourier-sparse interpolation without a frequency gap. In *2016 ieee 57th annual symposium on foundations of computer science (focs)*, 741–750. IEEE.

Chen, Yifang, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. 2024d. The computational limits of state-space models and mamba via the lens of circuit complexity. *arXiv preprint arXiv:2412.06148*.

Chen, Zhengdao, Eric Vanden-Eijnden, and Joan Bruna. 2022. On feature learning in neural networks with global convergence guarantees. In *International conference on learning representations*.

Cheng, Xiang, Yuxin Chen, and Suvrit Sra. 2023. Transformers implement functional gradient descent to learn non-linear functions in context. *arXiv preprint arXiv:2312.06528*.

Chi, Po-Han, Pei-Hung Chung, Tsung-Han Wu, Chun-Cheng Hsieh, Yen-Hao Chen, Shang-Wen Li, and Hung-yi Lee. 2021. Audio albert: A lite bert for self-supervised learning of audio representation. In *2021 ieee spoken language technology workshop (slt)*, 344–350. IEEE.

Chizat, Lenaic, and Francis Bach. 2018a. A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*.

———. 2018b. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems* 31.

———. 2020. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on learning theory*, 1305–1338. PMLR.

Chizat, Lenaic, Edouard Oyallon, and Francis Bach. 2019. On lazy training in differentiable programming. In *Advances in neural information processing systems*.

Chorowski, Jan, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. In *Advances in neural information processing systems 28: Annual conference on neural information processing systems 2015, december 7-12, 2015, montreal, quebec, canada*, ed. Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, 577–585.

Chowdhery, Aakanksha, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Chughtai, Bilal, Lawrence Chan, and Neel Nanda. 2023. A toy model of universality: Reverse engineering how networks learn group operations. In *International conference on machine learning*, 6243–6267. PMLR.

Chung, Hyung Won, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Cohen, Gregory, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. 2017. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (ijcnn)*, 2921–2926. IEEE.

Cole, Elijah, Xuan Yang, Kimberly Wilber, Oisin Mac Aodha, and Serge Belongie. 2022. When does contrastive visual representation learning work? In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 14755–14764.

Conneau, Alexis, and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)*. European Language Resources Association (ELRA).

Dai, Damai, Yutao Sun, Li Dong, Yaru Hao, Zhifang Sui, and Furu Wei. 2022. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. *arXiv preprint arXiv:2212.10559*.

Damian, Alexandru, Jason Lee, and Mahdi Soltanolkotabi. 2022. Neural networks can learn representations with gradient descent. In *Conference on learning theory*. PMLR.

Daniely, Amit, and Eran Malach. 2020. Learning parities with neural networks. *Advances in Neural Information Processing Systems* 33:20356–20365.

Dao, Tri. 2023. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*.

Dao, Tri, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems* 35:16344–16359.

Darlow, Luke N, Elliot J Crowley, Antreas Antoniou, and Amos J Storkey. 2018. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*.

Dass, Jyotikrishna, Shang Wu, Huihong Shi, Chaojian Li, Zhifan Ye, Zhongfeng Wang, and Yingyan Lin. 2023. Vitality: Unifying low-rank and sparse approximation for vision transformer acceleration with a linear taylor attention. In *2023 ieee international symposium on high-performance computer architecture (hpca)*. IEEE.

Dehghani, Mostafa, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. 2018. Universal transformers. *arXiv preprint arXiv:1807.03819*.

Demirel, Mehmet F, Shengchao Liu, Siddhant Garg, Zhenmei Shi, and Yingyu Liang. 2022. Attentive walk-aggregating graph neural networks. *Transactions on Machine Learning Research*.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition*, 248–255. Ieee.

Deng, Li. 2012. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine* 29(6):141–142.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies*. Association for Computational Linguistics.

Diakonikolas, Ilias, Surbhi Goel, Sushrut Karmalkar, Adam R Klivans, and Mahdi Soltanolkotabi. 2020. Approximation schemes for relu regression. In *Conference on learning theory*.

Doersch, Carl, Abhinav Gupta, and Alexei A. Efros. 2015. Unsupervised visual representation learning by context prediction. In *2015 IEEE international conference on computer vision, ICCV 2015, santiago, chile, december 7-13, 2015*, 1422–1430. IEEE Computer Society.

Dolan, William B., and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the third international workshop on paraphrasing (IWP2005)*.

Dong, Qingxiu, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*.

Dong, Weisheng, Guangming Shi, Xin Li, Yi Ma, and Feng Huang. 2014. Compressive sensing via nonlocal low-rank regularization. *IEEE transactions on image processing* 23(8):3618–3632.

Doshi, Darshil, Tianyu He, Aritra Das, and Andrey Gromov. 2024. Grokking modular polynomials. *arXiv preprint arXiv:2406.03495*.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg

Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*.

Dou, Qi, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. 2019. Domain generalization via model-agnostic learning of semantic features. *Advances in Neural Information Processing Systems* 32.

Dou, Xialiang, and Tengyuan Liang. 2020. Training neural networks as learning data-adaptive kernels: Provable representation and approximation benefits. *Journal of the American Statistical Association* 1–14.

Du, Simon, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. 2019. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*.

Du, Simon S, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. 2018. Gradient descent provably optimizes over-parameterized neural networks. In *International conference on learning representations*.

Dubois, Yann, Yangjun Ruan, and Chris J Maddison. 2022. Optimal representations for covariate shifts. In *International conference on learning representations*.

Dwibedi, Debidatta, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. 2021. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the ieee/cvf international conference on computer vision*, 9588–9597.

Elhage, Nelson, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. 2022. Toy models of superposition. *arXiv preprint arXiv:2209.10652*.

Elhage, Nelson, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread* 1.

Ericsson, Linus, Henry Gouk, and Timothy M. Hospedales. 2021. How well do self-supervised models transfer? In *IEEE conference on computer vision and pattern recognition, CVPR 2021, virtual, june 19-25, 2021*, 5414–5423. Computer Vision Foundation / IEEE.

Fan, Ky. 1951. Maximum properties and inequalities for the eigenvalues of completely continuous operators. *Proceedings of the National Academy of Sciences* 37(11): 760–766.

Fan, Xinyan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. 2021. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In *Proceedings of the 44th international acm sigir conference on research and development in information retrieval*.

Fang, Chen, Ye Xu, and Daniel N Rockmore. 2013. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the ieee international conference on computer vision*, 1657–1664.

Fang, Cong, Hanze Dong, and Tong Zhang. 2019. Over parameterized two-level neural networks can learn near optimal feature representations. 1910.11508.

———. 2021. Mathematical models of overparameterized neural networks. *Proceedings of the IEEE* 109(5):683–703.

Feng, Yu, and Yuhai Tu. 2021. Phases of learning dynamics in artificial neural networks: in the absence or presence of mislabeled data. *Machine Learning: Science and Technology*.

Foret, Pierre, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. 2021. Sharpness-aware minimization for efficiently improving generalization. In *International conference on learning representations*.

Frankle, Jonathan, and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International conference on learning representations*.

Frankle, Jonathan, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. 2019. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*.

Frei, Spencer, Yuan Cao, and Quanquan Gu. 2020. Agnostic learning of a single neuron with gradient descent. In *Advances in neural information processing systems*.

———. 2021. Provable generalization of sgd-trained neural networks of any width in the presence of adversarial label noise. *arXiv preprint arXiv:2101.01152*.

Frei, Spencer, Niladri S Chatterji, and Peter Bartlett. 2022a. Benign overfitting without linearity: Neural network classifiers trained by gradient descent for noisy linear data. In *Conference on learning theory*, 2668–2703. PMLR.

Frei, Spencer, Niladri S Chatterji, and Peter L Bartlett. 2022b. Random feature amplification: Feature learning and generalization in neural networks. *arXiv preprint arXiv:2202.07626*.

Frei, Spencer, and Quanquan Gu. 2021. Proxy convexity: A unified framework for the analysis of neural networks trained by gradient descent. *Advances in Neural Information Processing Systems* 34.

Frei, Spencer, Gal Vardi, Peter L Bartlett, and Nathan Srebro. 2023a. Benign overfitting in linear classifiers and leaky relu networks from kkt conditions for margin maximization. *arXiv preprint arXiv:2303.01462*.

———. 2023b. The double-edged sword of implicit bias: Generalization vs. robustness in relu networks. *arXiv preprint arXiv:2303.01456*.

Frei, Spencer, Gal Vardi, Peter L Bartlett, Nathan Srebro, and Wei Hu. 2022c. Implicit bias in leaky relu networks trained on high-dimensional data. *arXiv preprint arXiv:2210.07082*.

Fu, Yao, Hao Peng, Tushar Khot, and Mirella Lapata. 2023. Improving language model negotiation with self-play and in-context learning from ai feedback. *arXiv preprint arXiv:2305.10142*.

Ganin, Yaroslav, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research* 17(1):2096–2030.

Gao, Peng, Jiaming Han, Renrui Zhang, Ziyi Lin, Shijie Geng, Aojun Zhou, Wei Zhang, Pan Lu, Conghui He, Xiangyu Yue, et al. 2023. Llama-adapter v2: Parameter-efficient visual instruction model. *arXiv preprint arXiv:2304.15010*.

Gao, Peng, Chiori Hori, Shijie Geng, Takaaki Hori, and Jonathan Le Roux. 2020. Multi-pass transformer for machine translation. *arXiv preprint arXiv:2009.11382*.

Gao, Tianyu, Adam Fisch, and Danqi Chen. 2021a. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing*.

Gao, Tianyu, Xingcheng Yao, and Danqi Chen. 2021b. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, 6894–6910.

Gao, Yeqi, Zhao Song, and Baocheng Sun. 2022. An $O(k \log n)$ time fourier set query algorithm. *arXiv preprint arXiv:2208.09634*.

Garg, Shivam, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*.

Garg, Siddhant, and Yingyu Liang. 2020. Functional regularization for representation learning: A unified theoretical perspective. *arXiv preprint arXiv:2008.02447*.

Garrigos, Guillaume, and Robert M Gower. 2023. Handbook of convergence theorems for (stochastic) gradient methods. *arXiv preprint arXiv:2301.11235*.

Gatmiry, Khashayar, Nikunj Saunshi, Sashank J Reddi, Stefanie Jegelka, and Sanjiv Kumar. 2024. Can looped transformers learn to implement multi-step gradient descent for in-context learning? In *Forty-first international conference on machine learning*.

Ge, Suyu, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. 2023. Model tells you what to discard: Adaptive kv cache compression for llms. *arXiv preprint arXiv:2310.01801*.

Geiger, Mario, Leonardo Petrini, and Matthieu Wyart. 2021. Landscape and training regimes in deep learning. *Physics Reports* 924:1–18.

Geiger, Mario, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. 2020. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment* 2020(11):113301.

Geirhos, Robert, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. 2019. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International conference on learning representations*.

Geva, Mor, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, 5484–5495.

Ghorbani, Behrooz, Song Mei, Theodor Misiakiewicz, and Andrea Montanari. 2019. Limitations of lazy training of two-layers neural networks. *arXiv preprint arXiv:1906.08899*.

———. 2020. When do neural networks outperform kernel methods? In *Advances in neural information processing systems*.

Giannou, Angeliki, Liu Yang, Tianhao Wang, Dimitris Papailiopoulos, and Jason D Lee. 2024. How well can transformers emulate in-context newton's method? *arXiv preprint arXiv:2403.03183*.

Gidel, Gauthier, Francis Bach, and Simon Lacoste-Julien. 2019. Implicit regularization of discrete gradient dynamics in linear neural networks. *Advances in Neural Information Processing Systems* 32.

Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer vision and pattern recognition*.

Goldt, Sebastian, Madhu Advani, Andrew M Saxe, Florent Krzakala, and Lenka Zdeborová. 2019. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. *Advances in neural information processing systems* 32.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, 2672–2680.

Goodfellow, Ian J, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, et al. 2013. Challenges in representation learning: A report on three machine learning contests. In *International conference on neural information processing*, 117–124. Springer.

Grill, Jean-Bastien, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. 2020. Bootstrap your own latent - A new approach to self-supervised learning. In *Advances in neural information processing systems 33: Annual conference on neural information processing systems 2020, neurips 2020, december 6-12, 2020, virtual*.

Gromov, Andrey. 2023. Grokking modular arithmetic. *arXiv preprint arXiv:2301.02679*.

Gu, Jiuxiang, Yingyu Liang, Zhizhou Sha, Zhenmei Shi, and Zhao Song. 2024. Differential privacy mechanisms in neural tangent kernel regression. *arXiv preprint arXiv:2407.13621*.

Gu, Shuhang, Lei Zhang, Wangmeng Zuo, and Xiangchu Feng. 2014. Weighted nuclear norm minimization with application to image denoising. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 2862–2869.

Gulrajani, Ishaan, and David Lopez-Paz. 2021. In search of lost domain generalization. In *International conference on learning representations*.

Gunasekar, Suriya, Jason Lee, Daniel Soudry, and Nathan Srebro. 2018a. Characterizing implicit bias in terms of optimization geometry. In *International conference on machine learning*, 1832–1841. PMLR.

Gunasekar, Suriya, Jason D Lee, Daniel Soudry, and Nati Srebro. 2018b. Implicit bias of gradient descent on linear convolutional networks. *Advances in Neural Information Processing Systems* 31.

Guo, Tianyu, Wei Hu, Song Mei, Huan Wang, Caiming Xiong, Silvio Savarese, and Yu Bai. 2024. How do transformers learn in-context beyond simple functions? a case study on learning with representations. In *The twelfth international conference on learning representations*.

Hanin, Boris, and Mihai Nica. 2019. Finite depth and width corrections to the neural tangent kernel. In *International conference on learning representations*.

Hanna, Michael, Ollie Liu, and Alexandre Variengien. 2023. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *Advances in Neural Information Processing Systems* 36.

Hannun, Awni Y., Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep speech: Scaling up end-to-end speech recognition. *CoRR* abs/1412.5567. 1412.5567.

HaoChen, Jeff Z, Colin Wei, Adrien Gaidon, and Tengyu Ma. 2021. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in Neural Information Processing Systems* 34.

Haviv, Ishay, and Oded Regev. 2017. The restricted isometry property of subsampled fourier matrices. In *Geometric aspects of functional analysis*, 163–179. Springer.

He, Kaiming, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. 2022. Masked autoencoders are scalable vision learners. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 16000–16009.

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020a. Momentum contrast for unsupervised visual representation learning. In *Computer vision and pattern recognition*.

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020b. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF conference on computer vision and pattern recognition, CVPR 2020, seattle, wa, usa, june 13-19, 2020*, 9726–9735. Computer Vision Foundation / IEEE.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 770–778.

Hendrycks, Dan, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Hotelling, Harold. 1933. Analysis of a complex of statistical variables into principal components. *Journal of educational psychology* 24(6):417.

Howard, Jeremy, and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th annual meeting of the association for computational linguistics, ACL 2018, melbourne, australia, july 15-20, 2018, volume*

*1: Long papers*, ed. Iryna Gurevych and Yusuke Miyao, 328–339. Association for Computational Linguistics.

Hu, Edward J, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International conference on learning representations*.

Hu, Jerry Yao-Chieh, Pei-Hsuan Chang, Haozheng Luo, Hong-Yu Chen, Weijian Li, Wei-Po Wang, and Han Liu. 2024a. Outlier-efficient hopfield layers for large transformer-based models. In *Forty-first international conference on machine learning* (*icml*).

Hu, Jerry Yao-Chieh, Bo-Yu Chen, Dennis Wu, Feng Ruan, and Han Liu. 2024b. Nonparametric modern hopfield models. *arXiv preprint arXiv:2404.03900*.

Hu, Jerry Yao-Chieh, Thomas Lin, Zhao Song, and Han Liu. 2024c. On computational limits of modern hopfield models: A fine-grained complexity analysis. In *Forty-first international conference on machine learning* (*icml*).

Hu, Jerry Yao-Chieh, Dennis Wu, and Han Liu. 2024d. Provably optimal memory capacity for modern hopfield models: Tight analysis for transformer-compatible dense associative memories. In *Advances in neural information processing systems* (*neurips*), vol. 37.

Hu, Jerry Yao-Chieh, Donglin Yang, Dennis Wu, Chenwei Xu, Bo-Yu Chen, and Han Liu. 2023. On sparse modern hopfield model. In *Thirty-seventh conference on neural information processing systems* (*neurips*).

Huang, Jiaoyang, and Horng-Tzer Yau. 2020. Dynamics of deep neural networks and neural tangent hierarchy. In *International conference on machine learning*, 4542–4551. PMLR.

Huang, Yu, Yuan Cheng, and Yingbin Liang. 2023. In-context convergence of transformers. *arXiv preprint arXiv:2310.05249*.

Huang, Zeyi, Haohan Wang, Eric P Xing, and Dong Huang. 2020. Self-challenging improves cross-domain generalization. In *European conference on computer vision*, 124–140. Springer.

Indyk, Piotr, and Michael Kapralov. 2014. Sample-optimal Fourier sampling in any constant dimension. In *Ieee 55th annual symposium onfoundations of computer science (focs)*, 514–523. IEEE.

Indyk, Piotr, Michael Kapralov, and Eric Price. 2014. (nearly) sample-optimal sparse fourier transform. In *Proceedings of the twenty-fifth annual acm-siam symposium on discrete algorithms (soda)*, 480–499. SIAM.

Iyer, Srinivasan, Xi Victoria Lin, Ramakanth Pasunuru, Todor Mihaylov, Daniel Simig, Ping Yu, Kurt Shuster, Tianlu Wang, Qing Liu, Punit Singh Koura, et al. 2022. Opt-iml: Scaling language model instruction meta learning through the lens of generalization. *arXiv preprint arXiv:2212.12017*.

Jacot, Arthur. 2023. Implicit bias of large depth networks: a notion of rank for nonlinear functions. In *The eleventh international conference on learning representations*.

Jacot, Arthur, Franck Gabriel, and Clément Hongler. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*.

Jelassi, Samy, Michael Sander, and Yuanzhi Li. 2022. Vision transformers provably learn spatial structure. *Advances in Neural Information Processing Systems*.

Ji, Ziwei, and Matus Telgarsky. 2019a. The implicit bias of gradient descent on nonseparable data. In *Conference on learning theory*, 1772–1798. PMLR.

———. 2019b. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow relu networks. In *International conference on learning representations*.

———. 2020. Directional convergence and alignment in deep learning. *Advances in Neural Information Processing Systems* 33:17176–17186.

Jiang, Albert Q., Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023a. Mistral 7b. 2310.06825.

Jiang, Huiqiang, Yucheng Li, Chengruidong Zhang, Qianhui Wu, Xufang Luo, Surin Ahn, Zhenhua Han, Amir H Abdi, Dongsheng Li, Chin-Yew Lin, et al. 2024a. Minference 1.0: Accelerating pre-filling for long-context llms via dynamic sparse attention. *arXiv preprint arXiv:2407.02490*.

Jiang, Huiqiang, Qianhui Wu, , Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024b. LongLLMLingua: Accelerating and enhancing LLMs in long context scenarios via prompt compression. In *Proceedings of the 62nd annual meeting of the association for computational linguistics (volume 1: Long papers)*, ed. Lun-Wei Ku, Andre Martins, and Vivek Srikumar, 1658–1677. Bangkok, Thailand: Association for Computational Linguistics.

Jiang, Huiqiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023b. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, ed. Houda Bouamor, Juan Pino, and Kalika Bali, 13358–13376. Singapore: Association for Computational Linguistics.

Jiang, Ziyan, Xueguang Ma, and Wenhu Chen. 2024c. Longrag: Enhancing retrieval-augmented generation with long-context llms. *arXiv preprint arXiv:2406.15319*.

Jin, Yaonan, Daogao Liu, and Zhao Song. 2023. Super-resolution and robust sparse continuous fourier transform in any constant dimension: Nearly linear time and sample complexity. In *Acm-siam symposium on discrete algorithms (soda)*.

Jing, Longlong, and Yingli Tian. 2020. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Kalibhat, Neha Mukund, Kanika Narang, Hamed Firooz, Maziar Sanjabi, and Soheil Feizi. 2022. Towards better understanding of self-supervised representations. In *Icml 2022: Workshop on spurious correlations, invariance and stability*.

Kamath, Pritish, Omar Montasser, and Nathan Srebro. 2020. Approximate is good enough: Probabilistic variants of dimensional and margin complexity. In *Conference on learning theory*.

Kamradt, Greg. 2024. Needle in a haystack - pressure testing llms.

Kapralov, Michael. 2016. Sparse Fourier transform in any constant dimension with nearly-optimal sample complexity in sublinear time. In *Symposium on theory of computing conference (stoc)*.

———. 2017. Sample efficient estimation and recovery in sparse FFT via isolation on average. In *58th annual ieee symposium on foundations of computer science (focs)*.

Karimi, Hamed, Julie Nutini, and Mark Schmidt. 2016. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint european conference on machine learning and knowledge discovery in databases*, 795–811. Springer.

Ke, Yekun, Xiaoyu Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. 2024a. Advancing the understanding of fixed point iterations in deep neural networks: A detailed analytical study. *arXiv preprint arXiv:2410.11279*.

Ke, Yekun, Yingyu Liang, Zhenmei Shi, Zhao Song, and Chiwun Yang. 2024b. Curse of attention: A kernel-based perspective for why transformers fail to generalize on time series forecasting and beyond. *arXiv preprint arXiv:2412.06061*.

Kearns, Michael. 1998. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*.

Khattab, Omar, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. 2022. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv preprint arXiv:2212.14024*.

Kim, Daehee, Youngjun Yoo, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. 2021. Selfreg: Self-supervised contrastive regularization for domain generalization. In *Proceedings of the ieee/cvf international conference on computer vision*, 9619–9628.

Kirichenko, Polina, Pavel Izmailov, and Andrew Gordon Wilson. 2023. Last layer re-training is sufficient for robustness to spurious correlations. In *The eleventh international conference on learning representations*.

Ko, Ching-Yun, Jeet Mohapatra, Sijia Liu, Pin-Yu Chen, Luca Daniel, and Lily Weng. 2022. Revisiting contrastive learning through the lens of neighborhood component analysis: an integrated framework. In *International conference on machine learning*, 11387–11412. PMLR.

Koehler, Frederic, and Andrej Risteski. 2018. The comparative power of relu networks and polynomial kernels in the presence of sparse latent structure. In *International conference on learning representations*.

Koh, Pang Wei, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. 2021. Wilds: A benchmark of in-the-wild distribution shifts. In *International conference on machine learning*, 5637–5664. PMLR.

Kohler, Jonas Moritz, and Aurelien Lucchi. 2017. Sub-sampled cubic regularization for non-convex optimization. In *International conference on machine learning*, 1895–1904. PMLR.

Kolesnikov, Alexander, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. 2020. Big transfer (bit): General visual representation learning. In *Computer vision - ECCV 2020 - 16th european conference, glasgow,* *uk, august 23-28, 2020, proceedings, part V*, vol. 12350 of *Lecture Notes in Computer Science*, 491–507. Springer.

Koren, Yehuda, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42(8):30–37.

Korlakai Vinayak, Ramya, Samet Oymak, and Babak Hassibi. 2014. Graph clustering with missing data: Convex algorithms and analysis. *Advances in Neural Information Processing Systems* 27.

Kornblith, Simon, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*, 3519–3529. PMLR.

Kotar, Klemen, Gabriel Ilharco, Ludwig Schmidt, Kiana Ehsani, and Roozbeh Mottaghi. 2021. Contrasting contrastive self-supervised representation learning pipelines. In *2021 IEEE/CVF international conference on computer vision, ICCV 2021, montreal, qc, canada, october 10-17, 2021*, 9929–9939. IEEE.

Krizhevsky, Alex. 2012. Learning multiple layers of features from tiny images. *University of Toronto*.

Krizhevsky, Alex, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. *arXiv*.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Krueger, David, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. 2021. Out-of-distribution generalization via risk extrapolation (rex). In *International conference on machine learning*, 5815–5826. PMLR.

Kumar, Abhishek, and Hal Daume III. 2012. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*.

Kumar, Tanishq, Blake Bordelon, Samuel J Gershman, and Cengiz Pehlevan. 2023. Grokking as the transition from lazy to rich training dynamics. *arXiv preprint arXiv:2310.06110*.

Le, Ya, and Xuan Yang. 2015. Tiny imagenet visual recognition challenge. *CS 231N*.

LeCun, Yann, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Lee, Doyup, Sungwoong Kim, Ildoo Kim, Yeongjae Cheon, Minsu Cho, and Wook-Shin Han. 2022. Contrastive regularization for semi-supervised learning. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 3911–3920.

Lee, Jaehoon, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2018. Deep neural networks as gaussian processes. In *International conference on learning representations*.

Lee, Jaehoon, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. 2020. Finite versus infinite neural networks: an empirical study. *Advances in Neural Information Processing Systems* 33: 15156–15172.

Lee, Jaehoon, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. 2019a. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*.

Lee, Jason D, Qi Lei, Nikunj Saunshi, and Jiacheng Zhuo. 2021. Predicting what you already know helps: Provable self-supervised learning. *Advances in Neural Information Processing Systems* 34:309–323.

Lee, Yin Tat, Zhao Song, and Qiuyi Zhang. 2019b. Solving empirical risk minimization in the current matrix multiplication time. In *Conference on learning theory*, 2140–2157. PMLR.

Lester, Brian, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*. Association for Computational Linguistics.

Lewkowycz, Aitor, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems* 35:3843–3857.

Li, Bo, Yifei Shen, Jingkang Yang, Yezhen Wang, Jiawei Ren, Tong Che, Jun Zhang, and Ziwei Liu. 2023a. Sparse mixture-of-experts are domain generalizable learners. In *The eleventh international conference on learning representations*.

Li, Chenyang, Yingyu Liang, Zhenmei Shi, and Zhao Song. 2024a. Exploring the frontiers of softmax: Provable optimization, applications in diffusion model, and beyond. *manuscript*.

Li, Chenyang, Yingyu Liang, Zhenmei Shi, Zhao Song, and Tianyi Zhou. 2024b. Fourier circuits in neural networks: Unlocking the potential of large language models in mathematical reasoning and modular arithmetic. *arXiv preprint arXiv:2402.09469*.

Li, Da, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. 2018a. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the aaai conference on artificial intelligence*.

Li, Da, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. 2017. Deeper, broader and artier domain generalization. In *Proceedings of the ieee international conference on computer vision*, 5542–5550.

Li, Haoliang, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. 2018b. Domain generalization with adversarial feature learning. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 5400–5409.

Li, Hongkang, Meng Wang, Sijia Liu, and Pin-Yu Chen. 2023b. A theoretical understanding of shallow vision transformers: Learning, generalization, and sample complexity. In *The eleventh international conference on learning representations*.

Li, Hongkang, Meng Wang, Songtao Lu, Hui Wan, Xiaodong Cui, and Pin-Yu Chen. 2023c. Transformers as multi-task feature selectors: Generalization analysis of in-context learning. In *Neurips 2023 workshop on mathematics of modern machine learning*.

Li, Jingyao, Han Shi, Xin Jiang, Zhenguo Li, Hong Xu, and Jiaya Jia. 2024c. Quickllama: Query-aware inference acceleration for large language models. *arXiv preprint arXiv:2406.07528*.

Li, Xiang Lisa, and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing*. Association for Computational Linguistics.

Li, Xiaodong, Yudong Chen, and Jiaming Xu. 2021. Convex relaxation methods for community detection. *Statistical Science* 36(1):2–15.

Li, Xiaoyu, Yuanpeng Li, Yingyu Liang, Zhenmei Shi, and Zhao Song. 2024d. On the expressive power of modern hopfield networks. *arXiv preprint arXiv:2412.05562*.

Li, Xiaoyu, Yingyu Liang, Zhenmei Shi, and Zhao Song. 2024e. A tighter complexity analysis of sparsegpt. *arXiv preprint arXiv:2408.12151*.

Li, Xiaoyu, Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. 2024f. Fine-grained attention i/o complexity: Comprehensive analysis for backward passes. *arXiv preprint arXiv:2410.09397*.

Li, Ya, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. 2018c. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the european conference on computer vision (eccv)*, 624–639.

Li, Yingcong, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. 2023d. Transformers as algorithms: Generalization and stability in in-context learning. In *Proceedings of the 40th international conference on machine learning*. Proceedings of Machine Learning Research, PMLR.

Li, Yingcong, Kartik Sreenivasan, Angeliki Giannou, Dimitris Papailiopoulos, and Samet Oymak. 2023e. Dissecting chain-of-thought: Compositionality through in-context filtering and learning. In *Thirty-seventh conference on neural information processing systems*.

Li, Yuanzhi, and Yingyu Liang. 2018. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in neural information processing systems*.

Li, Yuanzhi, Tengyu Ma, and Hongyang R Zhang. 2020. Learning over-parametrized two-layer neural networks beyond ntk. In *Conference on learning theory*.

Li, Yuanzhi, Colin Wei, and Tengyu Ma. 2019. Towards explaining the regularization effect of initial large learning rate in training neural networks. *Advances in Neural Information Processing Systems*.

Li, Yuchen, Yuanzhi Li, and Andrej Risteski. 2023f. How do transformers learn topic structure: Towards a mechanistic understanding. In *Proceedings of the 40th international conference on machine learning*. Proceedings of Machine Learning Research, PMLR.

Li, Yucheng, Bo Dong, Chenghua Lin, and Frank Guerin. 2023g. Compressing context to enhance inference efficiency of large language models. *arXiv preprint arXiv:2310.06201*.

Li, Yuhong, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. 2024g. Snapkv: Llm knows what you are looking for before generation. *arXiv preprint arXiv:2404.14469*.

Liang, Yingyu, Heshan Liu, Zhenmei Shi, Zhao Song, and Junze Yin. 2024a. Conv-basis: A new paradigm for efficient attention inference and gradient computation in transformers. *arXiv preprint arXiv:2405.05219*.

Liang, Yingyu, Jiangxuan Long, Zhenmei Shi, Zhao Song, and Yufa Zhou. 2024b. Beyond linear approximations: A novel pruning approach for attention matrix.

Liang, Yingyu, Zhizhou Sha, Zhenmei Shi, Zhao Song, and Yufa Zhou. 2024c. Looped relu mlps may be all you need as practical programmable computers. *arXiv preprint arXiv:2410.09375*.

———. 2024d. Multi-layer transformers gradient can be approximated in almost linear time. *arXiv preprint arXiv:2408.13233*.

Liang, Yingyu, Zhenmei Shi, Zhao Song, and Chiwun Yang. 2024e. Toward infinite-long prefix in transformer. *arXiv preprint arXiv:2406.14036*.

Liang, Yingyu, Zhenmei Shi, Zhao Song, and Yufa Zhou. 2024f. Differential privacy of cross-attention with provable guarantee. *arXiv preprint arXiv:2407.14717*.

———. 2024g. Tensor attention training: Provably efficient learning of higher-order transformers. *arXiv preprint arXiv:2405.16411*.

———. 2024h. Unraveling the smoothness properties of diffusion models: A gaussian mixture perspective. *arXiv preprint arXiv:2405.16418*.

Liu, Evan Z, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021a. Just train twice: Improving group robustness without training group information. In *International conference on machine learning*, 6781–6792. PMLR.

Liu, Hong, Jeff Z HaoChen, Adrien Gaidon, and Tengyu Ma. 2021b. Self-supervised learning is more robust to dataset imbalance. In *Neurips 2021 workshop on distribution shifts: Connecting methods and applications*.

Liu, Ziming, Ouail Kitouni, Niklas S Nolte, Eric Michaud, Max Tegmark, and Mike Williams. 2022. Towards understanding grokking: An effective theory of representation learning. *Advances in Neural Information Processing Systems* 35: 34651–34663.

Loshchilov, Ilya, and Frank Hutter. 2018. Decoupled weight decay regularization. In *International conference on learning representations*.

Lu, Canyi, Jiashi Feng, Yudong Chen, Wei Liu, Zhouchen Lin, and Shuicheng Yan. 2019. Tensor robust principal component analysis with a new tensor nuclear norm. *IEEE transactions on pattern analysis and machine intelligence* 42(4):925–938.

Luo, Chuanchen, Chunfeng Song, and Zhaoxiang Zhang. 2020. Generalizing person re-identification by camera-aware invariance learning and cross-domain mixup. In *European conference on computer vision*.

Luo, Renqian, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. 2022. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics* 23(6).

Luo, Tao, Zhi-Qin John Xu, Zheng Ma, and Yaoyu Zhang. 2021. Phase diagram for two-layer relu neural networks at infinite-width limit. *Journal of Machine Learning Research* 22(71):1–47.

Luo, Zeping, Shiyou Wu, Cindy Weng, Mo Zhou, and Rong Ge. 2023. Understanding the robustness of self-supervised learning through topic modeling. In *The eleventh international conference on learning representations*.

Lyu, Kaifeng, Jikai Jin, Zhiyuan Li, Simon S Du, Jason D Lee, and Wei Hu. 2024. Dichotomy of early and late phase implicit biases can provably induce grokking. In *The twelfth international conference on learning representations*.

Lyu, Kaifeng, and Jian Li. 2019. Gradient descent maximizes the margin of homogeneous neural networks. In *International conference on learning representations*.

Lyu, Kaifeng, Zhiyuan Li, Runzhe Wang, and Sanjeev Arora. 2021. Gradient descent on two-layer nets: Margin maximization and simplicity bias. *Advances in Neural Information Processing Systems* 34:12978–12991.

Ma, Kaili, Haochen Yang, Han Yang, Tatiana Jin, Pengfei Chen, Yongqiang Chen, Barakeel Fanseu Kamhoua, and James Cheng. 2021. Improving graph representation learning by contrastive regularization. *arXiv preprint arXiv:2101.11525*.

Maas, Andrew L., Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, 142–150. Portland, Oregon, USA: Association for Computational Linguistics.

Van der Maaten, Laurens, and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research* 9(11).

Mahankali, Arvind, Tatsunori B Hashimoto, and Tengyu Ma. 2023. One step of gradient descent is provably the optimal in-context learner with one layer of linear self-attention. *arXiv preprint arXiv:2307.03576*.

Malach, Eran, Pritish Kamath, Emmanuel Abbe, and Nathan Srebro. 2021. Quantifying the benefit of using differentiable learning over tangent kernels. *arXiv preprint arXiv:2103.01210*.

Malladi, Sadhika, Tianyu Gao, Eshaan Nichani, Alex Damian, Jason D Lee, Danqi Chen, and Sanjeev Arora. 2023. Fine-tuning language models with just forward passes. *Advances in Neural Information Processing Systems*.

Manning, Christopher D, Kevin Clark, John Hewitt, Urvashi Khandelwal, and Omer Levy. 2020. Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences* 117(48):30046–30054.

Matthews, Alexander G de G, Mark Rowland, Jiri Hron, Richard E Turner, and Zoubin Ghahramani. 2018. Gaussian process behaviour in wide deep neural networks. In *International conference on learning representations*.

Mei, Song, Theodor Misiakiewicz, and Andrea Montanari. 2019. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. In *Conference on learning theory*, 2388–2464. PMLR.

Mei, Song, Andrea Montanari, and Phan-Minh Nguyen. 2018. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences* 115(33):E7665–E7671.

Meng, Kevin, David Bau, Alex Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems* 35:17359–17372.

Meng, Rang, Xianfeng Li, Weijie Chen, Shicai Yang, Jie Song, Xinchao Wang, Lei Zhang, Mingli Song, Di Xie, and Shiliang Pu. 2022b. Attention diversification for domain generalization. In *Computer vision–eccv 2022: 17th european conference, tel aviv, israel, october 23–27, 2022, proceedings, part xxxiv*, 322–340. Springer.

Merrill, William, Nikolaos Tsilivis, and Aman Shukla. 2023. A tale of two circuits: Grokking as competition of sparse and dense subnetworks. *arXiv preprint arXiv:2303.11873*.

Michalowicz, JV, JM Nichols, F Bucholtz, and CC Olson. 2009. An isserlis' theorem for mixed gaussian variables: Application to the auto-bispectral density. *Journal of Statistical Physics*.

Millidge, Beren. 2022. Grokking'grokking'.

Min, Sewon, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

Min, Sewon, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 conference on empirical methods in natural language processing*.

Ming, Yifei, Hang Yin, and Yixuan Li. 2022. On the impact of spurious correlation for out-of-distribution detection. In *The aaai conference on artificial intelligence (aaai)*.

Mishra, Swaroop, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. In *Proceedings of the 60th annual meeting of the association for computational linguistics*.

Moroshko, Edward, Blake E Woodworth, Suriya Gunasekar, Jason D Lee, Nati Srebro, and Daniel Soudry. 2020. Implicit bias in deep linear classification: Initialization scale vs training accuracy. *Advances in Neural Information Processing Systems* 33.

Morwani, Depen, Benjamin L Edelman, Costin-Andrei Oncescu, Rosie Zhao, and Sham Kakade. 2024. Feature emergence via margin maximization: case studies in algebraic tasks. In *The twelfth international conference on learning representations*.

Mousavi-Hosseini, Alireza, Sejun Park, Manuela Girotti, Ioannis Mitliagkas, and Murat A Erdogdu. 2022. Neural networks efficiently learn low-dimensional representations with sgd. *arXiv preprint arXiv:2209.14863*.

Murty, Shikhar, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. 2023. Grokking of hierarchical structure in vanilla transformers. *arXiv preprint arXiv:2305.18741*.

Nacson, Mor Shpigel, Suriya Gunasekar, Jason Lee, Nathan Srebro, and Daniel Soudry. 2019a. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In *International conference on machine learning*, 4683–4692. PMLR.

Nacson, Mor Shpigel, Jason Lee, Suriya Gunasekar, Pedro Henrique Pamplona Savarese, Nathan Srebro, and Daniel Soudry. 2019b. Convergence of gradient descent on separable data. In *The 22nd international conference on artificial intelligence and statistics*, 3420–3428. PMLR.

Nacson, Mor Shpigel, Nathan Srebro, and Daniel Soudry. 2019c. Stochastic gradient descent on separable data: Exact convergence with a fixed learning rate. In *The 22nd international conference on artificial intelligence and statistics*, 3051–3059. PMLR.

Nagarajan, Vaishnavh, and J Zico Kolter. 2019. Uniform convergence may be unable to explain generalization in deep learning. *Advances in Neural Information Processing Systems* 32.

Nakkiran, Preetum, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. 2020. Deep double descent: Where bigger models and more data hurt. In *International conference on learning representations*.

Nakkiran, Preetum, Gal Kaplun, Dimitris Kalimeris, Tristan Yang, Benjamin L Edelman, Fred Zhang, and Boaz Barak. 2019. Sgd on neural networks learns functions of increasing complexity. *arXiv preprint arXiv:1905.11604*.

Nakos, Vasileios, Zhao Song, and Zhengyu Wang. 2019. (nearly) sample-optimal sparse fourier transform in any dimension; ripless and filterless. In *2019 ieee 60th annual symposium on foundations of computer science (focs)*, 1568–1577. IEEE.

Nam, Hyeonseob, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. 2021. Reducing domain gap by reducing style bias. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 8690–8699.

Nanda, Neel, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023a. Progress measures for grokking via mechanistic interpretability. In *The eleventh international conference on learning representations*.

Nanda, Neel, Andrew Lee, and Martin Wattenberg. 2023b. Emergent linear representations in world models of self-supervised sequence models. *arXiv preprint arXiv:2309.00941*.

Netzer, Yuval, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning. *Advances in Neural Information Processing Systems*.

Newell, Alejandro, and Jia Deng. 2020. How useful is self-supervised pretraining for visual tasks? In *2020 IEEE/CVF conference on computer vision and pattern recognition, CVPR 2020, seattle, wa, usa, june 13-19, 2020*, 7343–7352. Computer Vision Foundation / IEEE.

Neyshabur, Behnam. 2017. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*.

Ng, Hong-Wei, and Stefan Winkler. 2014. A data-driven approach to cleaning large face datasets. In *2014 ieee international conference on image processing (icip)*, 343–347. IEEE.

Novak, Roman, Lechao Xiao, Jaehoon Lee, Yasaman Bahri, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. 2019. Bayesian convolutional neural networks with many channels are gaussian processes. In *International conference on learning representations*.

Nye, Maxwell, Anders Johan Andreassen, Gur AriGuy, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, et al. 2021. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*.

O'Donnell, Ryan. 2014. *Analysis of boolean functions*. Cambridge University Press.

Olah, Chris, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill* 5(3):e00024–001.

Olshausen, B., and D. Field. 1997. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research* 37:3311–3325.

Olsson, Catherine, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

van den Oord, Aäron, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *CoRR* abs/1807.03748. 1807.03748.

OpenAI. 2022. Introducing ChatGPT. https://openai.com/blog/chatgpt. Accessed: 2023-09-10.

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*.

Oymak, Samet, Zalan Fabian, Mingchen Li, and Mahdi Soltanolkotabi. 2019. Generalization guarantees for neural networks via harnessing the low-rank structure of the jacobian. *arXiv preprint arXiv:1906.05392*.

Oymak, Samet, and Mahdi Soltanolkotabi. 2019. Overparameterized nonlinear learning: Gradient descent takes the shortest path? In *International conference on machine learning*, 4951–4960. PMLR.

———. 2020. Toward moderate overparameterization: Global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory* 1(1):84–105.

Pan, Jane, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning 'learns' in-context: Disentangling task recognition and task learning. In *Findings of association for computational linguistics (acl)*.

Pan, Zhuoshi, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Ruhle, Yuqing Yang, Chin-Yew Lin, H. Vicky Zhao, Lili Qiu, and Dongmei Zhang. 2024. LLMLingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. In *Findings of the association for computational linguistics acl 2024*, ed. Lun-Wei Ku, Andre Martins, and Vivek Srikumar, 963–981. Bangkok, Thailand and virtual meeting: Association for Computational Linguistics.

Panigrahi, Abhishek, Sadhika Malladi, Mengzhou Xia, and Sanjeev Arora. 2023. Trainable transformer in transformer. *arXiv preprint arXiv:2307.01189*.

Papyan, Vardan, XY Han, and David L Donoho. 2020. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences* 117(40):24652–24663.

Parascandolo, Giambattista, Alexander Neitz, ANTONIO ORVIETO, Luigi Gresele, and Bernhard Schölkopf. 2020. Learning explanations that are hard to vary. In *International conference on learning representations*.

Pearson, Karl. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science* 2(11):559–572.

Peng, Xingchao, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the ieee/cvf international conference on computer vision*, 1406–1415.

Peters, Matthew E, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Pourreza, Mohammadreza, and Davood Rafiei. 2023. Din-sql: Decomposed in-context learning of text-to-sql with self-correction. *arXiv preprint arXiv:2304.11015*.

Power, Alethea, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*.

Prato, Gabriele, Ella Charlaix, and Mehdi Rezagholizadeh. 2020. Fully quantized transformer for machine translation. In *Findings of the association for computational linguistics: Emnlp 2020*, 1–14.

Quirke, Philip, and Fazl Barez. 2023. Understanding addition in transformers. In *The twelfth international conference on learning representations*.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th international conference on machine learning, ICML 2021, 18-24 july 2021, virtual event*, vol. 139 of *Proceedings of Machine Learning Research*, 8748–8763. PMLR.

Radhakrishnan, Adityanarayanan, Daniel Beaglehole, Parthe Pandit, and Mikhail Belkin. 2023. Mechanism of feature learning in deep fully connected networks and kernel machines that recursively learn features. 2212.13881.

Rahimi, Ali, and Benjamin Recht. 2008. Random features for large-scale kernel machines. In *Advances in neural information processing systems*.

Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 conference on empirical methods in natural language processing, EMNLP 2016, austin, texas, usa, november 1-4, 2016*, ed. Jian Su, Xavier Carreras, and Kevin Duh, 2383–2392. The Association for Computational Linguistics.

Rame, Alexandre, Corentin Dancette, and Matthieu Cord. 2022. Fishr: Invariant gradient variances for out-of-distribution generalization. In *International conference on machine learning*, 18347–18377. PMLR.

Ramesh, Aditya, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Raventos, Allan, Mansheej Paul, Feng Chen, and Surya Ganguli. 2023. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. In *Thirty-seventh conference on neural information processing systems*.

Recht, Benjamin, Maryam Fazel, and Pablo A Parrilo. 2010. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review* 52(3):471–501.

Reddy, Gautam. 2024. The mechanistic basis of data dependence and abrupt learning in an in-context classification task. In *The twelfth international conference on learning representations*.

Refinetti, Maria, Sebastian Goldt, Florent Krzakala, and Lenka Zdeborov. 2021. Classifying high-dimensional gaussian mixtures: Where kernel methods fail and neural networks succeed. In *International conference on machine learning*, 8936–8947. PMLR.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, 91–99.

Ren, Yunwei, Mo Zhou, and Rong Ge. 2023. Depth separation with multilayer mean-field networks. In *The eleventh international conference on learning representations*.

Ridnik, Tal, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. 2021. Imagenet-21k pretraining for the masses. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 1)*.

Rosenfeld, Elan, Pradeep Ravikumar, and Andrej Risteski. 2021. The risks of invariant risk minimization. In *International conference on learning representations*, vol. 9.

———. 2022. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv preprint arXiv:2202.06856*.

Rubin, Noa, Inbar Seroussi, and Zohar Ringel. 2023. Droplets of good representations: Grokking as a first order phase transition in two layer networks. *arXiv preprint arXiv:2310.03789*.

Sagawa, Shiori, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *International Conference on Learning Representations, ICLR*.

Sanford, Clayton, Daniel Hsu, and Matus Telgarsky. 2023. Representational strengths and limitations of transformers. In *Thirty-seventh conference on neural information processing systems*.

Saunshi, Nikunj, Jordan Ash, Surbhi Goel, Dipendra Misra, Cyril Zhang, Sanjeev Arora, Sham Kakade, and Akshay Krishnamurthy. 2022. Understanding contrastive learning requires incorporating inductive biases. In *Proceedings of the 39th international conference on machine learning*, ed. Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, vol. 162 of *Proceedings of Machine Learning Research*, 19250–19286. PMLR.

Saunshi, Nikunj, Orestis Plevrakis, Sanjeev Arora, Mikhail Khodak, and Hrishikesh Khandeparkar. 2019. A theoretical analysis of contrastive unsupervised representation learning. In *International conference on machine learning*, 5628–5637.

Saxena, Eshika, Alberto Alfarano, Emily Wenger, and Kristin Lauter. 2024. Teaching transformers modular arithmetic at scale. *arXiv preprint arXiv:2410.03569*.

Saxton, David, Edward Grefenstette, Felix Hill, and Pushmeet Kohli. 2018. Analysing mathematical reasoning abilities of neural models. In *International conference on learning representations*.

Schlag, Imanol, Kazuki Irie, and Jürgen Schmidhuber. 2021. Linear transformers are secretly fast weight programmers. In *International conference on machine learning*. PMLR.

Schulman, John, Barret Zoph, Christina Kim, Jacob Hilton, Jacob Menick, Jiayi Weng, Juan Felipe Ceron Uribe, Liam Fedus, Luke Metz, Michael Pokorny, et al. 2022. Chatgpt: Optimizing language models for dialogue. *OpenAI blog* 2(4).

Shah, Harshay, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. 2020. The pitfalls of simplicity bias in neural networks. In *Neurips*.

Shah, Jay, Ganesh Bikshandi, Ying Zhang, Vijay Thakkar, Pradeep Ramani, and Tri Dao. 2024. Flashattention-3: Fast and accurate attention with asynchrony and low-precision. *arXiv preprint arXiv:2407.08608*.

Shalev-Shwartz, Shai, Ohad Shamir, and Shaked Shammah. 2017. Failures of gradient-based deep learning. In *International conference on machine learning*, 3067–3075. PMLR.

Sharma, Piyush, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)*, 2556–2565.

Shen, Kendrick, Robbie M Jones, Ananya Kumar, Sang Michael Xie, Jeff Z HaoChen, Tengyu Ma, and Percy Liang. 2022. Connect, not collapse: Explaining contrastive learning for unsupervised domain adaptation. In *International conference on machine learning*, 19847–19878. PMLR.

Shi, Freda, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023a. Large language models can be easily distracted by irrelevant context. In *International conference on machine learning*. PMLR.

Shi, Yuge, Jeffrey Seely, Philip Torr, Siddharth N, Awni Hannun, Nicolas Usunier, and Gabriel Synnaeve. 2022a. Gradient matching for domain generalization. In *International conference on learning representations*.

Shi, Zhenmei, Jiefeng Chen, Kunyang Li, Jayaram Raghuram, Xi Wu, Yingyu Liang, and Somesh Jha. 2023b. The trade-off between universality and label efficiency of representations from contrastive learning. In *The eleventh international conference on learning representations*.

Shi, Zhenmei, Yifei Ming, Ying Fan, Frederic Sala, and Yingyu Liang. 2023c. Domain generalization via nuclear norm regularization. In *Conference on parsimony and learning (proceedings track)*.

Shi, Zhenmei, Yifei Ming, Xuan-Phi Nguyen, Yingyu Liang, and Shafiq Joty. 2024a. Discovering the gems in early layers: Accelerating long-context llms with 1000x input token reduction. *arXiv preprint arXiv:2409.17422*.

Shi, Zhenmei, Fuhao Shi, Wei-Sheng Lai, Chia-Kai Liang, and Yingyu Liang. 2022b. Deep online fused video stabilization. In *Proceedings of the ieee/cvf winter conference on applications of computer vision*, 1250–1258.

Shi, Zhenmei, Junyi Wei, and Yingyu Liang. 2022c. A theoretical analysis on feature learning in neural networks: Emergence from inputs and advantage over fixed features. In *International conference on learning representations*.

———. 2023d. Provable guarantees for neural networks via gradient feature learning. *Advances in Neural Information Processing Systems* 36.

Shi, Zhenmei, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. 2024b. Why larger language models do in-context learning differently? *arXiv preprint arXiv:2405.19592*.

Silver, David, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of go with deep neural networks and tree search. *nature* 529(7587):484.

Silver, David, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy P. Lillicrap, Karen Simonyan, and Demis Hassabis. 2017. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *CoRR* abs/1712.01815. 1712.01815.

Sirignano, Justin, and Konstantinos Spiliopoulos. 2020. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications* 130(3):1820–1852.

Song, Zhao. 2019. Matrix theory: Optimization, concentration and algorithms. Ph.D. thesis, The University of Texas at Austin.

Song, Zhao, Baocheng Sun, Omri Weinstein, and Ruizhe Zhang. 2022. Sparse fourier transform over lattices: A unified approach to signal reconstruction. *arXiv preprint arXiv:2205.00658*.

———. 2023a. Quartic samples suffice for fourier interpolation. In *2023 ieee 64th annual symposium on foundations of computer science (focs)*, 1414–1425. IEEE.

Song, Zhao, Mingquan Ye, Junze Yin, and Lichen Zhang. 2023b. A nearly-optimal bound for fast regression with $\ell_\infty$, guarantee. In *ICML*, vol. 202 of *Proceedings of Machine Learning Research*, 32463–32482. PMLR.

Song, Zhao, Mingquan Ye, and Lichen Zhang. 2023c. Streaming semidefinite programs: $o(\sqrt{n})$ passes, small space and fast runtime. 2309.05135.

Soudry, Daniel, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. 2018. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research* 19(1):2822–2878.

Stallkamp, Johannes, Marc Schlipsing, Jan Salmen, and Christian Igel. 2012. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks* 32:323–332.

Stander, Dashiell, Qinan Yu, Honglu Fan, and Stella Biderman. 2023. Grokking group multiplication with cosets. *arXiv preprint arXiv:2312.06581*.

Stöger, Dominik, and Mahdi Soltanolkotabi. 2021. Small random initialization is akin to spectral learning: Optimization and generalization guarantees for overparameterized low-rank matrix reconstruction. *Advances in Neural Information Processing Systems* 34:23831–23843.

Su, Jianlin, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing* 568:127063.

Sun, Baochen, and Kate Saenko. 2016. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, 443–450. Springer.

Sun, Tianxiang, Yunfan Shao, Hong Qian, Xuanjing Huang, and Xipeng Qiu. 2022. Black-box tuning for language-model-as-a-service. In *International conference on machine learning*. PMLR.

Sun, Yiyou, Zhenmei Shi, and Yixuan Li. 2023a. A graph-theoretic framework for understanding open-world semi-supervised learning. *Advances in Neural Information Processing Systems* 36.

Sun, Yiyou, Zhenmei Shi, Yingyu Liang, and Yixuan Li. 2023b. When and how does known class help discover unknown ones? provable understanding through spectral analysis. In *International conference on machine learning*, 33014–33043. PMLR.

Sung, Yi-Lin, Jaemin Cho, and Mohit Bansal. 2022. Vl-adapter: Parameter-efficient transfer learning for vision-and-language tasks. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 5227–5237.

Team, Gemini, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Team, Gemma, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette

Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Telgarsky, Matus. 2022. Feature selection with gradient descent on two-layer networks in low-rotation regimes. *arXiv preprint arXiv:2208.02789*.

Thilak, Vimal, Etai Littwin, Shuangfei Zhai, Omid Saremi, Roni Paiss, and Joshua Susskind. 2022. The slingshot mechanism: An empirical study of adaptive optimizers and the grokking phenomenon. *arXiv preprint arXiv:2206.04817*.

Tian, Yuandong. 2022. Deep contrastive learning is provably (almost) principal component analysis. *arXiv preprint arXiv:2201.12680*.

Tian, Yuandong, Yiping Wang, Beidi Chen, and Simon Du. 2023a. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. *Advances in Neural Information Processing Systems*.

Tian, Yuandong, Yiping Wang, Zhenyu Zhang, Beidi Chen, and Simon Du. 2023b. Joma: Demystifying multilayer transformers via joint dynamics of mlp and attention. *arXiv preprint arXiv:2310.00535*.

Tigges, Curt, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. 2023. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*.

Torralba, Antonio. 2003. Contextual priming for object detection. *International journal of computer vision* 53(2):169–191.

Tosh, Christopher, Akshay Krishnamurthy, and Daniel Hsu. 2021. Contrastive learning, multi-view redundancy, and linear models. In *Algorithmic learning theory*, 1179–1206. PMLR.

Touvron, Hugo, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Touvron, Hugo, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Tsai, Yao-Hung Hubert, Yue Wu, Ruslan Salakhutdinov, and Louis-Philippe Morency. 2020. Self-supervised learning from a multi-view perspective. In *International conference on learning representations*.

Tsigler, Alexander, and Peter L Bartlett. 2023. Benign overfitting in ridge regression. *Journal of Machine Learning Research* 24(123):1–76.

Van Gansbeke, Wouter, Simon Vandenhende, Stamatios Georgoulis, and Luc V Gool. 2021. Revisiting contrastive methods for unsupervised learning of visual representations. *Advances in Neural Information Processing Systems* 34:16238–16250.

Vapnik, Vladimir N. 1999. An overview of statistical learning theory. *IEEE transactions on neural networks* 10(5):988–999.

Vardi, Gal. 2023. On the implicit bias in deep-learning algorithms. *Communications of the ACM* 66(6):86–93.

Varma, Vikrant, Rohin Shah, Zachary Kenton, János Kramár, and Ramana Kumar. 2023. Explaining grokking through circuit efficiency. *arXiv preprint arXiv:2309.02390*.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30.

Veiga, Rodrigo, Ludovic Stephan, Bruno Loureiro, Florent Krzakala, and Lenka Zdeborová. 2022. Phase diagram of stochastic gradient descent in high-dimensional two-layer neural networks. *arXiv preprint arXiv:2202.00293*.

Venkateswara, Hemanth, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. 2017. Deep hashing network for unsupervised domain adaptation.

In *Proceedings of the ieee conference on computer vision and pattern recognition*, 5018–5027.

Vinje, William E, and Jack L Gallant. 2000. Sparse coding and decorrelation in primary visual cortex during natural vision. *Science* 287(5456):1273–1276.

Von Oswald, Johannes, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *International conference on machine learning*. PMLR.

Wan, Zhongwei, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. 2024. Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. *arXiv preprint arXiv:2406.18139*.

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*. Association for Computational Linguistics.

Wang, Jiayu, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Yixuan Li, and Neel Joshi. 2024. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. *Advances in Neural Information Processing Systems* 36.

Wang, Tongzhou, and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, 9929–9939. PMLR.

Wang, Yifei, Jonathan Lacotte, and Mert Pilanci. 2020a. The hidden convex optimization landscape of two-layer relu neural networks: an exact characterization of the optimal solutions. *arXiv e-prints* arXiv–2006.

Wang, Yizhong, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, et al. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, 5085–5109.

Wang, Yufei, Haoliang Li, and Alex C Kot. 2020b. Heterogeneous domain generalization via domain mixup. In *Icassp 2020-2020 ieee international conference on acoustics, speech and signal processing (icassp)*, 3622–3626. IEEE.

Wei, Colin, Jason D Lee, Qiang Liu, and Tengyu Ma. 2019. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. *Advances in Neural Information Processing Systems* 32.

Wei, Jason, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *International conference on learning representations*.

Wei, Jason, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022b. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Wei, Jason, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022c. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022d. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*.

Wei, Jerry, Le Hou, Andrew Kyle Lampinen, Xiangning Chen, Da Huang, Yi Tay, Xinyun Chen, Yifeng Lu, Denny Zhou, Tengyu Ma, et al. 2023a. Symbol tuning improves in-context learning in language models. In *The 2023 conference on empirical methods in natural language processing*.

Wei, Jerry, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. 2023b. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*.

Wen, Zixin, and Yuanzhi Li. 2021. Toward understanding the feature learning process of self-supervised contrastive learning. In *International conference on machine learning*, 11112–11122. PMLR.

Wibisono, Kevin Christian, and Yixin Wang. 2023. On the role of unstructured training data in transformers' in-context learning capabilities. In *Neurips 2023 workshop on mathematics of modern machine learning*.

Wick, Gian-Carlo. 1950. The evaluation of the collision matrix. *Physical review*.

Williams, Adina, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies, NAACL-HLT 2018, new orleans, louisiana, usa, june 1-6, 2018, volume 1 (long papers)*, ed. Marilyn A. Walker, Heng Ji, and Amanda Stent, 1112–1122. Association for Computational Linguistics.

Woodworth, Blake, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. 2020. Kernel and rich regimes in overparametrized models. In *Conference on learning theory*.

Wu, Dennis, Jerry Yao-Chieh Hu, Teng-Yun Hsiao, and Han Liu. 2024a. Uniform memory retrieval with larger capacity for modern hopfield models. In *Forty-first international conference on machine learning (icml)*.

Wu, Dennis, Jerry Yao-Chieh Hu, Weijian Li, Bo-Yu Chen, and Han Liu. 2024b. STanhop: Sparse tandem hopfield model for memory-enhanced time series prediction. In *The twelfth international conference on learning representations (iclr)*.

Wu, Jingfeng, Difan Zou, Zixiang Chen, Vladimir Braverman, Quanquan Gu, and Peter L Bartlett. 2024c. How many pretraining tasks are needed for in-context learning of linear regression? In *The twelfth international conference on learning representations*.

Xiao, Guangxuan, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Xiao, Han, Kashif Rasul, and Roland Vollgraf. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Xiao, Kai Yuanqing, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. 2021. Noise or signal: The role of image backgrounds in object recognition. In *International conference on learning representations*.

Xie, Sang Michael, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International conference on learning representations*.

Xu, Chenwei, Yu-Chao Huang, Jerry Yao-Chieh Hu, Weijian Li, Ammar Gilani, Hsi-Sheng Goan, and Han Liu. 2024a. Bishop: Bi-directional cellular learning for tabular data with generalized sparse modern hopfield model. In *Forty-first international conference on machine learning (icml)*.

Xu, Peng, Wei Ping, Xianchao Wu, Lawrence McAfee, Chen Zhu, Zihan Liu, Sandeep Subramanian, Evelina Bakhturina, Mohammad Shoeybi, and Bryan Catanzaro. 2024b. Retrieval meets long context large language models. 2310.03025.

Xu, Yuhui, Zhanming Jie, Hanze Dong, Lei Wang, Xudong Lu, Aojun Zhou, Amrita Saha, Caiming Xiong, and Doyen Sahoo. 2024c. Think: Thinner key cache by query-driven pruning. *arXiv preprint arXiv:2407.21018*.

Xu, Zhiwei, Yutong Wang, Spencer Frei, Gal Vardi, and Wei Hu. 2024d. Benign overfitting and grokking in relu networks for xor cluster data. In *The twelfth international conference on learning representations*.

Xu, Zhuoyan, Zhenmei Shi, and Yingyu Liang. 2024e. Do large language models have compositional ability? an investigation into limitations and scalability. In *Iclr 2024 workshop on mathematical and empirical understanding of foundation models*.

Xu, Zhuoyan, Zhenmei Shi, Junyi Wei, Yin Li, and Yingyu Liang. 2023. Improving foundation models for few-shot learning via multitask finetuning. In *Iclr 2023 workshop on mathematical and empirical understanding of foundation models*.

Xu, Zhuoyan, Zhenmei Shi, Junyi Wei, Fangzhou Mu, Yin Li, and Yingyu Liang. 2024f. Towards few-shot adaptation of foundation models via multitask finetuning. In *The twelfth international conference on learning representations*.

Yair, Noam, and Tomer Michaeli. 2018. Multi-scale weighted nuclear norm image restoration. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 3165–3174.

Yan, Shen, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. 2020. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*.

Yang, Greg. 2019. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*.

Yang, Greg, and Edward J Hu. 2020. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*.

Yang, Jianwei, Chunyuan Li, Pengchuan Zhang, Bin Xiao, Ce Liu, Lu Yuan, and Jianfeng Gao. 2022. Unified contrastive learning in image-text-label space. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 19163–19173.

Yang, Liu, Kangwook Lee, Robert Nowak, and Dimitris Papailiopoulos. 2023. Looped transformers are better at learning learning algorithms. *arXiv preprint arXiv:2311.12424*.

Yang, Xingyi, Xuehai He, Yuxiao Liang, Yue Yang, Shanghang Zhang, and Pengtao Xie. 2020. Transfer learning or self-supervised learning? A tale of two pretraining paradigms. *CoRR* abs/2007.04234. 2007.04234.

Yang, Zhaoyang, Zhenmei Shi, Xiaoyong Shen, and Yu-Wing Tai. 2019. Sf-net: Structured feature network for continuous sign language recognition. *arXiv preprint arXiv:1908.01341*.

Yao, Huaxiu, Yu Wang, Sai Li, Linjun Zhang, Weixin Liang, James Zou, and Chelsea Finn. 2022a. Improving out-of-distribution robustness via selective augmentation. In *Proceeding of the thirty-ninth international conference on machine learning*.

Yao, Shunyu, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik R Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. In *Thirty-seventh conference on neural information processing systems*.

Yao, Xufeng, Yang Bai, Xinyun Zhang, Yuechen Zhang, Qi Sun, Ran Chen, Ruiyu Li, and Bei Yu. 2022b. Pcl: Proxy-based contrastive learning for domain generalization. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 7097–7107.

Yehudai, Gilad, and Shamir Ohad. 2020. Learning a single neuron with gradient methods. In *Conference on learning theory*.

Yehudai, Gilad, and Ohad Shamir. 2019. On the power and limitations of random features for understanding neural networks. *Advances in Neural Information Processing Systems*.

Yousefzadeh, Roozbeh, and Xuenan Cao. 2023. Large language models' understanding of math: Source criticism and extrapolation. *arXiv preprint arXiv:2311.07618*.

Zbontar, Jure, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. 2021. Barlow twins: Self-supervised learning via redundancy reduction. In *Proceedings of the 38th international conference on machine learning, ICML 2021, 18-24 july 2021, virtual event*, vol. 139 of *Proceedings of Machine Learning Research*, 12310–12320. PMLR.

Zech, John R, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. 2018. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine* 15(11).

Zeiler, Matthew D, and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*.

Zhang, Chiyuan, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *International conference on learning representations*.

Zhang, Chiyuan, Samy Bengio, and Yoram Singer. 2019. Are all layers created equal? *arXiv preprint arXiv:1902.01996*.

Zhang, Hanlin, Yi-Fan Zhang, Yaodong Yu, Dhruv Madeka, Dean Foster, Eric Xing, Hima Lakkaraju, and Sham Kakade. 2023a. A study on the calibration of in-context learning. *arXiv preprint arXiv:2312.04021*.

Zhang, Hongyi, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *International conference on learning representations*.

Zhang, Marvin, Henrik Marklund, Abhishek Gupta, Sergey Levine, and Chelsea Finn. 2020. Adaptive risk minimization: A meta-learning approach for tackling group shift. *arXiv preprint arXiv:2007.02931* 8:9.

Zhang, Michael, Nimit S. Sohoni, Hongyang R. Zhang, Chelsea Finn, and Christopher Ré. 2022a. Correct-n-contrast: A contrastive approach for improving robustness to spurious correlations. In *International conference on machine learning*.

Zhang, Renrui, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. 2023b. Llama-adapter: Efficient fine-tuning of language models with zero-init attention. *arXiv preprint arXiv:2303.16199*.

Zhang, Ruiqi, Spencer Frei, and Peter L Bartlett. 2023c. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*.

Zhang, Shizhuo Dylan, Curt Tigges, Stella Biderman, Maxim Raginsky, and Talia Ringer. 2023d. Can transformers learn to solve problems recursively? *arXiv preprint arXiv:2305.14699*.

Zhang, Xiang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Nips*.

Zhang, Xingxuan, Linjun Zhou, Renzhe Xu, Peng Cui, Zheyan Shen, and Haoxin Liu. 2022b. Towards unsupervised domain generalization. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 4910–4920.

Zhang, Zhenyu, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. 2023e. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in Neural Information Processing Systems* 36.

Zhao, Zihao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International conference on machine learning*. PMLR.

Zheng, Huaixiu Steven, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H Chi, Quoc V Le, and Denny Zhou. 2024. Step-back prompting enables reasoning via abstraction in large language models. In *The twelfth international conference on learning representations*.

Zhong, Ziqian, Ziming Liu, Max Tegmark, and Jacob Andreas. 2023. The clock and the pizza: Two stories in mechanistic explanation of neural networks. *Advances in Neural Information Processing Systems* 36.

Zhou, Chunting, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, LILI YU, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. 2023a. LIMA: Less is more for alignment. In *Thirty-seventh conference on neural information processing systems*.

Zhou, Hattie, Arwen Bradley, Etai Littwin, Noam Razin, Omid Saremi, Joshua Susskind, Samy Bengio, and Preetum Nakkiran. 2023b. What algorithms can transformers learn? a study in length generalization. In *The 3rd workshop on mathematical reasoning and ai at neurips'23*.

Zhou, Hattie, Azade Nova, Hugo Larochelle, Aaron Courville, Behnam Neyshabur, and Hanie Sedghi. 2022. Teaching algorithmic reasoning via in-context learning. *arXiv preprint arXiv:2211.09066*.

Zhou, Kaiyang, Yongxin Yang, Timothy Hospedales, and Tao Xiang. 2020. Learning to generate novel domains for domain generalization. In *Computer vision–eccv 2020: 16th european conference, glasgow, uk, august 23–28, 2020, proceedings, part xvi 16*, 561–578. Springer.

Zhou, Kaiyang, Yongxin Yang, Yu Qiao, and Tao Xiang. 2021a. Domain generalization with mixstyle. In *International conference on learning representations*.

Zhou, Mo, Rong Ge, and Chi Jin. 2021b. A local convergence theory for mildly over-parameterized two-layer neural network. In *Colt*.

Zhu, Zhuotun, Lingxi Xie, and Alan Yuille. 2017. Object recognition with and without objects. In *Proceedings of the twenty-sixth international joint conference on artificial intelligence, IJCAI-17*, 3609–3615.

Zimmermann, Roland S, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. 2021. Contrastive learning inverts the data generating process. In *International conference on machine learning*, 12979–12990. PMLR.

Zou, Difan, Yuan Cao, Dongruo Zhou, and Quanquan Gu. 2018. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888*.

———. 2020. Gradient descent optimizes over-parameterized deep ReLU networks. *Machine Learning* 109(3):467–492.