

**TOWARDS BETTER FOUNDATION MODELS: THEORY AND METHODS FOR
ADAPTATION AND DEPLOYMENT**

by

Zhuoyan Xu

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

Doctor of Philosophy

(Statistics)

at the

UNIVERSITY OF WISCONSIN–MADISON

2026

Date of final oral examination: 12/16/2025

The dissertation is approved by the following members of the Final Oral Committee:

Yin Li, Associate Professor, Biostatistics & Medical Informatics and Computer Sciences

Yingyu Liang, Associate Professor, Computer Sciences

Yiqiao Zhong, Assistant Professor, Statistics and Computer Sciences

Maja Waldron, Assistant Professor, Statistics

Junjie Hu, Assistant Professor, Computer Sciences and Biostatistics & Medical Informatics

© Copyright by Zhuoyan Xu 2026
All Rights Reserved

*To my parents,
Juan Shen and Lin Xu.*

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisors: Prof. Yin Li, Prof. Yingyu Liang and Prof. Yiqiao Zhong, for their invaluable guidance, deep expertise and continuous support throughout my Ph.D. journey. I began working with Prof. Liang and Prof. Li in 2022. I can still vividly remember the excitement when I received the email from Prof. Li and Prof. Liang offering to advise me and support me with a research assistantship. Their trust, before I had produced any concrete results, inspired me to work harder. Over the past several years, they consistently provided me with research opportunities, resources, and the freedom to explore ideas deeply. We worked for nearly two years on my first project, which eventually resulted in a successful publication and laid the foundation for many subsequent projects. Their research taste, depth of knowledge, and working style have had a lasting influence on how I approach research, and continually inspired me to push myself further. I started working with Prof. Zhong in 2023. His dedication to research, openness to new and rapidly evolving fields encouraged me to broaden my perspective in research. His guidance played an important role in shaping my development during the mid-to-later stages of my Ph.D. Together, my advisors' mentorship and genuine kindness has been instrumental in shaping not only my research trajectory, but also my growth as an independent researcher.

I would also like to thank my collaborators and mentors during the early-to-mid stages of my Ph.D., especially Prof. Kris Sankaran and Dr. Zhenmei Shi. I worked with Prof. Sankaran during the early stages of my Ph.D. He is an exceptionally kind and thoughtful mentor, and our conversations were always pressure-free and encouraging. His support during my early Ph.D. years helped me navigate periods of uncertainty and self-doubt. Dr. Shi is a former Ph.D. student in Prof. Liang's group, who played a pivotal role in my development as a researcher. During our collaboration, he patiently mentored me through many aspects of academic research, especially when I was at an early stage. He set an example for me through his diligence, rigor, and professionalism. Working with him significantly accelerated my learning curve.

I would like to express my appreciation to my thesis committee members, Prof. Junjie Hu and Prof. Maja Waldron for their insightful feedback and valuable comments. I would also like to thank all the collaborators I worked with during my Ph.D. journey, including Junyi Wei, Fangzhou Mu, Khoi Duc Nguyen, Jiajun Song, Haoyang Fang, and others. Working with them was a valuable learning experience, where I benefited greatly from their their insights, feedback, and collaboration. I am also grateful to many individuals who supported me in various ways, including Zihao Wang, Yuheng Li, Weizheng Wang, Xinran Miao and many others.

Last but not least, I would like to express my deepest gratitude to my family. I am profoundly thankful to my parents for their unconditional support throughout my life. They have always encouraged me to pursue my goals and have provided steadfast support in every way they could.

I feel extremely fortunate to have received such generous support from so many people throughout my Ph.D. journey. This achievement would not have been possible without them.

CONTENTS

Contents iv

List of Tables vi

List of Figures x

Abstract xiii

1 Introduction 1

2 Background and Related Work 6

2.1 *Adaptation of Foundation Models* 7

2.2 *Compositional Ability of LLMs* 9

2.3 *Adaptive Inference of multimodal LLMs (MLLMs)* 11

3 Few-Shot Adaptation via Multitask Finetuning 14

3.1 *Introduction* 14

3.2 *Preliminaries: Multitask Finetuning for Few-Shot Learning* 15

3.3 *Theoretical Analysis: Benefit of Multitask Finetuning* 17

3.4 *Experiments* 25

3.5 *Conclusion* 30

4 Understanding Compositional Abilities 32

4.1 *Introduction* 32

4.2 *Warm-up: A Failure Case for Composition* 35

4.3 *Variability of Compositional Performance* 36

4.4 *Theoretical Analysis* 43

4.5 *Conclusion* 48

5 Adaptive Runtime Inference 50

5.1 *Introduction* 50

| | | |
|----------|--|-----|
| 5.2 | <i>Adaptive Inference of MLLMs</i> | 53 |
| 5.3 | <i>Experiments and Results</i> | 61 |
| 5.4 | <i>Conclusion and Discussion</i> | 69 |
| 6 | Conclusion | 71 |
| 6.1 | <i>Additional collaborative work</i> | 72 |
| 6.2 | <i>Future Research Directions</i> | 73 |
| A | Appendix of Chapter 3 | 76 |
| A.1 | <i>Limitation</i> | 76 |
| A.2 | <i>Deferred Proofs</i> | 76 |
| A.3 | <i>Multi-class Classification</i> | 98 |
| A.4 | <i>Linear Case Study</i> | 101 |
| A.5 | <i>Vision Experimental Results</i> | 112 |
| A.6 | <i>NLP Experimental Results</i> | 127 |
| A.7 | <i>Vision Language Tasks</i> | 136 |
| B | Appendix of Chapter 4 | 140 |
| B.1 | <i>Logical Tasks</i> | 140 |
| B.2 | <i>Formal Language Translation Tasks</i> | 144 |
| B.3 | <i>Theory for Confined Support</i> | 147 |
| B.4 | <i>Deferred Proof</i> | 149 |
| C | Appendix of Chapter 5 | 163 |
| C.1 | <i>Further Implementation Details</i> | 163 |
| C.2 | <i>Detailed Results</i> | 164 |
| C.3 | <i>Additional Ablation Studies</i> | 166 |
| C.4 | <i>Additional Results on Adaptivity</i> | 169 |
| C.5 | <i>Further Discussion</i> | 170 |
| | References | 179 |

LIST OF TABLES

| | | |
|-----|---|----|
| 3.1 | Results evaluating our task selection algorithm on Meta-dataset using ViT-B backbone. No Con.: Ignore consistency. No Div.: Ignore diversity. Random: Ignore both consistency and diversity. | 28 |
| 3.2 | Results of few-shot image classification. We report average classification accuracy (%) with 95% confidence intervals on test splits. Adaptation: Direction adaptation without finetuning; Standard FT: Standard finetuning; Ours: Our multitask finetuning; 1-/5-shot: number of labeled images per class in the target task. | 30 |
| 4.1 | Examples of two settings on composite tasks. Composite: in-context examples are about simple tasks, while the test input is about the composite task. Composite in-context: both in-context examples and the test input are about the composite task. | 35 |
| 4.2 | This table contains a collection of simple logical tasks. The <i>Words</i> category encompasses tasks that modify words at the character or structural level. The <i>Numerical</i> category is devoted to tasks that involve arithmetic computations performed on numbers. | 38 |
| 4.3 | Examples of the two logical composite tasks. Full examples can be found in Section B.1. | 38 |
| 4.4 | Results are evaluated composite tasks on various models. The accuracy is in %. | 40 |
| 5.1 | Efficiency metrics on MLLM benchmarks. Budget (%): input latency budget w.r.t. the base model latency. AdaLLaVA-L: switches on selecting different Transformer blocks. AdaLLaVA-H: switches on selecting different attention heads and MLP activations. Prumerge+ and FastV both use LLaVA 1.5. | 63 |
| 5.2 | VQA benchmark results. VQA ^{v2} : VQAv2 set. SQA ^I : ScienceQA set. VQA ^T : TextVQA set. AdaLLaVA enables a base MLLM to adapt to varying latency budgets with competitive performance. | 64 |

| | | |
|-----|---|-----|
| 5.3 | Hallucination and general benchmark results. POPE evaluates object hallucination. MME and MMBench assess general multimodal understanding. AdaLLaVA can be further integrated with token selection methods to enhance overall efficiency. | 65 |
| 5.4 | Generalization of AdaLLaVA to Mipha-3B. | 69 |
| 5.5 | Ablation on deterministic vs. probabilistic modeling for the scheduler. Results reported using 7B model on VQAv2. | 69 |
| A.1 | Class diversity on ViT-B32 backbone on miniImageNet. | 116 |
| A.2 | Class diversity on ViT-B32 backbone on Omniglot. | 116 |
| A.3 | The performance of the ViT-B backbone using different pretraining methods on tieredImageNet, varying the number of classes accessible to the model during the finetuning stage. Each column represents the number of classes within the training data. | 117 |
| A.4 | Finetuning data selection on model performance. FT data: dataset we select for multitask finetuning. Report the accuracy on the test-split of DomainNet. | 118 |
| A.5 | Results evaluating on DomainNet test-split using ViT-B backbone. First column shows performance where model finetune on data from DomainNet train-split alone, second column shows the performance of the model finetuned using a blend of the same data from DomainNet, combined with additional data from ImageNet. | 119 |
| A.6 | Results evaluating on DomainNet test-split using ViT-B backbone. Adaptation: Direction adaptation without finetuning; SFT: Standard finetuning; Ours: Our multitask finetuning. Col-1 shows performance without any finetuning, Col-2,3,4,5 shows performance with different finetuning methods and data. | 120 |
| A.7 | Results evaluating our task selection algorithm on Meta-dataset using ViT-B backbone. | 121 |
| A.8 | Comparison on 15-way classification. Average few-shot classification accuracies (%) with 95% confidence intervals clip encoder. | 123 |
| A.9 | Accuracy with a varying number of tasks and samples (ViT-B32 backbone). | 124 |

| | | |
|------|---|-----|
| A.10 | Few-shot effect on ViT-B32 backbone on miniImageNet. | 125 |
| A.11 | Accuracy with a varying number of tasks and samples (ViT-B32 backbone). | 125 |
| A.12 | Results of few-shot image classification. We report average classification accuracy (%) with 95% confidence intervals on test splits. Adaptation: Direction adaptation without finetuning; Standard FT: Standard finetuning; MAML: MAML algorithm in Finn et al. (2017); Ours: Our multitask finetuning; 1-/5-shot: number of labeled images per class in the target task. | 127 |
| A.13 | Results of few-shot learning with NLP benchmarks. All results are obtained using RoBERTa-large. We report mean (and standard deviation) of metrics over 5 different splits. †: Result in Gao et al. (2021a); FT: finetuning; task selection: select multitask data from customized datasets. | 129 |
| A.14 | Manual templates and label words that we used in our experiments, following Gao et al. (2021a). | 131 |
| A.15 | Dataset selection. | 133 |
| A.16 | Results of few-shot learning with NLP benchmarks. All results are obtained using RoBERTa-large. We report the mean (and standard deviation) of metrics over 5 different splits. †: Result in Gao et al. (2021a) in our paper; FT: finetuning; task selection: select multitask data from customized datasets. | 134 |
| A.17 | Our main results using simCSE (Gao et al., 2021b). We report mean (and standard deviation) performance over 5 splits of few-shot examples. FT: fine-tuning; task selection: select multitask data from customized dataset. | 135 |
| A.18 | Multitask finetune on zero-shot performance with CLIP model. | 137 |
| A.19 | Multitask finetune on zero-shot performance with ViT-B32 backbone on tieredImageNet. | 138 |
| A.20 | Multitask finetune on zero-shot performance with ViT-B32 backbone on tieredImageNet. | 139 |
| B.1 | Examples of the four logical composite tasks. Note that in (G) + (H), the output of the composite task can be either 4 or 11 depending on the order of operations, and we denote both as correct. | 141 |

| | | |
|-----|---|-----|
| B.2 | Results evaluating composite tasks on Llama3. The accuracy is shown in %. | 142 |
| B.3 | Demonstration in An et al. (2023a) showing examples with the original grammar and the new chain-structured grammar. | 144 |
| B.4 | Testing examples of Passive to Active and Object to Subject, red text shows the verbs changing from passive to active voice in simple tasks, and blue text shows the nouns from objective to subjective. | 145 |
| B.5 | Testing examples of Phrase Recombination and Longer Chain, red text shows the phrase serving as primitives in sentences in simple tasks, and blue text shows the logical structures as sub-sentences in long sentences. | 146 |
| C.1 | Results on broader benchmarks. | 165 |
| C.2 | Efficiency metrics of MLLMs. Budget (%): input latency requirement. AdaLLaVA-L: switches on selecting different transformer blocks. AdaLLaVA-H: switches on selecting different attention heads and MLP activations. | 171 |
| C.3 | VQA benchmark results. VQA ^{v2} : VQAv2 set. SQA ^I : ScienceQA set. VQA ^T : TextVQA set. | 172 |
| C.4 | Hallucination and general benchmark results. POPE evaluates object hallucination. MME and MMBench assess general multimodal understanding. | 173 |
| C.5 | AdaLLaVA-L on LLaVA 1.5-7B model (Example 1), generating appropriate responses while adapting to varying computational budgets. | 174 |
| C.6 | AdaLLaVA-L on LLaVA 1.5-7B model (Example 2), continued from Table C.5. | 175 |
| C.7 | AdaLLaVA-L on LLaVA 1.5-7B model (Example 3), continued from Table C.6. | 176 |

LIST OF FIGURES

| | | |
|-----|--|----|
| 3.1 | Illustration of features in linear data. Blue are the features encoded in \mathcal{C} while red is not. | 22 |
| 3.2 | Illustration of the similarity and coverage. Target tasks (\mathcal{T}_0) with the most similar tasks in yellow and the rest in blue. The ellipsoid spanned by yellow tasks is the coverage for the target task. Adding more tasks in blue to the ellipsoid does not increase the coverage boundary. | 24 |
| 3.3 | Results on ViT-B backbone pretrained by MoCo v3. (a) Accuracy v.s. number of shots per finetuning task. Different curves correspond to different total numbers of samples Mm . (b) Accuracy v.s. the number of tasks M . Different curves correspond to different numbers of samples per task m . (c) Accuracy v.s. number of samples per task m . Different curves correspond to different numbers of tasks M | 27 |
| 4.1 | Inconsistent performance in GPT-4. Consider two simple tasks: If a word is followed by an asterisk (*), capitalize the letter. If two words are surrounded by parentheses, swap the positions. GPT-4 correctly solves two simple tasks based on demonstrations (left). The composite tasks have test inputs with both asterisk (*) and parenthesis. The correct answer should be <i>output: SPORTS PIE</i> . However, GPT-4 fails to solve the composite tasks (right). The same failure was observed in Claude 3. | 33 |
| 4.2 | The exact match accuracy (y-axis) vs the model scale (x-axis, “b” stands for billion) for Capitalization & Swap tasks (example in Figure 4.1). Line <i>capital</i> : performance on the simple task of capitalization; <i>swap</i> : on the simple task of swap; <i>composite</i> : in-context examples are from simple tasks while test input from the composite task. <i>composite incontext</i> : in-context examples and test input are all from the composite task (example in Table 4.1). | 36 |

| | | |
|-----|--|----|
| 4.3 | The word error rate (WER) vs the model scale on composite linguistic translation tasks. Dashed lines: simple tasks. Solid lines: composite tasks. Rows: (T1) Phrase Recombination with Longer Chain; (T2) Passive to Active and Object to Subject Transformation. Columns: different models. Lines: performance in different evaluation settings, e.g., the two simple tasks, the composite setting, and the composite in-context setting (examples are shown in Section B.2). | 42 |
| 5.1 | Top: AdaLLaVA empowers a base LLaVA model with the ability to adapt to varying compute budgets at inference time while maintaining minimal performance loss. Bottom: Given an image, a text query and a latency budget, AdaLLaVA learns to reconfigure operations within a base MLLM, generating appropriate responses while sticking to the budget. | 51 |
| 5.2 | Overview of AdaLLaVA . (a) Model architecture: Our latency encoder embeds an input latency budget into a latency token, which is further processed by the early part of the LLM. The resulting embedding is then fed into the scheduler, leading to the output of an execution plan that controls individual operations in the remaining part of the LLM. Our latency encoder and scheduler are jointly trained with the MLLM. (b) AdaLLaVA-L: the scheduler controls the execution of entire Transformer blocks. (c) AdaLLaVA-H: the scheduler controls the execution of attention heads and MLP neurons, by masking out their activation values and the corresponding weights. | 55 |
| 5.3 | Accuracy-latency tradoffs of AdaLLaVA with LLaVA-1.5-7B and additional token selection techniques (PruMerge+ / FastV). Results reported on VQAv2. | 66 |
| 5.4 | Visualization of attention between the input latency token and visual tokens with a 100% latency budget. | 67 |

| | | |
|-----|---|-----|
| 5.5 | Visualization of execution plans for different input. The plan is color-coded with enable or disable for the 16th to 32th Transformer blocks (left to right). The latency budget is 75%. | 68 |
| A.1 | Dataset selection based on consistency and diversity on domainNet. Figure A.1a shows the consistency. Figure A.1b shows the diversity. | 122 |
| A.2 | Finetuning with different selection of domain datasets, where <i>rp</i> : <i>real</i> and <i>painting</i> ; <i>rps</i> : <i>real</i> and <i>painting</i> and <i>sketch</i> and so on. | 123 |
| A.3 | Finetuning using tieredImageNet train-split, test on test-split. | 126 |
| A.4 | Linear similarity among features vectors among 14 language datasets. | 132 |
| B.1 | The accuracy v.s. model scale on composite logical rule tasks. Dashed lines: simple tasks. Solid lines: composite tasks. Rows: (A) + (C) Capitalization & Two Sum; (G) + (H) Modular & Two Sum Plus; (A) + (F) Capitalization & Plus One. Columns: different models. Lines: performance in different evaluation settings, i.e., the two simple tasks, the composite setting, and the composite in-context setting (examples for the last two are shown in Table 4.1). | 143 |
| C.1 | Training logs | 165 |
| C.2 | Relative performance of applying AdaLLaVA-L to Mipha-3B under various latency budget. The center of the radar corresponds to 60% performance of the base Mipha-3B. | 167 |
| C.3 | Ablation studies on switch design choices. | 168 |
| C.4 | Comparison to random sampling | 168 |
| C.5 | Visualization of execution plan | 169 |
| C.8 | Evolution of latency token across layers in AdaLLaVA-L on 7b model. | 170 |
| C.6 | The key-query attention scores between latency token and visual tokens. The latency input is 1.0 in these examples. | 177 |
| C.7 | The key-query attention scores between latency token and visual tokens. The latency input is 1.0 in these examples. | 178 |

ABSTRACT

Foundation models, trained on broad data at scale and adaptable to diverse downstream tasks, have transformed artificial intelligence. Yet a persistent *specialization gap* remains between these general-purpose models and the level of performance demanded of domain experts. This gap manifests through three interconnected barriers: a *data-efficiency barrier* when tasks require fine-grained domain knowledge under limited supervision, a *compositional reasoning barrier* when tasks necessitate multi-step reasoning that interleaves several operations, and an *adaptivity barrier* when deployment environments impose varying computational constraints.

This dissertation addresses these barriers through three key contributions that combine theoretical analysis, empirical validation, and practical algorithms. *First*, I develop a theoretical framework for multitask finetuning, revealing that diverse and consistent intermediate tasks significantly reduce target task error compared to direct adaptation. I introduce novel metrics to quantify task relationships and derive a practical task selection algorithm. *Second*, I investigate compositional abilities in large language models (LLMs) through systematic empirical and theoretical studies. I find that models succeed at composite tasks applying distinct operations to separate input segments, but struggle with multi-step reasoning requiring sequential operations—a distinction my theoretical analysis explains. *Third*, I develop an adaptive inference framework for multimodal LLMs, grounded in the insight that such models can be viewed as collections of shallower sub-models. Through a learned scheduler, my approach dynamically reconfigures computation based on input content and resource constraints, maintaining strong performance while adapting to fluctuating computing budgets.

Together, these contributions advance both the theoretical understanding and practical deployment of foundation models, providing principled methods for bridging the specialization gap and enabling the transformation of general-purpose models into capable, efficient, and deployable task-specific systems.

1 INTRODUCTION

The emergence of foundation models (Bommasani et al., 2021) has fundamentally transformed the landscape of artificial intelligence (AI), enabling unprecedented capabilities across diverse domains. Foundation models are trained on broad data at scale and can be adapted to a wide range of downstream tasks, as exemplified by large language models (LLMs) (e.g. BERT (Devlin et al., 2019), Llama (Touvron et al., 2023a,b), GPTs (Brown et al., 2020; OpenAI, 2023)), vision models (e.g. CLIP (Radford et al., 2021) and DINOv2 (Oquab et al., 2024)), and multimodal large language models (MLLMs) (e.g. GPT-4V(ision) (OpenAI, 2022a), Claude (Anthropic, 2024, 2025), Llama-3 (Meta, 2024), LLaVA (Liu et al., 2023)). These models have demonstrated remarkable abilities across multiple modalities and tasks—from basic classification and recognition to complex understanding and reasoning, leading to some of the most exciting developments in AI to date.

A central promise of foundation models is their adaptability: a single pretrained model can generalize to several downstream tasks. For tasks that align well with the pretraining distribution, foundation models can perform effectively through zero-shot inference alone without modification. However, when target tasks deviate from the pretraining regime, such as data domain, task structure, or reasoning patterns, additional adaptation becomes necessary. Practitioners have developed several techniques for such cases. *Finetuning* updates model parameters on task-specific data. *In-context learning* (ICL) enables models to perform new tasks by conditioning on a few demonstrations, with no parameter updates. These approaches have extended foundation models’ reach beyond their zero-shot capabilities, enabling strong performance on a broader range of applications.

Despite these successes, a persistent *specialization gap* remains between general-purpose foundation models and the level of performance and robustness required of domain experts. Models that perform well on generic benchmarks often degrade substantially on tasks that deviate from pretraining. This degradation manifests through three key barriers.

First, a **data-efficiency barrier** arises when tasks require fine-grained domain

knowledge under limited supervision. Pretraining endows models with broad representations, but these do not readily transfer to specialized domains without sufficient labeled data. For example, a vision model trained on web images struggles to recognize rare medical conditions from limited case studies or classify newly discovered species from few reference images.

Second, a **compositional reasoning barrier** emerges when tasks demand multi-step reasoning that interleaves several operations. Models can often perform individual tasks in isolation, yet fail when asked to compose them, such as combining question answering with translation, retrieval, and numerical computation in a single query. The challenge is that existing models often struggle to recognize compositional structure and correctly sequence operations when only shown simple demonstrations of individual components.

Third, the computational demands of foundation models pose a distinct **adaptivity barrier** for practical deployment. Foundation models operate with fixed computational costs regardless of input difficulty or available resources. Yet real-world deployment environments are inherently variable: compute resources fluctuate due to contention on shared infrastructure, edge devices impose strict memory limitations, and interactive applications require responsiveness under changing system loads. Current models cannot dynamically reconfigure their computation in response to these varying constraints, limiting their practical utility even when their raw capabilities suffice for the task at hand.

These challenges share a common characteristic: they require capabilities that lie outside the model's pretraining regime, whether in terms of data distribution, task structure, or inference cost. Standard adaptation techniques often prove insufficient because they do not address the underlying causes of these failures. Closing this gap is critical for realizing the full potential of foundation models.

To this end, my research addresses the fundamental question: *How can we bridge this specialization gap through adaptive learning techniques that overcome these barriers simultaneously?*

My central intuition is that bridging the specialization gap requires reshaping how models acquire, organize, and deploy their knowledge and computation, so

that task structure, supervision, and resource constraints are reflected in both training and inference. This perspective motivates the methods developed in this dissertation.

Thesis Statement. In this dissertation, I assert the following statement:

Foundation models can be effectively specialized for downstream tasks through data-centric multitask adaptation, understanding of compositional reasoning mechanisms, and dynamic inference strategies.

Thesis Overview. My dissertation makes three targeted contributions: data-centric adaptation techniques for improving generalization, systematic analysis of compositional reasoning mechanisms, and adaptive frameworks for resource-flexible deployment. In doing so, my thesis work combines theoretical analysis that reveals the principles underlying successful adaptation and composition, empirical studies that validate these principles across diverse benchmarks, and practical algorithms that translate theoretical insights into deployable solutions.

Data-efficient adaptation. First, I tackle the challenge of adapting foundation models to new tasks with limited labeled data. While these models excel at many tasks, their effective adaptation, especially in few-shot scenarios, remains both a practical challenge and a theoretical mystery. I present a theoretical framework for analyzing multitask finetuning, revealing that with a diverse set of relevant tasks, this approach can significantly reduce error in target tasks compared to direct adaptation. My analysis quantifies the relationship between finetuning tasks and target tasks through novel diversity and consistency metrics, leading to practical algorithms for task selection that substantially improve adaptation performance.

Understanding Compositional Ability. Second, I delve into understanding the cognitive capabilities of large language models, specifically focusing on their ability to handle composite tasks. While these models demonstrate remarkable in-context learning capabilities, their approach to solving unseen complex tasks that combine

multiple simple tasks remains poorly understood. Through systematic empirical studies, I uncover that models exhibit divergent behaviors: they show promising performance on simpler composite tasks that apply distinct operations to different input segments but struggle with multi-step reasoning tasks. My theoretical analysis in a simplified setting provides insights into why models succeed in certain compositional scenarios while failing in others, contributing to our fundamental understanding of these systems.

Adaptive Inference. Third, I address the challenge of deploying foundation models under varying computational constraints. While multimodal LLMs achieve strong performance across diverse tasks, their fixed computational cost makes practical deployment difficult when memory and compute budgets fluctuate dynamically—as is common on edge devices, mobile platforms, or servers handling heterogeneous requests. I develop a framework grounded in the insight that a multimodal LLM can be conceptualized as a collection of shallower models, enabling dynamic re-configuration during inference. Through a learned scheduler and probabilistic modeling approach, I demonstrate that models can maintain comparable performance to full inference while adapting computation to input content and runtime resource constraints.

Other contributions not included in this thesis. Beyond these primary contributions, I have also advanced the understanding of foundation models through several complementary investigations through collaborations. We investigated how model scale affects in-context learning behavior (Shi et al., 2024b). We extend our compositional reasoning analysis, where we examined in-context skill composition and found that models struggle to recognize and assemble skills from simple demonstrations even with Chain-of-Thought (Liu et al., 2025b). On the architectural understanding, we studied the role of induction heads in out-of-distribution generalization (Song et al., 2025). Connecting to our efficiency goals, we developed Conv-Basis, a convolution-based attention approximation achieving nearly linear time complexity (Liang et al., 2025). These investigations collectively enhance our understanding of model behavior, capabilities, and deployment considerations.

Thesis Organization. The remainder of this dissertation is organized as follows: Chapter 2 provides necessary background and related work. Chapter 3 presents my theoretical analysis of multitask finetuning and its empirical validation. Chapter 4 explores the compositional abilities of large language models through both empirical studies and theoretical analysis. Chapter 5 presents my adaptive inference framework for multimodal LLMs. Chapter 6 concludes the thesis by summarizing key findings, discussing additional collaborative work, and outlining 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 BACKGROUND AND RELATED WORK

Foundation models are models trained on broad data at scale that can be adapted to a wide range of downstream tasks (Bommasani et al., 2021). Prominent examples including large language models (LLMs) (e.g. BERT (Devlin et al., 2019), Llama (Touvron et al., 2023a,b), GPTs (Brown et al., 2020; OpenAI, 2023)), vision models (e.g., CLIP (Radford et al., 2021) and DINOv2 (Oquab et al., 2024)), and multimodal large language models (MLLMs) (e.g., GPT-4V(ision) (OpenAI, 2022a), Claude (Anthropic, 2024, 2025), Llama-3 (Meta, 2024), LLaVA (Liu et al., 2023)). In this chapter, I review the key concepts and prior work on foundation models that contextualize and position the contributions of this dissertation. I first introduce the the background knowledge on foundation model, including definition and common forms of foundation models. Then I organize the related work into three main areas following the structure of my three core challenges.

Adaptation of Foundation Models. Section 2.1 covers the landscape of foundation model adaptation, including pretraining methods, standard adaptation techniques, multitask learning, and few-shot learning. This background contextualizes my theoretical analysis and empirical investigations of multitask finetuning in Chapter 3.

Compositional Ability of LLMs. Section 2.2 reviews research on large language models and their compositional abilities, with particular attention to in-context learning and emergent capabilities. This literature motivates my investigation of compositional reasoning in Chapter 4.

Efficient Inference of Multimodal LLMs (MLLMs). Section 2.3 surveys multimodal large language models and approaches to efficient inference, including adaptive computation and token reduction techniques. This background informs my development of adaptive inference framework in Chapter 5.

For each area, I highlight both the advances that enable my work and the gaps that my contributions address. Detailed technical background specific to each contribution is provided in the corresponding chapters.

2.1 Adaptation of Foundation Models

Training Foundation Models. Foundation models (Bommasani et al., 2021) are typically trained using self-supervised learning over broad data. The most commonly used training approaches include *contrastive learning* in vision and *masked modeling* in NLP. My theoretical analysis considers both approaches under a unified framework. Here I briefly review these approaches.

- *Contrastive learning*, in a self-supervised setting, aims to group randomly augmented versions of the same data point while distinguishing samples from diverse groups. The success of this approach in vision and multi-modal training tasks (Oord et al., 2018; Chen et al., 2020; He et al., 2020; Tian et al., 2020a; Grill et al., 2020; Radford et al., 2021) has spurred considerable interest. Several recent studies (Arora et al., 2019; HaoChen et al., 2021; Tosh et al., 2021; Zimmermann et al., 2021; Wei et al., 2021; Wang and Isola, 2020; Wen and Li, 2021; Wang et al., 2022; Shi et al., 2023a; Huang et al., 2023; Sun et al., 2023b,a) seek to develop its theoretical understanding. Arora et al. (2019) established theoretical guarantees on downstream classification performance. HaoChen et al. (2021) provided analysis on spectral contrastive loss. Their analysis assumes the pretraining and target tasks share the same data distribution and focus on the effect of contrastive learning on direct adaptation. My work (details in Chapter 3) focuses on the novel class setting and investigates further finetuning the pretrained model with multitask to improve performance.
- *Masked modeling* seeks to predict masked tokens in an input sequence. This self-supervised approach is the foundation of many large language models (Devlin et al., 2019; Liu et al., 2019; Chowdhery et al., 2022; Ni et al., 2022; Touvron et al., 2023a), and has been recently explored in vision (He et al., 2022). In the theoretical frontier, Zhao et al. (2023) formulated masked language modeling as standard supervised learning with labels from the input text. They further investigated the relationship between pretrained data and testing data by

diversity statement. My work (details in Chapter 3) subsumes their work as a special case, and can explain a broader family of pretraining methods.

Adapting Foundation Models. Adapting foundation models to downstream tasks has recently received significant attention. The conventional wisdom, mostly adopted in vision (Vinyals et al., 2016; Ge and Yu, 2017; Chen et al., 2020; He et al., 2020, 2022; Shi et al., 2023b), involves learning a simple function, such as linear probing, on the representation from a foundation model, while keeping the model frozen or minorly finetuning the whole model. In NLP, prompt-based finetuning (Gao et al., 2021a; Chung et al., 2024; Song et al., 2022; Zhou et al., 2022b; Xie et al., 2023; Zhang et al., 2023a) was developed and widely used, in which a prediction task is transformed into a masked language modeling problem during finetuning. With the advances in large language models, parameter-efficient tuning has emerged as an attractive solution. Prompt tuning (Lester et al., 2021; Li and Liang, 2021; Roberts et al., 2023) learns an extra prompt token for a new task, while updating minimal or no parameters in the model backbone. Another promising approach is in-context learning (Min et al., 2022b; Wei et al., 2022a,b; Shi et al., 2023d; Xu et al., 2024b), where the model is tasked to make predictions based on contexts supplemented with a few examples, with no parameter updates. In my work, I consider adapting foundation models to new tasks with limited labels. Parameter-efficient tuning, such as in-context learning, might face major challenges (Xie et al., 2022) when the distribution of the new task deviates from those considered in pretraining. Instead, my approach finetunes the model using multiple relevant tasks. I empirically verify that doing so leads to better adaptation.

Multitask Learning. Multitask supervised learning has been considered for transfer learning to a target task (Zhong et al., 2021; Sanh et al., 2022; Chen et al., 2022; Min et al., 2022a; Wang et al., 2023b). Multitask has been shown to induce zero-shot generalization in large language models (Sanh et al., 2022), and also enable parameter efficient tuning by prompt tuning (Wang et al., 2023b). My prior work (details in Chapter 3) leverages multitask learning to unlock better zero-shot and

few-shot performance of pretrained models. Min et al. (2022a); Chen et al. (2022) primarily focus on in-context learning, Zhong et al. (2021) focuses on the idea of task conversion where transfer classification task as question-answer format, my approach is based on utilizing original examples, in alignment with my theoretical framework. A line of theoretical work provides the error bound of the target task in terms of sample complexity (Du et al., 2021; Tripuraneni et al., 2021; Shi et al., 2023a; Xu et al., 2023). Tripuraneni et al. (2020) established a framework of multitask learning centered around the notion of task diversity for the training data. Their work mainly analyzed representations from supervised pretraining using multitasks. In contrast, my prior work (details in Chapter 3) considers representations from self-supervised pretraining, and focuses on multitask finetuning. My approach and analysis guarantee that limited but diverse and consistent finetuning task can improve the prediction performance on a target task with novel classes.

Few-shot Learning and Meta Learning. Few-shot learning necessitates the generalization to new tasks with only a few labeled samples (Wang et al., 2020; Vu et al., 2021; Murty et al., 2021; Liu et al., 2021; Yang et al., 2022; Galanti et al., 2022). Direct training with limited data is prone to overfitting. Meta learning offers a promising solution that allows the model to adapt to the few-shot setting (Finn et al., 2017; Raghu et al., 2020). This solution has been previously developed for vision tasks (Vinyals et al., 2016; Snell et al., 2017; Chen et al., 2021b; Hu et al., 2022b). Inspired by meta learning in the few-shot setting, my analysis extends the idea of multitask finetuning by providing sound theoretic justifications and demonstrating strong empirical results. I further introduce a task selection algorithm that bridges my theoretical findings with practical applications in multitask finetuning.

2.2 Compositional Ability of LLMs

Large Language Model. LLMs are often Transformer-based (Vaswani et al., 2017) equipped with the enormous size of parameters and pretrained on vast training data. Typical LLMs includes BERT (Devlin et al., 2019), PaLM (Chowdhery et al.,

2022), LLaMA (Touvron et al., 2023a), ChatGPT (OpenAI, 2022b), GPT4 (OpenAI, 2023). Pretraining methods include masked language modeling (Devlin et al., 2019; Liu et al., 2019), contrastive learning (Gao et al., 2021b; Shi et al., 2023a; Sun et al., 2023b,a) and auto-regressive pretraining (Radford et al., 2018, 2019). Several works (Madasu and Srivastava, 2022; Alajrami et al., 2023) investigate the effects of pretraining on language models. Adapting LLMs to various downstream tasks has received significant attention, e.g., adaptor (Hu et al., 2022a, 2023; Zhang et al., 2023a; Luo et al., 2024), prompt tuning (Lester et al., 2021; Li and Liang, 2021; Wei et al., 2023a; Gu et al., 2024c), multitask finetuning (Sanh et al., 2022; Wang et al., 2023b; Xu et al., 2023, 2024c), instruction tuning (Chung et al., 2024; Mishra et al., 2022), in-context learning (Min et al., 2022b; Dong et al., 2024; Yao et al., 2023), low-rank adaptation (Zeng and Lee, 2024; Hu et al., 2024), reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022) and inference acceleration (Gu et al., 2024b,d; Xu et al., 2024a).

In-context Learning. LLM exhibits a remarkable ability for in-context learning (ICL) (Brown et al., 2020), particularly for generative models. Given a sequence of labeled examples and a testing example (combined as a prompt), the model can construct new predictors for testing examples without further parameter updates. Several empirical studies investigate the behavior of ICLs. Zhao et al. (2021); Holtzman et al. (2021); Lu et al. (2022b) formulate the problems and report the sensitivity. Rubin et al. (2022); Liu et al. (2022); Hongjin et al. (2023); Wang et al. (2023a) provide methods to better choose in-context learning examples. Chen et al. (2021b); Min et al. (2022a) use meta training with an explicit in-context learning object to boost performance. Theoretically, Xie et al. (2022); Garg et al. (2022) provide a framework to explain the working mechanism of in-context learning. Von Oswald et al. (2023); Akyürek et al. (2023); Mahankali et al. (2023); Zhang et al. (2023b), investigating with linear models, show how transformers can represent gradient descent and conduct linear regression. Based on these works, I provide an analysis showing how LLM can exhibit compositional ability in ICL.

Emergence of Compositional Ability. Scaling law was first proposed by Kaplan et al. (2020) and then followed up by Hoffmann et al. (2022), emphasizing both the scale of models and training data. Sometimes, increasing scale can lead to new behaviors of LLMs, termed *emergent abilities* (Wei et al., 2022a; Arora and Goyal, 2023), such as domain generalization (Shi et al., 2024a), math reasoning (Gu et al., 2024a), spatial reasoning (Wang et al., 2024a) and so on. Recent works show LLMs with larger scales have distinct behavior compared to smaller language models (Wei et al., 2023b; Shi et al., 2023d, 2024b). These behaviors can have positive or negative effects on performance. Solving complex tasks and reasoning is an active problem in the AI community (Huang and Chang, 2022). There is a line of empirical works investigating the compositional ability in linguistic fashion (Kim and Linzen, 2020; Levy et al., 2022; An et al., 2023a,b; Xu et al., 2024b). LLMs are capable of learning abstract reasoning (e.g., grammar) to perform new tasks when finetuned or given suitable in-context examples. In my work (details in Chapter 4), I include linguistic experiments as part of my testing suite, illustrating LLMs’ compositional ability. Ye et al. (2023); Berglund et al. (2023); Dziri et al. (2023) show LLMs will have difficulties solving tasks that require reasoning. Berglund et al. (2023) studies that LLMs trained on “A is B” fail to learn “B is A”. In my work (details in Chapter 4), I conduct similar experiments showing LLMs will fail on composite if different steps of logical rules are mixed.

2.3 Adaptive Inference of multimodal LLMs (MLLMs)

Multimodal Large Language Models. With the success of LLMs, increasing research focus on extends LLMs from pure text modality to other modalities such as image (Liu et al., 2023), video (Li et al., 2024), and audio (Latif et al., 2023). Such development leads to the emergence of MLLMs, often involving combine vision encoders with existing LLMs. Flamingo (Alayrac et al., 2022) inserts gated cross-attention dense blocks between vision encoder and LLMs, align vision and

language modality. BLIP2 (Li et al., 2023b) introduce Q-former with two-stage pretraining, bridge frozen image encoders and LLMs to enable visual instruction capability. LLaVA (Liu et al., 2023, 2024a) and MiniGPT-4 (Zhu et al., 2024a) use simple MLP to connect vision embedding space and text token space and show state-of-art performance on a variety of tasks. My adaptive inference work (details in Chapter 5) builds on these developments and aims to enable adaptive inference of MLLMs.

Adaptive Inference. Adaptive inference refers to the capability in which the computational complexity of making predictions is dynamically adjusted based on the input data, latency budget, or desired accuracy levels (Han et al., 2021). Early works focus on the selection of hand-crafted features in multi-stage prediction pipelines (Karayev et al., 2014; Xu et al., 2012; Grubb and Bagnell, 2012). More recent works have extended these ideas to deep models. For convolutional networks, methods have been developed to downsample the input, skip layers or exist early during inference (Figurnov et al., 2017; Li et al., 2021; Wang et al., 2018b; Bengio et al., 2015; Wu et al., 2018; Hu et al., 2019; Jie et al., 2019; Meng et al., 2020). For vision transformers, various approaches have been proposed to enhance efficiency, such as selecting different patches of images (Wang et al., 2021; Rao et al., 2021; Pan et al., 2021), and using different attention heads and blocks (Meng et al., 2022). Similar ideas have also been explored for LLMs, where models selectively process tokens (Raposo et al., 2024) or execute a subset of the operations (Du et al., 2022; Rotem et al., 2023) during inference.

Efficient Inference for MLLMs. MLLMs face a major challenge in deployment, due to their high computational costs during inference. Several recent works design lightweight model architectures to reduce the costs. Examples include Phi-2 (Javaheripi et al., 2023), Tinygpt-v (Yuan et al., 2023) and LLaVA- ϕ (Zhu et al., 2024c).

Vary-toy (Wei et al., 2024) enhanced performance through specialized vision vocabulary in smaller models. TinyLLaVA (Zhou et al., 2024) and LLaVA-OneVision (Li et al., 2024) learn small-scale models with better training data and pipeline. MoE-

LLaVA (Lin et al., 2024) and LLaVA-MoD (Shu et al., 2024) improve efficiency by incorporating mixture-of-experts architectures and parameter sparsity techniques. Another line of research investigates the selection of input tokens to improving efficiency. An input image or video can lead to a large number of vision tokens. To address this, MADTP (Cao et al., 2024) and LLaVA-PruMerge (Shang et al., 2024) introduce token pruning and merging technique to reduce the tokens counts. Pham et al. (Pham et al., 2024) propose to selectively disabling attention mechanisms for visual tokens in MLLMs.

While my adaptive inference work (details in Chapter 5) also aims to improve the efficiency of MLLMs, it focuses dynamically adjusting an MLLM to fit varying latency budget during inference. This makes my approach orthogonal to prior efforts centered on developing inherently efficient MLLMs. Through experiments, I will demonstrate that my approach is compatible with smaller models and integrates seamlessly with existing token-pruning techniques *e.g.*, LLaVA-PruMerge (Shang et al., 2024).

3 FEW-SHOT ADAPTATION VIA MULTITASK FINETUNING

Contribution statement. This chapter is joint work with Zhenmei Shi, Junyi Wei, Fangzhou Mu, Yin Li and Yingyu Liang. I proposed the method, contributed to part of the theoretical analysis, and completed all the experiments. This chapter was published in ICLR 2024 (Xu et al., 2024c).

3.1 Introduction

In this chapter, we focus on the problem of adapting a pretrained foundation model to a new task with a few labeled samples, where the target task can differ significantly from pretraining and the limited labeled data are insufficient for finetuning. This few-shot learning problem has been a long-standing challenge in machine learning (Wang et al., 2020). Prior approaches include learning from examples in the context prompt (in-context learning) (Brown et al., 2020), constructing simple classifiers based on the pretrained representation (Zhang et al., 2020), or finetuning the model using text prompts converted from labeled data (Gao et al., 2021a). An emerging solution involves finetuning a pretrained model on multiple auxiliary tasks pertaining to the target task. This multitask finetuning approach, related to meta learning (Hospedales et al., 2021), has been recently explored in NLP and vision (Murty et al., 2021; Vu et al., 2021; Zhong et al., 2021; Hu et al., 2022b; Chen et al., 2022; Min et al., 2022a). For example, latest studies (Sanh et al., 2022; Muennighoff et al., 2023) show that finetuning language models on a large set of tasks enables strong zero-shot generalization on unseen tasks. Nonetheless, the lack of sound theoretical explanations behind these previous approaches raises doubts about their ability to generalize on real-world tasks (Perez et al., 2021).

To bridge the gap, we study the theoretical justification of multitask finetuning. We consider an intermediate step that finetunes a pretrained model with a set of relevant tasks before adapting to a target task. Each of these auxiliary tasks might have a small number of labeled samples, and categories of these samples might not

overlap with those on the target task. Our key intuition is that a sufficiently diverse set of relevant tasks can capture similar latent characteristics as the target task, thereby producing meaningful representation and reducing errors in the target task. To this end, we present rigorous theoretical analyses, provide key insight into conditions necessary for successful multitask finetuning, and introduce a novel algorithm for selecting tasks suitable for finetuning.

Our key contributions are three folds. *Theoretically*, we present a framework for analyzing pretraining followed by multitask finetuning. Our analysis (Section 3.3) reveals that with limited labeled data from diverse tasks, finetuning can improve the prediction performance on a downstream task. *Empirically*, we perform extensive experiments on both vision and language tasks (Section 3.4) to verify our theorem. Our results suggest that our theorem successfully predicts the behavior of multitask finetuning across datasets and models. *Practically*, inspired by our theorem, we design a *task selection algorithm* for multitask finetuning. On the Meta-Dataset (Triantafillou et al., 2020), our algorithm shows significantly improved results in comparison to finetuning using all possible tasks.

3.2 Preliminaries: Multitask Finetuning for Few-Shot Learning

This section reviews the pretraining of foundation models and adaptation for few-shot learning, and then formalizes the multitask finetuning approach.

Pretraining Foundation Models. We consider three common pretraining methods: contrastive learning, masked language modeling, and supervised pretraining. *Contrastive learning* is widely considered in vision and multi-modal tasks. This approach pretrains a model ϕ from a hypothesis class Φ of foundation models via loss on contrastive pairs generated from data points x . First sample a point x and then apply some transformation to obtain x^+ ; independently sample another point x^- . The population contrastive loss is then $\mathcal{L}_{\text{con-pre}}(\phi) := \mathbb{E} [\ell_u(\phi(x)^\top (\phi(x^+) - \phi(x^-)))]$,

where the loss function ℓ_u is a non-negative decreasing function. In particular, logistic loss $\ell_u(v) = \log(1 + \exp(-v))$ recovers the typical contrastive loss in most empirical work (Logeswaran and Lee, 2018; Oord et al., 2018; Chen et al., 2020). *Masked language modeling* is a popular self-supervised learning approach in NLP. It can be regarded as a kind of *supervised pretraining*: the masked word is viewed as the class (see Section A.2 for more details). In what follows we provide a unified formulation. On top of the representation function ϕ , there is a linear function $f \in \mathcal{F} \subset \{\mathbb{R}^d \rightarrow \mathbb{R}^K\}$ predicting the labels where K is the number of classes. The supervised loss is: $\mathcal{L}_{\text{sup-pre}}(\phi) := \min_{f \in \mathcal{F}} \mathbb{E}[\ell(f \circ \phi(x), y)]$, where $\ell(\cdot, y)$ is the cross-entropy loss. To simplify the notation, we unify $\mathcal{L}_{\text{pre}}(\phi)$ as the pretraining loss.

Adapting Models for Few-shot Learning. A pretrained foundation model ϕ can be used for downstream target tasks \mathcal{T} by learning linear classifiers on ϕ . We focus on binary classification (the general multiclass setting is in Section A.3). A linear classifier on ϕ is given by $\mathbf{w}^\top \phi(x)$ where $\mathbf{w} \in \mathbb{R}^d$. The supervised loss of ϕ w.r.t the task \mathcal{T} is then:

$$\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi) := \min_{\mathbf{w}} \mathbb{E}_{(x,y) \sim \mathcal{D}_{\mathcal{T}}} [\ell(\mathbf{w}^\top \phi(x), y)], \quad (3.1)$$

where $\mathcal{D}_{\mathcal{T}}(x, y)$ is the distribution of data (x, y) in task \mathcal{T} . In few-shot learning with novel classes, there are *limited labeled data points* for learning the linear classifier. Further, the target task \mathcal{T}_0 may contain *classes different from those in pretraining*. We are interested in obtaining a model ϕ such that $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi)$ is small.

Multitask Finetuning. In the challenging setting of few-shot learning, the data in the target task is limited. On the other hand, we can have prior knowledge of the target task characteristics and its associated data patterns, and thus can collect additional data from relevant and accessible sources when available. Such data may cover the patterns in target task and thus can be used as auxiliary tasks to finetune the pretrained model before adaptation to the target task. Here we

formalize this idea in a general form and provide analysis in later sections. Formally, suppose we have M auxiliary tasks $\{\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_M\}$, each with m labeled samples $\mathcal{S}_i := \{(x_j^i, y_j^i) : j \in [m]\}$. The finetuning data are $\mathcal{S} := \cup_{i \in [M]} \mathcal{S}_i$. Given a pretrained model $\hat{\phi}$, we further finetune it using the objective:

$$\min_{\phi \in \Phi} \frac{1}{M} \sum_{i=1}^M \hat{\mathcal{L}}_{\text{sup}}(\mathcal{T}_i, \phi), \quad (3.2)$$

$$\text{where } \hat{\mathcal{L}}_{\text{sup}}(\mathcal{T}_i, \phi) := \min_{\mathbf{w}_i \in \mathbb{R}^d} \frac{1}{m} \sum_{j=1}^m \ell(\mathbf{w}_i^\top \phi(x_j^i), y_j^i).$$

This can be done via gradient descent from the initialization $\hat{\phi}$ (see Algorithm 2 in the Appendix). Multitask finetuning is conceptually simple, and broadly applicable to different models and datasets. While its effectiveness has been previously demonstrated (Murty et al., 2021; Vu et al., 2021; Zhong et al., 2021; Hu et al., 2022b; Chen et al., 2022; Min et al., 2022a; Sanh et al., 2022; Muennighoff et al., 2023), the theoretical justification remains to be fully investigated and understood.

3.3 Theoretical Analysis: Benefit of Multitask Finetuning

To understand the potential benefit of multitask finetuning, we will compare the performance of $\hat{\phi}$ (from pretraining) and ϕ' (from pretraining and multitask finetuning) on a target task \mathcal{T}_0 . That is, we will compare $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi})$ and $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi')$, where $\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)$ is the population supervised loss of ϕ on the task \mathcal{T} defined in Equation (3.1). For the analysis, we first formalize the data distributions and learning models, then introduce the key notions, and finally present the key theorems.

Data Distributions. Let \mathcal{X} be the input space and $\bar{\mathcal{Y}} \subseteq \mathbb{R}^d$ be the output space of the foundation model. Following Arora et al. (2019), suppose there is a set of latent classes \mathcal{C} with $|\mathcal{C}| = K$, and a distribution η over the classes; each class $y \in \mathcal{C}$

has a distribution $\mathcal{D}(\mathbf{y})$ over inputs \mathbf{x} . In pretraining using contrastive learning, the distribution $\mathcal{D}_{\text{con}}(\eta)$ of the contrastive data $(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-)$ is given by: $(\mathbf{y}, \mathbf{y}^-) \sim \eta^2$ and $\mathbf{x}, \mathbf{x}^+ \sim \mathcal{D}(\mathbf{y}), \mathbf{x}^- \sim \mathcal{D}(\mathbf{y}^-)$. In masked self-supervised or fully supervised pretraining, (\mathbf{x}, \mathbf{y}) is generated by $\mathbf{y} \sim \eta, \mathbf{x} \sim \mathcal{D}(\mathbf{y})$. In a task \mathcal{T} with binary classes $\{\mathbf{y}_1, \mathbf{y}_2\}$, the data distribution $\mathcal{D}_{\mathcal{T}}(\mathbf{x}, \mathbf{y})$ is by first uniformly drawing $\mathbf{y} \in \{\mathbf{y}_1, \mathbf{y}_2\}$ and then drawing $\mathbf{x} \sim \mathcal{D}(\mathbf{y})$. Finally, let ζ denote the conditional distribution of $(\mathbf{y}_1, \mathbf{y}_2) \sim \eta^2$ conditioned on $\mathbf{y}_1 \neq \mathbf{y}_2$, and suppose the tasks in finetuning are from ζ . Note that in few-shot learning with novel classes, the target task's classes may not be the same as those in the pretraining. Let \mathcal{C}_0 be the set of possible classes in the target task, which may or may not overlap with \mathcal{C} .

Learning Models. Recall that Φ is the hypothesis class of foundation models $\phi : \mathcal{X} \rightarrow \bar{\mathcal{Z}}$. To gauge the generalization performance, let $\phi^* \in \Phi$ denote the model with the lowest target task loss $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*)$ and $\phi_{\zeta}^* \in \Phi$ denote the model with the lowest average supervised loss over the set of auxiliary tasks $\mathcal{L}_{\text{sup}}(\phi_{\zeta}^*) := \mathbb{E}_{\mathcal{T} \sim \zeta}[\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi_{\zeta}^*)]$. Note that if all $\phi \in \Phi$ have high supervised losses, we cannot expect the method to lead to a good generalization performance, and thus we need to calibrate w.r.t. ϕ^* and ϕ_{ζ}^* . We also need some typical regularity assumptions.

Assumption 3.3.1 (Regularity Assumptions). $\|\phi\|_2 \leq R$ and linear operator $\|\mathbf{w}\|_2 \leq B$. The loss ℓ_u is bounded in $[0, C]$ and L -Lipschitz. The supervised loss $\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)$ is \tilde{L} -Lipschitz with respect to ϕ .

Diversity and Consistency. Central to our theoretical analysis lies in the definitions of *diversity* in auxiliary tasks used for finetuning and their *consistency* with the target task.

Definition 3.3.2 (Diversity). The averaged representation difference for two model $\phi, \tilde{\phi}$ on a distribution ζ over tasks is $\bar{d}_{\zeta}(\phi, \tilde{\phi}) := \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi) - \mathcal{L}_{\text{sup}}(\mathcal{T}, \tilde{\phi})] = \mathcal{L}_{\text{sup}}(\phi) - \mathcal{L}_{\text{sup}}(\tilde{\phi})$. The worst-case representation difference between representations $\phi, \tilde{\phi}$ on the family of classes \mathcal{C}_0 is $d_{\mathcal{C}_0}(\phi, \tilde{\phi}) := \sup_{\mathcal{T}_0 \subseteq \mathcal{C}_0} |\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \tilde{\phi})|$. We say the

model class Φ has ν -diversity (for ζ and \mathcal{C}_0) with respect to ϕ_ζ^* , if for any $\phi \in \Phi$, $d_{\mathcal{C}_0}(\phi, \phi_\zeta^*) \leq \bar{d}_\zeta(\phi, \phi_\zeta^*)/\nu$.

Such diversity notion has been proposed and used to derive statistical guarantees (e.g., Tripuraneni et al. (2020); Zhao et al. (2023)). Intuitively, diversity measures whether the data from ζ covers the characteristics of the target data in \mathcal{C}_0 , e.g., whether the span of the linear mapping solutions \mathbf{w} 's for tasks from ζ can properly cover the solutions for tasks from \mathcal{C}_0 (Zhao et al., 2023). Existing work showed that diverse pretraining data will lead to a large diversity parameter ν and can improve the generalization in the target task. Our analysis will show the diversity in finetuning tasks from ζ can benefit the performance of a target task from \mathcal{C}_0 .

Definition 3.3.3 (Consistency). *We say the model class Φ has κ -consistency (for ζ and \mathcal{C}_0) with respect to ϕ^* and ϕ_ζ^* , where $\kappa := \sup_{\mathcal{T}_0 \subseteq \mathcal{C}_0} [\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*)]$.*

This consistency notion measures the similarity between the data in tasks from ζ and the data in the target task from \mathcal{C}_0 . Intuitively, when the tasks from ζ are similar to the target task \mathcal{T}_0 , their solutions ϕ_ζ^* and ϕ^* will be similar to each other, resulting in a small κ . Below we will derive guarantees based on the diversity ν and consistency κ to explain the gain from multitask finetuning.

Key Results. We now present the results for a uniform distribution η , and include the full proof and results for general distributions in Section A.2 and Section A.3. Recall that we will compare the performance of $\hat{\phi}$ (the model from pretraining) and ϕ' (the model from pretraining followed by multitask finetuning) on a target task \mathcal{T}_0 . For $\hat{\phi}$ without multitask finetuning, we have:

Theorem 3.3.4. (No Multitask Finetuning) *Assume assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^* and ϕ_ζ^* . Suppose $\hat{\phi}$ satisfies $\hat{\mathcal{L}}_{\text{pre}}(\hat{\phi}) \leq \epsilon_0$. Let $\tau := \Pr_{(y_1, y_2) \sim \eta^2} \{y_1 = y_2\}$. Then for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,*

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\frac{2\epsilon_0}{1-\tau} - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa. \quad (3.3)$$

In Theorem 3.3.4, $\widehat{\mathcal{L}}_{\text{pre}}(\phi)$ is the empirical loss of $\mathcal{L}_{\text{pre}}(\phi)$ with pretraining sample size N . We now consider ϕ' obtained by multitask finetuning. Define the subset of models with pretraining loss smaller than $\tilde{\epsilon}$ as $\Phi(\tilde{\epsilon}) := \left\{ \phi \in \Phi : \widehat{\mathcal{L}}_{\text{pre}}(\phi) \leq \tilde{\epsilon} \right\}$. Recall the Rademacher complexity of Φ on n points is

$$\mathcal{R}_n(\Phi) := \mathbb{E}_{\{\sigma_j\}_{j=1}^n, \{x_j\}_{j=1}^n} \left[\sup_{\phi \in \Phi} \sum_{j=1}^n \sigma_j \phi(x_j) \right].$$

Theorem 3.3.5 below showing that the target prediction performance of the model ϕ' from multitask finetuning can be significantly better than that of $\hat{\phi}$ without multitask finetuning. In particular, achieves an error reduction $\frac{1}{\nu} \left[(1 - \alpha) \frac{2\epsilon_0}{1-\tau} \right]$. The reduction is achieved when multitask finetuning is solved to a small loss ϵ_1 for a small α on sufficiently many finetuning data.

Theorem 3.3.5. (With Multitask Finetuning) *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^* and ϕ_ζ^* . Suppose for some constant $\alpha \in (0, 1)$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} \frac{2\epsilon_0}{1-\tau}$ and obtain ϕ' . For any $\delta > 0$, if for $\tilde{\epsilon} = \widehat{\mathcal{L}}_{\text{pre}}(\phi')$,*

$$\begin{aligned} M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{L}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ Mm &\geq \frac{1}{\epsilon_1} \left[16LB\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \end{aligned}$$

then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\alpha \frac{2\epsilon_0}{1-\tau} - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa. \quad (3.4)$$

The requirement is that the number of tasks M and the total number of labeled samples Mm across tasks are sufficiently large. This implies when M is above the threshold, the total size Mm determines the performance, and increasing either M or m while freezing the other can improve the performance. We shall verify these findings in our experiments (Section 3.4.1).

Theorem 3.3.5 also shows the conditions for successful multitask finetuning, in particular, the impact of the diversity and consistency of the finetuning tasks. Besides small finetuning loss on sufficiently many data, a large diversity parameter ν and a small consistency parameter κ will result in a small target error bound. Ideally, data from the finetuning tasks should be similar to those from the target task, but also sufficiently diverse to cover a wide range of patterns that may be encountered in the target task. This inspires us to perform finer-grained analysis of diversity and consistency using a simplified data model (Section 3.3.1), which sheds light on the design of an algorithm to select a subset of finetuning tasks with better performance (Section 3.3.2).

3.3.1 Case Study of Diversity and Consistency

Our main results, rooted in notions of diversity and consistency, state the general conclusion of multitask finetuning on downstream tasks. A key remaining question is how relevant tasks should be selected for multitask finetuning in practice. Our intuition is that this task selection should promote both diversity (encompassing the characteristics of the target task) and consistency (focusing on the relevant patterns in achieving the target task’s objective). To illustrate such theoretical concepts and connect them to practical algorithms, we specialize the general conclusion to settings that allow easy interpretation of diversity and consistency. In this section, we provide a toy linear case study and we put the proof and also the analysis of a more general setting in Section A.4, e.g., more general latent class $\mathcal{C}, \mathcal{C}_0$, more general distribution ζ , input data with noise.

In what follows, we specify the data distributions and function classes under consideration, and present an analysis for this case study. Our goal is to explain the intuition behind diversity and consistency notions: *diversity is about coverage, and consistency is about similarity in the latent feature space*. This can facilitate the design of task selection algorithms.

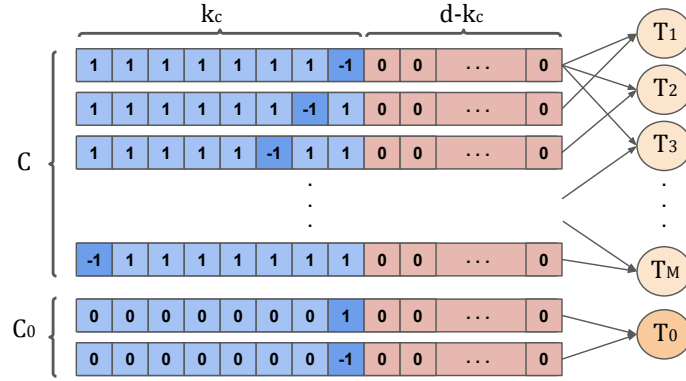


Figure 3.1: Illustration of features in linear data. Blue are the features encoded in C while red is not.

Linear Data and Tasks. Inspired by classic dictionary learning and recent analysis on representation learning (Wen and Li, 2021; Shi et al., 2023a), we consider the latent class/representation setting where each latent class $z \in \{0, -1, +1\}^d$ is represented as a feature vector. We focus on individual binary classification tasks, where $\mathcal{Y} = \{-1, +1\}$ is the label space. Thus, each task has two latent classes z, z' (denote the task as $\mathcal{T}_{z,z'}$) and we randomly assign -1 and $+1$ to each latent class. Namely, $\mathcal{T}_{z,z'}$ is defined as: $x = \begin{cases} z, & \text{if } y = -1 \\ z', & \text{if } y = +1 \end{cases}$. We show a diagram in Figure 3.1, we denote each task containing two latent classes, namely (z, z') . Each task in diagram can be represented as $(T_1$ to T_{z_1, z'_1}, T_2 to $T_{z_2, z'_2})$. We further assume a balanced class setting in all tasks, i.e., $p(y = -1) = p(y = +1) = \frac{1}{2}$. Now, we define the latent classes seen in multitask finetuning tasks:

$$\mathcal{C} = \left\{ \underbrace{(1, 1, \dots, 1, 1, -1, 0, \dots, 0)}_{k_e}^{\top}, \underbrace{(1, 1, \dots, 1, -1, 1, 0, \dots, 0)}_{d-k_e}^{\top}, \dots, \underbrace{(-1, 1, \dots, 1, 1, 1, 0, \dots, 0)}_{k_e}^{\top}, \underbrace{(1, 1, \dots, 1, 1, 1, 0, \dots, 0)}_{d-k_e}^{\top} \right\}.$$

Note that their feature vectors only encode the first k_c features, and $|\mathcal{C}| = k_c$. We let $\mathcal{C}_0 := \{z^{(1)}, z^{(2)}\} \subseteq \{0, -1, +1\}^d$ which is used for the target task, and assume that $z^{(1)}$ and $z^{(2)}$ only differ in 1 dimension, i.e., the target task can be done using this one particular dimension. Let ζ be a distribution uniformly sampling two different latent classes from \mathcal{C} . Then, our data generation pipeline for getting a multitask finetuning task is (1) sample two latent classes $(z, z') \sim \zeta$; (2) assign label $-1, +1$ to two latent classes.

Linear Model and Loss Function. We consider a linear model class with regularity Assumption 3.3.1, i.e., $\Phi = \{\phi \in \mathbb{R}^{d \times d} : \|\phi\|_F \leq 1\}$ and linear head $w \in \mathbb{R}^d$ where $\|w\|_2 \leq 1$. Thus, the final output of the model and linear head is $w^\top \phi x$. We use the loss in Shi et al. (2023a), i.e., $\ell(w^\top \phi x, y) = -y w^\top \phi x$.

Remark 3.3.1. *Although we have linear data, linear model, and linear loss, $\mathcal{L}_{\text{sup}}(\phi)$ is a non-linear function on ϕ as the linear heads are different across tasks, i.e., each task has its own linear head.*

Now we can link our diversity and consistency to features encoded by training or target tasks.

Theorem 3.3.6 (Diversity and Consistency). *If \mathcal{C} encodes the feature in \mathcal{C}_0 , i.e., the different entry dimension of $z^{(1)}$ and $z^{(2)}$ in \mathcal{C}_0 is in the first k_c dimension, then we have ν is lower bounded by constant $\tilde{c} \geq \frac{2\sqrt{2}-2}{k_c-1}$ and $\kappa \leq 1 - \sqrt{\frac{1}{k_c}}$. Otherwise, we have $\nu \rightarrow 0$ and $\kappa \geq 1$.*

Theorem 3.3.6 establishes \tilde{c} -diversity and κ -consistency in Definition 3.3.2 and Definition 3.3.3. The analysis shows that diversity can be intuitively understood as the coverage of the finetuning tasks on the target task in the latent feature space: If the key feature dimension of the target task is covered by the features encoded by finetuning tasks, then we have lower-bounded diversity ν ; if not covered, then the diversity ν tends to 0 (leading to vacuous error bound in Theorem 3.3.5). Also, consistency can be intuitively understood as similarity in the feature space: when k_c is small, a large fraction of the finetuning tasks are related to the target task,

leading to a good consistency (small κ); when k_c is large, we have less relevant tasks, leading to a worse consistency. Such an intuitive understanding of diversity and consistency will be useful for designing practical task selection algorithms.

3.3.2 Task Selection

Our analysis suggests that out of a pool of candidate tasks, a subset S with good consistency (i.e., small κ) and large diversity (i.e., large ν) will yield better generalization to a target task. To realize this insight, we present a greedy selection approach, which sequentially adds tasks with the best consistency, and stops when there is no significant increase in the diversity of the selected subset. In doing so, our approach avoids enumerating all possible subsets and thus is highly practical.

A key challenge is to compute the consistency and diversity of the data. While the exact computation deems infeasible, we turn to approximations that capture the key notions of consistency and diversity. We show a simplified diagram for task selection in Figure 3.2. Specifically, given a foundation model ϕ , we assume any task data $\mathcal{T} = \{x_j\}$ follows a Gaussian distribution in the representation space: let $\phi(\mathcal{T}) = \{\phi(x_j)\}$ denote the representation vectors obtained by applying ϕ on the data points in \mathcal{T} ; compute the sample mean $\mu_{\mathcal{T}}$ and covariance $C_{\mathcal{T}}$ for $\phi(\mathcal{T})$, and view it as the Gaussian $\mathcal{N}(\mu_{\mathcal{T}}, C_{\mathcal{T}})$. Further, following the intuition shown in the case study, we simplify consistency to similarity: for the target task \mathcal{T}_0 and a candidate task \mathcal{T}_i , if the cosine similarity

$\text{CosSim}(\mathcal{T}_0, \mathcal{T}_i) := \mu_{\mathcal{T}_0}^\top \mu_{\mathcal{T}_i} / (\|\mu_{\mathcal{T}_0}\|_2 \|\mu_{\mathcal{T}_i}\|_2)$ is large, we view \mathcal{T}_i as consistent with \mathcal{T}_0 . Next, we simplify diversity to coverage: if a dataset D (as a collection of finetuning tasks) largely “covers” the target data \mathcal{T}_0 , we view D as diverse for \mathcal{T}_0 .

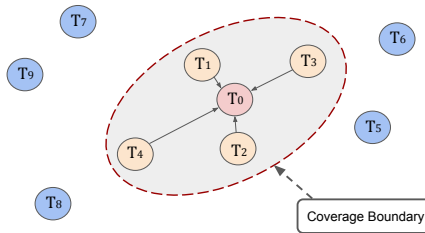


Figure 3.2: Illustration of the similarity and coverage. Target tasks (\mathcal{T}_0) with the most similar tasks in yellow and the rest in blue. The ellipsoid spanned by yellow tasks is the coverage for the target task. Adding more tasks in blue to the ellipsoid does not increase the coverage boundary.

Regarding the task data as Gaussians, we note that the covariance ellipsoid of D covers the target data $\mu_{\mathcal{T}_0}$ iff $(\mu_D - \mu_{\mathcal{T}_0})^\top C_D^{-1}(\mu_D - \mu_{\mathcal{T}_0}) \leq 1$. This inspires us to define the following coverage score as a heuristic for diversity: $\text{coverage}(D; \mathcal{T}_0) := 1/(\mu_D - \mu_{\mathcal{T}_0})^\top C_D^{-1}(\mu_D - \mu_{\mathcal{T}_0})$.

Using these heuristics, we arrive at the following selection algorithm: sort the candidate task in descending order of their cosine similarities to the target data; sequentially add tasks in the sorted order to L until $\text{coverage}(L; \mathcal{T}_0)$ has no significant increase. Algorithm 1 illustrates this key idea.

Algorithm 1 Consistency-Diversity Task Selection

Input: Target task \mathcal{T}_0 , candidate finetuning tasks: $\{\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_M\}$, model ϕ , threshold p .

- 1: Compute $\phi(\mathcal{T}_i)$ and $\mu_{\mathcal{T}_i}$ for $i = 0, 1, \dots, M$.
- 2: Sort \mathcal{T}_i 's in descending order of similarity($\mathcal{T}_0, \mathcal{T}_i$). Denote the sorted list as $\{\mathcal{T}'_1, \mathcal{T}'_2, \dots, \mathcal{T}'_M\}$.
- 3: $L \leftarrow \{\mathcal{T}'_1\}$
- 4: **for** $i = 2, \dots, M$ **do**
- 5: If $\text{coverage}(L \cup \mathcal{T}'_i; \mathcal{T}_0) \geq (1+p) \cdot \text{coverage}(L; \mathcal{T}_0)$, then $L \leftarrow L \cup \mathcal{T}'_i$; otherwise, break.
- 6: **end for**

Output: Selected data L for multitask finetuning.

3.4 Experiments

We now present our main results, organized in three parts. Section 3.4.1 explores how different numbers of finetuning tasks and samples influence the model's performance, offering empirical backing to our theoretical claims. Section 3.4.2 investigates whether our task selection algorithm can select suitable tasks for multitask finetuning. Section 3.4.3 provides a more extensive exploration of the effectiveness of multitask finetuning on various datasets and pretrained models. We defer other results to the appendix. Specifically, Section A.5.4 shows that better diversity and consistency of finetuning tasks yield improved performance on target tasks under

same sample complexity. Section A.5.4 shows that finetuning tasks satisfying diversity yet without consistency lead to no performance gain even with increased sample complexity. Further, Section A.6 and Section A.7 present additional experiments using NLP and vision-language models, respectively.

Experimental Setup. We use four few-shot learning benchmarks: miniImageNet (Vinyals et al., 2016), tieredImageNet (Ren et al., 2018), DomainNet (Peng et al., 2019) and Meta-dataset (Triantafillou et al., 2020). We use foundation models with different pretraining schemes (MoCo-v3 (Chen et al., 2021a), DINO-v2 (Oquab et al., 2024), and supervised learning with ImageNet (Russakovsky et al., 2015)) and architectures (ResNet (He et al., 2016) and ViT (Dosovitskiy et al., 2021)). We consider few-shot tasks consisting of N classes with K support samples and Q query samples per class (known as N -way K -shot). The goal is to classify the query samples based on the support samples. Tasks used for finetuning are constructed by samples from the training split. Each task is formed by randomly sampling 15 classes, with every class drawing 1 or 5 support samples and 10 query samples. Target tasks are similarly constructed from the test set. We follow (Chen et al., 2021b) for multitask finetuning and target task adaptation. During multitask finetuning, we update all parameters in the model using a nearest centroid classifier, in which all samples are encoded, class centroids are computed, and cosine similarity between a query sample and those centroids are treated as the class logits. For adaptation to a target task, we only retain the model encoder and consider a similar nearest centroid classifier. This multitask finetuning protocol applies to all experiments (Sections 3.4.1 to 3.4.3). We provide full experimental set up in Section A.5.

3.4.1 Verification of Theoretical Analysis

We conduct experiments on the tieredImageNet dataset to confirm the key insight from our theorem — the impact of the number of finetuning tasks (M) and the number of samples per task (m).

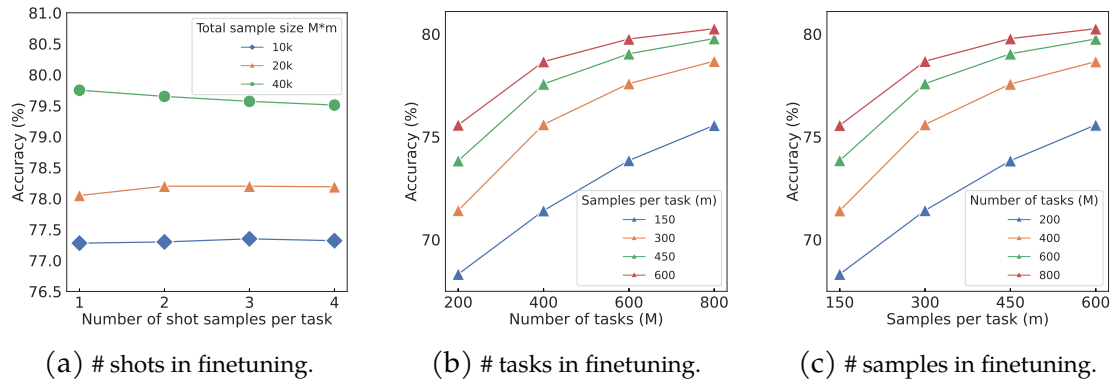


Figure 3.3: Results on ViT-B backbone pretrained by MoCo v3. (a) Accuracy v.s. number of shots per finetuning task. Different curves correspond to different total numbers of samples Mm . (b) Accuracy v.s. the number of tasks M . Different curves correspond to different numbers of samples per task m . (c) Accuracy v.s. number of samples per task m . Different curves correspond to different numbers of tasks M .

Results. We first investigate the influence of the number of shots. We fix the target task as a 1-shot setting but vary the number of shots from 1 to 4 in finetuning, and vary the total sample size $Mm = [10k, 20k, 40k]$. The results in Figure 3.3a show no major change in accuracy with varying the number of shots in finetuning. It is against the common belief that meta-learning like Prototypical Networks (Snell et al., 2017) has to mimic the exact few-shot setting and that a mismatch will hurt the performance. The results also show that rather than the number of shots, the total sample size Mm determines the performance, which is consistent with our theorem. We next investigate the influence of M and m . We vary the number of tasks ($M = [200, 400, 600, 800]$) and samples per task ($m = [150, 300, 450, 600]$) while keeping all tasks have one shot sample. Figure 3.3b shows increasing M with fixed m improves accuracy, and Figure 3.3c shows increasing m with fixed M has similar behavior. Furthermore, different configurations of M and m for the same total sample size Mm have similar performance (e.g., $M = 400, m = 450$ compared to $M = 600, m = 300$ in Figure 3.3b). These again align with our theorem.

| Pretrained | Selection | INet | Omglot | Acraft | CUB | QDraw | Fungi | Flower | Sign | COCO |
|------------|-----------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| CLIP | Random | 56.29 | 65.45 | 31.31 | 59.22 | 36.74 | 31.03 | 75.17 | 33.21 | 30.16 |
| | No Con. | 60.89 | 72.18 | 31.50 | 66.73 | 40.68 | 35.17 | 81.03 | 37.67 | 34.28 |
| | No Div. | 56.85 | 73.02 | 32.53 | 65.33 | 40.99 | 33.10 | 80.54 | 34.76 | 31.24 |
| | Selected | 60.89 | 74.33 | 33.12 | 69.07 | 41.44 | 36.71 | 80.28 | 38.08 | 34.52 |
| DINOv2 | Random | 83.05 | 62.05 | 36.75 | 93.75 | 39.40 | 52.68 | 98.57 | 31.54 | 47.35 |
| | No Con. | 83.21 | 76.05 | 36.32 | 93.96 | 50.76 | 53.01 | 98.58 | 34.22 | 47.11 |
| | No Div. | 82.82 | 79.23 | 36.33 | 93.96 | 55.18 | 52.98 | 98.59 | 35.67 | 44.89 |
| | Selected | 83.21 | 81.74 | 37.01 | 94.10 | 55.39 | 53.37 | 98.65 | 36.46 | 48.08 |
| MoCo v3 | Random | 59.66 | 60.72 | 18.57 | 39.80 | 40.39 | 32.79 | 58.42 | 33.38 | 32.98 |
| | No Con. | 59.80 | 60.79 | 18.75 | 40.41 | 40.98 | 32.80 | 59.55 | 34.01 | 33.41 |
| | No Div. | 59.57 | 63.00 | 18.65 | 40.36 | 41.04 | 32.80 | 58.67 | 34.03 | 33.67 |
| | Selected | 59.80 | 63.17 | 18.80 | 40.74 | 41.49 | 33.02 | 59.64 | 34.31 | 33.86 |

Table 3.1: Results evaluating our task selection algorithm on Meta-dataset using ViT-B backbone. No Con.: Ignore consistency. No Div.: Ignore diversity. Random: Ignore both consistency and diversity.

3.4.2 Task Selection

Setup. To evaluate our task selection Algorithm 1, we use the Meta-Dataset (Triantafillou et al., 2020). It contains 10 extensive public image datasets from various domains, each partitioned into train/val/test splits. For each dataset except Describable Textures due to small size, we conduct an experiment, where the test-split of that dataset is used as the target task while the train-split from all the other datasets are used as candidate finetuning tasks. Each experiment follows the experiment protocol in Section 3.4. We performed ablation studies on the task selection algorithm, concentrating on either consistency or diversity, while violating the other. See details in Section A.5.4.

Results. Table 3.1 compares the results from finetuning with tasks selected by our algorithm to those from finetuning with tasks selected by other methods. Our algorithm consistently attains performance gains. For instance, on Omniglot, our algorithm leads to significant accuracy gains over random selection of 8.9%, 19.7%, and 2.4% with CLIP, DINO v2, and MoCo v3, respectively. Violating consistency or diversity conditions generally result in a reduced performance compared to our

approach. These results are well aligned with our expectations and affirm our diversity and consistency conclusions. We provide more ablation study on task selection in Table A.7 in Section A.5.4. We also apply task selection algorithm on DomainNet in Section A.5.5. Furthermore, in Section A.6, we employ our algorithm for NLP models on the GLUE dataset.

3.4.3 Effectiveness of Multitask Finetuning

Setup. We also conduct more extensive experiments on large-scale datasets across various settings to confirm the effectiveness of multitask finetuning. We compare to two baselines: *direct adaptation* where we directly adapt pretrained model encoder on target tasks without any finetuning, and *standard finetuning* where we append encoder with a linear head to map representations to class logits and finetune the whole model. During testing, we removed the linear layer and used the same few-shot testing process with the finetuned encoders. Please refer Table A.12 in Section A.5.8 for full results.

Results. Table 3.2 presents the results for various pretraining and finetuning methods, backbones, datasets, and few-shot learning settings. Multitask finetuning consistently outperforms the baselines in different settings. For example, in the most challenging setting of 1-shot on DomainNet, it attains a major gain of 7.1% and 9.3% in accuracy over standard finetuning and direct adaptation, respectively, when considering self-supervised pretraining with DINO v2 and using a Transformer model (ViT-S).

Interestingly, multitask finetuning achieves more significant gains for models pretrained with supervised learning than those pretrained with contrastive learning. For example, on DomainNet, multitask finetuning on supervised pretrained ViT-B achieves a relative gain of 67% and 35% for 1- and 5-shot, respectively. In contrast, multitask finetuning on DINO v2 pretrained ViT-B only shows a relative gain of 10% and 4%. This suggests that models from supervised pretraining might face a

| pretrained | backbone | method | miniImageNet | | tieredImageNet | | DomainNet | |
|------------------------------------|----------|-------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| | | | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot |
| MoCo v3 | ViT-B | Adaptation | 75.33 (0.30) | 92.78 (0.10) | 62.17 (0.36) | 83.42 (0.23) | 24.84 (0.25) | 44.32 (0.29) |
| | | Standard FT | 75.38 (0.30) | 92.80 (0.10) | 62.28 (0.36) | 83.49 (0.23) | 25.10 (0.25) | 44.76 (0.27) |
| | | Ours | 80.62 (0.26) | 93.89 (0.09) | 68.32 (0.35) | 85.49 (0.22) | 32.88 (0.29) | 54.17 (0.30) |
| | ResNet50 | Adaptation | 68.80 (0.30) | 88.23 (0.13) | 55.15 (0.34) | 76.00 (0.26) | 27.34 (0.27) | 47.50 (0.28) |
| | | Standard FT | 68.85 (0.30) | 88.23 (0.13) | 55.23 (0.34) | 76.07 (0.26) | 27.43 (0.27) | 47.65 (0.28) |
| | | Ours | 71.16 (0.29) | 89.31 (0.12) | 58.51 (0.35) | 78.41 (0.25) | 33.53 (0.30) | 55.82 (0.29) |
| DINO v2 | ViT-S | Adaptation | 85.90 (0.22) | 95.58 (0.08) | 74.54 (0.32) | 89.20 (0.19) | 52.28 (0.39) | 72.98 (0.28) |
| | | Standard FT | 86.75 (0.22) | 95.76 (0.08) | 74.84 (0.32) | 89.30 (0.19) | 54.48 (0.39) | 74.50 (0.28) |
| | | Ours | 88.70 (0.22) | 96.08 (0.08) | 77.78 (0.32) | 90.23 (0.18) | 61.57 (0.40) | 77.97 (0.27) |
| | ViT-B | Adaptation | 90.61 (0.19) | 97.20 (0.06) | 82.33 (0.30) | 92.90 (0.16) | 61.65 (0.41) | 79.34 (0.25) |
| | | Standard FT | 91.07 (0.19) | 97.32 (0.06) | 82.40 (0.30) | 93.07 (0.16) | 61.84 (0.39) | 79.63 (0.25) |
| | | Ours | 92.77 (0.18) | 97.68 (0.06) | 84.74 (0.30) | 93.65 (0.16) | 68.22 (0.40) | 82.62 (0.24) |
| Supervised pretraining on ImageNet | ViT-B | Adaptation | 94.06 (0.15) | 97.88 (0.05) | 83.82 (0.29) | 93.65 (0.13) | 28.70 (0.29) | 49.70 (0.28) |
| | | Standard FT | 95.28 (0.13) | 98.33 (0.04) | 86.44 (0.27) | 94.91 (0.12) | 30.93 (0.31) | 52.14 (0.29) |
| | | Ours | 96.91 (0.11) | 98.76 (0.04) | 89.97 (0.25) | 95.84 (0.11) | 48.02 (0.38) | 67.25 (0.29) |
| | ResNet50 | Adaptation | 81.74 (0.24) | 94.08 (0.09) | 65.98 (0.34) | 84.14 (0.21) | 27.32 (0.27) | 46.67 (0.28) |
| | | Standard FT | 84.10 (0.22) | 94.81 (0.09) | 74.48 (0.33) | 88.35 (0.19) | 34.10 (0.31) | 55.08 (0.29) |
| | | Ours | 87.61 (0.20) | 95.92 (0.07) | 77.74 (0.32) | 89.77 (0.17) | 39.09 (0.34) | 60.60 (0.29) |

Table 3.2: **Results of few-shot image classification.** We report average classification accuracy (%) with 95% confidence intervals on test splits. Adaptation: Direction adaptation without finetuning; Standard FT: Standard finetuning; Ours: Our multitask finetuning; 1-/5-shot: number of labeled images per class in the target task.

larger domain gap than models from DINO v2, and multitask finetuning helps to bridge this gap.

3.5 Conclusion

In this work, we studied the theoretical justification of multitask finetuning for adapting pretrained foundation models to downstream tasks with limited labels. Our analysis shows that, given sufficient sample complexity, finetuning using a diverse set of pertinent tasks can improve the performance on the target task. This claim was examined in our theoretical framework and substantiated by the empirical evidence accumulated throughout our study. Built on this theoretical insight, we further proposed a practical algorithm for selecting tasks for multitask finetuning, leading to significantly improved results when compared to using all possible tasks.

This work directly addresses the first challenge that adapting foundation models

to new tasks when labeled data is scarce outlined in Chapter 1. By providing theoretical foundations and practical algorithms for multitask finetuning, I demonstrate how diverse intermediate tasks can bridge the gap between generic pretrained representations and specialized downstream requirements. These insights lay the groundwork for my subsequent investigations into model adaptation mechanisms, particularly in understanding how knowledge transfer occurs across tasks and how we can leverage this understanding to develop more effective adaptation techniques.

4 UNDERSTANDING COMPOSITIONAL ABILITIES

Contribution statement. This chapter is joint work with Zhenmei Shi and Yingyu Liang. I proposed the method, contributed to part of the theoretical analysis, and completed all the experiments. This chapter was published in COLM 2024 (Xu et al., 2024b).

4.1 Introduction

While multitask finetuning enables foundation models to acquire knowledge across diverse tasks, real-world deployment demands more than isolated task competence. Many practical problems require models to integrate multiple capabilities within a single inference—a challenge that exposes potential limitations in how these models compose learned operations. This gap between acquiring individual skills and combining them motivates our investigation into compositional abilities.

LLMs have revolutionized the natural language processing (NLP) and general AI community. Recent advances have shown success in various fields. As model scale increases, larger models exhibit new behavior known as emergence ability. One remarkable emergence is the in-context learning ability (ICL) (Brown et al., 2020), where a model can solve new tasks given only a few examples as input, without any parameter updates. However, despite recent success, how LLMs solve complex reasoning tasks, particularly not seen in pre-training, remains an open question and largely lacks understanding.

In this chapter, we focus on the problem of how LLMs tackle composite tasks that incorporate multiple simple tasks. Specifically, we investigate whether a model trained and in-context learned on individual tasks can effectively integrate these skills to tackle combined challenges, which are intuitive and simple for humans. For instance, in Figure 4.1, if a human is given examples where words following an asterisk (*) will be capitalized and words surrounded by parenthesis will be permuted, one can also understand words following an asterisk (*) surrounded by parenthesis

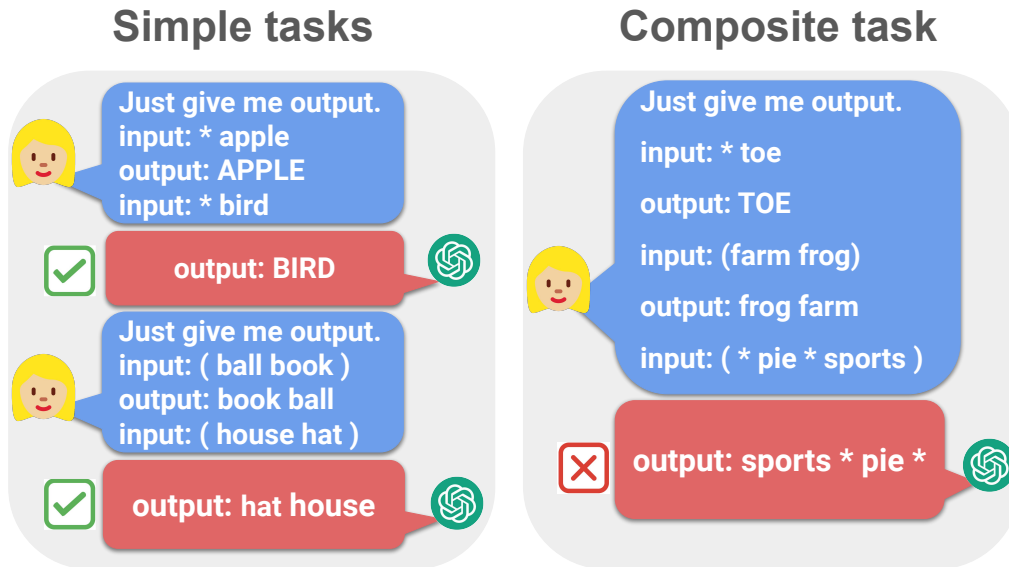


Figure 4.1: Inconsistent performance in GPT-4. Consider two simple tasks: If a word is followed by an asterisk (*), capitalize the letter. If two words are surrounded by parentheses, swap the positions. GPT-4 correctly solves two simple tasks based on demonstrations (left). The composite tasks have test inputs with both asterisk (*) and parenthesis. The correct answer should be *output: SPORTS PIE*. However, GPT-4 fails to solve the composite tasks (right). The same failure was observed in Claude 3.

will be capitalized and permuted simultaneously. This basic generalization seems trivial, yet we observe that LLMs fail to generalize in this way.

Compositional ability is an active problem in the AI community and is crucial for advancing Artificial General Intelligence (AGI). Recent studies have made significant contributions to the understanding of this area. Dziri et al. (2023) formulate compositional tasks as computation graphs to quantify each task’s complexity level. Power et al. (2022) show that models may develop generalization capabilities when trained extensively, beyond the point of overfitting, highlighting a phenomenon known as “grokking”. An et al. (2023b) examines how LLMs acquire abstract reasoning and achieve compositional generalization in a linguistic context through ICL by testing LLMs on tasks that involve translating a formal language with a custom grammar. Although these studies offer insight, how LLMs compose tasks

together is still not fully understood, especially in the ICL setting. Moreover, the absence of a solid theoretical framework in these discussions needs to be investigated concerning the underlying mechanisms of such behaviors.

Inspired by these seminal works, we further evaluate LLMs on a series of compositional tasks through ICL. The models were presented with examples of simple tasks and then asked to tackle composite tasks that they had not encountered during pretraining or in-context learning. We observe various behaviors: (1) for some composite tasks, the models showed a reasonable level of compositional skill, a capability that improved with larger model sizes; (2) for more complex composite tasks requiring sequential reasoning, the model struggle, and increasing the model size typically did not lead to better performance.

Our key intuition is that if the simple tasks that form a composite task can be easily separated into subtasks based on the inputs (e.g., performed separately on different parts of the input sentence), the model is more likely to complete such a composite task successfully (we call it “a separable composite task”). The performance of the model depends on how it connects and uses the information given for each part of the task. To clarify this insight, we present theoretical analyses in a simplified setting and provide key insights into conditions needed for success in such separable composite tasks.

Our contributions are twofold. *Empirically*, we introduce a variety of composite tasks from both the linguistic and logical domains to explore how the nature of these tasks influences the compositional performance of LLMs through ICL. *Theoretically*, we provide analysis on a simple yet insightful model: a one-layer single-head linear self-attention network (Von Oswald et al., 2023; Akyürek et al., 2023; Mahankali et al., 2023; Zhang et al., 2023b). This framework allows us to demonstrate a clear separation in input embedding, effectively breaking down composite tasks into simpler components. We delve into the scaling of language models by examining the structure of the key and query matrices in the attention mechanism, arguing that larger models with a more complex internal structure exhibit enhanced performance on individual tasks, thereby improving their overall compositional capabilities on such separable tasks.

4.2 Warm-up: A Failure Case for Composition

Our goal is to understand the behavior of LLMs on compositional reasoning tasks. As a warm-up, we evaluate the **Capitalization & Swap** tasks (Figure 4.1) on different models. Recall the tasks: given words of common objects, * represents the operation of capitalizing the letters, () represents swapping the positions of the two words. We consider the standard in-context learning setting, which concatenates input-output examples $K = 10$ and one test input as the prompt for LLM. We perform experiments across various LLM families, e.g., Llama families (Touvron et al., 2023a) and GPTs (Radford et al., 2019; Black et al., 2021), see model details in Section B.1.

Evaluation settings. To make thorough evaluations, we consider four settings: (1) capital: only on the capitalization task; (2) swap: only on swap; (3) composite: in-context examples are from simple tasks while the test input is about the composite task; (4) composite in-context: in-context examples and the test input are all drawn from the composite task. The composite in-context setting reduces the evaluation to another simple task, not requiring the model to composite the simple task ability but directly learning from the in-context examples. It serves as the gold standard performance for the composite task. See Table 4.1 for illustration.

Results. In Figure 4.2, somewhat surprisingly, we observe that LLMs cannot solve the composite task, although they perform well on simple tasks. There is a signifi-

| | Composite | Composite in-context |
|--------|---|--|
| Prompt | <i>input: * apple</i> <i>output: APPLE</i> <i>input: (farm frog)</i> <i>output: frog farm</i> <i>input: (* bell * ford)</i> | <i>input: (* good * zebra)</i> <i>output: ZEBRA GOOD</i> <i>input: (* bicycle * add)</i> |
| Truth | <i>output: FORD BELL</i> | <i>output: ADD BICYCLE</i> |

Table 4.1: Examples of two settings on composite tasks. Composite: in-context examples are about simple tasks, while the test input is about the composite task. Composite in-context: both in-context examples and the test input are about the composite task.

cant gap between the performance in these settings. Models in Llama families can solve capital and swap with nearly ~90% accuracy but only achieve around 20% or below on the composite task. We also observe that composite in-context examples will significantly improve the performance: The accuracy of Llama families can go up to match the simple task accuracy. These observations show that *the models fail to compose the knowledge from the simple tasks, although they do have the representation power to solve the composite task* (which can only be exploited when provided composite in-context examples) and *scaling up may not help*.

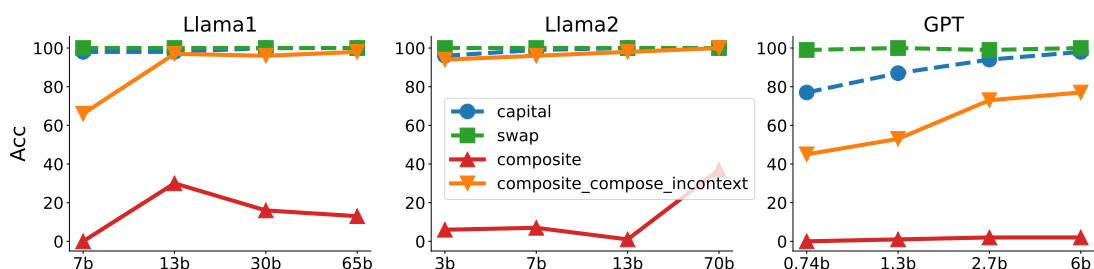


Figure 4.2: The exact match accuracy (y-axis) vs the model scale (x-axis, “b” stands for billion) for **Capitalization & Swap** tasks (example in Figure 4.1). Line *capital*: performance on the simple task of capitalization; *swap*: on the simple task of swap; *composite*: in-context examples are from simple tasks while test input from the composite task. *composite incontext*: in-context examples and test input are all from the composite task (example in Table 4.1).

4.3 Variability of Compositional Performance

The experiment on Capitalization & Swap shows failure cases while existing studies reported some successful composite abilities (Levy et al., 2022; An et al., 2023b). This observation suggests a more refined perspective: LLMs exhibit variable compositional abilities, excelling in certain composite tasks while struggling in others. This section expands our exploration to additional composite tasks to further examine and understand this variability.

We introduce more composite tasks, including linguistic and logical challenges, wrapped as a testing suite. Similar to the Capitalization & Swap experiment, we design composite tasks that compose two simple tasks and evaluate the model in four settings: the two simple tasks, the composite setting, and the composite in-context setting (Table 4.1 show examples for the latter two). We consider two kinds of task: logical rules and linguistic translation. We first choose two simple tasks and compose them to construct a composite task.

To address concerns about data leakage and the possibility that models encounter similar tasks during pretraining, we opt for synthetic data in this work. While it is challenging to guarantee that test data has never been seen during pretraining, we take significant steps to mitigate this risk. Specifically, we construct our compositional test data using a unique syntax and mapping mechanism. This approach substantially shifts the data distribution away from existing web-scale data, making it highly improbable that our test data has been encountered during pretraining. By doing so, we aim to create novel composite tasks that comprehensively evaluate the models’ compositional abilities. Section 4.3.1 investigates logical tasks and Section 4.3.2 investigates translation tasks.

We perform experiments to answer the following questions: **(Q1)** How do LLMs perform in various tasks, where models might perform well in some scenarios while failing in others? **(Q2)** Does scaling up the model help in general? **(Q3)** Is the variability in performance relevant to the nature of tasks? Our experiments provide the following answers: **(A1)** A pattern of variable performance is observable across a range of composite tasks. **(A2)** Scaling-up helps when the model exhibits compositional ability for certain tasks but may not help when the model initially struggles. **(A3)** In tasks that involve processing inputs from varied segments or perspectives, especially simpler ones, the model tends to demonstrate compositional capabilities.

4.3.1 Composite Logical Rules

We enhance our suite of logical tasks by introducing a series of straightforward tasks that process either simple words or numerical values, with the output being a

specific functional transformation of the input. These tasks are detailed in Table 4.2.

| Tasks | Task | Input | Output |
|------------------|----------------------|-----------------|------------|
| Words | (A) Capitalization | apple | APPLE |
| | (B) Swap | bell ford | ford bell |
| | (C) Two Sum | twenty @ eleven | thirty-one |
| | (D) Past Tense | pay | paid |
| | (E) Opposite | Above | Below |
| Numerical | (F) Plus One | 435 | 436 |
| | (G) Modular | 15 @ 6 | 3 |
| | (H) Two Sum Plus One | 12 # 5 | 18 |

Table 4.2: This table contains a collection of simple logical tasks. The *Words* category encompasses tasks that modify words at the character or structural level. The *Numerical* category is devoted to tasks that involve arithmetic computations performed on numbers.

| Tasks | Simple Task | Simple Task | Composite |
|-----------|--------------------------------------|---|---|
| (A) + (B) | input: * apple output: AP- PLE | input: (farm frog) output: frog farm | input: (* bell * ford) output: FORD BELL |
| (A) + (F) | input: 435 output: 436 | input: cow output: COW | input: 684 cat output: 685 CAT |

Table 4.3: Examples of the two logical composite tasks. Full examples can be found in Section B.1.

Composite tasks are created by merging two simple tasks. We conceptualize simple tasks as functions, $f(\cdot)$ and $g(\cdot)$ that map inputs to their respective outputs. We identify two distinct approaches to creating composite tasks: (1) **Compose by**

parts: For inputs x, y , the result is $f(x), g(y)$. One example is **(A) + (F)** in Table 4.3. If a numerical number is given, it will increment by one; if the word is given, the letters will be capitalized; if both are given, perform both operations. **(2) Compose by steps:** Given input x , the result is $f(g(x))$. One example is **(A) + (B)** in Table 4.3. We use customized symbols as function mapping for composing two simple tasks. Examples are in Figure 4.1 and Table 4.3. Following existing work, we use exact match accuracy to evaluate the performance since the output for these tasks is usually simple and short.

Results. We provide our main results on composite tasks in Table 4.4. For the composed by parts tasks **(A) + (F)** and **(D) + (F)**, the models show strong compositional ability: the composite accuracy is high, improves with increasing scale, and eventually reaches similar performance as the “gold standard” composite in-context setting, as highlighted in red numbers. We refer to these tasks as “separable composite tasks”, which are relatively easy for the model to solve. On the compose-by-step tasks, we observe the models have various performances. For composite tasks with sequential reasoning steps, the models exhibit various performances. For tasks involving capitalization **(A)** or swap **(B)**, the model has poor performance on a small scale (7b or lower) but has increased performance in increased model scale, such as 44% accuracy in **(A) + (C)** and 66% accuracy in **(B) + (D)**. One exception is Llama1-65b, which has lower accuracy than a smaller-scale model. We conjecture it is due to some unknown inductive bias during the pretraining. On composite steps tasks involving the arithmetic calculation of numerical numbers **(G) + (H)**, the model has the worst performance, and increasing the model scale does not provide benefits. A key observation is that compose-by-part tasks are separable compositions where the input can be broken down into two distinct segments. Such tasks are typically straightforward for a model to address. In all experiments, providing composed examples as in-context demonstrations will help the model understand and solve the composite tasks well, such as **Com. in-context** rows in all task combinations. We conclude that models fail to compose mechanisms of two simple tasks together; however, given composite examples, models can learn the composed mechanism efficiently. We also experimented with prompt demon-

strations and found instructions provide no direct results; see more experimental details in Section B.1.2. See more experimental results (including Llama3 (Meta, 2024)) and visualizations in Section B.1.3.

| Tasks | | Mistral | | Llama2 | | | Llama1 | | | |
|-----------|-----------------|---------|------|--------|-----|-----|--------|-----|-----|-----|
| | | 7B | 8x7B | 7B | 13B | 70B | 7B | 13B | 30B | 65B |
| (A) + (B) | Capitalization | 99 | 98 | 99 | 100 | 100 | 98 | 98 | 100 | 100 |
| | swap | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| | Compose | 16 | 42 | 7 | 1 | 37 | 0 | 30 | 16 | 13 |
| | Com. in-context | 95 | 96 | 96 | 98 | 100 | 66 | 97 | 96 | 98 |
| (A) + (C) | twoSum | 71 | 100 | 72 | 93 | 99 | 62 | 56 | 98 | 99 |
| | Capitalization | 98 | 99 | 100 | 95 | 99 | 97 | 98 | 99 | 99 |
| | Compose | 8 | 19 | 3 | 23 | 44 | 3 | 3 | 31 | 2 |
| | Com. in-context | 31 | 65 | 52 | 77 | 100 | 9 | 22 | 93 | 69 |
| (A) + (F) | Capitalization | 97 | 99 | 98 | 77 | 99 | 84 | 96 | 99 | 98 |
| | PlusOne | 100 | 99 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| | Compose | 92 | 96 | 74 | 69 | 97 | 57 | 60 | 69 | 99 |
| | Com. in-context | 99 | 98 | 99 | 100 | 100 | 99 | 99 | 100 | 100 |
| (B) + (D) | Swap | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| | Past Tense | 97 | 99 | 97 | 100 | 99 | 97 | 98 | 100 | 100 |
| | Compose | 6 | 12 | 0 | 1 | 62 | 57 | 34 | 46 | 5 |
| | Com. in-context | 92 | 98 | 86 | 95 | 98 | 86 | 95 | 89 | 94 |
| (B) + (E) | Swap | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 | 100 |
| | Opposite | 61 | 62 | 58 | 68 | 65 | 51 | 58 | 64 | 63 |
| | Compose | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Com. in-context | 35 | 32 | 12 | 37 | 37 | 0 | 9 | 7 | 9 |
| (D) + (F) | Past Tense | 100 | 100 | 98 | 100 | 100 | 100 | 100 | 100 | 100 |
| | Plus One | 100 | 100 | 100 | 100 | 100 | 99 | 100 | 100 | 100 |
| | Compose | 71 | 46 | 32 | 80 | 80 | 40 | 44 | 14 | 74 |
| | Com. in-context | 98 | 100 | 98 | 99 | 100 | 95 | 96 | 98 | 100 |
| (G) + (H) | Modular | 25 | 22 | 5 | 23 | 43 | 9 | 16 | 29 | 29 |
| | twoSumPlus | 38 | 42 | 3 | 77 | 90 | 14 | 10 | 40 | 87 |
| | Compose | 4 | 5 | 0 | 1 | 1 | 0 | 0 | 0 | 5 |
| | Com. in-context | 4 | 8 | 13 | 13 | 12 | 11 | 13 | 7 | 12 |

Table 4.4: Results are evaluated composite tasks on various models. The accuracy is in %.

4.3.2 Composite Linguistic Translation

Inspired by previous work in compositional generalization (An et al., 2023b; Levy et al., 2022; An et al., 2023a; Kim and Linzen, 2020), here we design our composite

tasks by formal language translation tasks.

Our translation tasks are mainly derived from semantic parsing task COGS (Kim and Linzen, 2020) and compositional generalization task COFE An et al. (2023b). These two datasets contain input as natural English sentences and output as a chain-ruled sentence following a customized grammar (see details in Section B.2). We construct two composite tasks centered on compositional generalization utilizing the training datasets to create in-context examples. See details in Section B.2.

We use the word error rate (WER) as the metric. It measures the minimum number of editing operations (deletion, insertion, and substitution) required to transform one sentence into another and is common for speech recognition or machine translation evaluations.

(T1) Phrase Recombination with Longer Chain. COFE proposed two compositional generalization tasks (Figure 2 in An et al. (2023b)). *Phrase Recombination*: integrate a prepositional phrase (e.g., “A in B”) into a specific grammatical role (e.g., “subject”, “object”); *Longer Chain*: Extend the tail of the logical form in sentences. We see them as simple tasks, and merge them to form a composite task: substitute the sentence subject in the Longer Chain task with a prepositional phrase from the Phrase Recombination task. Details and examples are in Table B.5 of Section B.2.

(T2) Passive to Active and Object to Subject Transformation. We consider two tasks from Kim and Linzen (2020). *Passive to Active*: Transitioning sentences from passive to active voice. *Object to Subject*: Changing the same object (a common noun) from objective to subjective. They are merged to form our composite task, where both transformations are applied simultaneously to the input sentence. Details and examples are in Table B.4 of Section B.2.

Results. Figure 4.3 shows that LLMs can handle these composite tasks. The WER on the composite task is decent and improves with increasing model scale, particularly in Llama2 models. These confirm the composite abilities of the models in these tasks. Here, we notice that both composite tasks are separable composite tasks. If we break down these sentences into sub-sentences and phrases, the simple task operations occur in different parts or perspectives of the input sentences. So, the results here provide further support for composite abilities on separable composite

tasks, where simple tasks that form the composite task are related to inputs from different parts or perspectives.

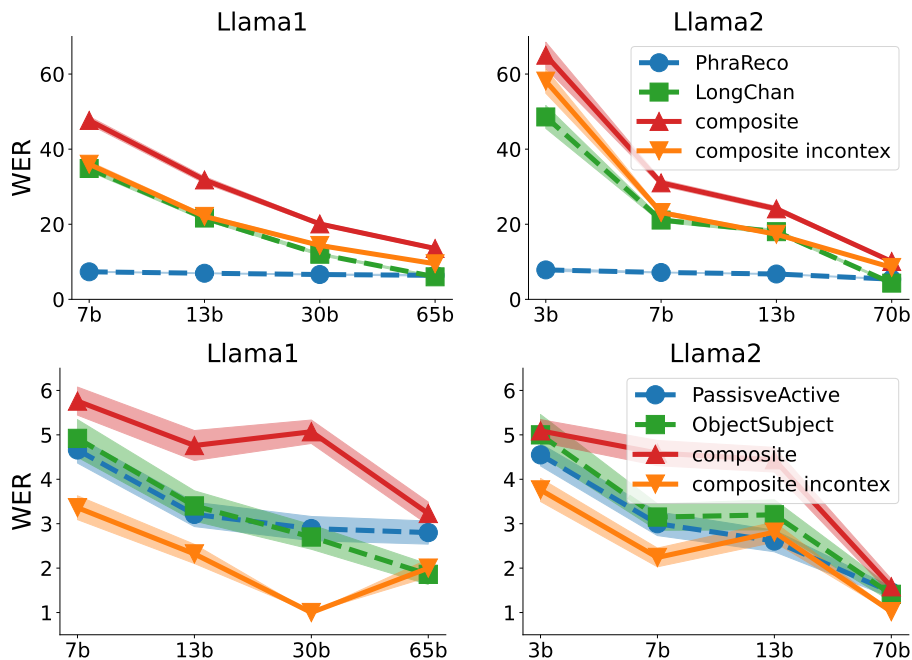


Figure 4.3: The word error rate (WER) vs the model scale on composite linguistic translation tasks. Dashed lines: simple tasks. Solid lines: composite tasks. Rows: (T1) Phrase Recombination with Longer Chain; (T2) Passive to Active and Object to Subject Transformation. Columns: different models. Lines: performance in different evaluation settings, e.g., the two simple tasks, the composite setting, and the composite in-context setting (examples are shown in Section B.2).

We also observed the LLM exhibits better compositional ability on linguistic tasks than on logical tasks. We conclude natural language inputs can indeed help language models understand concepts better than special symbols or code. Natural language provides a richer context, which aligns better with how these models are trained on large text corpora. In contrast, logical and numerical tasks often rely on more rigid structures, which makes it harder for models to generalize without explicit training on such patterns.

Discussion. We observe the capability of models to handle composite tasks is significantly influenced by the task characteristics. If composite tasks contain simple tasks related to different parts or perspectives of the input, the model will tackle the composite tasks well.

One natural explanation is that the model processes the input in some hidden embedding space and decomposes the embedding of the input into different “regions”. Here, each region is dedicated to specific types of information and thus related to different tasks, such as word-level modifications, arithmetic calculations, mapping mechanisms, semantic categorization, linguistic acceptability, or sentiment analysis. Then, suppose the two simple tasks correspond to two different task types that relate to separate regions of the embedding. In that case, the model can effectively manage the composite task by addressing each simple task operation within its corresponding region. As the model scales, its ability to handle individual tasks improves, leading to enhanced performance on composite tasks in such scenarios. For separable composite tasks, the inputs are divided into distinct regions and also reflected in embeddings, resulting in the model’s high performance. However, when the simple tasks are not separable (e.g., requiring sequential steps in reasoning), their information mixes together, complicating the model’s ability to discern and process them distinctly. Such overlap often leads to the model’s inability to solve the composite task. This intuition is formalized in the following sections in a stylized theoretical setting.

4.4 Theoretical Analysis

4.4.1 Problem Setup

Despite the complex nature of non-linearity in transformers in LLMs, we note it is useful to appeal to the simple case of linear models to see if there are parallel insights that can help us better understand the phenomenon. In this section, we analyze a linear attention module and aim to provide rigorous proof about why LLMs can achieve compositional ability in some simple cases that could shed light

on the more intricate behaviors observed in LLMs.

In-context learning. We follow existing work (Akyürek et al., 2023; Garg et al., 2022; Mahankali et al., 2023) with slight generalization to K simple tasks. A labeled example is denoted as (x, y) where $x \in \mathbb{R}^d, y \in \mathbb{R}^K$. In a simple task $k \in [K]$, y has only one non-zero entry $y^{(k)}$. In a composite task, y can have non-zero entries in dimensions corresponding to the combined simple tasks. The model takes a prompt $(x_1, y_1, \dots, x_N, y_N, x_q)$ as input, which contains N in-context examples (x_i, y_i) 's and a query x_q , and aims to predict \hat{y}_q close to the true label y_q for x_q . The prompt is usually stacked into an embedding matrix: $E := \begin{pmatrix} x_1 & x_2 & \dots & x_N & x_q \\ y_1 & y_2 & \dots & y_N & 0 \end{pmatrix} \in \mathbb{R}^{d_e \times (N+1)}$

where $d_e = d + K$. In in-context learning, we first pretrain the model using training prompts and then evaluate the model with evaluation prompts; see details below.

Pretraining procedure. We have B training data indexed by τ , each containing an input prompt $(x_{\tau,1}, y_{\tau,1}, \dots, x_{\tau,N}, y_{\tau,N}, x_{\tau,q})$ and a corresponding true label $y_{\tau,q}$. Consider the following empirical loss: $\hat{L}(\theta) = \sum_{k=1}^K \hat{L}_k(\theta) = \frac{1}{2B} \sum_{\tau=1}^B \|\hat{y}_{\tau,q} - y_{\tau,q}\|^2$ and the population loss (i.e., $B \rightarrow \infty$): $L(\theta) = \frac{1}{2} \mathbb{E}_{x_{\tau,1}, y_{\tau,1}, \dots, x_{\tau,N}, y_{\tau,N}, x_{\tau,q}} \left[(\hat{y}_{\tau,q} - y_{\tau,q})^2 \right]$.

Evaluation procedure. We now detail how to evaluate the model on downstream *composite* tasks. We consider the downstream classification task to be a multi-class classification problem, where the output label is a K -dimensional vector, and each entry corresponds to a simple binary classification task. For any given simple task k , the binary classification label is given by $\text{sgn}(y_q^{(k)})$, where sgn is the sign function. Similarly, our prediction is $\tilde{y}_q^{(k)} = \text{sgn}(\hat{y}_q^{(k)})$. The accuracy of a composite task is defined as $\text{Acc}_\theta(x_1, \dots, y_N, x_q) = \frac{1}{K} \sum_{k=1}^K \mathbb{1} \left(\text{sgn}(\hat{y}_q^{(k)}) = \text{sgn}(y_q^{(k)}) \right)$. We denote it as $\text{Acc}_\theta(\{x_i, y_i\}_{i=1}^N)$. Here we denote the model performance on each task as separate dimension, (e.g. letter capitalization, numbers increment), and the performance of composite tasks as the aggregation of multiple dimensions.

Data. Assume xi.i.d. $\mathcal{N}(0, \Lambda)$, where $\Lambda \in \mathbb{R}^{d \times d}$ is the covariance matrix. Assume $y = Wx$, where $W \in \mathbb{R}^{K \times d}$. For any simple task $k \in [K]$, its label is the k -th entry of y , which is $y^{(k)} = \langle w^{(k)}, x \rangle$, where $w^{(k)}$ is the k -th row of W . We assume each task weight $w^{(k)}$ i.i.d. $\mathcal{N}(0, I_d)$.

Linear self-attention networks. These networks are widely studied (Von Oswald

et al., 2023; Akyürek et al., 2023; Garg et al., 2022; Zhang et al., 2023b; Shi et al., 2023d). Following them, we consider the following linear self-attention network with parameters $\theta = (W^{PV}, W^{KQ})$: $f_{\text{LSA},\theta}(E) = E + W^{PV}E \cdot \frac{E^T W^{KQ} E}{N}$. The prediction of the model for x_q is $\hat{y}_q = [f_{\text{LSA},\theta}(E)]_{(d+1):(d+K),N+1}$, the bottom rightmost sub-vector of $f_{\text{LSA},\theta}(E)$ with length K .

Compositional ability. We now provide a formal definition about *compositional ability* of an LLM on composite tasks.

Definition 4.4.1 (Compositional Ability). *Consider a composite task \mathcal{T} that combines two simple tasks k and g . Let S_k denote N labeled examples from task k , and similarly for S_g . Given an x_q from composite task \mathcal{T} , we say that the model has compositional ability on \mathcal{T} if the model has higher accuracy using in-context examples from $S_k \cup S_g$ than from either single one, i.e. $\max\{Acc_\theta(S_k), Acc_\theta(S_g)\} \leq Acc_\theta(S_k \cup S_g)$.*

4.4.2 Theoretical Results

In this section, we present our theoretical results. We explain the observation in empirical results through the lens of confined supports in input embeddings corresponding to separate subspaces (modeling separable composition). We provide theoretical justification showing that separable composite task composite tasks whose inputs are composed by components adhere to certain conditions where models exhibit satisfactory performance. Models will fail when such conditions are violated. We first introduce the basic setup and definitions.

Disjoint subspaces of simple tasks. Recall that x lies in a d -dimensional space where each dimension represents a different characteristic. A simple task may depend only on a subset of these dimensions since its label only depends on a few features. Let $\mathbb{S} = [d]$ represent the dimensions of x . For a task k , the output $y^{(k)} = \langle w_k, x \rangle$ depends on a subset of dimensions in x . Denote this subset by $\mathbb{K} \subseteq \mathbb{S}$ and call it the active index set for task k .

In the following, we always assume that the K tasks have disjoint subspaces: for any two tasks $k \neq g$, their active index sets \mathbb{K} , and \mathbb{G} are disjoint, i.e., $\mathbb{K} \cap \mathbb{G} = \emptyset$. In practice, the dimensions within \mathbb{K} could be associated with numerical arithmetic

operations, while those in \mathbb{G} might pertain to semantic analysis. This illustrates the model’s approach to address these tasks in their respective subspaces.

We now introduce a mild assumption regarding the distribution of input embeddings.

Assumption 4.4.2. *Given two disjoint subspaces \mathbb{K} and \mathbb{G} , the covariance matrix Λ of the input distribution can be segmented into block matrices $\Lambda_{\mathbb{K}\mathbb{K}}$, $\Lambda_{\mathbb{K}\mathbb{G}}$, $\Lambda_{\mathbb{G}\mathbb{K}}$, and $\Lambda_{\mathbb{G}\mathbb{G}}$, then we assume $\sigma_{\max}(\Lambda_{\mathbb{K}\mathbb{G}}) = \sigma_{\max}(\Lambda_{\mathbb{G}\mathbb{K}}) \leq \epsilon$ for constant ϵ , where $\sigma(\cdot)$ denote the singular value of matrix.*

Assumption 4.4.2 implies that for two separate simple tasks, each associated with its respective feature subspace \mathbb{K} and \mathbb{G} , the covariance between these two sets of features is bounded by a constant value. This is a natural assumption when inputs of composite tasks can be decomposed into parts. Suppose we have input embeddings from two tasks: arithmetic computations and semantic analysis. This assumption suggests that the feature subspaces of the input embeddings for two tasks are almost independent.

We now define confined support, which means that each task’s input embedding only has support within its feature subspace.

Definition 4.4.3 (Confined Support). *We say a task has confined support if the input x only has larger singular values within its active index set. The norm of entries outside the active index set is bounded by a small constant δ .*

This definition shows that each simple task only has large values within its corresponding subsets of dimensions of input embeddings. For example, let \mathbb{K} represent the first d_1 dimensions of an input vector x , and \mathbb{G} account for the remaining d_2 dimensions, with the total dimension being $d = d_1 + d_2$. The examples from task k will have input as $x = (x_1, x_{\delta_1})$ where $x_1 \in \mathbb{R}^{d_1}$, $x_{\delta_1} \in \mathbb{R}^{d_2}$, $\|x_{\delta_1}\| \leq \delta$. Similarly, the examples from task g will have inputs as $x = (x_{\delta_2}, x_2)$. We now present our results of the compositional ability under a confined support of x .

Theorem 4.4.4. Consider distinct tasks k and g with corresponding examples $\mathcal{S}_k, \mathcal{S}_g$. If two tasks have confined support, and Assumption 4.4.2 is true, then with high probability, the model has the compositional ability as defined in Definition 4.4.1. Moreover,

$$Acc_{\theta}(\mathcal{S}_k) + Acc_{\theta}(\mathcal{S}_g) \leq Acc_{\theta}(\mathcal{S}_{k \cup g}).$$

Theorem 4.4.4 shows the compositional ability of LLMs to handle composite tasks that integrate two simple tasks, which have confined support in their own feature subspace.

An illustrative case involves the tasks of Capitalization (A) & Plus One (F) and Past Tense (D) & Plus One (F), as depicted in Table 4.4. These two simple tasks involve word-level modification and arithmetic operation on separate parts of the input. Due to this separation, each task correlates with a specific segment of the input embedding. Therefore, it is observed that these tasks possess confined supports.

We further provide theory illustrating the **necessity of the confined supports**, we demonstrate that when the confined support is violated, simple tasks begin to show variations (indicated by large singular values) across the entire feature subspace of the input embedding. For instance, the composite task of Capitalization (A) & Swap (B), which involves mixed steps in reasoning as shown in Figure 4.2, shows poor performance of LLMs given both simple tasks' examples as in-context demonstrations. Another example is Modular (G) & Two Sum Plus (H) as shown in the last row of Table 4.4, where both simple tasks involve multisteps arithmetic operation. These two tasks share the same embedding space support, mixing their variations and causing the model to be unable to effectively address the composite tasks that integrate them. We further substantiate this observation with Section 4.4.2, which establishes that when two tasks share overlapping support in the embedding space, a scenario can arise where the model fails to demonstrate compositional ability.

If two tasks do not have confined support, there exists one setting in which we

have

$$\text{Acc}_\theta(\mathcal{S}_k) = \text{Acc}_\theta(\mathcal{S}_g) = \text{Acc}_\theta(\mathcal{S}_{k \cup g}).$$

Section 4.4.2 demonstrates that a model’s failure to solve tasks with mixed steps reasoning, which contains overlapping input embedding spaces, thereby diminishing the model’s ability to solve them when presented together.

We also show the **scaling effect**: if simple tasks have confined support, the compositional ability of language models will increase as the model scale increases in Theorem B.3.1 in Section B.3.1. We demonstrate this by showing that the model’s accuracy on each simple task improves with a larger model scale. We finally provide a **case study** on confined support for illustration in Section B.3.2. We defer the full proof in Section B.4.

4.5 Conclusion

In this chapter, we presented a distinct pattern in LLMs’ behaviors when tackling composite tasks. We observed that if the composite task can be separated into two simple tasks whose inputs are from distinct perspectives, the models exhibit decent compositional ability. Otherwise, LLMs will struggle, and scaling up the model size may not offer improvement. We illustrated this behavior across a variety of logical and linguistic challenges. We extended our discussion to the role of input embeddings in affecting model performance, providing a theoretical backup that connects the nature of tasks to how inputs are processed. We anticipate that our research will shed light on the compositional capabilities and reasoning of LLMs, and stimulate further exploration in this direction.

This work addresses the challenge of understanding when and why foundation models succeed or fail at compositional reasoning I proposed in Chapter 1. My findings reveal fundamental patterns in how these models handle composite tasks: they excel when distinct operations apply to separate input segments, but struggle when tasks require sequential, interdependent reasoning steps. These discoveries directly inform my collaborative work on analyzing induction heads and their

role in out-of-distribution generalization (Song et al., 2025), as understanding the mechanisms behind successful composition can help develop more robust architectures for complex reasoning tasks. Furthermore, this improved understanding of compositional behavior informs how we design prompts and task decompositions, connecting to the broader goal of effectively deploying foundation models on tasks that require multi-step reasoning.

5 ADAPTIVE RUNTIME INFERENCE

Contribution statement. This chapter is joint work with Khoi Duc Nguyen, Preeti Mukherjee, Saurabh Bagchi, Somali Chaterji, Yingyu Liang and Yin Li. I proposed the method and completed part of the experiments. This chapter was published in ICCV 2025 (Xu et al., 2025).

5.1 Introduction

While we have investigated how foundation models acquire task-specific knowledge and how they combine capabilities for complex reasoning, a critical challenge remains: deploying these models under the computational constraints of real-world systems. Server farms face fluctuating loads, edge devices contend with limited resources, and applications demand varying latency guarantees. Our work on adaptive inference addresses this final barrier, developing frameworks that dynamically adjust model computation based on both input complexity and available resources.

Large language models (LLMs) (OpenAI, 2023; Anthropic, 2025) have recently been extended to connect visual and textual data, giving rise to multimodal large language models (MLLMs). Exemplified by LLaVA (Liu et al., 2023, 2024a) and other works (Liu et al., 2024b; Li et al., 2024, 2023b; Zhu et al., 2024a; Alayrac et al., 2022; Wang et al., 2024b), MLLMs have shown impressive visual reasoning capabilities, but come with significant computational costs. Several efforts have sought to improve the efficiency of MLLMs by exploring lightweight architectures, mixture of experts, or token selection techniques (Zhu et al., 2024a; Yao et al., 2024; Lin et al., 2024; Shang et al., 2024; Chen et al., 2024). However, prior approaches typically exhibit a fixed accuracy and latency footprint during inference, rendering them incapable of adapting to varying compute budget or input content.

We argue that MLLMs with fixed computational footprints are insufficient for real-world deployment. Consider an example of deploying an MLLM on a server farm. Different requests may have different latency requirements, *e.g.*, requests

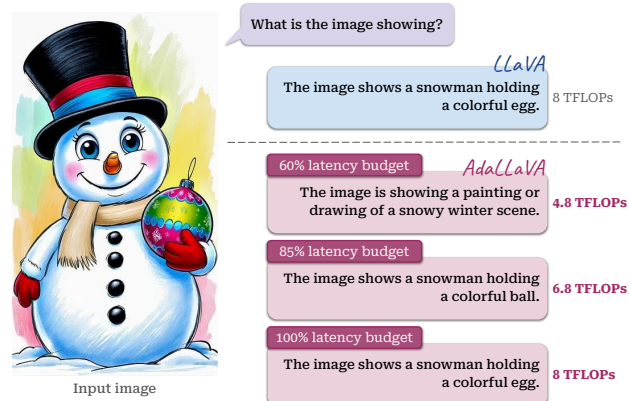
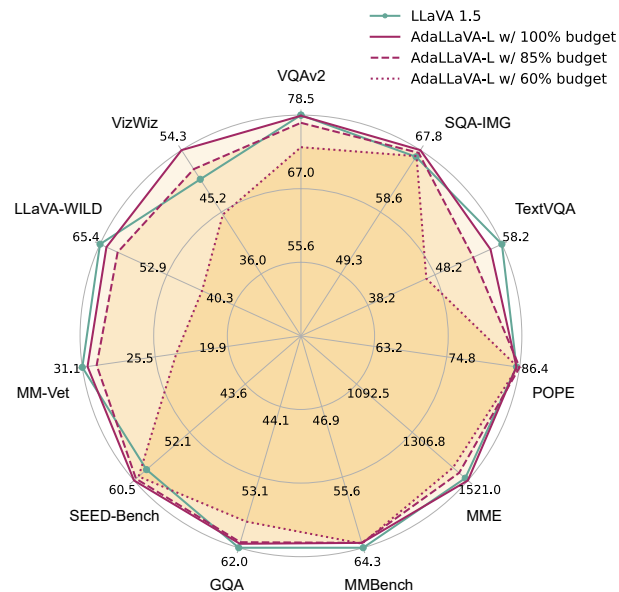


Figure 5.1: **Top:** AdaLLaVA empowers a base LLaVA model with the ability to adapt to varying compute budgets at inference time while maintaining minimal performance loss. **Bottom:** Given an image, a text query and a latency budget, AdaLLaVA learns to reconfigure operations within a base MLLM, generating appropriate responses while sticking to the budget.

from a mobile application require instant feedback to users, while asynchronous processing tasks such as video summarization can tolerate higher latency due to their non-interactive nature. Further, the available computing resources may vary over time as the overall load on the system fluctuates. Similarly, when deployed

on an edge device, the latency budget often remains constant, yet the computing resources may vary due to contention produced by other concurrent programs. In spite of this need, developing adaptive inference strategies for MLLMs that are robust across varying computational budgets (Karayev et al., 2014) remains an open research challenge.

To bridge this gap, we propose *latency-aware adaptive inference for MLLMs*, aiming to dynamically adjust a model’s computational load based on input content and a specified latency or compute budget.¹ This problem is of both conceptual interest and practical significance. Our key insight is that a modern MLLM can be viewed as a collection of shallower models, where choosing among these models enables dynamic reconfiguration during inference. For example, prior works have shown that Transformer blocks in an LLM and some attention heads within these blocks can be bypassed with minimal impact on accuracy, while reducing latency (Song et al., 2024; Cai et al., 2024; Khaki and Plataniotis, 2024). Therefore, strategically selecting these operations during inference results in a set of models with shared parameters but distinct accuracy-latency tradeoffs, allowing the MLLM to flexibly respond to varying latency budgets and content complexity.

To this end, we present **AdaLLaVA**, a learning-based framework for adaptive inference in MLLMs. As shown in fig. 5.1, given an input image, a text query, and a latency budget, AdaLLaVA empowers an MLLM to answer the query about the image while adhering to the specified budget — a capability unattainable with the base MLLM. The key to AdaLLaVA lies in a learned scheduler that dynamically generates an execution plan, selecting a subset of operations within the MLLM based on the input content and a specified latency budget. This execution plan ensures that inference is performed within the given budget while maximizing expected accuracy. To enable effective learning of the scheduler, we introduce a probabilistic formulation in tandem with a dedicated sampling strategy, designed to account for latency constraints at training time.

¹In this work, we measure a model’s latency and its budget using the number of floating-point operations (FLOPs). Thus, the terms “compute budget” and “latency budget” are used interchangeably throughout.

We conduct extensive experiments to evaluate AdaLLaVA. Our results demonstrate that AdaLLaVA can achieve a range of accuracy-latency tradeoffs at runtime. AdaLLaVA exhibits strong adaptability to different latency budgets, effectively trading accuracy for compute during inference. Across several benchmarks, AdaLLaVA retains comparable performance to its base MLLM while operating with higher efficiency (see fig. 5.1). For example, on several comprehensive benchmarks, AdaLLaVA can achieve 99.0% and 98.2% average performance of the baseline LLaVA model when using only 80% and 65% of the latency budget, respectively.

Importantly, it consistently adheres to specified latency constraints and generates content-aware execution plans tailored to input images. Furthermore, we show that AdaLLaVA can be integrated with existing token selection techniques designed to enhance efficiency, making it a versatile solution for adaptive inference in MLLMs.

Our key **contributions** are summarized as follows.

1. We present AdaLLaVA, a novel adaptive inference framework for MLLMs. Our method is among the first to enable dynamic execution of MLLMs based on a latency budget and the input content at inference time.
2. Our key technical innovation lies in (1) the design of a learning-based, latency-aware scheduler, which reconfigures a base MLLM model during inference; and (2) a probabilistic modeling approach, which incorporates hard latency constraints during MLLM training.
3. Through extensive experiments, we demonstrate that (1) AdaLLaVA can adapt to a range of latency requirements while preserving the performance of the base model; and (2) AdaLLaVA can be integrated with token selection techniques to further enhance efficiency.

5.2 Adaptive Inference of MLLMs

We now present **AdaLLaVA**, our adaptive inference framework for MLLMs. Given a latency budget and an input image-query pair at inference time, AdaLLaVA lever-

ages a scheduler learned from data to dynamically reconfigure the execution of MLLMs. Importantly, this scheduler strategically selects a subset of operations to execute, catered to the input budget and content. In doing so, AdaLLaVA ensures that the inference adheres to the latency constraint while preserving model accuracy. fig. 5.2 (a) provides an overview of our framework, where our designed scheduler takes an input of both multimodal sample and latency budget, and outputs an execution plan. In what follows, we introduce the background on MLLMs (section 5.2.1), outline our key idea for scheduling MLLMs (section 5.2.2), present our approach for training and inference with the scheduler (section 5.2.3), and further describe the details of our solution (section 5.2.5).

5.2.1 Preliminaries: MLLMs

An MLLM takes an image (or video) \mathbf{X}^v and a text query $\mathbf{X}^q = \{x^q\}$ as its input, and generates an answer $\mathbf{X}^a = \{x^a\}$ in text format. Specifically, \mathbf{X}^v is first encoded by a visual encoder $h_v(\cdot)$ (including a vision backbone and its projector) into a set of visual tokens $\{\mathbf{z}^v \in \mathbb{R}^d\}$. Similarly, \mathbf{X}^q is processed by a text encoder $h_t(\cdot)$, which embeds the words x^q into a set of text tokens $\{\mathbf{z}^q \in \mathbb{R}^d\}$ with $\mathbf{z}^q = h_t(x^q)$. These tokens are combined into $\{\mathbf{z}^{v|q}\} = [\{\mathbf{z}^v\}, \{\mathbf{z}^q\}]$, and processed by an LLM $f(\cdot)$, which decodes the answer \mathbf{X}^a in an autoregressive manner:

$$f([\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}]; \theta) \rightarrow x_i^a, \quad (5.1)$$

where $\{\mathbf{z}_{<i}^a\}$ are text tokens from previously generated answer $x_{<i}^a$, *i.e.* $\mathbf{z}^a = h_t(x^a)$, and θ denotes LLM parameters.

For the rest of our paper, we will primarily consider the learning of LLM parameters θ —the major portion of parameters within the MLLM. Yet we note that learning encoder parameters (in $h_v(\cdot)$ and $h_t(\cdot)$) can be done similarly.

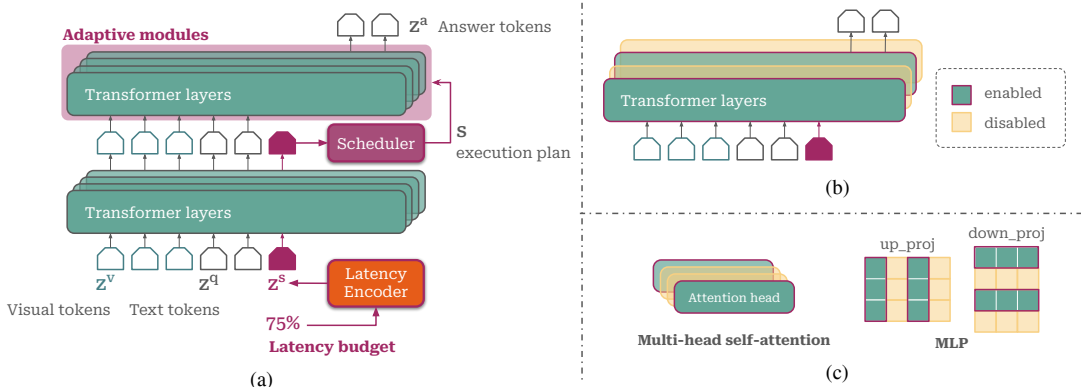


Figure 5.2: Overview of AdaLLaVA. (a) **Model architecture:** Our latency encoder embeds an input latency budget into a latency token, which is further processed by the early part of the LLM. The resulting embedding is then fed into the scheduler, leading to the output of an execution plan that controls individual operations in the remaining part of the LLM. Our latency encoder and scheduler are jointly trained with the MLLM. (b) **AdaLLaVA-L:** the scheduler controls the execution of entire Transformer blocks. (c) **AdaLLaVA-H:** the scheduler controls the execution of attention heads and MLP neurons, by masking out their activation values and the corresponding weights.

5.2.2 Reconfiguring and Scheduling MLLMs

Dynamic reconfiguration. Our key insight is that an MLLM can be conceptualized as a collection of shallower models with shared parameters, each offering a distinct accuracy-latency tradeoff. This perspective enables dynamic reconfiguration of the MLLM during inference to meet varying latency budgets. To this end, we propose equipping the LLM $f(\cdot)$ with K tunable binary switches $\mathbf{s} \in (0, 1)^K$, which control the execution of individual operations, such as Transformer blocks or attention heads, at runtime. Each switch determines whether a specific operation will be executed (1) or skipped (0). We defer the choice of these operations and the design of these switches to our model instantiation. Here, we focus on the concept of reconfigurable LLM decoding, expressed as

$$f([\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}; \theta) \rightarrow x_i^a. \tag{5.2}$$

Specifically, $f(\cdot)$ now takes the switches \mathbf{s} as an additional input, and selectively executes a subset of operations when generating its output. Note that the switches

\mathbf{s} do not depend on the decoding step i , *i.e.*, given the input tokens, a fixed set of operations is applied to generate all output tokens, although the operations may vary for different inputs.

Scheduler. The core of our method is a scheduler $g(\cdot)$ that controls the execution of $f(\cdot)$ during inference. The scheduler $g(\cdot)$ is trained to predict a configuration of switches \mathbf{s} based on the input tokens $\{\mathbf{z}^{v|q}\}$ and an inference latency budget l . This is written as

$$g(\{\mathbf{z}^{v|q}\}, l; \Phi) \rightarrow \mathbf{s}, \quad (5.3)$$

where Φ denotes the parameters of the scheduler $g(\cdot)$.

The goal of $g(\cdot)$ is to determine an execution plan that meets the latency requirement while maximizing the accuracy. This requires solving the following combinatorial optimization problem *for each input sample*:

$$\begin{aligned} \min_{\mathbf{s}} \quad & -\sum_i \log p(x_i^a = f([\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}; \theta)) \\ \text{s.t.} \quad & \text{Latency}(f([\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}; \theta)) \leq l. \end{aligned} \quad (5.4)$$

The objective here is to minimize the negative log likelihood of the target token—the standard loss used for training MLLMs, while the constraint ensures that the latency of executing the model falls within the budget.

5.2.3 Learning to Schedule Execution Plans

Learning the scheduler $g(\cdot)$ poses a major challenge. While it is tempting to pursue a fully supervised approach, in which $g(\cdot)$ is trained to predict the exact solution to eq. (5.4), doing so requires solving the optimization for each sample at every iteration during training. Even with a small number of switches, this is prohibitively expensive.

Deterministic modeling. One possible solution is to solve a relaxed version of the constrained optimization at training time. We initially explored this solution, where we task $g(\cdot)$ to predict a hard execution plan with binary switches \mathbf{s} and

attribute latency violation as part of the objective. This leads to the following loss

$$\arg \min_{\theta, \phi} -\sum_i \log p(x_i^a = f(\cdot)) + \lambda \max(0, \text{Latency}(f(\cdot)) - l),$$

where λ can be treated as the Lagrange multiplier. The execution of the LLM $f(\cdot)$ relies on the output from the scheduler $g(\cdot)$, allowing the joint optimization of $f(\cdot)$ and $g(\cdot)$.

We empirically found that this method fails to enforce a strict latency constraint on the scheduler and often produces suboptimal execution plans that under-utilize the available resources. We demonstrate this limitation through experimental results in section 5.3.3.

Probabilistic modeling. In contrast, we propose a probabilistic model to further relax the constraints, avoiding directly solving eq. (5.4) while stabilizing the joint training of the LLM and the scheduler. Specifically, we task $g(\cdot)$ to model a distribution over the choice of the switches \mathbf{s} , in lieu of making a hard decision:

$$g(\{\mathbf{z}^{v|q}\}, l; \Phi) \rightarrow p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi). \quad (5.5)$$

With slight abuse of notation, we denote $p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$ as the probability over the states \mathbf{s} of K binary switches given the input $\{\mathbf{z}^{v|q}\}$, latency budget l , and the scheduler parameters Φ . Ideally, $p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi) = 0$ if the execution latency of \mathbf{s} exceeds the budget l .

We now re-formulate the inference of MLLM as sampling from the following hierarchical distribution.

$$\begin{aligned} \mathbf{s} &\sim p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi), \\ x_i^a &\sim p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}, \theta). \end{aligned} \quad (5.6)$$

Conceptually, this formulation defines the following generative process: (1) the scheduler g considers the input and a latency budget and outputs the conditional probability of the execution plan $p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$; (2) an execution plan \mathbf{s} is then sampled from the predicted distribution without violating the latency constraint;

and (3) the plan is executed to sequentially decode x_i^a and generate the answer.

Modeling $p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$. Our design requires that the sampled execution plan strictly adheres to the latency budget while maximizing resource utilization. To achieve this, we restrict the support of $p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$ to the states \mathbf{s} that have exactly k activated switches, where k is the maximum number of switches allowed to be turned on without violating l . Specifically, to sample \mathbf{s} , $g(\{\mathbf{z}^{v|q}\}, l; \Phi)$ first outputs a categorical distribution over K available switches. Then, k switches are picked one by one without replacement, following the categorical distribution.

Training loss. The probabilistic model allows us to directly train the LLM and the scheduler with the following loss

$$\arg \min_{\theta, \Phi} \mathbb{E}_{\mathcal{D}} \left[-\log p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], l, \theta, \Phi) \right],$$

where \mathcal{D} is the data distribution approximated by the training set $(\mathbf{X}^v, \mathbf{X}^q, \mathbf{X}^a, l) \sim \mathcal{D}$. By marginalizing \mathbf{s} , we have

$$\begin{aligned} p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], l, \theta, \Phi) &= \\ \mathbb{E}_{p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)} [p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}, \theta)] &. \end{aligned} \quad (5.7)$$

Thus, the loss function is transformed into

$$\arg \min_{\theta, \Phi} \mathbb{E}_{\mathcal{D}, \mathbf{s} \sim p(\mathbf{s}|\cdot)} \left[-\log p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}, \theta) \right],$$

where $p(\mathbf{s}|\cdot) = p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$.

5.2.4 Training and Inference

Approximate training. We present an approximate training scheme in the context of stochastic gradient descent (SGD). Specifically, for each training sample within a mini-batch, a latency budget l is first sampled uniformly from a range of possible budgets, then an execution plan \mathbf{s} is sampled from $p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$. With the

sampled \mathbf{s} guaranteed to satisfy the budget l , the next token x_i^a can be decoded and the log-likelihood $\log p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}, \theta)$ (*i.e.*, the loss) can be readily computed. Optimizing this loss requires backpropagation through the sampling process $\mathbf{s} \sim p(\mathbf{s} | \{\mathbf{z}^{v|q}\}, l, \Phi)$, which we approximate using the Gumbel-Softmax trick Maddison et al. (2017); Jang et al. (2017). See the supplement for more details.

Adaptive inference. During inference, the scheduler outputs the probability $p(\mathbf{s} | \{\mathbf{z}^{v|q}\}, l, \Phi)$ over possible switch configurations \mathbf{s} , given the input $\{\mathbf{z}^{v|q}\}$ and the latency budget l . In theory, decoding the answer \mathbf{X}^a requires marginalizing over this distribution, which is infeasible due to the large number of configurations. In practice, we approximate the inference by selecting the most probable execution plan from the scheduler. This approximation bypasses the marginalization and thus remains highly efficient. We empirically verify its effectiveness. Formally, this approximation is given by

$$\begin{aligned} x_i^a &= \arg \max_{x_i^a} \mathbb{E}_{\mathbf{s} \sim p(\mathbf{s}|\cdot)} [p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}, \theta)] \\ &\approx \arg \max_{x_i^a} p(x_i^a | [\{\mathbf{z}^{v|q}\}, \{\mathbf{z}_{<i}^a\}], \mathbf{s}^*, \theta), \end{aligned}$$

where $\mathbf{s}^* = \arg \max_{\mathbf{s}} p(\mathbf{s} | \{\mathbf{z}^{v|q}\}, l, \Phi)$.

5.2.5 Model Instantiation

Design of tunable switches. We consider attaching binary switches to the LLM part of an MLLM, which accounts for the majority of computational costs. We explore two different designs of switches to select operations.

- **AdaLLaVA-L (layer-level):** This design attaches binary switches to entire Transformer blocks. When a switch is off, the corresponding block is bypassed through its residual connection, becoming an identity mapping. The execution plan thus determines whether each layer is computed or bypassed (see fig. 5.2(b)).

- AdaLLaVA-H (head/neuron-level): This design introduces binary switches within Transformer blocks, targeting individual attention heads in attention modules and specific neurons in MLP layers. When a switch is off, its computation is skipped, and its contribution is removed. In MLP, switches function similarly to dropout Srivastava et al. (2014), selectively disabling neuron activations (see fig. 5.2(c)).

Model architecture. Our goal is to design a lightweight scheduler that minimizes computational overhead yet remains expressive enough to support effective decision-making. To this end, we reuse part of the LLM $f(\cdot)$ to extract visual-language features and encode the latency constraint for the scheduler. Specifically, we first design a latency encoder that converts a latency budget into a token embedding, which is then appended to the original input sequence before being processed by the LLM layers. Within the LLM, the latency token is processed by a few Transformer blocks, attending to all visual-language tokens. The processed token is then passed to a lightweight scheduler that generates the execution plan for the rest of the LLM. Notably, the first few Transformer blocks in the LLM serve two purposes: it simultaneously processes regular MLLM tasks and learns resource allocation based on both content and budget constraints. This design is depicted in fig. 5.2 (a).

Implementation details. Our *latency encoder* uses the sinusoidal positional encoding Waswani et al. (2017) to map the scalar latency l to a 256-D vector. A two-layer MLP, with GELU and layer norm, then converts this vector to a latency token \mathbf{z}^s , ready to be appended to the input sequence of the LLM (see fig. 5.2(a)). Our *scheduler* is implemented as a linear layer that maps the processed latency token (from the bottom part of the LLM Transformer blocks) to logits, defining a categorical distribution over switch selection. We use *FLOPs* to quantify the theoretical latency budget, following the calculation in Yuan et al. (2024). Specifically, we report the average prefill FLOPs on a target dataset, isolating it from variations in decoding length to ensure a more consistent evaluation.

We split the LLM evenly into two parts unless otherwise specified. We use the

first part to process the latency token, and apply tunable switches exclusively to the latter part. In AdaLLaVA-H, we attach a switch to each attention head in the self-attention. For the MLP, channels are grouped to match the number of attention heads, with each group controlled by a single switch. This implementation reduces the design space while preserving control granularity. See ablation study on group size in the supplement.

5.3 Experiments and Results

We now present our experiments and results. We introduce our setup (section 5.3.1), present our main results (section 5.3.2), and provide further analyses (section 5.3.3). Additional experiments, including further ablations, are included in our supplement.

5.3.1 Experimental Setup

Experiment protocol. In most of our experiments, we build on LLaVA-1.5 Liu et al. (2024a). Training LLaVA and many other MLLMs typically involves two stages: (1) vision-language alignment pre-training; and (2) visual instruction tuning. We focus on the second stage and seek to jointly finetune the LLM within the MLLM and train our scheduler using visual instruction data, while keeping the vision encoder frozen. Once trained, we perform zero-shot inference across multiple benchmarks following the common practice in the community Liu et al. (2024a), yet under varying latency budgets.

Training details. Our model is initialized with the pretrained LLaVA-1.5 checkpoints. During finetuning, each training sample is paired with a randomly sampled latency budget ranging from 0.5 to 1.0, as by default we only operate on the top half of the layers in LLM. We set the learning rate to 10^{-5} for the original LLaVA model and the scheduler, while keeping other training hyperparameters consistent with the original LLaVA stage-2 finetuning protocol.

Benchmarks and metrics. We conduct a comprehensive evaluation across multiple visual understanding benchmarks, including VQAv2 Goyal et al. (2017), ScienceQA Lu et al. (2022a), TextVQA Singh et al. (2019), MME (Fu et al., 2023), and MMBench (Liu et al., 2025a). We also evaluate on hallucination benchmarks such as POPE (Li et al., 2023c). For TextVQA, we specifically focus on the image-based subset, where each question is paired with its corresponding image content. For each benchmark, we report the official metrics on the same dataset splits as in LLaVA-1.5. We report accuracy for VQAv2, ScienceQA, TextVQA and MMBench, perception score for MME, and F1 score for POPE. Additionally, we consider varying latency budgets (from 0.5 to 1.0) when evaluating AdaLLaVA. We report the Prefill FLOPs and time on MME benchmark.

Baselines and model variants. We mainly compare our model with base model LLaVA-1.5 Liu et al. (2024a). We evaluate AdaLLaVA with 7B and 13B (see supplement) models, and across two different designs: (a) AdaLLaVA-L for selecting Transformer blocks; and (b) AdaLLaVA-H for selecting attention heads and MLP activations. In our additional analyses, we also use Mipha-3B Zhu et al. (2024b) as the base model.

5.3.2 Main Results

Comparison to baselines. Our main results across six benchmarks are summarized in tables 5.1 to 5.3. AdaLLaVA demonstrates competitive performance with notable efficiency improvements across all benchmarks.

AdaLLaVA-L, when applied to LLaVA-1.5 7B, maintains comparable performance under full computational budgets. With reduced compute budgets, *AdaLLaVA-L* shows minimal performance degradation with an average accuracy drop of only 1.5% at 85% budget and 3.4% at 60% budget. Remarkably, at 60% compute budget, *AdaLLaVA-L* even has slightly better results than the base model on ScienceQA (66.9 vs. 66.8) and POPE (86.1 vs. 85.9).

AdaLLaVA-H shows similar results, with only 1% average performance drop at 85% budget, and 1.9% at 60% budget. The superior performance of *AdaLLaVA-H*

| Method | LLM | Budget (%) | FLOPs (T) | Prefill time (ms) |
|--------------------------------------|-----------|------------|-----------|-------------------|
| LLaVA-1.5 Liu et al. (2024a) | Vicuna-7B | 100 | 8.6 | 81 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 8.6 | 81 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 7.2 | 69 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 5.1 | 49 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 8.6 | 81 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 7.2 | 69 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 5.1 | 49 |
| Prumerge+ Shang et al. (2024) | Vicuna-7B | 100 | 3.0 | 29 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 3.0 | 29 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 2.6 | 24 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 1.8 | 17 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 3.0 | 29 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 2.6 | 24 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 1.8 | 17 |
| FastV (K=2,R=0.5) Chen et al. (2024) | Vicuna-7B | 100 | 4.9 | 47 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 4.9 | 47 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 4.2 | 40 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 3.0 | 29 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 4.9 | 47 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 4.2 | 40 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 3.0 | 29 |

Table 5.1: **Efficiency metrics on MLLM benchmarks.** Budget (%): input latency budget w.r.t. the base model latency. AdaLLaVA-L: switches on selecting different Transformer blocks. AdaLLaVA-H: switches on selecting different attention heads and MLP activations. Prumerge+ and FastV both use LLaVA 1.5.

compared to AdaLLaVA-L can be attributed to its head/neuron-level switching mechanism, allowing for more fine-grained control over computational resources than layer-level switches used in AdaLLaVA-L.

Importantly, for all results, AdaLLaVA adheres to the specified latency budgets (see section 5.3.3). We provide results on additional VQA benchmarks in section C.2 table C.1.

Integration with token selection. Token selection techniques have demonstrated recent success in improving the efficiency of MLLMs Shang et al. (2024); Chen et al. (2024). AdaLLaVA presents an orthogonal direction in adaptive inference.

| Method | LLM | Budget (%) | VQA ^{v2} | SQA ¹ | VQA ^T |
|--------------------------------------|-----------|------------|---------------------|-------------------|---------------------|
| | | | Goyal et al. (2017) | Lu et al. (2022a) | Singh et al. (2019) |
| LLaVA-1.5 Liu et al. (2024a) | Vicuna-7B | 100 | 78.5 | 66.8 | 58.2 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 78.4 | 67.8 | 57.0 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 77.1 | 67.4 | 54.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 75.0 | 66.9 | 47.7 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 77.9 | 68.5 | 57.1 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 76.8 | 68.2 | 55.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 74.2 | 68.1 | 48.7 |
| Prumerge+ Shang et al. (2024) | Vicuna-7B | 100 | 76.8 | 68.3 | 57.1 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 76.3 | 68.3 | 55.8 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 75.3 | 68.5 | 52.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 73.0 | 67.7 | 47.4 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 76.0 | 67.9 | 56.0 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 75.0 | 68.1 | 54.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 72.2 | 67.6 | 47.2 |
| FastV (K=2,R=0.5) Chen et al. (2024) | Vicuna-7B | 100 | 77.7 | 68.7 | 58.1 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 77.8 | 67.7 | 57.0 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 76.9 | 67.8 | 54.4 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 74.5 | 67.0 | 47.4 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 77.4 | 68.4 | 57.0 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 76.6 | 67.7 | 54.8 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 73.9 | 68.3 | 48.7 |

Table 5.2: **VQA benchmark results.** VQA^{v2}: VQAv2 set. SQA¹: ScienceQA set. VQA^T: TextVQA set. AdaLLaVA enables a base MLLM to adapt to varying latency budgets with competitive performance.

We now demonstrate that AdaLLaVA can be integrated with token selection to further enhance the efficiency. We combine AdaLLaVA with PruMerge+ Shang et al. (2024) and FastV Chen et al. (2024), two latest token selection methods designed for MLLMs. For FastV, we set filtering layer K=2 and filtering ratio R=50% to ensure consistent comparison. The results are shown in tables 5.1 to 5.3.

With the integration of PruMerge+ or FastV, AdaLLaVA shows significantly improved efficiency across board, when compared to AdaLLaVA with LLaVA-1.5. Again, AdaLLaVA adapts to varying latency budgets and achieves competitive performance relative to the base model (PruMerge+/FastV). For example, with PruMerge+, AdaLLaVA-H shows 2.45% average performance boost at 85% compute budget and only 1.01% performance drop at 60%. A surprising observation is that AdaLLaVA-H achieves strong performance at 85% latency budget, sometimes beating the base model with token pruning. Overall, our results suggest that AdaLLaVA complements to existing token selection approaches. When integrated

| Method | LLM | Budget (%) | POPE | MME | MMBench |
|--------------------------------------|-----------|------------|-------------------|------------------|--------------------|
| | | | Li et al. (2023c) | Fu et al. (2023) | Liu et al. (2025a) |
| LLaVA-1.5 Liu et al. (2024a) | Vicuna-7B | 100 | 85.9 | 1510.7 | 64.3 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 85.9 | 1521.0 | 63.7 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 86.4 | 1487.2 | 63.7 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 86.1 | 1463.8 | 63.8 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 86.9 | 1471.1 | 64.1 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 86.7 | 1494.9 | 64.3 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 85.0 | 1489.6 | 64.8 |
| Prumerge+ Shang et al. (2024) | Vicuna-7B | 100 | 84.0 | 1462.4 | 64.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 85.1 | 1455.5 | 61.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 85.7 | 1429.5 | 62.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 85.6 | 1450.9 | 61.3 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 86.6 | 1503.2 | 63.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 86.4 | 1511.8 | 63.6 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 86.4 | 1458.0 | 63.6 |
| FastV (K=2,R=0.5) Chen et al. (2024) | Vicuna-7B | 100 | 82.5 | 1516.2 | 64.3 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 82.8 | 1494.3 | 63.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 83.3 | 1478.1 | 63.7 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 83.8 | 1463.1 | 63.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 84.3 | 1484.2 | 63.8 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 83.9 | 1520.5 | 63.9 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 82.4 | 1452.8 | 65.3 |

Table 5.3: **Hallucination and general benchmark results.** POPE evaluates object hallucination. MME and MMBench assess general multimodal understanding. AdaLLaVA can be further integrated with token selection methods to enhance overall efficiency.

with Prumerge+ at an 85% latency budget, our approach reduces computational requirements by 70% while maintaining performance with only 1.7% drop in accuracy.

5.3.3 Additional Analyses

Latency adaptivity. We now evaluate the key capability of AdaLLaVA: its adaptivity to input latency budget, *i.e.*, the ability to complete inference under varying latency requirements using a single model. We report the accuracy-latency tradeoff of AdaLLaVA variants (*i.e.*, Pareto curves), both with and without token selection, on the VQAv2 benchmark. These results are shown in fig. 5.3.

Our results show that AdaLLaVA can empower a base MLLM with static compute footprint (*i.e.*, LLaVA-1.5, PruMerge+, or FastV as individual dots in the fig. 5.3) to adapt to varying accuracy-latency tradeoffs (*i.e.*, the corresponding

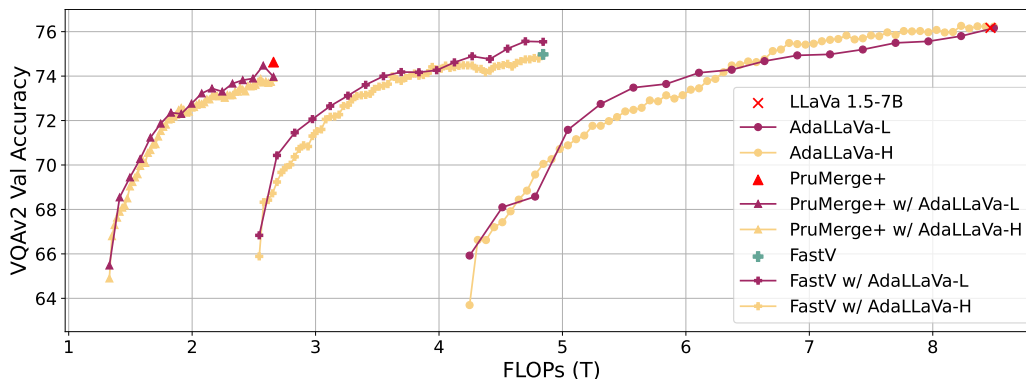


Figure 5.3: **Accuracy-latency tradeoffs of AdaLLaVA** with LLaVA-1.5-7B and additional token selection techniques (PruMerge+ / FastV). Results reported on VQAv2.

curves in fig. 5.3). With varying latency budgets from 50% to 100%, AdaLLaVA effectively trades compute with accuracy. Integrating with token selection methods (PruMerge+ / FastV) further improves the overall efficiency. Thanks to our sampling process in the probabilistic modeling, AdaLLaVA maintains 0% latency violation. We provide additional visualization of execution plans with different latency in fig. C.5 in section C.4.

Content adaptivity. It is worth noting that AdaLLaVA is also adaptive to the input content, *i.e.*, *with the same latency budget, its execution plan is dynamically adjusted based on input*. While not our main focus, we present results to illustrate our model’s content adaptivity, with the aim of providing insights into its behavior and aiding in its diagnosis.

We visualize attention maps from the latency token to all input visual tokens, computed right before the latency token is fed into the scheduler. This is shown in fig. 5.4. These attention maps highlight key regions in the input image for answering the target question. For example, in the top, attention concentrates on “Yes Man” for the movie title question but shifts to the actor name for actor identification question. Further, we visualize the execution plans of different input content given by our scheduler in fig. 5.5. Under the same latency budget, AdaLLaVA generates distinct execution plans conditioned on the different visual content. These results show AdaLLaVA’s ability to dynamically adjust its computational focus based on

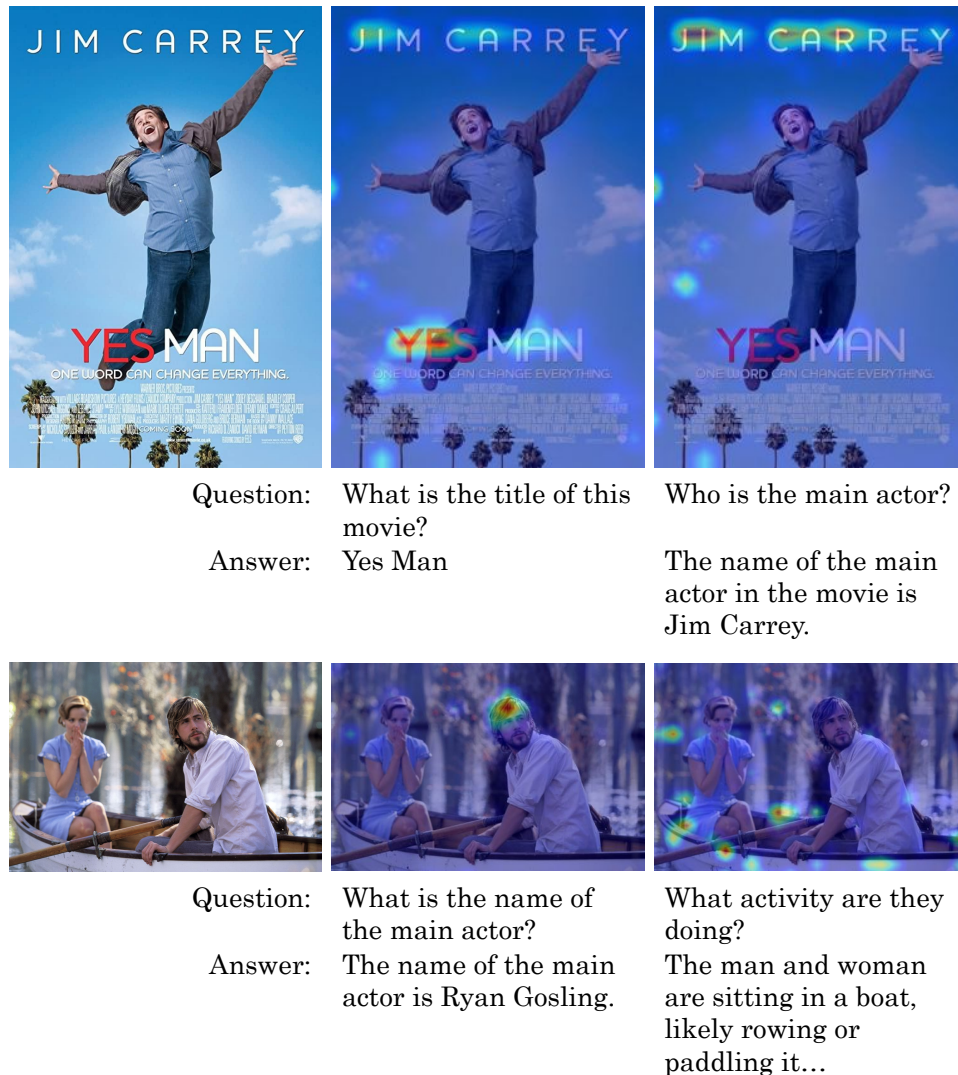


Figure 5.4: **Visualization of attention between the input latency token and visual tokens** with a 100% latency budget.

the input image and text query. See our section C.4 for additional visualizations.

Generalization across MLLMs. We further demonstrate that AdaLLaVA can generalize to other MLLMs beyond LLaVA. We consider Mipha-3B Zhu et al. (2024b), a lightweight MLLM built on Phi-2.7B Javaheripi et al. (2023). Specifically, we apply AdaLLaVA-L on Mipha-3B, following its training strategy Zhu et al. (2024b), and report the results on MME benchmark, shown in table 5.4. These results have

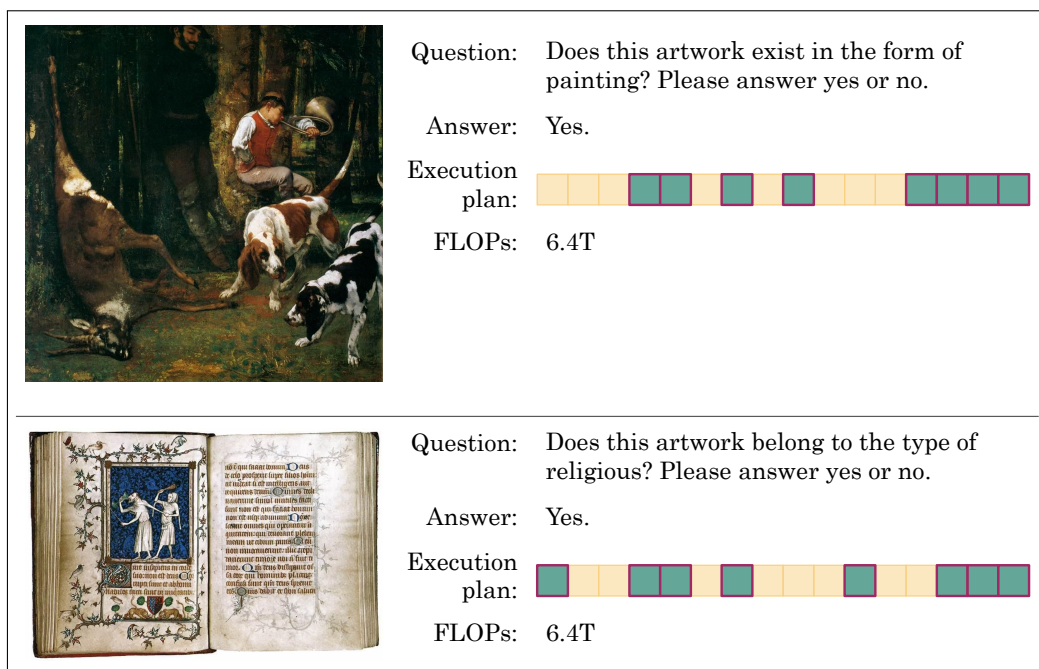


Figure 5.5: **Visualization of execution plans for different input.** The plan is color-coded with **enable** or **disable** for the 16th to 32th Transformer blocks (left to right). The latency budget is 75%.

similar trend to those with LLaVA-1.5 in tables 5.1 to 5.3. Complete results are presented in fig. C.2 in section C.2.

Ablation: probabilistic vs. deterministic modeling of the scheduler. We present two design choices of the scheduler: deterministic and probabilistic (see section 5.2.3). For our main results, we adopt the probabilistic version with conditional sampling (detailed in section 5.2.5). We now compare these two approaches across different latency budgets on the VQAv2 benchmark, using AdaLLaVA-L 7B model. The results are summarized in table 5.5. Our probabilistic model demonstrates superior adaptability across different latency budgets compared to the deterministic approach. We notice deterministic approach has noticeable performance drop given low latency budget due to under-utilization, and sometimes violates the latency budget. These results confirm our choice of the probabilistic modeling.

Additional ablations. Ablations on (1) the number and granularity of switches;

| Model | VQA ^{v2} | SQA ^I | VQA ^T | POPE | MME | MMBench |
|--------------------|-------------------|------------------|------------------|------|--------|---------|
| Mipha-3B | 81.3 | 70.9 | 56.6 | 86.7 | 1488.9 | 69.7 |
| w/ AdaLLaVA-L-100% | 81.1 | 70.9 | 55.3 | 87.7 | 1450.4 | 69.2 |
| w/ AdaLLaVA-L-85% | 80.4 | 71.0 | 53.0 | 87.8 | 1429.3 | 69.0 |
| w/ AdaLLaVA-L-60% | 77.2 | 68.4 | 44.8 | 88.0 | 1397.3 | 64.6 |

Table 5.4: **Generalization** of AdaLLaVA to Mipha-3B.

| Latency budget | AdaLLaVA-L (probabilistic scheduler) | | | AdaLLaVA-L (deterministic scheduler) | | |
|----------------|--------------------------------------|-------------|-----------------|--------------------------------------|-------------|-----------------|
| | Accuracy | Success (%) | Utilization (%) | Accuracy | Success (%) | Utilization (%) |
| 0.95 | 75.6 | 100.0 | 98.7 | 75.6 | 96.1 | 87.6 |
| 0.85 | 74.9 | 100.0 | 99.2 | 74.6 | 100.0 | 80.4 |
| 0.75 | 74.3 | 100.0 | 100.0 | 73.5 | 100.0 | 83.2 |
| 0.65 | 72.7 | 100.0 | 96.5 | 72.2 | 100.0 | 83.1 |

Table 5.5: **Ablation** on deterministic vs. probabilistic modeling for the scheduler. Results reported using 7B model on VQA^{v2}.

(2) different designs of switches (*i.e.*, AdaLLaVA-H vs. AdaLLaVA-L); and (3) sampling strategies are included in section C.3 due to space limit.

5.4 Conclusion and Discussion

In this paper, we introduced AdaLLaVA, a novel adaptive inference framework designed for MLLMs. AdaLLaVA features a lightweight, learning-based scheduler and a probabilistic modeling technique. It empowers a base MLLM with the ability to adapt to varying latency budgets at inference time. Extensive experiments across benchmarks demonstrated that AdaLLaVA is capable of producing latency- and content-aware execution plans, effectively achieving a range of accuracy-latency tradeoffs.

Adaptive inference of MLLMs. Unlike LLMs, MLLMs include a vision encoder and process a large number of redundant visual tokens. While our paper focuses on the scheduling of the LLM component, this adaptivity can be further extended to token selection and vision encoder. We hope this work will be a step toward making

MLLMs more viable for real-world applications where computational resources may be constrained and fluctuate significantly.

Relationship to other efficiency methods. This paper explores adaptive inference in MLLMs, emphasizing adaptability to varying latency budgets within a single model. Our approach is orthogonal to prior methods aimed at improving inference efficiency, such as sparse attention Child et al. (2019) and token selection Shang et al. (2024). Indeed, many of these techniques (*e.g.* token selection as shown in the paper) can be integrated with our framework to further enhance efficiency.

Practical deployment. Our work focuses on algorithm-level innovation, leaving system-level optimization as future work. Conceptually, serving AdaLLaVA is similar to serving MoE-based LLMs Huang et al. (2024), which also dynamically routes tokens to different execution paths based on the input. We express compute budgets as percentages of base model FLOPs to abstract hardware/software variations, leaving cross-device portability to future work. We invite joint effort from the vision, learning, and systems communities to further explore these directions.

This work addresses the challenge of deploying foundation models under varying computational constraints outlined in Chapter 1. By developing a principled framework for dynamic reconfiguration, I demonstrate that strong performance can be maintained while adapting to fluctuating resource budgets at runtime. This chapter forms the final piece of my thesis.

6 CONCLUSION

This dissertation addresses fundamental challenges in making foundation models more practical and capable through three complementary threads: knowledge, reasoning, and adaptivity.

I began by examining how foundation models acquire task-specific knowledge. My theoretical analysis of multitask finetuning (Chapter 3) revealed that diverse task sets significantly improve adaptation performance with limited labeled data. By developing novel diversity and consistency metrics to quantify task relationships, I provided both theoretical justification and a practical task selection algorithm for effective few-shot adaptation. Building on this understanding of knowledge acquisition, I investigated how models combine learned capabilities. Our study of compositional abilities in LLMs (Chapter 4) uncovered distinct patterns: models excel at composite tasks involving distinct input segments but struggle with multi-step reasoning requiring sequential operations. My theoretical analysis explained this behavior, showing that compositional capability emerges when tasks process different input parts separately—an insight with direct implications for understanding model limitations. Finally, I addressed the practical challenge of deploying capable models under real-world constraints. Our adaptive inference framework, AdaLLaVA (Chapter 5), enables multimodal LLMs to dynamically reconfigure their operations based on input content and latency budgets. Through a learned scheduler and probabilistic modeling approach, AdaLLaVA achieves effective accuracy-latency tradeoffs, maintaining comparable performance to full models while operating with significantly higher efficiency.

Together, these research works form a coherent approach to our central goal: transforming general-purpose foundation models into more effective and efficient task-specific experts. This progression from theoretical foundations to practical solutions addresses the critical challenges of adaptation, efficiency, and reasoning that currently limit the practical impact of foundation models.

6.1 Additional collaborative work

While my research primarily focuses customized foundation models, I have also contributed to several related research directions that further our understanding of LLMs. Below, I briefly summarize these works and discuss their connections to the main themes my dissertation.

Understanding Scale Effects in In-Context Learning. In collaborative work (Shi et al., 2024b) investigating why larger language models perform in-context learning differently, we investigated how model scale affects in-context learning behavior in LLMs. Through theoretical analysis and empirical studies, we found that larger and smaller models process in-context examples differently: larger models tend to be more sensitive to noise in test contexts, while smaller models show increased robustness. Our theoretical framework explains this behavior by showing that smaller models focus on key features while larger models distribute attention across a broader range of features, providing new insights into how model scale influences learning dynamics. Our experiments demonstrate how LLMs of varying sizes perform differently in in-context learning tasks. This work provides additional theoretical insights into how model scale affects learning behaviors, complementing our analysis of compositional abilities.

Investigating OOD Generalization Through Induction Heads. I also contributed to research examining out-of-distribution generalization in transformers through the lens of induction heads (Song et al., 2025). This work investigated the role of induction heads in transformer models through systematic ablation experiments. Using synthetic in-context learning tasks designed to test compositional reasoning, we examined how models perform when induction heads are removed or shuffled between layers. These experiments provided empirical evidence that induction heads are critical for compositional learning and out-of-distribution generalization. The experimental results showed clear performance degradation on compositional tasks when induction head functionality was disrupted, supporting the theoretical

framework about how transformers compose information across attention layers to achieve generalization.

In-Context Skill Composition. Extending our compositional reasoning analysis (Chapter 4), I contributed to research investigating whether language models can compose skills demonstrated only through in-context examples (Liu et al., 2025b). Through systematic experiments, we uncovered a surprising finding that goes beyond our previous observations—providing simple task examples can have a *negative impact* on composition performance, contrary to the intuition that such demonstrations should help models identify and combine relevant skills. Even when Chain-of-Thought examples are provided, models fail to correctly align the demonstrated skills with the appropriate reasoning steps, often mismatching skills to wrong steps in the composition. Our theoretical analysis reveals that this failure stems from models’ inability to recognize compositional structure and align examples appropriately, suggesting that the bottleneck lies not merely in executing multi-step reasoning but in identifying when and how demonstrated skills should be applied. These insights deepen our understanding of compositional limitations and point toward the importance of explicit step-alignment in prompting strategies.

6.2 Future Research Directions

An emerging paradigm in AI centers on autonomous agents that can plan, reason, and act over extended time horizons. Such agentic systems leverage foundation models, especially LLMs as their cognitive backbone to perform complex tasks autonomously, from conducting multi-step research and writing code to interacting with external tools and services on behalf of users. As foundation models increasingly serve this role, the questions of adaptation, composition, and efficiency become even more critical. The techniques developed throughout this dissertation address precisely these challenges: adapting models to specialized tasks, understanding their reasoning capabilities, and deploying them efficiently under resource constraints. These capabilities form essential building blocks for effective agentic

systems. Looking ahead, our findings point to several promising research directions at this intersection.

Memory and Adaptive Retrieval for Agents. A critical frontier involves developing memory systems that enable AI agents to maintain coherent long-term behavior. Agent memory refers to mechanisms that allow system to store, retrieve, and reason over information accumulated across interactions, moving beyond the fixed context window limitations of current LLMs. Current approaches demonstrate the potential to explicitly manage LLMs' context, but fundamental questions remain about how to structure memory for effective retrieval and reasoning over extended time horizons. Future work should examine the interplay between different memory types: working memory for immediate task context, episodic memory for conversation histories, and semantic memory for world knowledge (Chhikara et al., 2025; Rasmussen et al., 2025; Packer et al., 2023). Closely related is the challenge of adaptive retrieval. While retrieval-augmented generation (RAG) has become standard for grounding model outputs, future agentic systems require retrieval policies that can evolve. Research is needed to develop systems that can decide what to retrieve from memory or external sources, how much information to gather, and when retrieval is unnecessary based on task difficulty or uncertainty (Edge et al., 2024; Guo et al., 2025b). Extending these capabilities to multimodal settings, where agents must retrieve and reason over images, videos, and audio, introduces additional challenges in multimodal embedding, cross-modal relevance ranking, and latency-efficient processing for real-time applications (Guo et al., 2025a).

Reinforcement Learning and Alignment for Agentic Systems. Training agents capable of planning, tool use, and multi-step task execution requires fundamentally different approaches than current alignment methods. While existing techniques like RLHF optimize responses at the question-answer level, agentic systems such as deep research agents operate over extended trajectories spanning dozens of decisions, from query formulation to source selection to synthesis (OpenAI, 2025; Team et al., 2025). This shift from single-turn feedback to trajectory-level rewards

introduces severe credit assignment challenges: determining which intermediate decisions contributed to success or failure across a long action sequence is computationally intractable. Future work we can explore process reward models that provide dense feedback on intermediate steps and techniques for decomposing complex workflows into components where credit can be reliably assigned. Additionally, as agents gain autonomy to execute code and interact with external services, research is needed to develop safety monitoring systems that detect deviation from intended behavior before errors propagate through long action sequences.

Efficient Inference and Scalable Deployment. As multimodal foundation models become larger and more widely deployed, future work must address efficiency at both the model and system levels. At the model and algorithm level, promising directions include model compression and adaptive computation—such as quantization, pruning, distillation, dynamic depth, and basic KV-cache reuse—that reduce per-token cost without degrading accuracy. At the system level, research is needed on hardware-aware kernels and serving infrastructure, including FlashAttention-style implementations, efficient batching and scheduling for heterogeneous requests, and KV-cache management strategies (*e.g.*, paging, sharing, and offloading across devices). These algorithm–systems co-design problems become even more challenging in multimodal settings, where text, vision, and other modalities must share limited hardware memory and compute budgets. Developing principled methods to jointly optimize accuracy, latency, and cost under realistic workload distributions will be essential for making state-of-the-art foundation model practical in real-time and resource-constrained environments.

A APPENDIX OF CHAPTER 3

In this appendix, we state our limitation in Section A.1. The proof of our theoretical results for the binary case is presented in Section A.2, where we formalize the theoretical settings and assumptions and elaborate on the results to contrastive pretraining in Section A.2.1 and supervised pretraining in Section A.2.2. We prove the main theory in Section A.2.4, which is a direct derivative of A.2.1 and A.2.2. We generalize the setting to multiclass and provide proof in Section A.3. We include the full proof of the general linear case study in Section A.4. We provide additional experimental results of vision tasks in Section A.5, language tasks in Section A.6, and vision-language tasks in section A.7.

A.1 Limitation

We recognize an interesting phenomenon within multitask finetuning and dig into deeper exploration with theoretical analysis, while our experimental results may or may not beat state-of-the-art (SOTA) performance, as our focus is not on presenting multitask finetuning as a novel approach nor on achieving SOTA performance. On the other hand, the estimation of our diversity and consistency parameters accurately on real-world datasets is valuable but time-consuming. Whether there exists an efficient algorithm to estimate these parameters is unknown. We leave this challenging problem as our future work.

A.2 Deferred Proofs

In this section, we provide a formal setting and proof. We first formalize our setting in multiclass. Consider our task \mathcal{T} contains r classes where $r \geq 2$.

Contrastive Learning. In contrastive learning, we sampled one example x from any latent class y , then apply the data augmentation module that randomly trans-

forms such sample into another view of the original example denoted x^+ . We also sample other $r - 1$ examples $\{x_k^-\}_{k=1}^r$ from other latent classes $\{y_k^-\}_{k=1}^{r-1}$. We treat (x, x^+) as a positive pair and (x, x_k^-) as negative pairs. We define $\mathcal{D}_{\text{con}}(\eta)$ over sample $(x, x^+, x_1^-, \dots, x_{r-1}^-)$ by following sampling procedure

$$(y, y_1^-, \dots, y_{r-1}^-) \sim \eta^r \quad (\text{A.1})$$

$$x \sim \mathcal{D}(y), x^+ \sim \mathcal{D}(y), x_k^- \sim \mathcal{D}(y_k^-), k = 1, \dots, r - 1. \quad (\text{A.2})$$

We consider general contrastive loss $\ell_u \left(\left\{ \phi(x)^\top (\phi(x^+) - \phi(x_k^-)) \right\}_{k=1}^{r-1} \right)$, where loss function ℓ_u is non-negative decreasing function. Minimizing the loss is equivalent to maximizing the similarity between positive pairs while minimizing it between negative pairs. In particular, logistic loss $\ell_u(\mathbf{v}) = \log(1 + \sum_i \exp(-\mathbf{v}_i))$ for $\mathbf{v} \in \mathbb{R}^{r-1}$ recovers the one used in most empirical works:

$$-\log \left(\frac{\exp \{ \phi(x)^\top \phi(x^+) \}}{\exp \{ \phi(x)^\top \phi(x^+) \} + \sum_{i=1}^{r-1} \exp \{ \phi(x)^\top \phi(x_i^-) \}} \right).$$

The population contrastive loss is defined as

$$\mathcal{L}_{\text{con-pre}}(\phi) := \mathbb{E} \left[\ell_u \left(\left\{ \phi(x)^\top (\phi(x^+) - \phi(x_k^-)) \right\}_{k=1}^{r-1} \right) \right].$$

Let $\mathcal{S}_{\text{con-pre}} := \left\{ x_j, x_j^+, x_{j1}^-, \dots, x_{j(r-1)}^- \right\}_{j=1}^N$ denote our contrastive training set with N samples, sampled from $\mathcal{D}_{\text{con}}(\eta)$, we have empirical contrastive loss

$$\widehat{\mathcal{L}}_{\text{con-pre}}(\phi) := \frac{1}{N} \sum_{i=1}^N \left[\ell_u \left(\left\{ \phi(x)^\top (\phi(x^+) - \phi(x_k^-)) \right\}_{k=1}^{r-1} \right) \right].$$

Supervised Learning. In supervised learning we have a labeled dataset denoted as $\mathcal{S}_{\text{con-pre}} := \{x_j, y_j\}_{j=1}^N$ with N samples, by following sampling procedure:

$$y \sim \eta \quad (\text{A.3})$$

$$x \sim \mathcal{D}(y). \quad (\text{A.4})$$

There are in total K classes, denote \mathcal{C} as the set consists of all classes. On top of the representation function ϕ , there is a linear function $f \in \mathcal{F} \subset \{\mathbb{R}^d \rightarrow \mathbb{R}^K\}$ predicting the labels, denoted as $g(x) = f \circ \phi(x)$. We consider general supervised loss on data point (x, y) is

$$\ell(g(x), y) := \ell_u((g(x))_y - (g(x))_{y' \neq y, y' \in \mathcal{C}}). \quad (\text{A.5})$$

where loss function ℓ_u is non-negative decreasing function. In particular, logistic loss $\ell_u(\mathbf{v}) = \log(1 + \sum_i \exp(-\mathbf{v}_i))$ for $\mathbf{v} \in \mathbb{R}^{K-1}$ recovers the one used in most empirical works:

$$\ell(g(x), y) = \ell_u((g(x))_y - (g(x))_{y' \neq y, y' \in \mathcal{C}}) \quad (\text{A.6})$$

$$= \log \left\{ 1 + \sum_{k \neq y}^K \exp(-[(g(x))_y - (g(x))_k]) \right\} \quad (\text{A.7})$$

$$= -\log \left\{ \frac{\exp(g(x))_y}{\sum_{k=1}^K \exp(g(x))_k} \right\}. \quad (\text{A.8})$$

The population supervised loss is

$$\mathcal{L}_{\text{sup-pre}}(\phi) = \min_{f \in \mathcal{F}} \mathbb{E}_{x, y} [\ell(f \circ \phi(x), y)]. \quad (\text{A.9})$$

For training set $\mathcal{S}_{\text{sup-pre}} := \{x_i, y_i\}_{i=1}^N$ with N samples, the empirical supervised pretraining loss is $\widehat{\mathcal{L}}_{\text{sup-pre}}(\phi) := \min_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^N [\ell(f \circ \phi(x_i), y_i)]$.

Masked Language Modeling. Masked language modeling is a self-supervised learning method. It can be viewed as a specific form of supervised pretraining above. The pretraining data is a substantial dataset of sentences, often sourced from Wikipedia. In the pretraining phase, a random selection of words is masked within each sentence, and the training objective is to predict these masked words using the context provided by the remaining words in the sentence. This particular pretraining task can be viewed as a multi-class classification problem, where the number of classes (denoted as K) corresponds to the size of the vocabulary. Considering BERT and its variations, we have function ϕ as a text encoder. This encoder outputs a learned representation, often known as [CLS] token. The size of such learned representation is d , which is 768 for BERT_{BASE} or 1024 for BERT_{LARGE}.

Supervised Tasks. Given a representation function ϕ , we apply a task-specific linear transformation W to the representation to obtain the final prediction. Consider r -way supervised task \mathcal{T} consist a set of distinct classes $(y_1, \dots, y_r) \subseteq \mathcal{C}$. We define $\mathcal{D}_{\mathcal{T}}(y)$ as the distribution of randomly drawing $y \in (y_1, \dots, y_r)$, we denote this process as $y \sim \mathcal{T}$. Let $\mathcal{S}_{\mathcal{T}} := \{x_j, y_j\}_{j=1}^m$ denote our labeled training set with m samples, sampled i.i.d. from $y_j \sim \mathcal{T}$ and $x_j \sim \mathcal{D}(y_j)$. Define $g(\phi(\mathbf{x})) := W\phi(\mathbf{x}) \in \mathbb{R}^r$ as prediction logits, where $W \in \mathbb{R}^{r \times d}$. The typical supervised logistic loss is $\ell(g \circ \phi(\mathbf{x}), y) := \ell_u(\{g(\phi(\mathbf{x}))_y - g(\phi(\mathbf{x}))_{y'}\}_{y' \neq y})$. Similar to Arora et al. (2019), define supervised loss w.r.t the task \mathcal{T}

$$\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi) := \min_{W \in \mathbb{R}^{r \times d}} \mathbb{E}_{y \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(W \cdot \phi(x), y)]. \quad (\text{A.10})$$

Define supervised loss with mean classifier as

$$\mathcal{L}_{\text{sup}}^{\mu}(\mathcal{T}, \phi) := \mathbb{E}_{y \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(W^{\mu} \cdot \phi(x), y)],$$

where each row of W^{μ} is the mean of each class in \mathcal{T} , $W_{y_k}^{\mu} := \mu_{y_k} = \mathbb{E}_{x \sim y_k}(\phi(x))$, $k = 1, \dots, r$. In the target task, suppose we have r distinct classes from \mathcal{C} with equal weights. Consider \mathcal{T} follows a general distribution ζ . Define expected supervised

Algorithm 2 Multitask Finetuning

Input: Multitasks $\mathcal{T}_1, \dots, \mathcal{T}_M$, pretrained model $\hat{\phi}$ with parameter θ , step size γ

- 1: Initialize ϕ with $\hat{\phi}$
- 2: **repeat**
- 3: **for all** \mathcal{T}_i **do**
- 4: $\theta \leftarrow \theta - \gamma \nabla_{\theta} \widehat{\mathcal{L}}_{\text{sup}}(\mathcal{T}_i, \phi)$ $\triangleright \widehat{\mathcal{L}}_{\text{sup}}(\mathcal{T}_i, \phi)$ is defined in (3.2)
- 5: **end for**
- 6: **until** converge

Output: The final model, denoted as ϕ'

loss as $\mathcal{L}_{\text{sup}}(\phi) := \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)]$.

Multitask Finetuning. Suppose we have M auxiliary tasks $\{\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_M\}$, each with m labeled samples $\mathcal{S}_i := \{(x_j^i, y_j^i) : j \in [m]\}$. The finetuning data are $\mathcal{S} := \cup_{i \in [M]} \mathcal{S}_i$. Given a pretrained model $\hat{\phi}$, we further finetune it using the objective:

$$\min_{\phi \in \Phi} \frac{1}{M} \sum_{i=1}^M \widehat{\mathcal{L}}_{\text{sup}}(\mathcal{T}_i, \phi), \quad \text{where } \widehat{\mathcal{L}}_{\text{sup}}(\mathcal{T}_i, \phi) := \min_{\mathbf{w}_i \in \mathbb{R}^d} \frac{1}{m} \sum_{j=1}^m \ell(\mathbf{w}_i^\top \phi(x_j^i), y_j^i). \quad (\text{A.11})$$

This can be done via gradient descent from the initialization $\hat{\phi}$ (see Algorithm 2).

Algorithm 2 has similar pipeline as Raghu et al. (2020) where in the inner loop only a linear layer on top of the embeddings is learned. However, our algorithm is centered on multitask finetuning, where no inner loop is executed.

Finally, we formalize our assumption Assumption 3.3.1 below.

Assumption A.2.1 (Regularity Conditions). *The following regularity conditions hold:*

label=(A1) Representation function ϕ satisfies $\|\phi\|_2 \leq R$.

lbbel=(A2) Linear operator W satisfies bounded spectral norm $\|W\|_2 \leq B$.

lcbel=(A3) The loss function ℓ_u are bounded by $[0, C]$ and $\ell(\cdot)$ is L -Lipschitz.

ldbel=(A4) The supervised loss $\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)$ is \tilde{L} -Lipschitz with respect to ϕ for $\forall \mathcal{T}$.

A.2.1 Contrastive Pretraining

In this section, we will show how multitask finetuning improves the model from contrastive pretraining. We present pretraining error in binary classification and $\mathcal{D}_{\mathcal{T}}(\mathbf{y})$ as uniform. See the result for the general condition with multi-class in Section A.3.

Contrastive Pretraining and Direct Adaptation

In this section, we show the error bound of a foundation model on a target task, where the model is pretrained by contrastive loss followed directly by adaptation.

We first show how pretraining guarantees the expected supervised loss:

$$\mathcal{L}_{\text{sup}}(\phi) = \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)]. \quad (\text{A.12})$$

The error on the target task can be bounded by $\mathcal{L}_{\text{sup}}(\phi)$. We use ϵ^* denote $\mathcal{L}_{\text{sup}}(\phi_{\zeta}^*)$.

Lemma A.2.2 (Lemma 4.3 in Arora et al. (2019)). *For $\forall \phi \in \Phi$ pretrained in contrastive loss, we have $\mathcal{L}_{\text{sup}}(\phi) \leq \frac{1}{1-\tau}(\mathcal{L}_{\text{con-pre}}(\phi) - \tau)$.*

We state the theorem below.

Theorem A.2.3. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^*, ϕ_{ζ}^* . Suppose $\hat{\phi}$ satisfies $\hat{\mathcal{L}}_{\text{con-pre}}(\hat{\phi}) \leq \epsilon_0$. Let $\tau := \Pr_{(y_1, y_2) \sim \eta^2} \{y_1 = y_2\}$. Consider pretraining set $\mathcal{S}_{\text{con-pre}} = \{x_j, x_j^+, x_j^-\}_{j=1}^N$. For any $\delta \geq 0$, if*

$$N \geq \frac{1}{\epsilon_0} \left[8\text{LR}\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right].$$

Then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subset \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\frac{1}{1-\tau} (2\epsilon_0 - \tau) - \mathcal{L}_{\text{sup}}(\phi^*) \right] + \kappa. \quad (\text{A.13})$$

The pretraining sample complexity is $O(\frac{\mathcal{R}_N(\Phi)}{\epsilon_0} + \frac{\log(1/\delta)}{\epsilon_0^2})$. The first term is the Rademacher complexity of the entire representation space Φ with sample size N . The second term relates to the generalization bound. Pretraining typically involves a vast and varied dataset, sample complexity is usually not a significant concern during this stage.

Proof of Theorem A.2.3. Recall in binary classes, $\mathcal{S}_{\text{con-pre}} = \{x_j, x_j^+, x_j^-\}_{j=1}^N$ denote our contrastive training set, sampled from $\mathcal{D}_{\text{con}}(\eta)$. Then by Lemma A.2 in Arora et al. (2019), with a and c , we have for $\forall \phi \in \Phi$ with probability $1 - \delta$,

$$\mathcal{L}_{\text{con-pre}}(\phi) - \hat{\mathcal{L}}_{\text{con-pre}}(\phi) \leq \frac{4LR\mathcal{R}_N(\Phi)}{N} + C\sqrt{\frac{\log \frac{1}{\delta}}{N}}. \quad (\text{A.14})$$

To have above $\leq \epsilon_0$, we have sample complexity

$$N \geq \frac{1}{\epsilon_0} \left[8LR\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right].$$

In pretraining, we have $\hat{\phi}$ such that

$$\hat{\mathcal{L}}_{\text{con-pre}}(\hat{\phi}) \leq \epsilon_0.$$

Then with the above sample complexity, we have pretraining $\hat{\phi}$

$$\mathcal{L}_{\text{con-pre}}(\hat{\phi}) \leq 2\epsilon_0.$$

Recall ν -diversity and κ -consistency, for target task \mathcal{T}_0 , we have that for $\hat{\phi}$ and

ϕ^* ,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) = \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) \quad (\text{A.15})$$

$$+ \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \quad (\text{A.16})$$

$$\leq d_{\mathcal{E}_0}(\hat{\phi}, \phi_\zeta^*) + \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \quad (\text{A.17})$$

$$\leq \bar{d}_\zeta(\hat{\phi}, \phi_\zeta^*)/\nu + \kappa \quad (\text{A.18})$$

$$\leq \frac{1}{\nu} \left[\mathcal{L}_{\text{sup}}(\hat{\phi}) - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa \quad (\text{A.19})$$

$$= \frac{1}{\nu} \left[\frac{1}{1-\tau} (\mathcal{L}_{\text{con-pre}}(\hat{\phi}) - \tau) - \epsilon^* \right] + \kappa \quad (\text{A.20})$$

$$\leq \frac{1}{\nu} \left[\frac{1}{1-\tau} (2\epsilon_0 - \tau) - \epsilon^* \right] + \kappa, \quad (\text{A.21})$$

where the second to last inequality comes from Theorem A.2.2. \square

Contrastive Pretraining and Multitask Finetuning

In this section, we show the error bound of a foundation model on a target task can be further reduced by multitask finetuning. We achieve this by showing that expected supervised loss $\mathcal{L}_{\text{sup}}(\phi)$ can be further reduced after multitask finetuning. The error on the target task can be bounded by $\mathcal{L}_{\text{sup}}(\phi)$. We use ϵ^* denote $\mathcal{L}_{\text{sup}}(\phi_\zeta^*)$.

Following the intuition in Garg and Liang (2020), we first re-state the definition of representation space.

Definition A.2.4. *The subset of representation space is*

$$\Phi(\tilde{\epsilon}) = \left\{ \phi \in \Phi : \hat{\mathcal{L}}_{\text{pre}}(\phi) \leq \tilde{\epsilon} \right\}.$$

Recall $\mathcal{S} = \{(x_j^i, y_j^i) : i \in [M], j \in [m]\}$ as finetuning dataset.

We define two function classes and associated Rademacher complexity.

Definition A.2.5. *Consider function class*

$$\mathcal{G}_\ell(\tilde{\epsilon}) = \left\{ g_{W,\phi}(x, y) : g_{W,\phi}(x, y) = \ell(W\phi(x_j^i), y_j^i), \phi \in \Phi(\tilde{\epsilon}), \|W\|_2 \leq B \right\}.$$

We define Rademacher complexity as

$$\mathcal{R}_n(\mathcal{G}_\ell(\tilde{\epsilon})) = \mathbb{E}_{\{\sigma_i\}_{i=1}^n, \{\mathbf{x}_j, \mathbf{y}_j\}_{j=1}^n} \left[\sup_{\ell \in \mathcal{G}_\ell(\tilde{\epsilon})} \sum_{j=1}^n \sigma_j \ell(W \cdot \Phi(\mathbf{x}_j), \mathbf{y}_j) \right].$$

Definition A.2.6. Consider function class

$$\mathcal{G}(\tilde{\epsilon}) = \{g_\phi : g_\phi(\mathcal{T}) = \mathcal{L}_{\text{sup}}(\mathcal{T}, \phi), \phi \in \Phi(\tilde{\epsilon})\}.$$

We define Rademacher complexity as

$$\mathcal{R}_M(\mathcal{G}(\tilde{\epsilon})) = \mathbb{E}_{\{\sigma_i\}_{i=1}^M, \{\mathcal{T}_i\}_{i=1}^M} \left[\sup_{\phi \in \Phi(\tilde{\epsilon})} \sum_{i=1}^M \sigma_i \mathcal{L}_{\text{sup}}(\mathcal{T}_i, \phi) \right].$$

The key idea is multitask finetuning further reduce the expected supervised loss of a pretrained foundation model ϕ :

$$\mathcal{L}_{\text{sup}}(\phi) = \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)]. \quad (\text{A.22})$$

We first introduce some key lemmas. These lemmas apply to general r classes in a task \mathcal{T} .

Lemma A.2.7 (Bounded Rademacher complexity). *By b and c , we have for $\forall n$*

$$\mathcal{R}_n(\mathcal{G}_\ell(\tilde{\epsilon})) \leq 4\sqrt{r-1} \text{LB} \mathcal{R}_n(\Phi(\tilde{\epsilon})).$$

Proof of Theorem A.2.7. We first prove $\ell(g(\phi(\mathbf{x})), \mathbf{y})$ is $\sqrt{2(r-1)} \text{LB}$ -Lipschitz with respect to ϕ for all $\forall \mathbf{y} \in \mathcal{C}$. Consider

$$f_{\mathbf{y}}(g(\phi(\mathbf{x}))) = \{g(\phi(\mathbf{x}))_{\mathbf{y}} - g(\phi(\mathbf{x}))_{\mathbf{y}'}\}_{\mathbf{y}' \neq \mathbf{y}},$$

where $f_y : \mathbb{R}^r \rightarrow \mathbb{R}^{r-1}$. Note that

$$\begin{aligned} \ell(g \circ \phi(\mathbf{x}), \mathbf{y}) &= \ell\left(\{g(\phi(\mathbf{x}))_{\mathbf{y}} - g(\phi(\mathbf{x}))_{\mathbf{y}' \neq \mathbf{y}}\}\right) \\ &= \ell(f_y(g(\phi(\mathbf{x}))). \end{aligned}$$

By c , we have ℓ is L -Lipschitz. We then prove f_y is $\sqrt{2(r-1)}$ -Lipschitz. Without loss generality, consider $\mathbf{y} = r$. We have $f_y(\mathbf{y}) = [y_r - y_i]_{i=1}^{r-1}$. We have $\frac{\partial f_j}{\partial y_i} = -\mathbb{1}\{j = i\}$, $i = 1, \dots, r-1$, $\frac{\partial f_j}{\partial y_r} = 1$. The Jacobian J satisfies $\|J\|_2 \leq \|J\|_F = \sqrt{2(r-1)}$.

Since g is B -Lipschitz by $b: \|W\|_2 \leq B$. Then $\ell(g(\phi(\mathbf{x})), \mathbf{y})$ is $\sqrt{2(r-1)}LB$ -Lipschitz with respect to ϕ for all $\forall \mathbf{y} \in \mathcal{C}$. The conclusion follows Corollary 4 in Maurer (2016). \square

Lemma A.2.8 (Bounded $\tilde{\epsilon}$). *After finite steps in Multitask finetuning in Algorithm 2, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} \frac{1}{1-\tau} (2\epsilon_0 - \tau)$ and obtain ϕ' . Then there exists a bounded $\tilde{\epsilon}$ such that $\phi' \in \Phi(\tilde{\epsilon})$.*

Proof of Theorem A.2.8. Given finite number of steps and finite step size γ in Algorithm 2, we have bounded $\|\phi' - \hat{\phi}\|$. Then with b and c , using Theorem A.2.7 we have $\ell(g(\phi(\mathbf{x})), \mathbf{y})$ is $\sqrt{2(r-1)}LB$ -Lipschitz with respect to ϕ for all $\forall \mathbf{y}$, using theorem A.2 in Arora et al. (2019) we have l_u is LC -Lipschitz with respect to ϕ , we have $\hat{\mathcal{L}}_{\text{pre}}(\phi)$ is M -Lipschitz with respect to ϕ with bounded M . We have $\exists \epsilon$ such that $\hat{\mathcal{L}}_{\text{pre}}(\phi') - \hat{\mathcal{L}}_{\text{pre}}(\hat{\phi}) \leq \epsilon \|\phi' - \hat{\phi}\|$. We have $\hat{\mathcal{L}}_{\text{pre}}(\phi') \leq \epsilon_0 + \epsilon \|\phi' - \hat{\phi}\|$. Take $\tilde{\epsilon} = \epsilon_0 + \epsilon \|\phi' - \hat{\phi}\|$ yields the result. \square

Lemma A.2.9. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^*, ϕ_ζ^* . Suppose for some small constant $\alpha \in (0, 1)$ and $\tilde{\epsilon}$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} \frac{1}{1-\tau} (2\epsilon_0 - \tau)$ and obtain ϕ' . For any $\delta > 0$, if*

$$\begin{aligned} M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{L}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ Mm &\geq \frac{1}{\epsilon_1} \left[8\sqrt{r-1}LB\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \end{aligned}$$

then expected supervised loss $\mathcal{L}_{\text{sup}}(\phi') \leq \alpha \frac{1}{1-\tau} (2\epsilon_0 - \tau)$, with probability $1 - \delta$.

Proof of Theorem A.2.9. Recall $\mathcal{S} := \{(x_j^i, y_j^i) : i \in [M], j \in [m]\}$ as finetuning dataset. Consider in Equation (3.2) we have $\widehat{\mathbf{W}} := (\widehat{W}_1, \dots, \widehat{W}_M)$ and ϕ' such that

$$\frac{1}{M} \sum_{i=1}^M \frac{1}{m} \sum_{j=1}^m \ell(\widehat{W}_i \cdot \phi'(x_j^i), y_j^i) \leq \epsilon_1 < \frac{\alpha}{3} \epsilon_0.$$

We tried to bound

$$\mathcal{L}_{\text{sup}}(\phi') - \frac{1}{m} \sum_{j=1}^m \ell(\widehat{W}_i \cdot \phi'(x_j^i), y_j^i).$$

Recall that

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_i, \phi) = \min_{W \in \mathbb{R}^{r \times d}} \mathbb{E}_{y \sim \mathcal{T}_i} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(W \cdot \phi(x), y)].$$

For $\forall \phi \in \Phi(\tilde{\epsilon})$

$$\mathcal{L}_{\text{sup}}(\phi) = \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)] = \mathbb{E}_{\mathcal{T} \sim \zeta} \left[\min_{W \in \mathbb{R}^{r \times d}} \mathbb{E}_{y \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(W \cdot \phi(x), y)] \right].$$

We have for $\forall \phi \in \Phi(\tilde{\epsilon})$, by uniform convergence (see Mohri et al. (2018) Theorem 3.3), we have with probability $1 - \delta/2$

$$\mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)] - \frac{1}{M} \sum_{i=1}^M \mathcal{L}_{\text{sup}}(\mathcal{T}_i, \phi) \leq \frac{2\mathcal{R}_M(\mathcal{G}(\tilde{\epsilon}))}{M} + \sqrt{\frac{\log(2/\delta)}{M}} \quad (\text{A.23})$$

$$\leq \frac{2\sqrt{2}\tilde{\mathcal{L}}\mathcal{R}_M(\Phi(\tilde{\epsilon}))}{M} + \sqrt{\frac{\log(2/\delta)}{M}}, \quad (\text{A.24})$$

where the last inequality comes from d and Corollary 4 in Maurer (2016). To have above $\leq \epsilon_1/2$, we have sample complexity

$$M \geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{\mathcal{L}}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right].$$

Then we consider generalization bound for $\forall \phi$ and $\mathbf{W} := (W_1, \dots, W_M)$

$$\mathcal{L}_{\text{sup}}(\phi, \mathbf{W}) = \frac{1}{M} \sum_{i=1}^M \mathbb{E}_{y^i \sim \mathcal{T}_i} \mathbb{E}_{x^i \sim \mathcal{D}(y^i)} \ell(W_i \cdot \phi(x^i), y^i) \quad (\text{A.25})$$

$$\hat{\mathcal{L}}_{\text{sup}}(\phi, \mathbf{W}) = \frac{1}{M} \sum_{i=1}^M \frac{1}{m} \sum_{j=1}^m \ell(W_i \cdot \phi(x_j^i), y_j^i), \quad (\text{A.26})$$

where $\mathbf{W} = (W_1, \dots, W_M)$.

By uniform convergence (see Mohri et al. (2018) Theorem 3.3), we have with probability $1 - \delta/2$,

$$\begin{aligned} \mathcal{L}_{\text{sup}}(\phi, \mathbf{W}) - \hat{\mathcal{L}}_{\text{sup}}(\phi, \mathbf{W}) &\leq \frac{2\mathcal{R}_{Mm}(\mathcal{G}_\ell)}{Mm} + \sqrt{\frac{\log(2/\delta)}{Mm}} \\ &\leq \frac{8\sqrt{r-1}\text{LB}\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon}))}{Mm} + C\sqrt{\frac{\log(2/\delta)}{Mm}}, \end{aligned}$$

where the last inequality comes from Theorem A.2.7. To have above $\leq \epsilon_1/2$, we have sample complexity

$$Mm \geq \frac{1}{\epsilon_1} \left[8\sqrt{r-1}\text{LB}\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right],$$

satisfying $\forall \phi \in \Phi(\tilde{\epsilon})$

$$\begin{aligned} \frac{1}{M} \sum_{i=1}^M \mathcal{L}_{\text{sup}}(\mathcal{T}_i, \phi) &= \frac{1}{M} \sum_{i=1}^M \min_{W \in \mathbb{R}^{r \times d}} \mathbb{E}_{y \sim \mathcal{T}_i} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(W \cdot \phi(x), y)] \\ &\leq \frac{1}{M} \sum_{i=1}^M \mathbb{E}_{y \sim \mathcal{T}_i} \mathbb{E}_{x \sim \mathcal{D}(y)} \left[\ell(\widehat{W}_i \cdot \phi(x), y) \right] \\ &= \mathcal{L}_{\text{sup}}(\phi, \widehat{\mathbf{W}}) \\ &\leq \hat{\mathcal{L}}_{\text{sup}}(\phi, \widehat{\mathbf{W}}) + \epsilon_1/2. \end{aligned}$$

Then combine above with Equation (A.23)

$$\begin{aligned}\mathcal{L}_{\text{sup}}(\Phi) &= \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \Phi)] \\ &\leq \hat{\mathcal{L}}_{\text{sup}}(\Phi, \widehat{\mathbf{W}}) + \epsilon_1.\end{aligned}$$

We have

$$\begin{aligned}\mathcal{L}_{\text{sup}}(\Phi') - \frac{1}{m} \sum_{j=1}^m \ell(\widehat{\mathbf{W}}_i \cdot \Phi'(x_j^i), y_j^i) &\leq \epsilon_1 \\ \mathcal{L}_{\text{sup}}(\Phi') &\leq 2\epsilon_1 \leq \alpha \frac{1}{1-\tau} (2\epsilon_0 - \tau).\end{aligned}$$

The boundedness of $\tilde{\epsilon}$ follows Theorem A.2.8. \square

We state the theorem below.

Theorem A.2.10. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to Φ^*, Φ_ζ^* . Suppose for some small constant $\alpha \in (0, 1)$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} \frac{1}{1-\tau} (2\epsilon_0 - \tau)$ and obtain Φ' . For any $\delta > 0$, if*

$$\begin{aligned}M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{\mathcal{L}}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ Mm &\geq \frac{1}{\epsilon_1} \left[8\text{LB}\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right],\end{aligned}$$

then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \leq \frac{1}{\nu} \left[\alpha \frac{1}{1-\tau} (2\epsilon_0 - \tau) - \mathcal{L}_{\text{sup}}(\Phi_\zeta^*) \right] + \kappa. \quad (\text{A.27})$$

Proof of Theorem A.2.10. Recall with ν -diversity and κ -consistency with respect to

ϕ^*, ϕ_ζ^* , for target task \mathcal{T}_0 , we have that for ϕ' and ϕ^* ,

$$\begin{aligned}
\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) &= \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) \\
&\quad + \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \\
&\leq \frac{1}{\nu} \bar{d}_\zeta(\phi', \phi_\zeta^*) + \kappa \\
&\leq \frac{1}{\nu} [\mathcal{L}_{\text{sup}}(\phi') - \mathcal{L}_{\text{sup}}(\phi_\zeta^*)] + \kappa \\
&\leq \frac{1}{\nu} \left[\alpha \frac{1}{1-\tau} (2\epsilon_0 - \tau) - \epsilon^* \right] + \kappa,
\end{aligned}$$

where the last inequality comes from Theorem A.2.9, where taking $r = 2$. \square

A.2.2 Supervised Pretraining

In this section, we will show how multitask finetuning improves the model from supervised pretraining. We present pretraining error in binary classification and $\mathcal{D}_\mathcal{T}(\mathbf{y})$ as uniform. See the result for the general condition with multi-class in Section A.3.

Supervised Pretraining and Direct Adaptation

In this section, we show the error bound of a foundation model on a target task, where the model is pretrained by supervised loss followed directly by adaptation. For general $\mathbf{y} \sim \eta$. Let $p_i := \Pr_{\mathbf{y} \sim \eta} \{\mathbf{y} = \mathbf{y}_i\}$, where $\sum_{i=1}^K p_i = 1$.

Lemma A.2.11. *Suppose $\mathbf{y} \sim \eta$ and $l \leq \Pr_{\mathbf{y} \sim \eta} \{\mathbf{y} = \mathbf{y}_i\} \leq u$. Consider a task \mathcal{T} containing r classes, which is a subset of the total class set \mathcal{C} . We have $\forall \phi \in \Phi$,*

$$\mathcal{L}_{\text{sup}}(\phi) \leq \left(\frac{u}{l} \right)^r \mathcal{L}_{\text{sup-pre}}(\phi),$$

where

$$\mathcal{L}_{\text{sup-pre}}(\phi) = \min_{f \in \mathcal{F}} \mathbb{E}_{\mathbf{x}, \mathbf{y}} [\ell(f \circ \phi(\mathbf{x}), \mathbf{y})]. \quad (\text{A.28})$$

Proof of Section A.2.2. We first prove $r = 3$, where $\mathcal{T} = \{y_1, y_2, y_3\}$. Then in supervised pretraining, we have:

$$\mathcal{L}_{\text{sup-pre}}(\phi) = \min_{f \in \mathcal{F}} \mathbb{E}_{y \sim \mathcal{T}} \mathbb{E}_{x \sim y} [\ell(f \circ \phi(x), y)]. \quad (\text{A.29})$$

Let $f = (f_1, f_2, f_3)^\top$ be the best linear classifier on top of ϕ , the prediction logits are $g(x) = f \circ \phi(x) = (g_1(x), g_2(x), g_3(x))^\top$. Then we have:

$$\mathbb{E}_{x \sim y_1} [\ell(g \circ \phi(x), y)] = -\log \frac{\exp(g_1(x))}{\sum_{k=1}^3 \exp(g_k(x))}.$$

We let $y_k(x) = \exp(g_k(x))$, $k = 1, 2, 3$. Then

$$\begin{aligned} & \mathcal{L}_{\text{sup-pre}}(\phi) \\ &= - \left[p_1 \mathbb{E}_{x \sim y_1} \left(\log \frac{y_1(x)}{\sum_{k=1}^3 y_k(x)} \right) + p_2 \mathbb{E}_{x \sim y_2} \left(\log \frac{y_2(x)}{\sum_{k=1}^3 y_k(x)} \right) + p_3 \mathbb{E}_{x \sim y_3} \left(\log \frac{y_3(x)}{\sum_{k=1}^3 y_k(x)} \right) \right] \\ &= p_1 \mathbb{E}_{x \sim y_1} \left(\log \frac{\sum_{k=1}^3 y_k(x)}{y_1(x)} \right) + p_2 \mathbb{E}_{x \sim y_2} \left(\log \frac{\sum_{k=1}^3 y_k(x)}{y_2(x)} \right) + p_3 \mathbb{E}_{x \sim y_3} \left(\log \frac{\sum_{k=1}^3 y_k(x)}{y_3(x)} \right). \end{aligned}$$

Recall

$$\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi) := \min_{\mathbf{w}} \mathbb{E}_{y \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(\mathbf{w}^\top \phi(x), y)]. \quad (\text{A.30})$$

Consider

$$\mathcal{L}_{\text{sup}}^*(\mathcal{T}, \phi) := \mathbb{E}_{y \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}(y)} [\ell(\mathbf{w}^\top \phi(x), y)], \quad (\text{A.31})$$

where \mathbf{w} is the corresponding sub-vector of f according to task (for e.g., $\mathbf{w} =$

$(f_1, f_2)^\top$ if $\mathcal{T} = \{y_1, y_2\}$). Then we have

$$\begin{aligned}
\mathcal{L}_{\text{sup}}^*(\mathcal{T}, \phi) &= -\frac{p_1 p_2}{p_1 p_2 + p_1 p_3 + p_2 p_3} \cdot \frac{1}{2} \left[\mathbb{E}_{x \sim y_1} \left(\log \frac{y_1(x)}{y_1(x) + y_2(x)} \right) + \mathbb{E}_{x \sim y_2} \left(\log \frac{y_2(x)}{y_1(x) + y_2(x)} \right) \right] \\
&\quad -\frac{p_1 p_3}{p_1 p_2 + p_1 p_3 + p_2 p_3} \cdot \frac{1}{2} \left[\mathbb{E}_{x \sim y_1} \left(\log \frac{y_1(x)}{y_1(x) + y_3(x)} \right) + \mathbb{E}_{x \sim y_3} \left(\log \frac{y_3(x)}{y_1(x) + y_3(x)} \right) \right] \\
&\quad -\frac{p_2 p_3}{p_1 p_2 + p_1 p_3 + p_2 p_3} \cdot \frac{1}{2} \left[\mathbb{E}_{x \sim y_2} \left(\log \frac{y_2(x)}{y_2(x) + y_3(x)} \right) + \mathbb{E}_{x \sim y_3} \left(\log \frac{y_3(x)}{y_2(x) + y_3(x)} \right) \right] \\
&= \frac{p_1 p_2}{p_1 p_2 + p_1 p_3 + p_2 p_3} \cdot \frac{1}{2} \left[\mathbb{E}_{x \sim y_1} \left(\log \frac{y_1(x) + y_2(x)}{y_1(x)} \right) + \mathbb{E}_{x \sim y_2} \left(\log \frac{y_1(x) + y_2(x)}{y_2(x)} \right) \right] \\
&\quad + \frac{p_1 p_3}{p_1 p_2 + p_1 p_3 + p_2 p_3} \cdot \frac{1}{2} \left[\mathbb{E}_{x \sim y_1} \left(\log \frac{y_1(x) + y_3(x)}{y_1(x)} \right) + \mathbb{E}_{x \sim y_3} \left(\log \frac{y_1(x) + y_3(x)}{y_3(x)} \right) \right] \\
&\quad + \frac{p_2 p_3}{p_1 p_2 + p_1 p_3 + p_2 p_3} \cdot \frac{1}{2} \left[\mathbb{E}_{x \sim y_2} \left(\log \frac{y_2(x) + y_3(x)}{y_2(x)} \right) + \mathbb{E}_{x \sim y_3} \left(\log \frac{y_2(x) + y_3(x)}{y_3(x)} \right) \right].
\end{aligned}$$

By observing the terms with $y_1(x)$ as denominator (similar as $y_2(x), y_3(x)$), we want to prove:

$$p_1 \left(\frac{u}{l} \right)^2 \geq \frac{1}{2} \left(\frac{p_1 p_2 + p_1 p_3}{p_1 p_2 + p_1 p_3 + p_2 p_3} \right).$$

This obtained by $\left(\frac{u}{l} \right)^2 \geq \frac{1}{3} \frac{u}{l^2}$.

We have

$$\mathcal{L}_{\text{sup}}^*(\mathcal{T}, \phi) \leq \left(\frac{u}{l} \right)^2 \mathcal{L}_{\text{sup-pre}}(\phi).$$

For the general K-class setting, we follow similar steps, we have

$$\mathcal{L}_{\text{sup-pre}}(\phi) = - \left[\sum_{i=1}^r p_i \mathbb{E}_{x \sim y_i} \left(\log \frac{y_i(x)}{\sum_{k=1}^K y_k(x)} \right) \right].$$

We denote J as all possible r product of $p_i \in \{p_1, \dots, p_K\}$, $J = \{p_1 \cdots p_r, \dots\}$.

Similarly, we have

$$\mathcal{L}_{\text{sup}}^*(\mathcal{T}, \Phi) = -\frac{1}{r} \left\{ \sum_{\mathcal{T} \subset \mathcal{C}} \left[\frac{\prod_{i \in \mathcal{T}} p_i}{J} \sum_{i \in \mathcal{T}} \mathbb{E}_{x \sim y_i} \left(\log \frac{y_i(x)}{\sum_{j \in \mathcal{T}} y_j(x)} \right) \right] \right\}$$

where \mathcal{T} are all tasks with r classes. By observing, inside the summation there are in total $\binom{K-1}{r-1}$ terms with $y_1(x)$ as the numerator, where corresponding probability is

$$\frac{p_1 \prod_{i \in \mathcal{T}, i \neq 1} p_i}{J},$$

where each term can be upper bounded by $-\left(\frac{u}{l}\right)^r p_1 \mathbb{E}_{x \sim y_i} \left(\log \frac{y_i(x)}{\sum_{k=1}^K y_k(x)} \right)$ (similar as $y_j(x), j \in \mathcal{T}$). \square

We state the theorem below.

Theorem A.2.12. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^*, ϕ_ζ^* . Suppose $\hat{\phi}$ satisfies $\hat{\mathcal{L}}_{\text{sup-pre}}(\hat{\phi}) \leq \epsilon_0$. Let $p_i := \Pr_{y \sim \eta}\{y = y_i\}$, where $\sum_{i=1}^K p_i = 1$. Let $\rho := \frac{\max_i p_i}{\min_j p_j}$. Consider pretraining set $\mathcal{S}_{\text{sup-pre}} := \{x_i, y_i\}_{i=1}^N$, for any $\delta \geq 0$, if*

$$N \geq \frac{1}{\epsilon_0} \left[8LR\sqrt{K}\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right].$$

Then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subset \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} [2\rho^2 \epsilon_0 - \epsilon^*] + \kappa. \quad (\text{A.32})$$

Proof of Theorem A.2.12. The proof follows similar steps in Theorem A.2.3. For supervised pretraining, the sample complexity is similar to Theorem A.2.3, note that there is an extra \sqrt{K} term. We show how we have this term below:

Consider function class

$$\mathcal{G}_\ell = \{g_{W, \phi}(x, y) : g_{W, \phi}(x, y) = \ell(W^\top \phi(x_j^i), y_j^i), \phi \in \Phi, \|W\|_2 \leq B\}.$$

The Rademacher complexity is

$$\mathcal{R}_n(\mathcal{G}_\ell) = \mathbb{E}_{\{\sigma_i\}_{i=1}^n, \{x_j, y_j\}_{j=1}^n} \left[\sup_{\ell \in \mathcal{G}_\ell} \sum_{j=1}^n \sigma_j \ell(W \cdot \phi(x_j), y_j) \right].$$

Then from Theorem A.2.7, the pretraining is a large task with classification among K classes.

$$\mathcal{R}_n(\mathcal{G}_\ell) \leq 4\sqrt{K}LB\mathcal{R}_n(\Phi).$$

Then by Theorem 3.3 in Mohri et al. (2018), with a and c , we have for $\forall \phi \in \Phi$ with probability $1 - \delta$,

$$\mathcal{L}_{\text{sup-pre}}(\phi) - \widehat{\mathcal{L}}_{\text{sup-pre}}(\phi) \leq \frac{4LR\sqrt{K}\mathcal{R}_N(\Phi)}{N} + C\sqrt{\frac{\log \frac{1}{\delta}}{N}}. \quad (\text{A.33})$$

To have above $\leq \epsilon_0$, we have sample complexity

$$N \geq \frac{1}{\epsilon_0} \left[8LR\sqrt{K}\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right].$$

With the above sample complexity of $\mathcal{S}_{\text{sup-pre}} = \{x_i, y_i\}_{i=1}^N$, we have pretraining $\hat{\phi}$

$$\mathcal{L}_{\text{sup-pre}}(\hat{\phi}) \leq 2\epsilon_0.$$

Recall ν -diversity and κ -consistency, with respect to ϕ^*, ϕ_ζ^* , for target task \mathcal{T}_0 , we have that for $\hat{\phi}$ and ϕ^* ,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\Phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \leq \mathbf{d}_{\mathcal{C}_0}(\hat{\Phi}, \Phi_\zeta^*) + \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \quad (\text{A.34})$$

$$\leq \bar{\mathbf{d}}_\zeta(\hat{\Phi}, \Phi_\zeta^*)/\nu + \kappa \quad (\text{A.35})$$

$$\leq \frac{1}{\nu} \left[\mathcal{L}_{\text{sup}}(\hat{\Phi}) - \mathcal{L}_{\text{sup}}(\Phi_\zeta^*) \right] + \kappa \quad (\text{A.36})$$

$$\leq \frac{1}{\nu} \left[\rho^2 \mathcal{L}_{\text{sup-pre}}(\hat{\Phi}) - \epsilon^* \right] + \kappa \quad (\text{A.37})$$

$$\leq \frac{1}{\nu} \left[2\rho^2 \epsilon_0 - \epsilon^* \right] + \kappa \quad (\text{A.38})$$

where the second to last inequality comes from Theorem A.2.11. □

Supervised Pretraining and Multitask Finetuning

In this section, we show the error bound of a supervised pretrained foundation model on a target task can be further reduced by multitask finetuning. We follow similar steps in Section A.2.1. Recall Definition A.2.4, similar to Theorem A.2.9, we introduce the following lemma under supervised pretraining loss.

Lemma A.2.13. *Assume Assumption 3.3.1 and that Φ has (ν, ϵ) -diversity for ζ and \mathcal{C}_0 . Suppose for some small constant $\alpha \in (0, 1)$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} 2\rho^2 \epsilon_0$ and obtain Φ' . For any $\delta > 0$, if*

$$\begin{aligned} \mathbf{M} &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{\mathbf{L}}\mathcal{R}_{\mathbf{M}}(\Phi(\tilde{\epsilon})) + \frac{4\mathbf{C}^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ \mathbf{Mm} &\geq \frac{1}{\epsilon_1} \left[16\mathbf{L}\mathbf{B}\mathcal{R}_{\mathbf{Mm}}(\Phi(\tilde{\epsilon})) + \frac{4\mathbf{C}^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \end{aligned}$$

then expected supervised loss $\mathcal{L}_{\text{sup}}(\Phi') \leq 2\alpha\rho^2\epsilon_0$, with probability $1 - \delta$.

Proof of Theorem A.2.13. The steps follow similar steps in Theorem A.2.9. □

We state the main theorem below.

Theorem A.2.14. *Assume Assumption 3.3.1 and that Φ has (ν, ϵ) -diversity for ζ and \mathcal{C}_0 . Suppose for some small constant $\alpha \in (0, 1)$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3}2\rho^2\epsilon_0$ and obtain ϕ' . For any $\delta > 0$, if*

$$\begin{aligned} M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{L}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ Mm &\geq \frac{1}{\epsilon_1} \left[16LB\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \end{aligned}$$

then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} (2\alpha\rho^2\epsilon_0 - \mathcal{L}_{\text{sup}}(\phi^*)) + \epsilon. \quad (\text{A.39})$$

Proof of Theorem A.2.14. Recall ν -diversity and κ -consistency, with respect to ϕ^*, ϕ_ζ^* , for target task \mathcal{T}_0 , we have that for $\hat{\phi}$ and ϕ^* ,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq d_{\mathcal{C}_0}(\phi', \phi_\zeta^*) + \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \quad (\text{A.40})$$

$$\leq \bar{d}_\zeta(\phi', \phi_\zeta^*)/\nu + \kappa \quad (\text{A.41})$$

$$\leq \frac{1}{\nu} [\mathcal{L}_{\text{sup}}(\phi') - \mathcal{L}_{\text{sup}}(\phi_\zeta^*)] + \kappa \quad (\text{A.42})$$

$$\leq \frac{1}{\nu} (2\alpha\rho^2\epsilon_0 - \epsilon^*) + \kappa, \quad (\text{A.43})$$

where the last inequality comes from Theorem A.2.13. □

A.2.3 Masked Language Pretraining

The theoretical guarantee in masked language pretraining follows the same error bound in supervised pretraining, with K representing the size of the vocabulary.

A.2.4 Unified Main Theory

We now prove the main theory below. We first re-state the theorem.

Theorem 3.3.4. (No Multitask Finetuning) *Assume assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^* and ϕ_ζ^* . Suppose $\hat{\phi}$ satisfies $\hat{\mathcal{L}}_{\text{pre}}(\hat{\phi}) \leq \epsilon_0$. Let $\tau := \Pr_{(y_1, y_2) \sim \eta^2} \{y_1 = y_2\}$. Then for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,*

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\frac{2\epsilon_0}{1-\tau} - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa. \quad (3.3)$$

Proof of Theorem 3.3.4. The result is a direct combination of Theorem A.2.3 and Theorem A.2.12. \square

Theorem 3.3.5. (With Multitask Finetuning) *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^* and ϕ_ζ^* . Suppose for some constant $\alpha \in (0, 1)$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} \frac{2\epsilon_0}{1-\tau}$ and obtain ϕ' . For any $\delta > 0$, if for $\tilde{\epsilon} = \hat{\mathcal{L}}_{\text{pre}}(\phi')$,*

$$\begin{aligned} M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{\mathcal{L}}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ Mm &\geq \frac{1}{\epsilon_1} \left[16LB\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \end{aligned}$$

then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\alpha \frac{2\epsilon_0}{1-\tau} - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa. \quad (3.4)$$

Proof of Theorem 3.3.5. Follow the similar steps in proof of Theorem A.2.9, we have

$$\mathcal{L}_{\text{sup}}(\phi') \leq 2\epsilon_1 \leq \alpha \frac{2\rho^2}{1-\tau} \epsilon_0.$$

Recall ν -diversity and κ -consistency, with respect to ϕ^* , ϕ_ζ^* , for target task \mathcal{T}_0 , we have that for ϕ' and ϕ^* ,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \mathbf{d}_{e_0}(\phi', \phi_\zeta^*) + \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \quad (\text{A.44})$$

$$\leq \bar{\mathbf{d}}_\zeta(\phi', \phi_\zeta^*)/\nu + \kappa \quad (\text{A.45})$$

$$\leq \frac{1}{\nu} [\mathcal{L}_{\text{sup}}(\phi') - \mathcal{L}_{\text{sup}}(\phi_\zeta^*)] + \kappa \quad (\text{A.46})$$

$$\leq \frac{1}{\nu} \left[\alpha \frac{2\rho^2}{1-\tau} \epsilon_0 - \epsilon^* \right] + \kappa. \quad (\text{A.47})$$

□

The sample complexity of finetuning depends on $\tilde{\epsilon} = \widehat{\mathcal{L}}_{\text{pre}}(\phi')$. Below we show that $\tilde{\epsilon}$ can be upper bounded in finite step finetuning.

Lemma A.2.15 (Bounded $\tilde{\epsilon}$). *After finite steps in Multitask finetuning in Algorithm 2, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} \frac{1}{1-\tau} (2\epsilon_0 - \tau)$ and obtain ϕ' . Then there exists a bounded $\tilde{\epsilon}$ such that $\phi' \in \Phi(\tilde{\epsilon})$.*

Proof of Theorem A.2.15. Given finite number of steps and finite step size γ in Algorithm 2, we have bounded $\|\phi' - \hat{\phi}\|$. Then with \mathbf{b} and \mathbf{c} , using Theorem A.2.7 and lemma A.3 in Arora et al. (2019), we have $\widehat{\mathcal{L}}_{\text{pre}}(\phi)$ is M -Lipschitz with respect to ϕ with bounded M . We have $\exists \epsilon$ such that $\widehat{\mathcal{L}}_{\text{pre}}(\phi') - \widehat{\mathcal{L}}_{\text{pre}}(\hat{\phi}) \leq \epsilon \|\phi' - \hat{\phi}\|$. We have $\widehat{\mathcal{L}}_{\text{pre}}(\phi') \leq \epsilon_0 + \epsilon \|\phi' - \hat{\phi}\|$. Take $\tilde{\epsilon} = \epsilon_0 + \epsilon \|\phi' - \hat{\phi}\|$ yields the result. □

A.2.5 Bounded Task Loss by Task Diversity

By the previous lemma and claim, we have the below corollary.

Corollary A.2.16. *Suppose we have ϕ in pretraining: for $\forall \phi \in \Phi$,*

$$\mathcal{L}_{\text{sup}}(\phi) \leq \frac{1}{1-\tau} \left(\frac{\mathbf{u}}{\mathbf{l}} \right)^r \mathcal{L}_{\text{pre}}(\phi),$$

where $\mathcal{L}_{\text{pre}}(\phi)$ is $\mathcal{L}_{\text{con-pre}}(\phi)$ if contrastive learning and $\mathcal{L}_{\text{sup-pre}}(\phi)$ if supervised learning.

Consider $\rho = \frac{\mu}{\nu}$ and Theorem A.2.16,

Recall ν -diversity and κ -consistency, with respect to ϕ^*, ϕ_ζ^* , for target task \mathcal{T}_0 , we have that for $\hat{\phi}$ and ϕ^* ,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq d_{e_0}(\hat{\phi}, \phi_\zeta^*) + \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \quad (\text{A.48})$$

$$\leq \bar{d}_\zeta(\hat{\phi}, \phi_\zeta^*)/\nu + \kappa \quad (\text{A.49})$$

$$\leq \frac{1}{\nu} \left[\mathcal{L}_{\text{sup}}(\hat{\phi}) - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa \quad (\text{A.50})$$

$$\leq \frac{1}{\nu} \left[\frac{\rho^\tau}{1-\tau} \mathcal{L}_{\text{pre}}(\hat{\phi}) - \mathcal{L}_{\text{sup}}(\phi^*) \right] + \kappa. \quad (\text{A.51})$$

A.3 Multi-class Classification

In this section, we provide a general result for multi-classes.

A.3.1 Contrastive Pretraining

Lemma A.3.1 (Theorem 6.1 in Arora et al. (2019)). *For multi-classes, we have*

$$\mathcal{L}_{\text{sup}}(\phi) \leq \mathcal{L}_{\text{sup}}^\mu(\phi) \leq \frac{1}{1-\tau_r} \mathcal{L}_{\text{con-pre}}(\phi), \quad (\text{A.52})$$

where $\tau_r = \mathbb{E}_{(\mathbf{y}, \mathbf{y}_1^-, \dots, \mathbf{y}_{r-1}^-) \sim \eta^\tau} \mathbb{1}\{\mathbf{y} \text{ does not appear in } (\mathbf{y}_1^-, \dots, \mathbf{y}_{r-1}^-)\}$.

Proof of Theorem A.3.1. The proof of Theorem A.3.1 follows the first two steps in the proof of Theorem B.1 of Arora et al. (2019). we denote distribution of $\mathbf{y} \sim \mathcal{T}$ as $\mathcal{D}_\mathcal{T}(\mathbf{y})$ and it's uniform distribution. \square

We first provide contrastive pretraining error similar to Theorem A.2.3 in a multiclass setting.

Theorem A.3.2. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to ϕ^*, ϕ_ζ^* . Suppose $\hat{\phi}$ satisfies $\hat{\mathcal{L}}_{\text{con-pre}}(\hat{\phi}) \leq \epsilon_0$. Consider a pretraining set*

$\mathcal{S}_{\text{un}} = \{x_j, x_j^+, x_{j1}^-, \dots, x_{j(r-1)}\}_{j=1}^N$. For target task \mathcal{T}_0 , with sample complexity

$$N \geq \frac{1}{\epsilon_0} \left[8LR\sqrt{r-1}\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right],$$

it's sufficient to learn an $\hat{\phi}$ with classification error

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\frac{2}{1-\tau_r} \epsilon_0 - \epsilon^* \right] + \epsilon,$$

with probability $1 - \delta$.

Proof of Theorem A.3.2. Following similar step of proof of Theorem A.2.3, we have with

$$N \geq \frac{1}{\epsilon_0} \left[8LR\sqrt{r-1}\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right].$$

Then pretraining $\hat{\phi}$

$$\mathcal{L}_{\text{con-pre}}(\hat{\phi}) \leq 2\epsilon_0.$$

Recall ν -diversity and κ -consistency, for target task \mathcal{T}_0 , we have that for $\hat{\phi}$ and ϕ^* ,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \bar{d}_\zeta(\hat{\phi}, \phi_\zeta^*)/\nu + \kappa \quad (\text{A.53})$$

$$\leq \frac{1}{\nu} \left[\mathcal{L}_{\text{sup}}(\hat{\phi}) - \mathcal{L}_{\text{sup}}(\phi_\zeta^*) \right] + \kappa \quad (\text{A.54})$$

$$(\text{A.55})$$

Consider Theorem A.3.1, we have:

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi^*) \leq \frac{1}{\nu} \left[\frac{1}{1-\tau_r} \mathcal{L}_{\text{con-pre}}(\hat{\phi}) - \epsilon^* \right] + \kappa \quad (\text{A.56})$$

$$= \frac{1}{\nu} \left(\frac{2\epsilon_0}{1-\tau_r} - \epsilon^* \right) + \kappa. \quad (\text{A.57})$$

□

Below, we provide our main result similar to Theorem A.2.10 for multi-classes setting.

Theorem A.3.3. *For target evaluation task \mathcal{T}_0 , consider the error bound in pretraining is $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\Phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \leq \frac{1}{\nu} \left[\frac{2\epsilon_0}{1-\tau_r} - \epsilon^* \right] + \kappa$. Consider α as any small constant, for any $\epsilon_1 < \frac{\alpha}{3} \frac{2\epsilon_0}{1-\tau_r}$, consider a multitask finetuning set $\mathcal{S} = \{(x_j^i, y_j^i) : i \in [M], j \in [m]\}$, with M number of tasks, and m number of samples in each task. Then, with sample complexity*

$$\begin{aligned} M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{L}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right] \\ Mm &\geq \frac{1}{\epsilon_1} \left[8\text{LB}\sqrt{r-1}\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right]. \end{aligned}$$

Solving Equation (3.2) with empirical risk lower than ϵ_1 is sufficient to learn an ϕ' with classification error $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \leq \frac{1}{\nu}(\alpha \frac{2\epsilon_0}{1-\tau_r} - \epsilon^*) + \kappa$, with probability $1 - \delta$.

Proof of Theorem A.3.3. Recalling Theorem A.2.7 and Theorem A.2.9, the proof follows the same steps in the proof of Theorem A.2.10, with different r .

□

A.3.2 Supervised Pretraining

We first provide contrastive pretraining error similar to Theorem A.2.12 in the multiclass setting.

Theorem A.3.4. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to Φ^*, Φ_ζ^* . Suppose $\hat{\Phi}$ satisfies $\hat{\mathcal{L}}_{\text{sup-pre}}(\hat{\Phi}) \leq \epsilon_0$. Let $p_i := \Pr_{y \sim \eta} \{y = y_i\}$, where $\sum_{i=1}^K p_i = 1$. Let $\rho := \frac{\max_i p_i}{\min_j p_j}$. Consider pretraining set $\mathcal{S}_{\text{sup-pre}} := \{x_i, y_i\}_{i=1}^N$, for any $\delta \geq 0$, if*

$$N \geq \frac{1}{\epsilon_0} \left[8\text{LR}\sqrt{K}\mathcal{R}_N(\Phi) + \frac{8C^2}{\epsilon_0} \log\left(\frac{2}{\delta}\right) \right].$$

Then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subset \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \hat{\Phi}) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \leq \frac{1}{\nu} [2\rho^r \epsilon_0 - \mathcal{L}_{\text{sup}}(\Phi_\zeta^*)] + \kappa. \quad (\text{A.58})$$

Proof of Theorem A.3.4. The proof follows similar steps of Theorem A.2.12. \square

Below, we provide our main result similar to Theorem A.2.14 for multi-classes setting.

Theorem A.3.5. *Assume Assumption 3.3.1 and that Φ has ν -diversity and κ -consistency with respect to Φ^*, Φ_ζ^* . Suppose for some small constant $\alpha \in (0, 1)$, we solve Equation (3.2) with empirical loss lower than $\epsilon_1 = \frac{\alpha}{3} 2\rho^r \epsilon_0$ and obtain Φ' . For any $\delta > 0$, if*

$$\begin{aligned} M &\geq \frac{1}{\epsilon_1} \left[4\sqrt{2}\tilde{L}\mathcal{R}_M(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \\ Mm &\geq \frac{1}{\epsilon_1} \left[8\text{LB}\sqrt{r-1}\mathcal{R}_{Mm}(\Phi(\tilde{\epsilon})) + \frac{4C^2}{\epsilon_1} \log\left(\frac{2}{\delta}\right) \right], \end{aligned}$$

then with probability $1 - \delta$, for any target task $\mathcal{T}_0 \subseteq \mathcal{C}_0$,

$$\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi') - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \Phi^*) \leq \frac{1}{\nu} (2\alpha\rho^r \epsilon_0 - \mathcal{L}_{\text{sup}}(\Phi_\zeta^*)) + \kappa. \quad (\text{A.59})$$

Proof of Theorem A.3.5. Recalling Theorem A.2.7 and Theorem A.2.9, the proof follows the same steps in the proof of Theorem A.2.14, with different r . \square

A.4 Linear Case Study

In this section, we provide a full analysis of the linear case study to provide intuition about our consistency, diversity, and task selection algorithm. Intuitively, we have multiple classes, each centered around its mean vector. Target data has a subset of classes, while training data has another subset of classes. Consistency and diversity are related to how these two subsets overlap, i.e., the number of shared features and the number of disjoint features. Then, we can link it to the task selection algorithm.

In this section, z_i means the i -th dimension of vector z rather than the sample index.

A.4.1 Problem Setup

Linear Data and Tasks. We consider dictionary learning or sparse coding settings, which is a classic data model (e.g., Olshausen and Field (1997); Vinje and Gallant (2000); Blei et al. (2003); Shi et al. (2022, 2023c)). Let $\mathcal{X} \subseteq \mathbb{R}^d$ be the input space and we have input data $x \in \mathcal{X}$. Suppose $Q \in \mathbb{R}^{d \times D}$ is an unknown dictionary with D columns that can be regarded as patterns or features. For simplicity, assume $d = D$ and Q is orthonormal. We have $z \in \{0, -1, +1\}^d$ as a latent class, where z is a hidden vector that indicates the presence of each pattern. Each latent class z has a distribution $\mathcal{D}_z(x)$ over inputs x . We assume $\mathcal{D}_z(x)$ be a distribution with mean Qz , i.e., $x = Qz + e_z$, where $e_z \in \mathbb{R}^d$ is some noise vector drawing from a zero-mean distribution.

For simplicity, we consider each task to be a binary classification task, where $\mathcal{Y} = \{-1, +1\}$ is the label space. In each task (in multitask finetuning or target task), we have two latent classes z, z' (denote the task as $\mathcal{T}_{z, z'}$) and we randomly assign -1 and $+1$ to each latent class. W.l.o.g., we have in $\mathcal{T}_{z, z'}$:

$$x = \begin{cases} Qz + e_{z'}, & \text{if } y = -1 \\ Qz' + e_{z'}, & \text{if } y = +1. \end{cases} \quad (\text{A.60})$$

For simplicity, we consider a balanced class setting in all tasks, i.e., $p(y = -1) = p(y = +1) = \frac{1}{2}$.

Now, we define multitask finetuning tasks and target tasks. Suppose there is a set of latent classes $\mathcal{C} \subseteq \{0, -1, +1\}^d$ used for multitask finetuning tasks, which has an index set $J_e \subseteq [d]$, $k_e := |J_e|$ such that for any $z \in \mathcal{C}$, we have $z_{J_e} \in \{-1, +1\}^{k_e}$ and $z_{[d] \setminus J_e} \in \{0\}^{d-k_e}$. Similarly, suppose there is a set of latent classes $\mathcal{C}_0 \subseteq \{0, -1, +1\}^d$ used for target tasks whose index set is $J_0 \subseteq [d]$, $k_0 := |J_0|$. Note that J_e may or may not overlap with J_0 and denote the set of features encoded both by \mathcal{C}_0 and \mathcal{C}

as $L_e := J_0 \cap J_e$, $l_e := |L_e|$. Intuitively, L_e represents the target features covered by training data. Let ζ over $\mathcal{C} \times \mathcal{C}$ be the distribution of multitask finetuning tasks. Then, in short, our data generation pipeline for multitask finetuning tasks is (1) sample two latent classes $(z, z') \sim \zeta$ as a task $\mathcal{T}_{z, z'}$; (2) assign label $-1, +1$ to two latent classes; (3) sample input data from $\mathcal{D}_z(x)$ and $\mathcal{D}_{z'}(x)$ with balanced probabilities.

For simplicity, we have a symmetric assumption and a non-degenerate assumption for ζ . Symmetric assumption means each dimension is equal important and non-degenerate assumption means any two dimensions are not determined by each other in all tasks.

Assumption A.4.1 (Symmetric). *We assume for any multitask finetuning tasks distribution ζ , for any $j, k \in J_e$, switching two dimensions z_j and z_k does not change the distribution ζ .*

Assumption A.4.2 (Non-degenerate). *We assume for any multitask finetuning tasks distribution ζ , for any $j, k \in J_e$, over ζ we have $\mathbb{P}(z_j = z'_j, z_k \neq z'_k) > 0$.*

Remark A.4.1. *There exists many ζ satisfying above assumptions, e.g., (1) z_{J_e} and z'_{J_e} uniformly sampling from $\{-1, +1\}^{k_e}$; or (2) let $k_e = 2$, z_{J_e} and z'_{J_e} uniformly sampling from $\{(+1, +1), (-1, +1), (+1, -1)\}$ (note that uniformly sampling from $\{(+1, +1), (-1, -1)\}$ does not satisfy non-degenerate assumption). Also, we note that even when $\mathcal{C} = \mathcal{C}_0$, the target latent class may not exist in the multitask finetuning tasks.*

Linear Model and Loss Function. Let Φ be the hypothesis class of models $\phi : \mathcal{X} \rightarrow \bar{\mathcal{Z}}$, where $\bar{\mathcal{Z}} \subseteq \mathbb{R}^d$ is the output space of the model. We consider a linear model class with regularity Assumption 3.3.1, i.e., $\Phi = \{\phi \in \mathbb{R}^{d \times d} : \|\phi\|_F \leq R\}$ and linear head $w \in \mathbb{R}^d$ where $\|w\|_2 \leq B$. Thus, the final output of the model and linear head is $w^\top \phi x$. We use linear loss in Shi et al. (2023a), i.e., $\ell(w^\top \phi x, y) = -y w^\top \phi x$ and

we have

$$\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi) := \min_{\mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_2 \leq B} \mathbb{E}_{z, \mathbf{y} \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}_z(x)} [\ell(\mathbf{w}^\top \phi x, \mathbf{y})] \quad (\text{A.61})$$

$$\mathcal{L}_{\text{sup}}(\phi) := \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi)] \quad (\text{A.62})$$

$$\phi_\zeta^* := \arg \min_{\phi \in \Phi} \mathcal{L}_{\text{sup}}(\phi), \quad (\text{A.63})$$

where ϕ_ζ^* is the optimal representation for multitask finetuning.

A.4.2 Diversity and Consistency Analysis

Optimal Representation for Multitask Finetuning

Lemma A.4.3. *Assume Assumption A.4.1 and Assumption A.4.2. We have $\phi_\zeta^* = \mathbf{U} \Lambda^* \mathbf{Q}^{-1}$, where \mathbf{U} is any orthonormal matrix, $\Lambda^* = \text{diag}(\lambda^*)$. For any $i \in \mathcal{J}_c$, $\lambda_i^* = \frac{R}{\sqrt{k_e}}$ and $\lambda_i^* = 0$ otherwise.*

Proof of Theorem A.4.3. We have the singular value decomposition $\phi = \mathbf{U} \Lambda \mathbf{V}^\top$,

where $\Lambda = \text{diag}(\lambda)$, where $\lambda \in \mathbb{R}^d$. Then, we have

$$\mathcal{L}_{\text{sup}}(\Phi) = \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \Phi)] \quad (\text{A.64})$$

$$= \mathbb{E}_{\mathcal{T} \sim \zeta} \left[\min_{\mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_2 \leq B} \mathbb{E}_{z, y \sim \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}_z(x)} [\ell(\mathbf{w}^\top \Phi x, y)] \right] \quad (\text{A.65})$$

$$= \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} \left[\min_{\mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_2 \leq B} \frac{1}{2} \left(\mathbb{E}_{x \sim \mathcal{D}_z(x)} [\ell(\mathbf{w}^\top \Phi x, -1)] + \mathbb{E}_{x \sim \mathcal{D}_{z'}(x)} [\ell(\mathbf{w}^\top \Phi x, +1)] \right) \right] \quad (\text{A.66})$$

$$= \frac{1}{2} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} \left[\min_{\mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_2 \leq B} \mathbb{E}_{x \sim \mathcal{D}_z(x)} [\mathbf{w}^\top \Phi x] + \mathbb{E}_{x \sim \mathcal{D}_{z'}(x)} [-\mathbf{w}^\top \Phi x] \right] \quad (\text{A.67})$$

$$= \frac{1}{2} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} \left[\min_{\mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\|_2 \leq B} \mathbf{w}^\top \Phi Q z - \mathbf{w}^\top \Phi Q z' \right] \quad (\text{A.68})$$

$$= -\frac{B}{2} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} [\|\Phi Q(z - z')\|_2] \quad (\text{A.69})$$

$$= -\frac{B}{2} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} [\|\Lambda V^\top Q(z - z')\|_2]. \quad (\text{A.70})$$

W.l.o.g., we can assume $V^\top = Q^{-1}$. As $\|\Phi\|_F = \|\Lambda\|_F = \|\lambda\|_2$ thus we have

$$\begin{aligned} \min_{\Phi \in \Phi} \mathcal{L}_{\text{sup}}(\Phi) &= -\frac{B}{2} \max_{\|\Lambda\|_F \leq R} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} [\|\Lambda(z - z')\|_2] \\ &= -\frac{B}{2} \max_{\|\lambda\|_2 \leq R} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} \left[\sqrt{\sum_{i=1}^d \lambda_i^2 (z_i - z'_i)^2} \right] \\ &= -\frac{B}{2} \max_{\|\lambda\|_2 = R} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} \left[\sqrt{\sum_{i=1}^d \lambda_i^2 (z_i - z'_i)^2} \right] \\ &= -B \max_{\|\lambda\|_2 = R} \mathbb{E}_{\mathcal{T}_{z, z'} \sim \zeta} \left[\sqrt{\sum_{i \in \mathcal{J}_e} \lambda_i^2 \mathbb{1}[z_i \neq z'_i]} \right], \end{aligned} \quad (\text{A.71})$$

where $\mathbb{1}[z_i \neq z'_i]$ is a Boolean function, mapping True to 1 and False to 0.

Let $\phi_\zeta^* = U\Lambda^*Q^{-1}$ with corresponding λ^* . Now, we use contradiction to prove

for any $j, k \in J_e$, we have $\lambda_j^* = \lambda_k^*$. W.l.o.g., suppose $\lambda_j^* < \lambda_k^*$,

$$\begin{aligned}
& \mathcal{L}_{\text{sup}}(\Phi_\zeta^*) \\
&= -\mathbf{B} \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_j^{*2} \mathbb{1}[z_j \neq z'_j] + \lambda_k^{*2} \mathbb{1}[z_k \neq z'_k] + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \right] \\
&= -\mathbf{B} \left\{ \mathbf{p}(z_j \neq z'_j, z_k \neq z'_k) \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_j^{*2} + \lambda_k^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \right] \middle| z_j \neq z'_j, z_k \neq z'_k \right] \right. \\
&\quad + \mathbf{p}(z_j = z'_j, z_k \neq z'_k) \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_k^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \right] \middle| z_j = z'_j, z_k \neq z'_k \right] \\
&\quad \left. + \mathbf{p}(z_j \neq z'_j, z_k = z'_k) \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_j^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \right] \middle| z_j \neq z'_j, z_k = z'_k \right] \right\}.
\end{aligned}$$

By symmetric Assumption A.4.1 and non-degenerate Assumption A.4.2, we have

$p(z_j = z'_j, z_k \neq z'_k) = p(z_j \neq z'_j, z_k = z'_k) > 0$, and

$$\begin{aligned}
& \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_k^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_j = z'_j, z_k \neq z'_k \right] \\
& + \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_j^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_j \neq z'_j, z_k = z'_k \right] \\
& = \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_k^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_j = z'_j, z_k \neq z'_k \right] \\
& + \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\lambda_j^{*2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_k \neq z'_k, z_j = z'_j \right] \\
& < 2 \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\frac{\lambda_j^{*2} + \lambda_k^{*2}}{2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_j = z'_j, z_k \neq z'_k \right] \\
& = \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\frac{\lambda_j^{*2} + \lambda_k^{*2}}{2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_j = z'_j, z_k \neq z'_k \right] \\
& + \mathbb{E}_{\mathcal{J}_{z, z'} \sim \zeta} \left[\sqrt{\frac{\lambda_j^{*2} + \lambda_k^{*2}}{2} + \sum_{i \in J_e \setminus \{j, k\}} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \middle| z_k \neq z'_k, z_j = z'_j \right].
\end{aligned}$$

where two equality follows Assumption A.4.1 and the inequality follows Jensen's inequality. Let $\phi' = \mathbb{U} \Lambda' \mathbb{Q}^{-1}$ with corresponding λ' , where $\lambda'_j = \lambda'_k = \sqrt{\frac{\lambda_j^{*2} + \lambda_k^{*2}}{2}}$ and for any $i \in J_e \setminus \{j, k\}$, $\lambda'_i = \lambda_i^*$. We have $\|\phi'\|_F = \|\phi_\zeta^*\|_F$ and $\mathcal{L}_{\text{sup}}(\phi_\zeta^*) > \mathcal{L}_{\text{sup}}(\phi')$ which is a contradiction as we have ϕ_ζ^* is the optimal solution. Thus, for any $j, k \in J_e$, we have $\lambda_j^* = \lambda_k^*$ and we finish the proof under simple calculation. \square

Now, we are ready to analyze consistency and diversity under this linear case study.

Consistency

The intuition is that ζ not only covers \mathcal{C}_0 but contains too much unrelated information. Recall that the consistent term in Definition 3.3.3 is

$$\kappa = \sup_{\mathcal{T}_0 \subseteq \mathcal{C}_0} [\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_0^*)].$$

We first define some notation we will use later. Let ζ_0 be a multitask finetuning tasks distribution over $\mathcal{C}_0 \times \mathcal{C}_0$ and denote the corresponding optimal representation model as ϕ_0^* . Suppose for any target task \mathcal{T}_0 contains two latent classes z, z' from \mathcal{C}_0 . W.l.o.g., denote z, z' differ in n_0 entries ($1 \leq n_0 \leq k_0$), whose n_e entries fall in L_e , where $0 \leq n_e \leq n_0$. Then, we get the lemma below:

Lemma A.4.4. *Assume Assumption A.4.1 and Assumption A.4.2. We have*

$$\kappa = \sup_{\mathcal{T}_0 \subseteq \mathcal{C}_0} [\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_0^*)] = \text{BR} \left(\sqrt{\frac{n_0}{k_0}} - \sqrt{\frac{n_e}{k_e}} \right). \quad (\text{A.72})$$

Proof of Theorem A.4.4. Recall $1 \leq n_0 \leq k_0$ and $0 \leq n_e \leq n_0$. By Theorem A.4.3, we have $\phi_\zeta^* = \mathbf{U}\Lambda^*\mathbf{Q}^{-1}$, where \mathbf{U} is any orthonormal matrix, $\Lambda^* = \text{diag}(\lambda^*)$. For any $i \in J_e$, $\lambda_i^* = \frac{R}{\sqrt{k_e}}$ and $\lambda_i^* = 0$ otherwise. We also have $\phi_0^* = \mathbf{U}_0\Lambda_0^*\mathbf{Q}^{-1}$, where \mathbf{U}_0 is any orthonormal matrix, $\Lambda_0^* = \text{diag}(\lambda^{0,*})$. For any $i \in J_0$, $\lambda_i^{0,*} = \frac{R}{\sqrt{k_0}}$ and $\lambda_i^{0,*} = 0$ otherwise. Thus, we have

$$\kappa = \sup_{\mathcal{T}_0 \subseteq \mathcal{C}_0} [\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_0^*)] \quad (\text{A.73})$$

$$= \text{BR} \left(\sqrt{\frac{n_0}{k_0}} - \sqrt{\frac{n_e}{k_e}} \right). \quad (\text{A.74})$$

□

Let $n'_e = k_e - n_e$. Note this k_e is an increasing factor if \mathcal{C} contains more features. Moreover, n_e is the number of features encoded by both target and training data, representing the information of target data covered by training data, n_e increases as more target information covered by training data, the loss will decrease. n'_e

is the number of features encoded in training data but not encoded by target data, representing the un-useful information, n'_e increases as more un-related information is covered by training data, the loss will increase.

Diversity

We first review some definitions in Definition 3.3.2. The **averaged representation difference** for two model $\phi, \tilde{\phi}$ on a distribution ζ over tasks is

$$\bar{d}_\zeta(\phi, \tilde{\phi}) := \mathbb{E}_{\mathcal{T} \sim \zeta} [\mathcal{L}_{\text{sup}}(\mathcal{T}, \phi) - \mathcal{L}_{\text{sup}}(\mathcal{T}, \tilde{\phi})] = \mathcal{L}_{\text{sup}}(\phi) - \mathcal{L}_{\text{sup}}(\tilde{\phi}). \quad (\text{A.75})$$

The **worst-case representation difference** between representations $\phi, \tilde{\phi}$ on the family of classes \mathcal{C}_0 is

$$d_{\mathcal{C}_0}(\phi, \tilde{\phi}) := \sup_{\mathcal{T}_0 \subseteq \mathcal{C}_0} |\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \tilde{\phi})|. \quad (\text{A.76})$$

We say the model class Φ has ν -diversity for ζ and \mathcal{C}_0 if for any $\phi \in \Phi$ and ϕ_ζ^* ,

$$d_{\mathcal{C}_0}(\phi, \phi_\zeta^*) \leq \bar{d}_\zeta(\phi, \phi_\zeta^*)/\nu. \quad (\text{A.77})$$

To find the minimum value of ν in Definition 3.3.2, we need further information about ζ . For simplicity, we have a fixed distance assumption, e.g., uniformly sampling from $\{(+1, +1, -1), (+1, -1, +1), (-1, +1, +1)\}$. Then, we consider two different cases below. We consider that all $\mathcal{T}_0 \subseteq \mathcal{C}_0$ such containing z, z' that differ in only 1 entry.

Assumption A.4.5 (Fixed Distance). *We assume for any multitask finetuning tasks distribution ζ , for any two latent classes $(z, z') \sim \zeta$, we have z, z' differ in n_k entries.*

Case $L_e \neq J_0$. In this case, $J_0 \setminus L_e \neq \emptyset$, we have the features learned in multitask finetuning that do not cover all features used in the target task. Then, we have the following lemma, which means if $L_e \neq J_0$ we can have infinitesimal ν to satisfy the diversity definition.

Lemma A.4.6. *Assume Assumption A.4.1, Assumption A.4.2 and Assumption A.4.5. When $L_e \neq J_0$, we have $\nu \rightarrow 0$.*

Proof of Theorem A.4.6. As features in \mathcal{C}_0 not covered by \mathcal{C} , we can always find a \mathcal{T}_0 such containing z, z' that only differ in entries in $J_0 \setminus L_e$, we say as entry \tilde{i} .

By Theorem A.4.3, we have $\phi_\zeta^* = U\Lambda^*Q^{-1}$, where U is any orthonormal matrix, $\Lambda^* = \text{diag}(\lambda^*)$. For any $i \in J_e$, $\lambda_i^* = \frac{R}{\sqrt{k_e}}$ and $\lambda_i^* = 0$ otherwise. We have $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) = 0$ and by Equation (A.71),

$$\mathcal{L}_{\text{sup}}(\phi_\zeta^*) = -B \mathbb{E}_{\mathcal{T}_{z,z'} \sim \zeta} \left[\sqrt{\sum_{i \in J_e} \lambda_i^{*2} \mathbb{1}[z_i \neq z'_i]} \right] \quad (\text{A.78})$$

$$= -BR \sqrt{\frac{n_k}{k_e}}. \quad (\text{A.79})$$

On the other hand, for any $\phi \in \Phi$, we have $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi) = -B|\lambda_{\tilde{i}}|$. Thus, we have

$$\begin{aligned} \nu &= \min_{\phi \in \Phi} \frac{\mathcal{L}_{\text{sup}}(\phi) - \mathcal{L}_{\text{sup}}(\phi_\zeta^*)}{|\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*)|} \\ &= \min_{\phi \in \Phi} \frac{\mathcal{L}_{\text{sup}}(\phi) + BR \sqrt{\frac{n_k}{k_e}}}{B|\lambda_{\tilde{i}}|} \\ &= \min_{\phi \in \Phi} \frac{-B \mathbb{E}_{\mathcal{T}_{z,z'} \sim \zeta} \left[\sqrt{\sum_{i \in J_e} \lambda_i^2 \mathbb{1}[z_i \neq z'_i]} \right] + BR \sqrt{\frac{n_k}{k_e}}}{B|\lambda_{\tilde{i}}|} \\ &\leq \frac{-\mathbb{E}_{\mathcal{T}_{z,z'} \sim \zeta} \left[\sqrt{\sum_{i \in J_e} \frac{R^2 - \lambda_i^2}{k_e} \mathbb{1}[z_i \neq z'_i]} \right] + R \sqrt{\frac{n_k}{k_e}}}{|\lambda_{\tilde{i}}|} \\ &= \frac{-\sqrt{\frac{(R^2 - \lambda_{\tilde{i}}^2)n_k}{k_e}} + R \sqrt{\frac{n_k}{k_e}}}{|\lambda_{\tilde{i}}|}, \end{aligned} \quad (\text{A.80})$$

where the first inequality is by constructing a specific ϕ . Note that Equation (A.80) $\rightarrow 0$ when $|\lambda_{\tilde{i}}| \rightarrow 0$. ϕ is constructed as: for any $i \in J_e$, $\lambda_i = \sqrt{\frac{R^2 - \lambda_{\tilde{i}}^2}{k_e}}$ and $|\lambda_{\tilde{i}}| \rightarrow 0$. Thus, we finish the proof. \square

Case $L_e = J_0$. In this case $J_0 \setminus L_e = \emptyset$, we have all features in \mathcal{C}_0 covered by \mathcal{C} .

Lemma A.4.7. *Assume Assumption A.4.1, Assumption A.4.2 and Assumption A.4.5. When all $\mathcal{T}_0 \subseteq \mathcal{C}_0$ such containing z, z' that differ in only 1 entry and $L_e = J_0$, we have ν is lower bounded by some constant $\tilde{c} = \sqrt{n_k} \left(1 - \sqrt{\frac{1}{k_e(k_e-1)}} \left(\sqrt{n_k(n_k-1)} + k_e - n_k \right) \right)$.*

Proof of Theorem A.4.7. We say the differ entry in \mathcal{T}_0 as entry \tilde{i} . By Theorem A.4.3, we have $\phi_\zeta^* = U\Lambda^*Q^{-1}$, where U is any orthonormal matrix, $\Lambda^* = \text{diag}(\lambda^*)$. For any $i \in J_e, \lambda_i^* = \frac{R}{\sqrt{k_e}}$ and $\lambda_i^* = 0$ otherwise. By Equation (A.71), we have $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*) = -\text{BR}\sqrt{\frac{n_k}{k_e}}$ and $\mathcal{L}_{\text{sup}}(\phi_\zeta^*) = -\text{BR}\sqrt{\frac{n_k}{k_e}}$.

On the other hand, for any $\phi \in \Phi$, we have $\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi) = -B|\lambda_{\tilde{i}}|$. Thus, by Assumption A.4.1, we have

$$\begin{aligned}
\nu &= \min_{\mathcal{T}_0 \subseteq \mathcal{C}_0, \phi \in \Phi} \frac{\mathcal{L}_{\text{sup}}(\phi) - \mathcal{L}_{\text{sup}}(\phi_\zeta^*)}{|\mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi) - \mathcal{L}_{\text{sup}}(\mathcal{T}_0, \phi_\zeta^*)|} \\
&= \min_{\mathcal{T}_0 \subseteq \mathcal{C}_0, \phi \in \Phi} \frac{\mathcal{L}_{\text{sup}}(\phi) + \text{BR}\sqrt{\frac{n_k}{k_e}}}{|-B|\lambda_{\tilde{i}}| + \text{BR}\sqrt{\frac{1}{k_e}}|} \\
&= \min_{\mathcal{T}_0 \subseteq \mathcal{C}_0, \phi \in \Phi} \frac{-B \mathbb{E}_{\mathcal{T}_{z,z'} \sim \zeta} \left[\sqrt{\sum_{i \in J_e} \lambda_i^2 \mathbb{1}[z_i \neq z'_i]} \right] + \text{BR}\sqrt{\frac{n_k}{k_e}}}{|-B|\lambda_{\tilde{i}}| + \text{BR}\sqrt{\frac{1}{k_e}}|} \\
&= \min_{\mathcal{T}_0 \subseteq \mathcal{C}_0, \phi \in \Phi} \frac{-\mathbb{E}_{\mathcal{T}_{z,z'} \sim \zeta} \left[\sqrt{\lambda_{\tilde{i}}^2 \mathbb{1}[z_{\tilde{i}} \neq z'_{\tilde{i}}]} + \sum_{i \in J_e \setminus \{\tilde{i}\}} \frac{R^2 - \lambda_i^2}{k_e - 1} \mathbb{1}[z_i \neq z'_i]} \right] + R\sqrt{\frac{n_k}{k_e}}}{|-|\lambda_{\tilde{i}}| + R\sqrt{\frac{1}{k_e}}|} \\
&= \min_{\mathcal{T}_0 \subseteq \mathcal{C}_0, \phi \in \Phi} \frac{-\left[\frac{n_k}{k_e} \sqrt{\lambda_{\tilde{i}}^2 + \frac{R^2 - \lambda_{\tilde{i}}^2}{k_e - 1} (n_k - 1)} + \frac{k_e - n_k}{k_e} \sqrt{\frac{n_k(R^2 - \lambda_{\tilde{i}}^2)}{k_e - 1}} \right] + R\sqrt{\frac{n_k}{k_e}}}{|-|\lambda_{\tilde{i}}| + R\sqrt{\frac{1}{k_e}}|} \\
&= \sqrt{n_k} \left(1 - \sqrt{\frac{1}{k_e(k_e-1)}} \left(\sqrt{n_k(n_k-1)} + k_e - n_k \right) \right), \tag{A.81}
\end{aligned}$$

where the last equality take $\lambda_{\tilde{i}} = 0$. □

A.4.3 Proof of Main Results

Proof of Theorem 3.3.6. Note that $R = B = n_0 = k_0 = 1, n_k = 2$.

We see that ζ satisfies Assumption A.4.1, Assumption A.4.2 and Assumption A.4.5. We finish the proof by Theorem A.4.4, Theorem A.4.6 and Theorem A.4.7 with some simple calculations. \square

Thus, we can link our diversity and consistency parameters to the number of features in z encoded by training tasks or target tasks. Based on this intuition, we propose a selection algorithm, where selection is based on x , we want to select data that encodes more relevant features of z , this can be achieved by comparing x from target data and training data either using cosine similarity or KDE.

A.5 Vision Experimental Results

We first provide a summary of dataset and protocol we use, we provide details in following sections.

Datasets and Models. We use four widely used few-shot learning benchmarks: miniImageNet (Vinyals et al., 2016), tieredImageNet (Ren et al., 2018), DomainNet (Peng et al., 2019) and Meta-dataset (Triantafillou et al., 2020), following the protocol in Chen et al. (2021b); Tian et al. (2020b). We use exemplary foundation models with different pretraining schemes (MoCo-v3 (Chen et al., 2021a), DINO-v2 (Oquab et al., 2024), and supervised learning with ImageNet (Russakovsky et al., 2015)) and architectures (ResNet (He et al., 2016) and ViT (Dosovitskiy et al., 2021)).

Experiment Protocol. We consider few-shot tasks consisting of N classes with K support samples and Q query samples per class (known as N -way K -shot). The goal is to classify the query samples into the N classes based on the support samples. Tasks used for finetuning are constructed by samples from the training split. Each task is formed randomly by sampling 15 classes, with every class drawing 1 or

5 support samples and 10 query samples. Target tasks are similarly constructed, yet from the test set. We follow (Chen et al., 2021b) for multitask finetuning and target task adaptation. During multitask finetuning, we update all parameters in the model using a nearest centroid classifier, in which all samples are encoded, class centroids are computed, and cosine similarity between a query sample and those centroids are treated as the class logits. For adaptation to a target task, we only retain the model encoder and consider a similar nearest centroid classifier. This experiment protocol applies to all three major experiments (Sections 3.4.1 to 3.4.3).

A.5.1 Datasets

The miniImageNet dataset is a common benchmark for few-shot learning. It contains 100 classes sampled from ImageNet, then is randomly split into 64, 16, and 20 classes as training, validation, and testing set respectively.

The tieredImageNet dataset is another widely used benchmark for few-shot learning. It contains 608 classes from 34 super-categories sampled from ImageNet. These categories are then subdivided into 20 training categories with 351 classes, 6 validation categories with 97 classes, and 8 testing categories with 160 classes

DomainNet is the largest domain adaptation benchmark with about 0.6 million images. It consists of around 0.6 million images of 345 categories from 6 domains: clipart (clp), infograph (inf), quickdraw (qdr), real (rel) and sketch (skt). We split it into 185, 65, 100 classes as training, validation, and testing set respectively. We conduct experiments on Sketch (skt) subsets.

Meta-Dataset encompasses ten publicly available image datasets covering a wide array of domains: ImageNet-1k, Omniglot, FGVC-Aircraft, CUB-200-2011, Describable Textures, QuickDraw, FGVCx Fungi, VGG Flower, Traffic Signs, and MSCOCO. Each of these datasets is split into training, validation, and testing subsets. For additional information on the Meta-Dataset can be found in Appendix 3 of Triantafillou et al. (2020).

A.5.2 Experiment Protocols

Our evaluation and the finetuning process take the form of few-shot tasks, where a target task consists of N classes with K support samples and Q query samples in each class. The objective is to classify the query samples into the N classes based on the support samples. To accomplish this, we take the support samples in each class and feed them through an image encoder to obtain representations for each sample. We then calculate the average of these representations within each class to obtain the centroid of each class. For a given query sample x , we compute the probability that x belongs to class y based on the cosine similarity between the representation of x and the centroid of class y .

In our testing stage, we constructed 1500 target tasks, each consisting of 15 classes randomly sampled from the test split of the dataset. Within each class, we randomly selected 1 or 5 of the available images as shot images and 15 images as query images. These tasks are commonly referred to as 1-shot or 5-shot tasks. We evaluated the performance of our model on these tasks and reported the average accuracy along with a 95% confidence interval.

During multitask finetuning, the image encoder is directly optimized on few-shot classification tasks. To achieve this, we construct multitasks in the same format as the target tasks and optimize from the same evaluation protocol. Specifically, we create a total of 200 finetuning tasks, each task consists of 15 classes sampled from the train split of data, where each class contains 1 support image and 9 query images, resulting in 150 images per task. The classes in a finetuning task are sampled from the train split of the data.

To ensure a fair comparison with the finetuning baseline, we used the same training and testing data, as well as batch size, and applied standard finetuning. During standard finetuning, we added a linear layer after the encoder and trained the model. We also utilized the linear probing then finetuning (LP-FT) technique proposed by Kumar et al. (2022), which has been shown to outperform finetuning alone on both in-distribution and out-of-distribution data. In the testing stage, we removed the linear layer and applied the same few-shot testing pipeline to the

finetuned encoders.

For task selection, we employ the CLIP ViT-B image encoder to obtain image embeddings. We assess consistency by measuring the cosine similarity of the mean embeddings and we evaluate diversity through a coverage score derived from the ellipsoid formula outlined in Section 3.3.2.

For optimization, we use the SGD optimizer with momentum 0.9, the learning rate is $1e-5$ for CLIP and moco v3 pretrained models, and is $2e-6$ for DINO v2 pretrained models. The models were finetuned over varying numbers of epochs in each scenario until they reached convergence.

A.5.3 Existence of Task Diversity

Task diversity is crucial for the foundation model to perform well on novel classes in target tasks.

In this section, we prove for task satisfying consistency, greater diversity in the related data can help reduce the error on the target task. Specifically, for the target task, where the target tasks data originates from the test split of a specific dataset, we utilized the train split of the same dataset as the finetuning tasks data. Then finetuning tasks satisfied consistency. In experiments, we varied the number of classes accessible to the model during the finetuning stage, while keeping the total sample number the same. This serves as a measure of the diversity of training tasks.

miniImageNet and Omniglot

We show the results of CLIP encoder on miniImageNet and Omniglot. We vary the number of classes model access to in finetuning stage. The number of classes varies from all classes, i.e., 64 classes, to 8 classes. Each task contains 5 classes. For finetuning tasks, each class contains 1 shot image and 10 query images. For target tasks, each class contains the 1-shot image and 15 query images.

Table A.1 shows the accuracy of ViT-B32 across different numbers of classes during the finetuning stage. The “Class 0” represents direct evaluation without any finetuning. We observe that finetuning the model leads to an average accuracy

| # limited classes | 64 | 32 | 16 | 8 | 0 |
|-------------------|------------------|------------------|------------------|------------------|------------------|
| Accuracy | 90.02 ± 0.15 | 88.54 ± 1.11 | 87.94 ± 0.22 | 87.07 ± 0.20 | 83.03 ± 0.24 |

Table A.1: Class diversity on ViT-B32 backbone on miniImageNet.

improvement of 4%. Furthermore, as the diversity of classes increases, we observe a corresponding increase in performance. This indicates that incorporating a wider range of classes during the finetuning process enhances the model’s overall accuracy.

For task diversity, we also use dataset Omniglot (Lake et al., 2015). The Omniglot dataset is designed to develop more human-like learning algorithms. It contains 1623 different handwritten characters from 50 different alphabets. The 1623 classes are divided into 964, 301, and 358 classes as training, validation, and testing sets respectively. We sample multitask in finetuning stage from training data and the target task from testing data.

| # limited classes | 964 | 482 | 241 | 50 | 10 | 0 |
|-------------------|------------------|------------------|------------------|------------------|------------------|------------------|
| Accuracy | 95.35 ± 0.14 | 95.08 ± 0.14 | 94.29 ± 0.15 | 88.48 ± 0.20 | 80.26 ± 0.24 | 74.69 ± 0.26 |

Table A.2: Class diversity on ViT-B32 backbone on Omniglot.

Table A.2 shows the accuracy of ViT-B32 on different numbers of classes in finetuning stage, where class 0 indicates direct evaluation without finetuning. Finetuning improves the average accuracy by 5.5%. As class diversity increases, performance increases.

tieredImageNet

We then show results on tieredImageNet across learning settings for the ViT-B backbone. We follow the same setting where we restrain each task that contains 15 classes.

We found that using more classes from related data sources during finetuning improves accuracy. This result indicates that upon maintaining consistency, a trend is observed where increased diversity leads to an enhancement in performance.

| Pretrained | 351 | 175 | 43 | 10 |
|-------------------|------------|------------|-----------|-----------|
| DINOv2 | 84.74 | 82.75 | 82.60 | 82.16 |
| CLIP | 68.57 | 67.70 | 67.06 | 63.52 |
| Supervised | 89.97 | 89.69 | 89.19 | 88.92 |

Table A.3: The performance of the ViT-B backbone using different pretraining methods on tieredImageNet, varying the number of classes accessible to the model during the finetuning stage. Each column represents the number of classes within the training data.

A.5.4 Ablation Study

In Section 3.4 and the result in Table 3.2, we utilize the train split from the same dataset to construct the finetuning data. It is expected that the finetuning data possess a diversity and consistency property, encompassing characteristics that align with the test data while also focusing on its specific aspects.

In the following ablation study, we explore the relationship between the diversity and consistency of data in finetuning tasks, sample complexity, and finetuning methods. We seek to answer the following questions: Does multitask finetuning benefit only from certain aspects? How do these elements interact with each other?

Violate both consistency and diversity: Altering Finetuning Task Data with Invariant Sample Complexity

In this portion, we examine the performance when the model is finetuned using data completely unrelated to the target task data. With the same finetuning sample complexity, the performance cannot be compared to the accuracy we have currently attained.

In this section, we present the performance of MoCo v3 with a ViT-B backbone on the DomainNet dataset. We finetuned the model using either ImageNet data or DomainNet train-split data and evaluated its performance on the test-split of DomainNet. We observed that finetuning the model with data selected from the DomainNet train-split resulted in improved performance on the target task. This

finding aligns with our expectations and highlights the significance of proper finetuning data selection.

When considering the results presented in Table A.4, we also noticed that for MoCo v3 with a ResNet50 backbone and DINO v2 with a ViT-S backbone, multitask finetuning on ImageNet led to a decrease in model performance compared to direct adaptation. This suggests that inappropriate data selection can have a detrimental effect on the final model performance. This conclusion is also supported by the findings of Kumar et al. (2022).

| pretrained | backbone | FT data | Accuracy |
|-------------------|-----------------|----------------|-----------------|
| MoCo v3 | ViT-B | ImageNet | 24.88 (0.25) |
| | | DomainNet | 32.88 (0.29) |
| | ResNet50 | ImageNet | 27.22 (0.27) |
| | | DomainNet | 33.53 (0.30) |
| DINO v2 | ViT-S | ImageNet | 51.69 (0.39) |
| | | DomainNet | 61.57 (0.40) |
| | ViT-B | ImageNet | 62.32 (0.40) |
| | | DomainNet | 68.22 (0.40) |
| Supervised | ViT-B | ImageNet | 31.16 (0.31) |
| | | DomainNet | 48.02 (0.38) |
| | ResNet50 | ImageNet | 29.56 (0.28) |
| | | DomainNet | 39.09 (0.34) |

Table A.4: Finetuning data selection on model performance. FT data: dataset we select for multitask finetuning. Report the accuracy on the test-split of DomainNet.

Violating consistency while retaining diversity: The Trade-Off between Task Consistency and Sample Complexity

Finetuning tasks with superior data are expected to excel under identical complexity, a natural question can be proposed: Does additional data enhance performance? Our results in this section negate this question. Testing the model on the DomainNet test-split, we employ two settings. In the first setting, we finetune the model on the DomainNet train-split. In the second, the model is finetuned with a combination of the same data from DomainNet as in the first setting, along with additional data from ImageNet.

Within our theoretical framework, mixing data satisfies diversity but fails consistency. The finetuning data, although containing related information, also encompasses excessive unrelated data. This influx of unrelated data results in a larger consistency parameter κ in our theoretical framework, adversely impacting model performance on the target task. We offer empirical evidence to affirm our theoretical conclusion.

| Pretrained | DomainNet | DomainNet + ImageNet |
|-------------------|------------------|-----------------------------|
| DINOv2 | 68.22 | 66.93 |
| CLIP | 64.97 | 63.48 |
| Supervised | 48.02 | 43.76 |

Table A.5: Results evaluating on DomainNet test-split using ViT-B backbone. First column shows performance where model finetune on data from DomainNet train-split alone, second column shows the performance of the model finetuned using a blend of the same data from DomainNet, combined with additional data from ImageNet.

Table A.5 shows mixed data of domainNet and ImageNet will doesn't provide the same advantages as using only DomainNet data. In this case, an increasing in data does not necessarily mean better performance.

Diversity and Consistency of Task Data and Finetuning Methods

To provide a more comprehensive understanding of the impact of task data and finetuning methods on model performance, we conduct additional experiments, utilizing varying finetuning methods and data. The model is tested on the DomainNet test split. We employ either multitask finetuning or standard finetuning, where a linear layer is added after the pretrained model. This linear layer maps the representations learned by encoders to the logits. The data of finetuning tasks derive from either the DomainNet train-split or ImageNet.

| Pretrained | 1 Adaptation | 2 ImageNet (SFT) | 3 ImageNet (Ours) | 4 DomainNet (SFT) | 5 DomainNet (Ours) |
|------------|-----------------|---------------------|----------------------|----------------------|-----------------------|
| DINOv2 | 61.65 | 59.80 | 62.32 | 61.84 | 68.22 |
| CLIP | 46.39 | 46.50 | 58.94 | 47.72 | 64.97 |
| Supervised | 28.70 | 28.52 | 31.16 | 30.93 | 48.02 |

Table A.6: Results evaluating on DomainNet test-split using ViT-B backbone. Adaptation: Direction adaptation without finetuning; SFT: Standard finetuning; Ours: Our multitask finetuning. Col-1 shows performance without any finetuning, Col-2,3,4,5 shows performance with different finetuning methods and data.

In Table A.6, we detail how data quality and finetuning methods of tasks impact the ultimate performance. Standard finetuning (SFT) with unrelated data diminishes performance compared to direct adaptation (col-1 vs col-2). On the other hand, multitask finetuning using unrelated data (ImageNet), or SFT with related data (DomainNet), both outperform direct adaptation. However, multitask finetuning with unrelated data proves more beneficial than the latter (col-3 vs col-4). The peak performance is attained through multitask finetuning on related data (col-5).

Ablation Study on Task Selection Algorithm

We show a simplified diagram for task selection in Figure 3.2.

We first provide some details of Table 3.1. We first create an array of finetuning tasks, and then apply our task selection algorithm to these tasks. Specifically, we

| Pretrained | Selection | INet | Omglot | Acraft | CUB | QDraw | Fungi | Flower | Sign | COCO |
|------------|-----------|-------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| CLIP | All | 60.87 | 70.53 | 31.67 | 66.98 | 40.28 | 34.88 | 80.80 | 37.82 | 33.71 |
| | Selected | 60.87 | 77.93 | 32.02 | 69.15 | 42.36 | 36.66 | 80.92 | 38.46 | 37.21 |
| DINOv2 | All | 83.04 | 72.92 | 36.52 | 94.01 | 49.65 | 52.72 | 98.54 | 34.59 | 47.05 |
| | Selected | 83.04 | 80.29 | 36.91 | 94.12 | 52.21 | 53.31 | 98.65 | 36.62 | 50.09 |
| MoCo v3 | All | 59.62 | 60.85 | 18.72 | 40.49 | 40.96 | 32.65 | 59.60 | 33.94 | 33.42 |
| | Selected | 59.62 | 63.08 | 19.03 | 40.74 | 41.16 | 32.89 | 59.64 | 35.25 | 33.51 |

Table A.7: Results evaluating our task selection algorithm on Meta-dataset using ViT-B backbone.

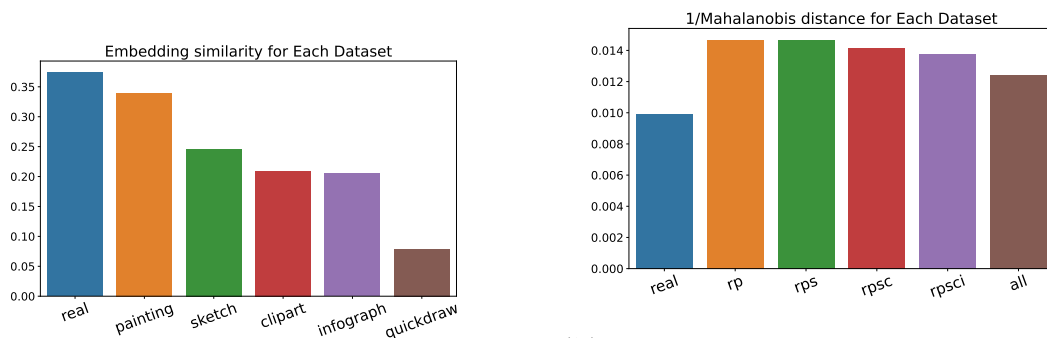
design 100 finetuning tasks by randomly selecting 15 classes, each providing 1 support sample and 10 query samples. The target tasks remain consistent with those discussed in Section 3.4. For a more comprehensive analysis of our algorithm, we performed ablation studies on the task selection algorithm, concentrating solely on either consistency or diversity, while violating the other. **Violate Diversity:** If the algorithm terminates early without fulfilling the stopping criteria, the data utilized in finetuning tasks fails to encompass all the attributes present in the target data. This leads to a breach of the diversity principle. **Violate Consistency:** Conversely, if the algorithm persists beyond the stopping criteria, the finetuning tasks become overly inclusive, incorporating an excessive amount of unrelated data, thus breaching the consistency.

This section details an ablation study on task selection for the dataset, we implement our task selection process on a meta-dataset, treating each dataset as a distinct task and choosing datasets to serve as data sources for the finetuning tasks. We show the result in Table A.7.

Table A.7 indicates that maintaining both consistency and diversity in the task selection algorithm is essential for optimal performance. This is evident from the comparison between the Random selection and the our approach, where the latter often shows improved performance across multiple datasets. ImageNet as the target task is an exception where the two approaches give the best results. Due to its extensive diversity, all samples from all other datasets are beneficial for finetuning. Consequently, the task selection algorithm tends to select all the candidate tasks.

A.5.5 Task Selection Algorithm on DomainNet

We verify our task selection algorithm by applying it on DomainNet. Here, the mini-ImageNet test-split is regarded as the target task source, and diverse domains (such as clipart (clp), infograph (inf), quickdraw (qdr), real (rel), and sketch (skt)) are considered as sources for finetuning tasks. We view different domain datasets as distinct finetuning tasks. With 6 domains in focus, our objective is to select a subset that optimizes model performance. We systematically apply Algorithm 1. Initially, we calculate the cosine similarity of mean embeddings between each domain and target tasks, ordering them from most to least similar: real, painting, sketch, clipart, infograph, and quickdraw. Sequentially adding datasets in this order, the process continues until the diversity score (1 over Mahalanobis distance) stops exhibiting significant increase.



(a) Mean embedding similarity for each data sort from most similar to least similar.

(b) Diversity score when adding task one by one, where *rp*: *real* and *painting*; *rps*: *real* and *painting* and *sketch* and so on.

Figure A.1: Dataset selection based on consistency and diversity on domainNet. Figure A.1a shows the consistency. Figure A.1b shows the diversity.

As we can see in Figure A.1, the diversity does not increase when we just select *real* and *painting* as our finetuning task data. For a comprehensive analysis, each combination is finetuned and the model performance accuracy on the target task is displayed.

As we can see in Figure A.2, the accuracy aligns with the conclusions drawn

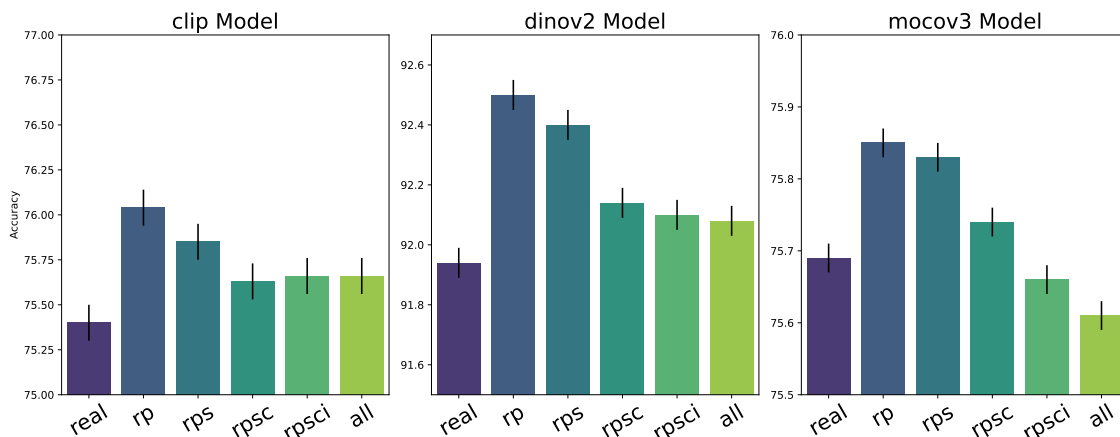


Figure A.2: Finetuning with different selection of domain datasets, where *rp*: *real* and *painting*; *rps*: *real* and *painting* and *sketch* and so on.

based on consistency and diversity. Remarkably, only *real* and *painting* suffice for the model to excel on the target task.

A.5.6 More Results with CLIP Encoder

In this section, we show additional results on CLIP (Radford et al., 2021) model.

| backbone | method | miniImageNet | | tieredImageNet | | DomainNet | |
|---------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|
| | | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot |
| CLIP-ViT32 | Direct Adaptation | 68.41 (0.54) | 87.43 (0.15) | 59.55 (0.21) | 79.51 (0.27) | 46.48 (0.37) | 72.01 (0.29) |
| | Standard FT | 69.39 (0.30) | 88.39 (0.15) | 61.20 (0.37) | 80.65 (0.27) | 47.72 (0.37) | 72.82 (0.29) |
| | Multitask FT (Ours) | 78.62 (0.15) | 93.22 (0.11) | 68.57 (0.37) | 84.79 (0.22) | 64.97 (0.39) | 80.49 (0.25) |
| CLIP-ResNet50 | Direct Adaptation | 61.31 (0.31) | 82.03 (0.18) | 51.76 (0.36) | 71.40 (0.30) | 40.55 (0.36) | 64.90 (0.31) |
| | Standard FT | 63.15 (0.31) | 83.45 (0.17) | 55.77 (0.35) | 75.28 (0.29) | 43.77 (0.38) | 67.30 (0.31) |
| | Multitask FT (Ours) | 67.03 (0.30) | 85.09 (0.17) | 57.56 (0.36) | 75.80 (0.28) | 52.67 (0.39) | 72.19 (0.30) |

Table A.8: **Comparison on 15-way classification.** Average few-shot classification accuracies (%) with 95% confidence intervals clip encoder.

We can observe from Table A.8 standard finetuning improves performance compared to direct adaptation. However, our proposed multitask finetuning approach consistently achieves even better results than the standard baseline.

Task (M) vs Sample (m). We vary the task size and sample size per task during finetuning. We verify the trend of different numbers of tasks and numbers of images per task. Each task contains 5 classes. For finetuning tasks, $m = 50$ indicates each class contains the 1-shot image and 9-query images. $m = 100$ indicates each class contains 2-shot and 18-query images. $m = 200$ indicates each class contains 4-shot and 36-query images. $M = m = 0$ indicates direct evaluation without finetuning. For target tasks, each class contains the 1-shot image and 15 query images.

| Task (M) \ Sample (m) | 0 | 50 | 100 | 200 |
|-----------------------|------------------|------------------------------------|------------------------------------|------------------------------------|
| 0 | 83.03 \pm 0.24 | | | |
| 200 | | 89.07 \pm 0.20 | 89.95 \pm 0.19 | 90.09 \pm 0.19 |
| 400 | | 89.31 \pm 0.19 | 90.11 \pm 0.19 | 90.70 \pm 0.18 |
| 800 | | 89.71 \pm 0.19 | 90.27 \pm 0.19 | 90.80 \pm 0.18 |

Table A.9: Accuracy with a varying number of tasks and samples (ViT-B32 backbone).

Table A.9 shows the results on the pretrained CLIP model using the ViT backbone. For direct adaptation without finetuning, the model achieves 83.03% accuracy. Multitask finetuning improves the average accuracy at least by 6%. For a fixed number of tasks or samples per task, increasing samples or tasks improves accuracy. These results suggest that the total number of samples ($M \times m$) will determine the overall performance, supporting our main theorem.

Few-shot Effect. We perform experiments on the few-shot effects of finetuning tasks. We aim to evaluate whether increasing the number of few-shot images in the finetuning task leads to significant improvements. Each finetuning task consists of 5 classes, and we maintain a fixed number of 10 query images per class while gradually increasing the number of shot images, as illustrated in Table A.10. As for the target tasks, we ensure each class contains 1 shot image and 15 query images for evaluation.

Table A.10 displays the accuracy results of ViT-B32 when varying the number of few-shot images in the finetuning tasks. We observe that increasing the number of few-shot images, thereby augmenting the sample size within each task, leads

| # shot images | 20 | 10 | 5 | 1 | 0 |
|---------------|------------------|------------------|------------------|------------------|------------------|
| Accuracy | 91.03 \pm 0.18 | 90.93 \pm 0.18 | 90.54 \pm 0.18 | 90.02 \pm 0.15 | 83.03 \pm 0.24 |

Table A.10: Few-shot effect on ViT-B32 backbone on miniImageNet.

| Task (M) | Sample (m) | 150 | 300 | 450 | 600 |
|----------|--------------|--------------|--------------|--------------|--------------|
| | 200 | 68.32 (0.35) | 71.42 (0.35) | 73.84 (0.35) | 75.58 (0.35) |
| 400 | 71.41 (0.35) | 75.60 (0.35) | 77.57 (0.34) | 78.66 (0.34) | |
| 600 | 73.85 (0.35) | 77.59 (0.34) | 79.04 (0.33) | 79.76 (0.33) | |
| 800 | 75.56 (0.35) | 78.68 (0.34) | 79.78 (0.33) | 80.26 (0.33) | |

Table A.11: Accuracy with a varying number of tasks and samples (ViT-B32 backbone).

to improved performance. This finding is quite surprising, considering that the finetuning tasks and target tasks have different numbers of shot images. However, this aligns with our understanding of sample complexity, indicating that having access to more training examples can enhance the model’s ability to generalize and perform better on unseen data.

A.5.7 Sample Complexity on Performance for tieredImageNet

We provide a table and visualization of the trend of the number of tasks and the number of samples per task for the MoCo v3 ViT model on tieredImageNet in Table A.11 and Figure A.3.

As demonstrated in the paper, we have observed that increasing the number of tasks generally leads to performance improvements, while keeping the number of samples per task constant. Conversely, when the number of samples per task is increased while maintaining the same number of tasks, performance also tends to improve. These findings emphasize the positive relationship between the number of tasks and performance, as well as the influence of sample size within each task.

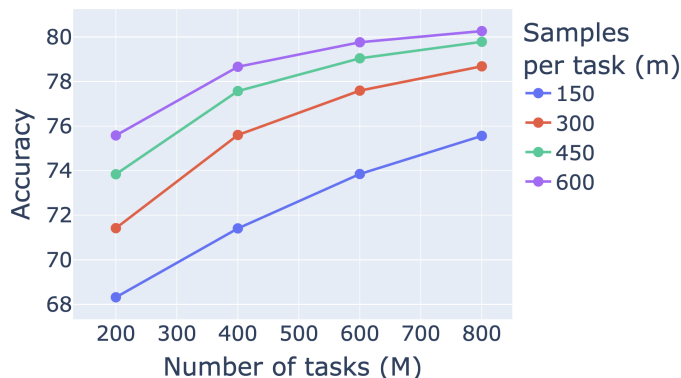


Figure A.3: Finetuning using tieredImageNet train-split, test on test-split.

A.5.8 Full results for Effectiveness of Multitask Finetuning

In this section, we provide another baseline in complement to the results in Section 3.4.3.

We incorporated the Model-Agnostic Meta-Learning (MAML) algorithm, as outlined by Finn et al. (2017), as another baseline for our few-shot tasks. MAML operates in a two-step process: it initially updates parameters based on within-episode loss (the inner loop), then it evaluates and updates loss based on learned parameters (the outer loop). We follow the pipeline in Triantafillou et al. (2020) to implement MAML for few-shot tasks. We show results in Table A.12.

Table A.12 reveals that MAML exhibits variable performance across different settings. For instance, it outperforms both Adaptation and Standard FT methods in scenarios like MoCo v3 ViT-B on miniImageNet, DomainNet, and ResNet 50 on supervised training for tieredImageNet. However, its performance is less impressive in other contexts, such as DINOv2 ViT-B on miniImageNet and ViT-B on supervised training for miniImageNet. This variability in performance is attributed to the constraints of our few-shot tasks, where the limited number of support samples restricts the model’s capacity to adapt to new tasks. Despite these fluctuations, our multitask finetuning approach consistently surpasses the mentioned baselines, often by a significant margin, across all evaluated scenarios.

| pretrained | backbone | method | miniImageNet | | tieredImageNet | | DomainNet | | |
|------------|-------------|------------------------------------|---------------------|---------------------|---------------------|---------------------|---------------------|---------------------|--------------|
| | | | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot | |
| MoCo v3 | ViT-B | Adaptation | 75.33 (0.30) | 92.78 (0.10) | 62.17 (0.36) | 83.42 (0.23) | 24.84 (0.25) | 44.32 (0.29) | |
| | | Standard FT | 75.38 (0.30) | 92.80 (0.10) | 62.28 (0.36) | 83.49 (0.23) | 25.10 (0.25) | 44.76 (0.27) | |
| | | MAML | 79.26 (0.28) | 93.02 (0.08) | 67.96 (0.32) | 84.66 (0.19) | 28.91 (0.39) | 51.12 (0.28) | |
| | | Ours | 80.62 (0.26) | 93.89 (0.09) | 68.32 (0.35) | 85.49 (0.22) | 32.88 (0.29) | 54.17 (0.30) | |
| | ResNet50 | Adaptation | 68.80 (0.30) | 88.23 (0.13) | 55.15 (0.34) | 76.00 (0.26) | 27.34 (0.27) | 47.50 (0.28) | |
| | | Standard FT | 68.85 (0.30) | 88.23 (0.13) | 55.23 (0.34) | 76.07 (0.26) | 27.43 (0.27) | 47.65 (0.28) | |
| | | MAML | 69.28 (0.26) | 88.78 (0.12) | 55.31 (0.32) | 75.51 (0.19) | 27.53 (0.39) | 47.73 (0.28) | |
| | | Ours | 71.16 (0.29) | 89.31 (0.12) | 58.51 (0.35) | 78.41 (0.25) | 33.53 (0.30) | 55.82 (0.29) | |
| DINO v2 | ViT-S | Adaptation | 85.90 (0.22) | 95.58 (0.08) | 74.54 (0.32) | 89.20 (0.19) | 52.28 (0.39) | 72.98 (0.28) | |
| | | Standard FT | 86.75 (0.22) | 95.76 (0.08) | 74.84 (0.32) | 89.30 (0.19) | 54.48 (0.39) | 74.50 (0.28) | |
| | | MAML | 86.67 (0.24) | 95.54 (0.08) | 74.63 (0.34) | 89.60 (0.19) | 52.72 (0.34) | 73.35 (0.28) | |
| | | Ours | 88.70 (0.22) | 96.08 (0.08) | 77.78 (0.32) | 90.23 (0.18) | 61.57 (0.40) | 77.97 (0.27) | |
| | ViT-B | Adaptation | 90.61 (0.19) | 97.20 (0.06) | 82.33 (0.30) | 92.90 (0.16) | 61.65 (0.41) | 79.34 (0.25) | |
| | | Standard FT | 91.07 (0.19) | 97.32 (0.06) | 82.40 (0.30) | 93.07 (0.16) | 61.84 (0.39) | 79.63 (0.25) | |
| | | MAML | 90.77 (0.18) | 97.20 (0.08) | 82.54 (0.32) | 92.88 (0.19) | 62.30 (0.39) | 79.01 (0.28) | |
| | | Ours | 92.77 (0.18) | 97.68 (0.06) | 84.74 (0.30) | 93.65 (0.16) | 68.22 (0.40) | 82.62 (0.24) | |
| | | Supervised pretraining on ImageNet | Adaptation | 94.06 (0.15) | 97.88 (0.05) | 83.82 (0.29) | 93.65 (0.13) | 28.70 (0.29) | 49.70 (0.28) |
| | | Standard FT | 95.28 (0.13) | 98.33 (0.04) | 86.44 (0.27) | 94.91 (0.12) | 30.93 (0.31) | 52.14 (0.29) | |
| ResNet50 | MAML | 95.35 (0.12) | 98.50 (0.08) | 86.79 (0.32) | 94.72 (0.19) | 30.53 (0.39) | 52.21 (0.28) | | |
| | Ours | 96.91 (0.11) | 98.76 (0.04) | 89.97 (0.25) | 95.84 (0.11) | 48.02 (0.38) | 67.25 (0.29) | | |
| | Adaptation | 81.74 (0.24) | 94.08 (0.09) | 65.98 (0.34) | 84.14 (0.21) | 27.32 (0.27) | 46.67 (0.28) | | |
| | Standard FT | 84.10 (0.22) | 94.81 (0.09) | 74.48 (0.33) | 88.35 (0.19) | 34.10 (0.31) | 55.08 (0.29) | | |
| | MAML | 82.07 (0.28) | 94.12 (0.08) | 75.69 (0.32) | 89.30 (0.19) | 35.10 (0.39) | 56.51 (0.28) | | |
| | Ours | 87.61 (0.20) | 95.92 (0.07) | 77.74 (0.32) | 89.77 (0.17) | 39.09 (0.34) | 60.60 (0.29) | | |

Table A.12: **Results of few-shot image classification.** We report average classification accuracy (%) with 95% confidence intervals on test splits. Adaptation: Direction adaptation without finetuning; Standard FT: Standard finetuning; MAML: MAML algorithm in Finn et al. (2017); Ours: Our multitask finetuning; 1-/5-shot: number of labeled images per class in the target task.

A.6 NLP Experimental Results

We first provide a summary of the experimental setting and results in the below subsection. Then we provide details in the following subsections.

A.6.1 Summary

To further validate our approach, we conducted prompt-based finetuning experiments on masked language models, following the procedure outlined in Gao et al. (2021a).

Datasets and Models. We consider a collection of 14 NLP datasets, covering 8 single-sentence and 6 sentence-pair English tasks. This collection includes tasks from the GLUE benchmark (Wang et al., 2018a), as well as 7 other popular sentence classification tasks. The objective is to predict the label based on a single sentence or a sentence-pair. Specifically, the goal is to predict sentiments for single sentences or to estimate the relationship between sentence pairs. Each of the datasets is split into training and test set. See details in Section A.6.2. We experiment with a pretrained model RoBERTa (Liu et al., 2019).

Experiment Protocols. We consider prompt-based finetuning for language models (Gao et al., 2021a). This approach turns a prediction task into a masked language modeling problem, where the model generates a text response to a given task-specific prompt as the label. Our experiment protocol follows Gao et al. (2021a). The experiments are divided into 14 parallel experiments, each corresponding to a dataset. For the few-shot experiment, we use test split data as the target task data and sample 16 examples per class from the train split as finetuning data. The evaluation metric is measured by prompt-based prediction accuracy.

During the testing stage, we conduct experiments in zero-shot and few-shot settings for a given dataset. In the zero-shot setting, we directly evaluate the model’s prompt-based prediction accuracy. In the few-shot setting, we finetune the model using support samples from the same dataset and assess its accuracy on the test split. For multitask finetuning, we select support samples from other datasets and construct tasks for prompt-based finetuning. We then evaluate the performance of the finetuned model on the target task. More details can be found in Section A.6.3.

Task Selection. We select datasets by using task selection algorithm of feature vectors, which are obtained by computing the representations of each dataset and analyzing their relationship. We first obtain text features for each data point in the dataset. We select few-shot samples for generating text features. For each example, we replace the masked word with the true label in its manual template, then we forward them through the BERT backbone. Then, we compute the first principal

| | SST-2 (acc) | SST-5 (acc) | MR (acc) | CR (acc) | MPQA (acc) | Subj (acc) | TREC (acc) | CoLA (Matt.) |
|---|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|------------------|
| Prompt-based zero-shot | 83.6 | 35.0 | 80.8 | 79.5 | 67.6 | 51.4 | 32.0 | 2.0 |
| Multitask FT zero-shot | 92.9 | 37.2 | 86.5 | 88.8 | 73.9 | 55.3 | 36.8 | -0.065 |
| Prompt-based FT [†] | 92.7 (0.9) | 47.4 (2.5) | 87.0 (1.2) | 90.3 (1.0) | 84.7 (2.2) | 91.2 (1.1) | 84.8 (5.1) | 9.3 (7.3) |
| Multitask Prompt-based FT + task selection | 92.0 (1.2) | 48.5 (1.2) | 86.9 (2.2) | 90.5 (1.3) | 86.0 (1.6) | 89.9 (2.9) | 83.6 (4.4) | 5.1 (3.8) |
| | 92.6 (0.5) | 47.1 (2.3) | 87.2 (1.6) | 91.6 (0.9) | 85.2 (1.0) | 90.7 (1.6) | 87.6 (3.5) | 3.8 (3.2) |
| | MNLI (acc) | MNLI-mm (acc) | SNLI (acc) | QNLI (acc) | RTE (acc) | MRPC (F1) | QQP (F1) | |
| Prompt-based zero-shot | 50.8 | 51.7 | 49.5 | 50.8 | 51.3 | 61.9 | 49.7 | |
| Multitask FT zero-shot | 63.2 | 65.7 | 61.8 | 65.8 | 74.0 | 81.6 | 63.4 | |
| Prompt-based FT [†] | 68.3 (2.3) | 70.5 (1.9) | 77.2 (3.7) | 64.5 (4.2) | 69.1 (3.6) | 74.5 (5.3) | 65.5 (5.3) | |
| Multitask Prompt-based FT + task selection | 70.9 (1.5) | 73.4 (1.4) | 78.7 (2.0) | 71.7 (2.2) | 74.0 (2.5) | 79.5 (4.8) | 67.9 (1.6) | |
| | 73.5 (1.6) | 75.8 (1.5) | 77.4 (1.6) | 72.0 (1.6) | 70.0 (1.6) | 76.0 (6.8) | 69.8 (1.7) | |

Table A.13: **Results of few-shot learning with NLP benchmarks.** All results are obtained using RoBERTa-large. We report mean (and standard deviation) of metrics over 5 different splits. †: Result in Gao et al. (2021a); FT: finetuning; task selection: select multitask data from customized datasets.

component to obtain a feature vector for each dataset. Dataset selection provides certain improvements on some datasets, as elaborated below. Further details can be found in Section A.6.4.

Results. Our results are presented in Table A.13. Again, our method outperforms direct adaptation on target tasks across most datasets. For zero-shot prediction, our method provides improvements on all datasets except CoLA. Our multitask finetuning approach results in performance improvements on 12 out of 15 target tasks for few-shot prediction, with the exceptions being SST-2, Subj, and CoLA. CoLA is also reported by Gao et al. (2021a) as an exception that contains non-grammatical sentences that are outside of the distribution of the pretrained language model. SST-2 already achieves high accuracy in zero-shot prediction, and our model performs best in such setting. Subj is unique in that its task is to predict whether a given sentence is subjective or objective, therefore multitasking with few-shot samples from other datasets may not provide significant improvement for this task.

A.6.2 Datasets and Models

The text dataset consisted of 8 single-sentence and 6 sentence-pair English tasks, including tasks from the GLUE benchmark (Wang et al., 2018a), as well as 7 other popular sentence classification tasks (SNLI (Bowman et al., 2015), SST-5 (Socher et al., 2013), MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), MPQA (Wiebe et al., 2005), Subj (Pang and Lee, 2004), TREC (Voorhees and Tice, 2000)). The objective was to predict the label based on a single sentence or a sentence-pair. Specifically, for single sentences, we aimed to predict their semantics as either positive or negative, while for sentence-pairs, we aimed to predict the relationship between them. We experiment with the pretrained model RoBERTa. We have 14 datasets in total. We split each dataset into train and test split, see details below. We experiment with the pretrained model RoBERTa.

We follow Gao et al. (2021a) in their train test split. We use the original development sets of SNLI and datasets from GLUE for testing. For datasets such as MR, CR, MPQA, and Subj that require a cross-validation evaluation, we randomly select 2,000 examples for testing and exclude them from training. For SST5 and TREC, we utilize their official test sets.

To construct multitask examples from support samples, we gather support samples from all datasets except the testing dataset. For each task, we randomly select ten support samples and prompt-based finetuning the model.

A.6.3 Experiment Protocols

Gao et al. (2021a) proposed a prompt-based finetuning pipeline for moderately sized language models such as BERT, RoBERTa. Prompt-based prediction converts the downstream prediction task as a (masked) language modeling problem, where the model directly generates a textual response also known as a label word, to a given prompt defined by a task-specific template. As an illustration, consider the SST-2 dataset, which comprises sentences expressing positive or negative sentiment. The binary classification task can be transformed into a masked prediction problem using the template `<S>, it was <MASK>.`, where `<S>` represents the input sentence

| Task | Template | Label words |
|-------|--|---|
| SST-2 | <S ₁ > It was [MASK] . | positive: great, negative: terrible |
| SST-5 | <S ₁ > It was [MASK] . | v.positive: great, positive: good, neutral: okay, negative: bad, v.negative: terrible |
| MR | <S ₁ > It was [MASK] . | positive: great, negative: terrible |
| CR | <S ₁ > It was [MASK] . | positive: great, negative: terrible |
| Subj | <S ₁ > This is [MASK] . | subjective: subjective, objective: objective |
| TREC | [MASK] : <S ₁ > | abbreviation: Expression, entity: Entity, description: Description human: Human, location: Location, numeric: Number |
| COLA | <S ₁ > This is [MASK] . | grammatical: correct, not_grammatical: incorrect |
| MNLI | <S ₁ > ? [MASK] , <S ₂ > | entailment: Yes, netural: Maybe, contradiction: No |
| SNLI | <S ₁ > ? [MASK] , <S ₂ > | entailment: Yes, netural: Maybe, contradiction: No |
| QNLI | <S ₁ > ? [MASK] , <S ₂ > | entailment: Yes, not_entailment: No |
| RTE | <S ₁ > ? [MASK] , <S ₂ > | entailment: Yes, not_entailment: No |
| MRPC | <S ₁ > [MASK] , <S ₂ > | equivalent: Yes, not_equivalent: No |
| QQP | <S ₁ > [MASK] , <S ₂ > | equivalent: Yes, not_equivalent: No |

Table A.14: Manual templates and label words that we used in our experiments, following Gao et al. (2021a).

and <MASK> is the label word (e.g., "great" or "terrible") that the model is supposed to predict, see full templates in Table A.14. Prompt-based finetuning updates the model with prompt-based prediction loss for a given example, such as a sentence or sentence-pair.

To conduct the few-shot experiment, we use all data from the test split as the target task data for each dataset, and sample 16 examples per class from the train split as the support samples. The experiments are divided into 14 parallel experiments, with each corresponding to one dataset. The evaluation accuracy is measured as the prompt-based prediction accuracy. We subsampled 5 different sets of few-shot examples to run replicates experiments and report average performance.

During the testing stage, for a given dataset (e.g. QNLI), we consider the entire test split as the target task and divide the experiment into zero-shot and few-shot settings. In the zero-shot setting, we directly evaluate the model by measuring the accuracy of prompt-based predictions. In the few-shot setting, we first prompt-based finetune the model with support samples from the same dataset (QNLI) and then evaluate the accuracy on the test split. This experimental protocol follows the same pipeline as described in Gao et al. (2021a).

To perform multitask finetuning for a target task on a particular dataset (e.g.

QNLI), we select support samples from other datasets (e.g. SST-2, Subj, QQP, etc.) as finetuning examples. We construct tasks using these examples and apply the same prompt-based finetuning protocol to multitask finetune the model on these tasks. Finally, we evaluate the performance of the finetuned model on the target task.

A.6.4 Task Selection

The importance of the relationship between the data used in the training tasks and the target task cannot be overstated in multitask finetuning. Our theory measures this relationship through diversity and consistency statements, which require that our finetuning data are diverse enough to capture the characteristics of the test data, while still focusing on the specific regions where the test data aligns. We visualize this diversity and relationship through the feature maps of the datasets.

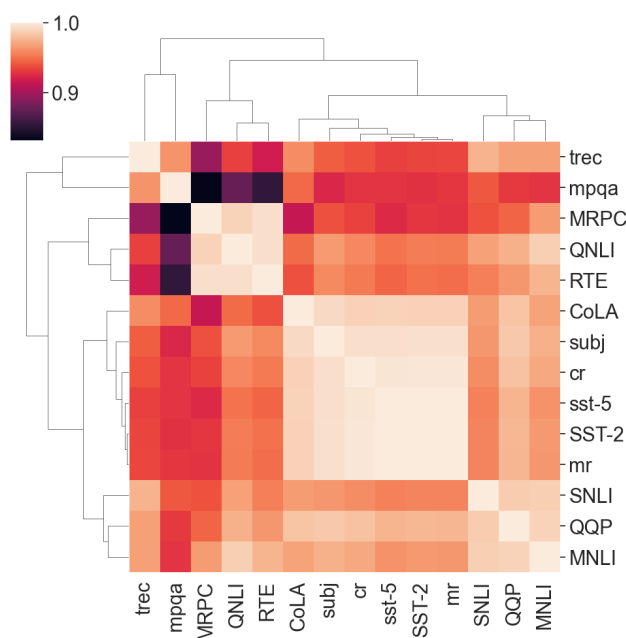


Figure A.4: Linear similarity among features vectors among 14 language datasets.

To visualize the relationship between feature vectors of different datasets, we

| |
|--------------------------------|
| cola: mr, cr,sst-2,sst-5,subj |
| sst-2: cola,mr, cr,sst-5,subj, |
| mrpc: qnli, rte |
| qqp: snli, mnli |
| mnli: snli, qqp |
| snli: qqp, mnli |
| qnli: mrpc, rte |
| rte: mrpc, qnli |
| mr: cola, cr,sst-2,sst-5,subj |
| sst-5: cola,mr, cr,sst-2,subj |
| subj: cola,mr, cr,sst-2,sst-5 |
| trec: mpqa |
| cr: cola,mr,sst-2,sst-5,subj |
| mpqa: trec |

Table A.15: Dataset selection.

first obtain text features for each data point in the dataset. We select few-shot samples for generating text features. For each example, we replace the masked word with the true label in its manual template, then we forward them through the BERT backbone. The reason for using BERT over RoBERTa is that the latter only has masked token prediction in pretraining, the [CLS] in pretrained RoBERTa model might not contain as much sentence information as BERT. Then, we compute the first principal component to obtain a feature vector for each dataset. We illustrate the relationship between these feature vectors in Figure A.4.

We further perform training data selection based on the task selection algorithm among the feature vectors, the selected dataset is shown in table Table A.15.

By performing task selection, we observed further improvements in multitask prompt-based finetuning on MR, CR, TREC, MNLI, QNLI, and QQP datasets. However, it's worth noting that the CoLA dataset is an exception, as it involves predicting

| | SST-2 (acc) | SST-5 (acc) | MR (acc) | CR (acc) | MPQA (acc) | Subj (acc) | TREC (acc) | CoLA (Matt.) |
|------------------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|------------------|
| Prompt-based zero-shot | 83.6 | 35.0 | 80.8 | 79.5 | 67.6 | 51.4 | 32.0 | 2.0 |
| Multitask FT zero-shot | 92.9 | 37.2 | 86.5 | 88.8 | 73.9 | 55.3 | 36.8 | -0.065 |
| + task selection | 92.5 | 34.2 | 87.1 | 88.7 | 71.8 | 72.0 | 36.8 | 0.001 |
| Prompt-based FT [†] | 92.7 (0.9) | 47.4 (2.5) | 87.0 (1.2) | 90.3 (1.0) | 84.7 (2.2) | 91.2 (1.1) | 84.8 (5.1) | 9.3 (7.3) |
| Multitask Prompt-based FT | 92.0 (1.2) | 48.5 (1.2) | 86.9 (2.2) | 90.5 (1.3) | 86.0 (1.6) | 89.9 (2.9) | 83.6 (4.4) | 5.1 (3.8) |
| + task selection | 92.6 (0.5) | 47.1 (2.3) | 87.2 (1.6) | 91.6 (0.9) | 85.2 (1.0) | 90.7 (1.6) | 87.6 (3.5) | 3.8 (3.2) |
| | MNLI (acc) | MNLI-mm (acc) | SNLI (acc) | QNLI (acc) | RTE (acc) | MRPC (F1) | QQP (F1) | |
| Prompt-based zero-shot | 50.8 | 51.7 | 49.5 | 50.8 | 51.3 | 61.9 | 49.7 | |
| Multitask FT zero-shot | 63.2 | 65.7 | 61.8 | 65.8 | 74.0 | 81.6 | 63.4 | |
| + task selection | 62.4 | 64.5 | 65.5 | 61.6 | 64.3 | 75.4 | 57.6 | |
| Prompt-based FT [†] | 68.3 (2.3) | 70.5 (1.9) | 77.2 (3.7) | 64.5 (4.2) | 69.1 (3.6) | 74.5 (5.3) | 65.5 (5.3) | |
| Multitask Prompt-based FT | 70.9 (1.5) | 73.4 (1.4) | 78.7 (2.0) | 71.7 (2.2) | 74.0 (2.5) | 79.5 (4.8) | 67.9 (1.6) | |
| + task selection | 73.5 (1.6) | 75.8 (1.5) | 77.4 (1.6) | 72.0 (1.6) | 70.0 (1.6) | 76.0 (6.8) | 69.8 (1.7) | |

Table A.16: **Results of few-shot learning with NLP benchmarks.** All results are obtained using RoBERTa-large. We report the mean (and standard deviation) of metrics over 5 different splits. †: Result in Gao et al. (2021a) in our paper; FT: finetuning; task selection: select multitask data from customized datasets.

the grammaticality of sentences, and its inputs may include non-grammatical sentences that are outside the distribution of masked language models, as noted in Gao et al. (2021a). Overall, our approach shows promising results for multitask learning in language tasks.

Full Results with Task Selection

To complement task selection in Table A.13, we provide full results here and explain each method thoroughly.

We first elaborate on what each method did in each stage. During the testing stage, we conducted experiments in zero-shot and few-shot settings for a given dataset following Gao et al. (2021a), who applied prompt-based methods on moderately sized language models such as RoBERTa. Prompt-based finetuning method updates the model with prompt-based prediction loss for a given example. The given example can either be from a testing dataset or other datasets.

Table A.16 shows our multitask finetuning and task selection provide helps on

target tasks, as detailed in Section A.6.1. We will elaborate on what each method did in the “Multitask finetuning phase” and “Downstream phase”.

In the “Multitask finetuning phase”: For prompt-based zero-shot (col-1) and prompt-based FT (col-4) we do not finetune any model. For Multitask Prompt-based finetuning (col-2,3,5,6), we conduct prompt-based finetuning methods using finetuning(auxiliary) tasks. The data of tasks are from datasets other than testing datasets. For instance, consider a model designated to adapt to a dataset (say SST-2), we choose data from other datasets (mr, cr, etc.) and combine these data together and form multiple auxiliary tasks, these tasks updated the model using prompt-based finetuning methods. In the “downstream phase” where we adapt the model: In the zero-shot setting (col-1,2,3), we directly evaluate the model’s prompt-based prediction accuracy. In the few-shot setting (col-4,5,6), we finetune the model using shot samples from the same dataset (sst-2) and assess its accuracy on the test split.

Additional Results on simCSE

| | SST-2 (acc) | SST-5 (acc) | MR (acc) | CR (acc) | MPQA (acc) | Subj (acc) | TREC (acc) | CoLA (Matt.) |
|---|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|------------------|
| Prompt-based zero-shot | 50.9 | 19.3 | 50 | 50 | 50 | 50.4 | 27.2 | 0 |
| Multitask FT zero-shot | 51.3 | 13.8 | 50 | 50 | 50 | 50.6 | 18.8 | 0 |
| Prompt-based FT [†] | 51.8 (2.6) | 20.5 (6.1) | 50.6 (0.8) | 50.8 (1.1) | 52.3 (1.9) | 55.4 (3.7) | 19.8 (7.3) | 0.8 (0.9) |
| Multitask Prompt-based FT + task selection | 50.6 (0.7) | 22.1 (6.2) | 50.5 (1.0) | 51.5 (1.7) | 53.4 (2.7) | 51.0 (1.4) | 26.4 (8.5) | 0.9 (1.3) |
| | 51.7 (1.7) | 19.7 (5.6) | 50.6 (0.8) | 51.6 (1.6) | 52.3 (2.7) | 54.7 (2.5) | 23.2 (9.9) | 0.5 (0.7) |
| | MNLI (acc) | MNLI-mm (acc) | SNLI (acc) | QNLI (acc) | RTE (acc) | MRPC (F1) | QQP (F1) | |
| Prompt-based zero-shot | 35.4 | 35.2 | 33.8 | 50.5 | 47.3 | 1.4 | 1.5 | |
| Multitask FT zero-shot | 35.4 | 35.2 | 33.6 | 49.5 | 47.3 | 53.8 | 53.8 | |
| Prompt-based FT [†] | 32.9 (0.8) | 33.0 (0.7) | 33.7 (0.6) | 50.6 (1.4) | 48.7 (3.7) | 79.2 (4.1) | 53.5 (2.7) | |
| Multitask Prompt-based FT + task selection | 32.5 (0.6) | 32.5 (0.7) | 33.5 (0.4) | 50.6 (2.4) | 50.0 (2.0) | 76.3 (6.5) | 54.2 (0.8) | |
| | 33.2 (1.2) | 33.2 (1.1) | 35.0 (0.8) | 50.3 (0.4) | 51.8 (2.0) | 72.2 (10.8) | 52.9 (3.0) | |

Table A.17: Our main results using simCSE (Gao et al., 2021b). We report mean (and standard deviation) performance over 5 splits of few-shot examples. FT: fine-tuning; task selection: select multitask data from customized dataset.

We present our results using the same approach as described in our paper.

However, we used a different pretrained loss, namely simCSE, as proposed by Gao et al. (2021b). However, the results are not promising, The reason is simCSE is trained with a contrastive loss instead of masked language prediction, making it less suitable for prompt-based finetuning.

A.7 Vision Language Tasks

Pretrained vision-language as another type of foundation model has achieved tremendous success across various downstream tasks. These models, such as CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021), align images and text in a shared space, enabling zero-shot classification in target tasks. Finetuning such models has resulted in state-of-the-art accuracy in several benchmarks.

Vision-language model enables the classification of images through prompting, where classification weights are calculated by a text encoder. The text encoder inputs text prompts containing class information, and outputs features aligned with features from the vision encoder in the same space.

However, standard finetuning can be affected by minor variations underperforming direct adaptation (Kumar et al., 2022; Wortsman et al., 2022). Additionally, standard finetuning can be computationally expensive, as it requires training the model on a large amount of target task data.

We perform our multitask finetuning pipeline on the vision-language model and observe certain improvements. It's worth mentioning although the vision-language model is pretrained using contrastive learning, the model does not align with our framework. Vision-language model computes contrastive loss between image and text encoder, whereas our pretraining pipeline formulates the contrastive loss between the same representation function ϕ for positive and negative sample pairs. Despite the discrepancy, we provide some results below.

A.7.1 Improving Zero-shot Performance

We investigate the performance of CLIP models in a zero-shot setting, following the established protocol for our vision tasks. Each task includes 50 classes, with one query image per class. We employ text features combined with class information as the centroid to categorize query images within the 50 classes. During adaptation, we classify among randomly selected classes in the test split, which consists of 50 classes.

We experimented with our methods on *tieredImageNet* and *DomainNet*. The text template utilized in *tieredImageNet* was adapted from the CLIP documentation. In adaptation, we classify among all classes in the test split (160 classes in *tieredImageNet* and 100 classes in *DomainNet*). For text features on *tieredImageNet*, we use 8 templates adapted from CLIP: a photo of a {}, itap of a {}, a bad photo of the {}, a origami {}, a photo of the large {}, a {} in a video game, art of the {}, a photo of the small {}. For templates on *DomainNet*, we simply use a photo of a {}. In the *DomainNet* The text template used for this experiment is "a photo of {}". We perform Locked-Text Tuning, where we fixed the text encoder and update the vision encoder alone.

| Backbone | Method | <i>tieredImageNet</i> | <i>DomainNet</i> |
|----------|------------|-----------------------|------------------|
| ViT-B | Adaptation | 84.43 (0.25) | 70.93 (0.32) |
| | Ours | 84.50 (0.25) | 73.31 (0.30) |
| ResNet50 | Adaptation | 81.01 (0.28) | 63.61 (0.34) |
| | Ours | 81.02 (0.27) | 65.55 (0.34) |

Table A.18: Multitask finetune on zero-shot performance with CLIP model.

Table A.18 demonstrates that CLIP already exhibits a high level of zero-shot performance. This is due to the model classifying images based on text information rather than relying on another image from the same class, which enables the model to utilize more accurate information to classify among query images. We show the effectiveness of zero-shot accuracy in *tieredImageNet* and *DomainNet*. It is

worth highlighting that our multitask finetuning approach enhances the model’s zero-shot performance, particularly on the more realistic DomainNet dataset. We have observed that our multitask finetuning pipeline yields greater improvements for tasks on which the model has not been extensively trained.

A.7.2 Updating Text Encoder and Vision Encoder

We also investigated whether updating the text encoder will provide better performance. On the tieredImageNet dataset, We finetune the text encoder and vision encoder simultaneously using the contrastive loss, following the protocol in Goyal et al. (2023).

| Method | Zero-shot | Multitask finetune |
|-----------------|--------------|--------------------|
| Accuracy | 84.43 (0.25) | 85.01 (0.76) |

Table A.19: Multitask finetune on zero-shot performance with ViT-B32 backbone on tieredImageNet.

In Table A.19, we observed slightly better improvements compared to updating the vision encoder alone. We anticipate similar performance trends across various datasets and backbone architectures. We plan to incorporate these findings into our future work.

A.7.3 CoCoOp

We also multitask finetune the vision language model following the protocol outlined in Zhou et al. (2022a). This approach involved prepending an image-specific token before the prompt to enhance prediction accuracy. To generate this token, we trained a small model on the input image. We evaluate the performance of our model on all classes in the test split, which corresponds to a 160-way classification task. This allows us to comprehensively assess the model’s ability to classify among a large number of categories.

| Method | Zero-shot | Multitask finetune |
|----------------|------------------|---------------------------|
| ViT-B32 | 69.9 | 71.4 |

Table A.20: Multitask finetune on zero-shot performance with ViT-B32 backbone on tieredImageNet.

Table A.20 showed the result of the performance of the CoCoOp method. We observed an improvement of 1.5% in accuracy on direct adaptation.

B APPENDIX OF CHAPTER 4

In this appendix, we provide more empirical settings and results for logical tasks in Section B.1 and linguistic translation tasks in Section B.2. We provide a theory for confined support and model scalability, along with a case study of a toy model in Section B.3. We provide full proof in Section B.4.

B.1 Logical Tasks

B.1.1 Task Setup

We provide a comprehensive explanation of logical composite tasks below. Examples can be seen in Table B.1.

- **(A) + (B) Capitalization & Swap**, as in Section 4.2.
- **(A) + (C) Capitalization & Two Sum**. Given words of numerical numbers, * represents the operation of capitalizing, @ represents summing the two numbers.
- **(G) + (H) Modular & Two Sum Plus**. Given numerical numbers, @ represents the operation of taking modular, # represents to sum the two numbers and then plus one.
- **(A) + (F) Capitalization & Plus One**. If numerical numbers are given, plus one; if words are given, capitalize the word; if both are given, perform both operations.

Among these, (A) + (F) performs the two operations on separable parts of the test inputs (i.e., separable composite task).

We design our logical tasks following the idea of math reasoning and logical rules. The details are shown in Table B.1. Our numerical numbers in Table 4.2 are uniformly randomly chosen from 1 to 1000. The words of numbers in task (C) are

| Tasks | Simple Task | Simple Task | Composite |
|-----------|---|---|---|
| (A) + (B) | input: * apple output: AP- PLE | input: (farm frog) output: frog farm | input: (* bell * ford) output: FORD BELL |
| (A) + (C) | input: * (<i>five</i>) output: FIVE | input: <i>twenty @</i> <i>eleven</i> output: thirty-one | input: * (<i>thirty-seven @ six-</i> <i>teen</i>) output: FIFTY-THREE |
| (G) + (H) | input: 15 @ 6 output: 3 | input: 12 # 5 output: 18 | input: 8 # 9 @ 7 Output: 4 |
| (A) + (F) | input: 435 output: 436 | input: cow output: COW | input: 684 cat output: 685 CAT |

Table B.1: Examples of the four logical composite tasks. Note that in (G) + (H), the output of the composite task can be either 4 or 11 depending on the order of operations, and we denote both as correct.

uniformly randomly chosen from one to one hundred. The words representing objects in Table 4.2 are uniformly randomly chosen from class names of ImageNet after dividing the phrase (if any) into words. We randomly chose 100 examples in composite testing data in our experiments and replicated the experiments in each setting three times. We fixed the number of in-context examples as $K = 10$ as demonstrations.

B.1.2 Experimental Setup

We use exact match accuracy to evaluate the performance between sequence outputs. The calculation of exact match accuracy divided the number of matched words by the length of ground truth.

For Llama models, we use official Llama1 and Llama2 models from Meta (Touvron et al., 2023a), we use open_llama_3b_v2 from open OpenLlama (Geng and Liu, 2023). For GPT models, we use GPT2-large from OpenAI (Radford et al., 2019), and we use GPT-neo models for GPT models in other scales from EleutherAI (Black et al., 2021).

We’ve experimented with prompt demonstrations. Instructional prompts do

help ChatGPT and Claude3 (although we haven't quantified the accuracy in large-scale experiments), but they offer limited benefits for current open-source models. On the other hand, we did not have prompt tuning or any other parameter updates during our evaluation.

In our experiments, we provided the model with instructions. Here are instructions of Figure 4.1, which were prepended to ICL examples. We refer to our codebase for full instructions and results.

* is a function before words for swapping the position of 2 words, # is another function after words for capitalizing letters of words.

B.1.3 Results

We show a visualization of some logical task accuracy along the increasing to model scale, complement to Table 4.4.

We also include results for the more recent model Llama3 (Meta, 2024) on the part of our logical tasks to demonstrate the idea. We show results in Table B.2.

| | | Llama3 | |
|-----------|-----------------|--------|-----|
| | | 8B | 70B |
| (A) + (B) | Capitalization | 100 | 100 |
| | swap | 100 | 100 |
| | Compose | 52 | 72 |
| | Com. in-context | 97 | 100 |
| (A) + (F) | Capitalization | 100 | 100 |
| | PlusOne | 100 | 100 |
| | Compose | 88 | 100 |
| | Com. in-context | 100 | 100 |

Table B.2: Results evaluating composite tasks on Llama3. The accuracy is shown in %.

As shown in the Table B.2, for the *separable composite tasks* which are relatively easy for model to solve (A) + (F), the models show strong compositional ability:

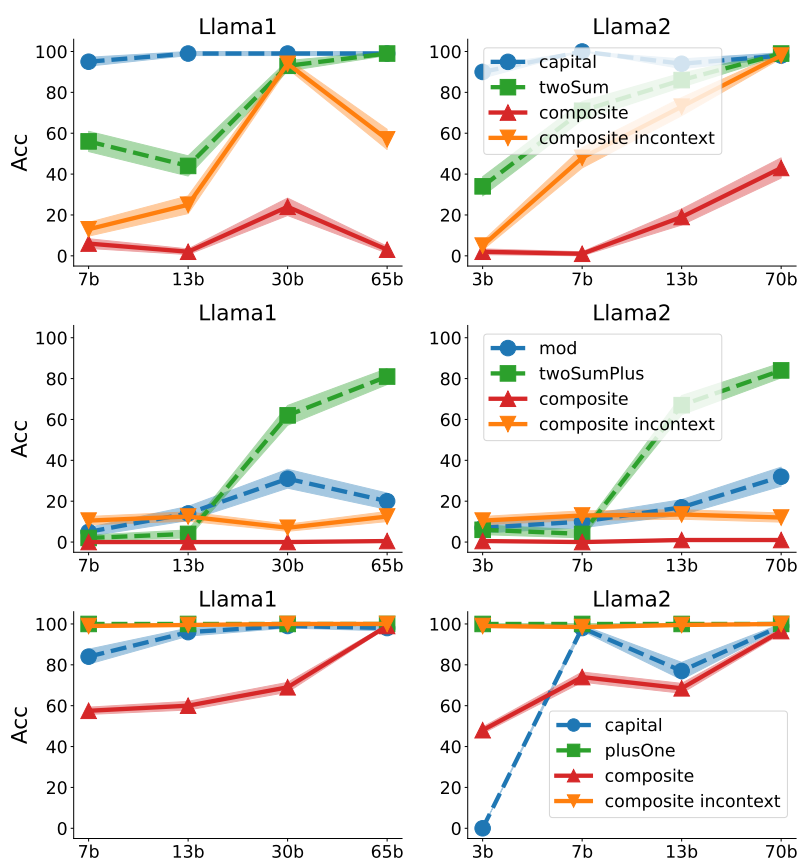


Figure B.1: The accuracy v.s. model scale on composite logical rule tasks. Dashed lines: simple tasks. Solid lines: composite tasks. Rows: (A) + (C) Capitalization & Two Sum; (G) + (H) Modular & Two Sum Plus; (A) + (F) Capitalization & Plus One. Columns: different models. Lines: performance in different evaluation settings, i.e., the two simple tasks, the composite setting, and the composite in-context setting (examples for the last two are shown in Table 4.1).

the composite accuracy is high, improves with increasing scale, and eventually reaches similar performance as the *gold standard* composite in-context setting. For composite tasks with sequential reasoning steps (A) + (B), the model has poor performance on a small scale but has increased performance on an increased model scale. Providing composed examples as in-context demonstrations will help the model understand and solve the composite tasks well.

B.2 Formal Language Translation Tasks

Our translation tasks mainly follow the compositional generalization tasks in COFE (An et al., 2023b). The details can be found in Section 4 in An et al. (2023a). We directly take the source grammar \mathcal{G}_s in COGS, which mimics the English natural language grammar, and reconstruct the target grammar \mathcal{G}_t in COGS to be chain-structured.

We follow the Primitive coverage principle proposed by An et al. (2023b) that primitives contained in each test sample should be fully covered by in-context examples. Here, primitives refer to the basic, indivisible elements of expressions, including subjects, objects, and verbs. Note that multiple sets of in-context examples can meet these criteria for each test case. Across all experimental conditions, we maintain a consistent number of test instances at 800.

We use the word error rate (WER) as the metric. It measures the differences between 2 sentences. It measures the minimum number of editing operations (deletion, insertion, and substitution) required to transform one sentence into another and is common for speech recognition or machine translation evaluations. The computation of WER is divided by the number of operations by the length of ground truth.

| Original Target Grammar | Chain-Structured Target Grammar |
|--|--|
| rose (x_1) AND help . theme (x_3 , x_1) AND help . | |
| agent (x_3 , x_6) AND dog (x_6) | HELP (DOG, ROSE, NONE) |
| * captain (x_1) ; eat . agent (x_2 , x_1) | EAT (CAPTION, NONE, NONE) |
| * dog (x_4) ; hope . agent (x_1 , Liam) AND hope . | |
| ccomp (x_1 , x_5) AND prefer . agent (x_5 , x_4) | HOPE (LIAM, NONE, NONE) CCOMP PREFER (DOG, NONE, NONE) |

Table B.3: Demonstration in An et al. (2023a) showing examples with the original grammar and the new chain-structured grammar.

In formal language tasks, as mentioned in Section 4.3.2, we change the original target grammar of COGS to be chain-structured. In Table B.3, we list some examples with the original target grammar and the new chain-structured grammar.

- First, to distinguish the input and output tokens, we capitalize all output tokens (e.g., from “rose” to “ROSE”).

- Second, we replace the variables (e.g., “x_1”) in the original grammar with their corresponding terminals (e.g., “ROSE”).
- Then, we group the terminals of AGENT (e.g., “DOG”), THEME (e.g., “ROSE”), and RECIPIENT with their corresponding terminal of PREDICATE (e.g., “HELP”) and combine this group of terminals in a function format, i.e., “PREDICATE (AGENT, THEME, RECIPIENT)”. If the predicate is not equipped with an agent, theme, or recipient in the original grammar, the corresponding new non-terminals (i.e., AGENT, THEME, and RECIPIENT, respectively) in the function format above will be filled with the terminal NONE (e.g., “HELP (DOG, ROSE, NONE)”). Such a function format is the minimum unit of a CLAUSE.
- Finally, each CLAUSE is concatenated with another CLAUSE by the terminal CCOMP (e.g., “HOPE (LIAM, NONE, NONE) CCOMP PREFER (DOG, NONE, NONE)”).

| Task | In-context Example | Testing Example |
|-------------------|--|--|
| Passive to Active | The book was squeezed . SQUEEZE (NONE , BOOK , NONE) | Sophia squeezed the donut . SQUEEZE (SOPHIA , DONUT , NONE) |
| Object to Subject | Henry liked a cockroach in a box . LIKE (HENRY , IN (COCKROACH , BOX) | A cockroach inflated a boy . INFLATE (COCKROACH , BOY , NONE) |
| Composite Task | The book was squeezed . SQUEEZE (NONE , BOOK , NONE) Henry liked a cockroach in a box . LIKE (HENRY , IN (COCKROACH , BOX) | A cockroach squeezed the hedgehog . SQUEEZE (COCKROACH , hedgehog , NONE) |

Table B.4: Testing examples of Passive to Active and Object to Subject, **red** text shows the verbs changing from passive to active voice in simple tasks, and **blue** text shows the nouns from objective to subjective.

In the following, we provide a detailed explanation of our two composite tasks in translation tasks.

Passive to Active and Object to Subject Transformation. Based on the generalization tasks identified in Kim and Linzen (2020)), we select two distinct challenges

| Task | | Example |
|----------------------|-----------------|--|
| Phrase Recombination | Input Output | The baby on a tray in the house screamed . SCREAM (ON (BABY , IN (TRAY , HOUSE)) , NONE , NONE) |
| Longer Chain | Input Output | A girl valued that Samuel admired that a monkey liked that Luna liked that Oliver respected that Savannah hoped that a penguin noticed that Emma noticed that the lawyer noticed that a cake grew . VALUE (GIRL , NONE , NONE) \ CCOMP ADMIRE (SAMUEL , NONE , NONE) \ CCOMP LIKE (MONKEY , NONE , NONE) \ CCOMP LIKE (LUNA , NONE , NONE) \ CCOMP RESPECT (OLIVER , NONE , NONE) \ CCOMP HOPE (SAVANNAH , NONE , NONE) \ CCOMP NOTICE (PENGUIN , NONE , NONE) \ CCOMP NOTICE (EMMA , NONE , NONE) \ CCOMP NOTICE (LAWYER , NONE , NONE) \ CCOMP GROW (NONE , CAKE , NONE) |
| Composite Task | Input Output | The baby on a tray in the house valued that Samuel admired that a monkey liked that Luna liked that Oliver respected that Savannah hoped that a penguin noticed that Emma noticed that the lawyer noticed that a cake grew . VALUE (ON (BABY , IN (TRAY , HOUSE)) , NONE , NONE) \ CCOMP ADMIRE (SAMUEL , NONE , NONE) \ CCOMP LIKE (MONKEY , NONE , NONE) \ CCOMP LIKE (LUNA , NONE , NONE) \ CCOMP RESPECT (OLIVER , NONE , NONE) \ CCOMP HOPE (SAVANNAH , NONE , NONE) \ CCOMP NOTICE (PENGUIN , NONE , NONE) \ CCOMP NOTICE (EMMA , NONE , NONE) \ CCOMP NOTICE (LAWYER , NONE , NONE) \ CCOMP GROW (NONE , CAKE , NONE) |

Table B.5: Testing examples of Phrase Recombination and Longer Chain, **red** text shows the phrase serving as primitives in sentences in simple tasks, and **blue** text shows the logical structures as sub-sentences in long sentences.

for our study as two simple tasks. *Passive to Active*: Transitioning sentences from Passive to Active voice. *Object to Subject*: Changing the focus from Object to Subject using common nouns. These tasks serve as the basis for our composite task, where both transformations are applied simultaneously to the same sentence. Examples illustrating this dual transformation can be found in Table B.4.

Enhanced Phrase Subject with Longer Chain. COFE proposed two compositional generalization tasks (Figure 2 in An et al. (2023b)): *Phrase Recombination (PhraReco)*: integrate a prepositional phrase (e.g., “A in B”) into a specific grammatical role (e.g.,

“subject”, “object”); *Longer Chain (LongChain)*: Extend the tail of the logical form in sentences. We consider these two generalization tasks as two simple tasks, merging them to form a composite task. In particular, we substitute the sentence subject in the Longer Chain task with a prepositional phrase from the Phrase Recombination task, creating a more complex task structure. Detailed examples of this combined task can be found in Table B.5.

B.3 Theory for Confined Support

B.3.1 Compositional Ability with Model Scale

We then show that if simple tasks have confined support, the compositional ability of language models will increase as the model scale increases. We demonstrate this by showing that the accuracy of the model on each simple task improves with a larger model scale.

Note that the optimal solutions of the parameter matrices are W^{*PV} and W^{*KQ} . We naturally consider that the rank of the parameter matrices W^{*PV} and W^{*KQ} can be seen as a measure of the model’s scale. A higher rank in these matrices implies that the model can process and store more information, thereby enhancing its capability. We state the following theorem.

Theorem B.3.1. *Suppose a composite task satisfies confined support. Suppose that we have $(x_1, y_1, \dots, x_N, y_N, x_q)$ as a testing input prompt and the corresponding W where $y_i = Wx_i$. As rank r decreases, $\mathbb{E}_{W, x_1, \dots, x_N} [Acc_\theta]$ will have a smaller upper bound.*

Theorem B.3.1 shows the expected accuracy of a model on composite tasks is subjected to a lower upper bound as the scale of the model diminishes. This conclusion explains why scaling up helps the performance when the model exhibits compositional ability for certain tasks (those we call “separable composite tasks”). One common characteristic of these tasks is their inputs display confined supports within the embeddings. This is evidenced by the model’s decent performance on tasks as presented in Table 4.4 and Figure 4.3, where inputs are composed of parts.

B.3.2 Case Study of Confined Support

Our theoretical conclusion shows the model behavior regarding input embedding. It states that the model will have compositional ability if tasks are under confined support of input embedding. To illustrate such theoretical concepts and connect them to empirical observations, we specialize the general conclusion to settings that allow easy interpretation of disjoint. In this section, we provide a toy linear case study on classification tasks, showing how confined support on embedding can be decomposed and composite tasks can be solved. We assume $\delta = \epsilon = 0$ in the following simple example.

Consider that there are only two simple tasks for some random objects with the color red and blue and the shape square and round: (1) binary classification based on the color red and blue. (2) binary classification based on shape: circle and square. However, during evaluation, the composite task is a four-class classification, including red circle, red square, blue circle, and blue square.

Then we have two simple tasks $K = 2$. Consider the input embedding $x = (a, b)$, where $a \in \mathbb{R}^2, b \in \mathbb{R}^2, d = 4$. Consider $W = \begin{pmatrix} 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{pmatrix}$ and $y = Wx$.

Consider the inputs from simple and composite tasks as:

- Task 1: Red: $x_1 = (1, 0, 0, 0), y_1 = (1, 0)$ and blue: $x_2 = (0, 1, 0, 0), y_2 = (-1, 0)$.
- Task 2: Circle $x_3 = (0, 0, 1, 0), y_3 = (0, 1)$ and square $x_4 = (0, 0, 0, 1), y_4 = (0, -1)$.
- Composed task: red circle $x_5 = (1, 0, 1, 0), y_5 = (1, 1)$, red square $x_6 = (1, 0, 0, 1), y_6 = (1, -1)$, blue circle $x_7 = (0, 1, 1, 0), y_7 = (-1, 1)$ and blue square $x_8 = (0, 1, 0, 1), y_8 = (-1, -1)$.

Suppose that we have the optimal solution \hat{y}_q as in Equation (B.1). Given $x_q = (1, 0, 1, 0)$ as a testing input for a red circle example, During the test, we have different predictions given different in-context examples:

1. Given only examples from Task 1 (red and blue): $[(x_1, y_1), (x_2, y_2)]$, we have $\hat{y}_q = (1, 0)$ can only classify the color as red.

2. Given only examples from Task 2 (square and circle): $[(x_4, y_4), (x_3, y_3)]$, we have $\hat{y}_q = (0, 1)$ only classify the shape as a circle.
3. Given a mixture of examples from Task 1 and 2 (red and circle): $[(x_1, y_1), (x_3, y_3)]$, we have $\hat{y}_q = (1, 1)$ can classify as red and circle.

We can see that in the final setting, the model shows compositional ability. This gives a concrete example for the analysis in Theorem 4.4.4.

B.4 Deferred Proof

In this section, we provide a formal setting and proof. We first formalize our model setup.

B.4.1 Linear self-attention networks.

These networks are widely studied (Von Oswald et al., 2023; Akyürek et al., 2023; Mahankali et al., 2023; Garg et al., 2022; Zhang et al., 2023b; Shi et al., 2023d). Following them, we consider the following linear self-attention network with parameters $\theta = (W^{PV}, W^{KQ})$:

$$f_{\text{LSA},\theta}(E) = E + W^{PV}E \cdot \frac{E^\top W^{KQ}E}{N}.$$

The prediction of the model for x_q is $\hat{y}_q = [f_{\text{LSA},\theta}(E)]_{(d+1):(d+K), N+1}$, the bottom rightmost sub-vector of $f_{\text{LSA},\theta}(E)$ with length K . Let

$$W^{PV} = \begin{pmatrix} W_{11}^{PV} & W_{12}^{PV} \\ (W_{21}^{PV})^\top & W_{22}^{PV} \end{pmatrix} \in \mathbb{R}^{(d+K) \times (d+K)},$$

$$W^{KQ} = \begin{pmatrix} W_{11}^{KQ} & W_{12}^{KQ} \\ (W_{21}^{KQ})^\top & W_{22}^{KQ} \end{pmatrix} \in \mathbb{R}^{(d+K) \times (d+K)},$$

where $W_{11}^{PV} \in \mathbb{R}^{d \times d}$, $W_{12}^{PV}, W_{21}^{PV} \in \mathbb{R}^{d \times K}$, and $W_{22}^{PV} \in \mathbb{R}^{K \times K}$; similar for W^{KQ} . Then the prediction is

$$\hat{y}_q = \begin{pmatrix} (W_{21}^{PV})^\top & W_{22}^{PV} \end{pmatrix} \begin{pmatrix} \mathbb{E}\mathbb{E}^\top \\ \mathbf{N} \end{pmatrix} \begin{pmatrix} W_{11}^{KQ} \\ (W_{21}^{KQ})^\top \end{pmatrix} x_q. \quad (\text{B.1})$$

We observe only part of the parameters affect our prediction, so we treat the rest of them as zero in our analysis.

B.4.2 Proof of Compositional Ability under Confined Support

Here, we provide the proof of our main conclusion regarding Theorem 4.4.4 and Section 4.4.2.

Without abuse of notation, we denote $\mathbf{U} = W_{11}^{KQ}$, $\mathbf{u} = W_{22}^{PV}$.

We also add some mild assumptions.

1. The covariance matrix Λ of simple tasks will have the same trace to prevent the scale effect of different simple tasks.
2. The spectral norm of Λ is bounded on both sides $m \leq \|\Lambda\| \leq M$.

We first introduce the lemma where the language model only pretrained on one simple task ($K = 1$). The pretraining loss $L(\theta)$ can be refactored, and the solution will have a closed form. We further discuss the following.

Lemma B.4.1 (Lemma 5.3 in Zhang et al. (2023b)). *Let $\Gamma := (1 + \frac{1}{N}) \Lambda + \frac{1}{N} \text{tr}(\Lambda) \mathbf{I}_{d \times d} \in \mathbb{R}^{d \times d}$. Let*

$$\tilde{\ell}(\mathbf{U}, \mathbf{u}) = \text{tr} \left[\frac{1}{2} \mathbf{u}^2 \Gamma \Lambda \mathbf{U} \Lambda \mathbf{U}^\top - \mathbf{u} \Lambda^2 \mathbf{U}^\top \right]$$

Then

$$\min_{\theta} L(\theta) = \min_{\mathbf{U}, \mathbf{u}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) + C = -\frac{1}{2} \text{tr}[\Lambda^2 \Gamma^{-1}] + C$$

where C is a constant independent with θ . For any global minimum of $\tilde{\ell}$, we have $\mathbf{u}\mathbf{U} = \Gamma^{-1}$.

As the above lemma construction, we denote the optimal solution as W^{*PV} and W^{*KQ} . Taking one solution as $\mathbf{U} = \Gamma^{-1}$, $\mathbf{u} = 1$, we observe the minimizer of global training loss is of the form:

$$W^{*PV} = \begin{pmatrix} 0_{d \times d} & 0_d \\ 0_d^\top & 1 \end{pmatrix}, W^{*KQ} = \begin{pmatrix} \Gamma^{-1} & 0_d \\ 0_d^\top & 0 \end{pmatrix}. \quad (\text{B.2})$$

We then prove our main theory Theorem 4.4.4 in Section 4.4.2, we first re-state below:

Theorem 4.4.4. *Consider distinct tasks k and g with corresponding examples $\mathcal{S}_k, \mathcal{S}_g$. If two tasks have confined support, and Assumption 4.4.2 is true, then with high probability, the model has the compositional ability as defined in Definition 4.4.1. Moreover,*

$$Acc_\theta(\mathcal{S}_k) + Acc_\theta(\mathcal{S}_g) \leq Acc_\theta(\mathcal{S}_{k \cup g}).$$

Proof of Theorem 4.4.4. WLOG, consider two simple tasks, $K = 2$. We have $\mathbf{x} = (\mathbf{a}, \mathbf{b})$, where $\mathbf{a} \in \mathbb{R}^{d_1}, \mathbf{b} \in \mathbb{R}^{d_2}, d_1 + d_2 = d$. Since \mathbf{x} only has large values on certain dimensions, it's equivalent to just consider corresponding dimensions in \mathbf{w} , i.e., for simple task 1, we have $\mathbf{w}^{(1)} = (\mathbf{w}_a, \mathbf{w}_{\delta b})$, for simple task 2, we have $\mathbf{w}^{(2)} = (\mathbf{w}_{\delta a}, \mathbf{w}_b)$.

We have $\mathbf{x} \sim \Lambda$, where:

$$\Lambda = \begin{pmatrix} \Lambda_{KK} & \Lambda_{KG} \\ \Lambda_{GK} & \Lambda_{GG} \end{pmatrix}$$

- Task 1: $\mathbf{x} = (\mathbf{a}, 0_{d_2})^\top + (0, \mathbf{b}_\delta)^\top$, $\mathbf{y} = (\mathbf{w}_a^\top \mathbf{a}, 0) + (0, \mathbf{w}_{\delta b}^\top \mathbf{b}_\delta)$.
- Task 2: $\mathbf{x} = (0_{d_1}, \mathbf{b})^\top + (\mathbf{a}_\delta, 0_{d_2})^\top$, $\mathbf{y} = (0, \mathbf{w}_b^\top \mathbf{b}) + (\mathbf{w}_{\delta a}^\top \mathbf{a}_\delta, 0)$.
- Composed task: $\mathbf{x} = (\mathbf{a}, \mathbf{b})^\top + (\mathbf{a}_\delta, \mathbf{b}_\delta)^\top$, $\mathbf{y} = (\mathbf{w}_a^\top \mathbf{a}, \mathbf{w}_b^\top \mathbf{b}) + (\mathbf{w}_{\delta a}^\top \mathbf{a}_\delta, \mathbf{w}_{\delta b}^\top \mathbf{b}_\delta)$.

The form of E is,

$$E := \begin{pmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \dots & \mathbf{a}_N & \mathbf{a}_q \\ \mathbf{b}_1 & \mathbf{b}_2 & \dots & \mathbf{b}_N & \mathbf{b}_q \\ \mathbf{y}_1 & \mathbf{y}_2 & \dots & \mathbf{y}_N & 0 \end{pmatrix} + E_r \in \mathbb{R}^{(d+2) \times (N+1)}.$$

where E_r represents the values caused by residual dimensions whose entries are bounded by δ .

Following Equation (4.3) in Zhang et al. (2023b), we have

$$EE^\top = \frac{1}{N} \begin{pmatrix} \sum_{i=1}^N \mathbf{a}_i \mathbf{a}_i^\top + \mathbf{a}_q \mathbf{a}_q^\top & \sum_{i=1}^N \mathbf{a}_i \mathbf{b}_i^\top + \mathbf{a}_q \mathbf{b}_q^\top & \sum_{i=1}^N \mathbf{a}_i \mathbf{y}_i^\top \\ \sum_{i=1}^N \mathbf{b}_i \mathbf{a}_i^\top + \mathbf{b}_q \mathbf{a}_q^\top & \sum_{i=1}^N \mathbf{b}_i \mathbf{b}_i^\top + \mathbf{b}_q \mathbf{b}_q^\top & \sum_{i=1}^N \mathbf{b}_i \mathbf{y}_i^\top \\ \sum_{i=1}^N \mathbf{y}_i \mathbf{a}_i^\top & \sum_{i=1}^N \mathbf{y}_i \mathbf{b}_i^\top & \sum_{i=1}^N \mathbf{y}_i \mathbf{y}_i^\top \end{pmatrix} + \delta \cdot o(EE^\top).$$

The W^{PV} can be presented in block matrix

$$W^{\text{PV}} = \begin{pmatrix} W_{11}^{\text{PV}} & W_{12}^{\text{PV}} & W_{13}^{\text{PV}} \\ (W_{21}^{\text{PV}})^\top & W_{22}^{\text{PV}} & W_{23}^{\text{PV}} \\ (W_{31}^{\text{PV}})^\top & (W_{32}^{\text{PV}})^\top & W_{33}^{\text{PV}} \end{pmatrix} \in \mathbb{R}^{(d_1+d_2+2) \times (d_1+d_2+2)}$$

We can apply Lemma B.4.1 into optimization and recall

$$W^{*KQ} = \begin{pmatrix} \Gamma_{\text{all}}^{-1} & 0_d \\ 0_d^\top & 0 \end{pmatrix}.$$

where $\Gamma_{\text{all}}^{-1} \in \mathbb{R}^{(d_1+d_2) \times (d_1+d_2)}$. Consider two tasks only related to disjoint dimension of \mathbf{x} , we also have $\sigma(\Lambda_{\text{KG}}) = \sigma(\Lambda_{\text{GK}}) \leq \epsilon$. Denote

$$\Lambda = \tilde{\Lambda} + \Lambda_r$$

where

$$\tilde{\Lambda} = \begin{pmatrix} \Lambda_{\text{KK}} & \\ & \Lambda_{\text{GG}} \end{pmatrix}, \Lambda_r = \begin{pmatrix} & \Lambda_{\text{KG}} \\ \Lambda_{\text{GK}} & \end{pmatrix}$$

We apply Lemma B.4.1 Recall $\Gamma := (1 + \frac{1}{N}) \Lambda + \frac{1}{N} \text{tr}(\Lambda) \mathbf{I}_{d \times d} \in \mathbb{R}^{d \times d}$, we have:

$$\begin{aligned} \Gamma &= \left(1 + \frac{1}{N}\right) \tilde{\Lambda} + \frac{1}{N} \text{tr}(\tilde{\Lambda}) \mathbf{I}_{d \times d} + \left(1 + \frac{1}{N}\right) \Lambda_r \\ &= \tilde{\Gamma} + \Gamma_r \end{aligned}$$

where denote $\Gamma_r = (1 + \frac{1}{N}) \Lambda_r$. We have:

$$\Gamma^{-1} = \tilde{\Gamma}^{-1} - \tilde{\Gamma}^{-1} \Gamma_r \tilde{\Gamma}^{-1} + \mathcal{O}(\Gamma_r)$$

We denote

$$\tilde{\Gamma} = \begin{pmatrix} \Gamma_1 & 0 \\ 0 & \Gamma_2 \end{pmatrix},$$

where $\Gamma_1 = (1 + \frac{1}{N}) \Lambda_{\mathbb{K}\mathbb{K}} + \frac{1}{N} \text{tr}(\Lambda) \mathbf{I}_{d_1} \in \mathbb{R}^{d_1 \times d_1}$ and $\Gamma_2 = (1 + \frac{1}{N}) \Lambda_{\mathbb{G}\mathbb{G}} + \frac{1}{N} \text{tr}(\Lambda) \mathbf{I}_{d_2} \in \mathbb{R}^{d_2 \times d_2}$. Then we have;

$$\Gamma^{-1} = \begin{pmatrix} \Gamma_1^{-1} & 0 \\ 0 & \Gamma_2^{-1} \end{pmatrix} + A$$

where $\sigma(A) \leq 2m^2\epsilon$.

Then, It's similar to applying Lemma B.4.1 for pretraining separately into dimensions corresponding to different tasks. We solve similarly to $W_{\mathbb{K}\mathbb{Q}}$.

We have:

$$f_{\theta}(\mathbf{E}) = \begin{pmatrix} 0_{d_1 \times d_1} & 0_{d_1 \times d_2} & 0_{d_1 \times 2} \\ 0_{d_2 \times d_1} & 0_{d_2 \times d_2} & 0_{d_2 \times 2} \\ 0_{2 \times d_1} & 0_{2 \times d_2} & \mathbf{I}_2 \end{pmatrix} \mathbf{E} \mathbf{E}^{\top} \begin{pmatrix} \Gamma_1^{-1} & 0_{d_1 \times d_2} & 0_{d_1 \times 2} \\ 0_{d_2 \times d_1} & \Gamma_2^{-1} & 0_{d_2 \times 2} \\ 0_{2 \times d_1} & 0_{2 \times d_2} & 0_{2 \times 2} \end{pmatrix} \begin{pmatrix} \mathbf{a}_q \\ \mathbf{b}_q \\ 0 \end{pmatrix} + \tilde{\mathbf{A}} \quad (\text{B.3})$$

$$\hat{\mathbf{y}}_q = \frac{1}{N} \left(\sum_{i=1}^N \mathbf{y}_i \mathbf{a}_i^{\top}, \sum_{i=1}^N \mathbf{y}_i \mathbf{b}_i^{\top}, \sum_{i=1}^N \mathbf{y}_i \mathbf{y}_i^{\top} \right) \begin{pmatrix} \Gamma_1^{-1} \mathbf{a}_q \\ \Gamma_2^{-1} \mathbf{b}_q \\ 0 \end{pmatrix} + \mathbf{v} \quad (\text{B.4})$$

$$= \left(\frac{1}{N} \sum_{i=1}^N \mathbf{y}_i \mathbf{a}_i^{\top} \right) \Gamma_1^{-1} \mathbf{a}_q + \left(\frac{1}{N} \sum_{i=1}^N \mathbf{y}_i \mathbf{b}_i^{\top} \right) \Gamma_2^{-1} \mathbf{b}_q + \mathbf{v} \quad (\text{B.5})$$

$$= \frac{1}{N} \left(\mathbf{a}_q^{\top} \Gamma_1^{-1} \sum_{i=1}^N \mathbf{y}_i^{(1)} \mathbf{a}_i \right) + \mathbf{v}. \quad (\text{B.6})$$

where $\tilde{\mathbf{A}}$ representing residual matrix whose norm can be bounded by $\mathcal{O}(m^2 \epsilon \delta)$. Recall $\mathbf{x} \sim \mathcal{N}(0, \Lambda)$, then with high probability each entry in \mathbf{v} will be bounded by $Cm^2 \delta \epsilon$ for some constant C .

WLOG, we write residual vectors as 0 vector for simplicity of notation, and only consider residuals for estimations $\hat{\mathbf{y}}$. Note that we composed example $\mathbf{x} = (\mathbf{a}, \mathbf{b})^{\top}$, $\mathbf{y} = (\mathbf{w}_a^{\top} \mathbf{a}, \mathbf{w}_b^{\top} \mathbf{b})$. For simplicity, we write $\hat{\mathbf{w}}_a = \frac{1}{N} \Gamma_1^{-1} \sum_{i=1}^N \mathbf{y}_i^{(1)} \mathbf{a}_i$, similarly, $\hat{\mathbf{w}}_b = \frac{1}{N} \Gamma_2^{-1} \sum_{i=1}^N \mathbf{y}_i^{(2)} \mathbf{b}_i$.

Given in-context examples from one simple task only, consider that we have N examples from simple task 1, $\mathcal{S}_1 = \left[\{(\mathbf{a}_i, 0), \mathbf{y}_i\}_{i=1}^N \right]$. We have $\hat{\mathbf{w}}^{(1)} = (\hat{\mathbf{w}}_a, 0_{d_2})$, $\hat{\mathbf{w}}^{(2)} = (0_d)$, and we also have $\hat{\mathbf{y}}_q = (\hat{\mathbf{y}}_q^{(1)}, 0)^{\top}$, where $\hat{\mathbf{y}}_q^{(1)} = \mathbf{a}_q^{\top} \Gamma_1^{-1} \left(\frac{1}{N} \sum_{i=1}^N \mathbf{y}_i^{(1)} \mathbf{a}_i \right) + Cm^2 \delta \epsilon$. We have $\text{Acc}_{\theta}(\mathcal{S}_1) = \frac{\mathbb{1}(\hat{\mathbf{y}}_q^{(1)} = \mathbf{y}_q^{(1)})}{2}$.

Similarly, for N in-context examples only from task 2, we have $\hat{\mathbf{w}}^{(1)} = (0_d)$, $\hat{\mathbf{w}}^{(2)} = (0_{d_1}, \hat{\mathbf{w}}_b)$, $\hat{\mathbf{y}}_q = (0, \hat{\mathbf{y}}_q^{(2)})^{\top}$, where $\hat{\mathbf{y}}_q^{(2)} = \mathbf{a}_q^{\top} \Gamma_2^{-1} \left(\frac{1}{N} \sum_{i=1}^N \mathbf{y}_i^{(2)} \mathbf{b}_i \right) + Cm^2 \delta \epsilon$. We have $\text{Acc}_{\theta}(\mathcal{S}_2) = \frac{\mathbb{1}(\hat{\mathbf{y}}_q^{(2)} = \mathbf{y}_q^{(2)})}{2}$.

Then we have $\mathcal{S}_{1 \cup 2}$ contains $2N$ in-context examples from both tasks, specifically,

we have N from task 1 and rest from task 2. We have $\hat{w}^{(1)} = (\hat{w}_a/2, 0_{d_2})$, $\hat{w}^{(2)} = (0_{d_1}, \hat{w}_b/2)$, $\hat{y}_q = (\hat{y}_q^{(1)}, \hat{y}_q^{(2)})^\top$.

Since $y_{\tau,q}^{(k)} = \text{sgn}(\langle w_\tau, x_{\tau,q} \rangle)$, $\tilde{y}_{\tau,q}^{(k)} = \text{sgn}(\hat{y}_{\tau,q}^{(k)})$, following the proof of Lemma B.4.2, where Acc_θ only concerns the direction of \hat{w} and w , we have

$$\text{Acc}_\theta(\mathcal{S}_{1 \cup 2}) = \frac{\mathbb{1}(\tilde{y}_q^{(1)} = y_q^{(1)}) + \mathbb{1}(\tilde{y}_q^{(2)} = y_q^{(2)})}{2}.$$

Extending the above analysis into any of two simple tasks, when the composite task integrates them, we have

$$\text{Acc}_\theta(\mathcal{S}_k) + \text{Acc}_\theta(\mathcal{S}_g) \leq \text{Acc}_\theta(\mathcal{S}_{k \cup g}). \quad (\text{B.7})$$

□

We then prove Section 4.4.2, and we first restate it below.

If two tasks do not have confined support, there exists one setting in which we have

$$\text{Acc}_\theta(\mathcal{S}_k) = \text{Acc}_\theta(\mathcal{S}_g) = \text{Acc}_\theta(\mathcal{S}_{k \cup g}).$$

Proof of Section 4.4.2. WLOG, consider two simple tasks, $K = 2$. We have $x = (a, b)$, where $a \in \mathbb{R}^{d_1}$, $b \in \mathbb{R}^{d_2}$, $d_1 + d_2 = d$. Consider the setting where w also have the same active dimensions, i.e., for simple task 1, we have $w^{(1)} = (w_a, 0)$, for simple task 2, we have $w^{(2)} = (0, w_b)$.

We have $x \sim \Lambda$. Consider tasks are overlapping on all dimensions, where:

- Task 1: $x = (a^{(1)}, b^{(1)})^\top$, $y = (w_a^\top a^{(1)}, w_b^\top b^{(1)})$.
- Task 2: $x = (a^{(2)}, b^{(2)})^\top$, $y = (w_a^\top a^{(2)}, w_b^\top b^{(2)})$.
- Composed task: $x = (a, b)^\top$, $y = (w_a^\top a, w_b^\top b)$.

Similarly, we have:

$$\hat{y}_q = \frac{1}{N} \left(\sum_{i=1}^N y_i \mathbf{a}_i^\top, \sum_{i=1}^N y_i \mathbf{b}_i^\top, \sum_{i=1}^N y_i y_i^\top \right) \begin{pmatrix} \Gamma_1^{-1} \mathbf{a}_q \\ \Gamma_2^{-1} \mathbf{b}_q \\ 0 \end{pmatrix} \quad (\text{B.8})$$

$$= \left(\frac{1}{N} \sum_{i=1}^N y_i \mathbf{a}_i^\top \right) \Gamma_1^{-1} \mathbf{a}_q + \left(\frac{1}{N} \sum_{i=1}^N y_i \mathbf{b}_i^\top \right) \Gamma_2^{-1} \mathbf{b}_q \quad (\text{B.9})$$

$$= \frac{1}{N} \left(\mathbf{a}_q^\top \Gamma_1^{-1} \sum_{i=1}^N y_i^{(1)} \mathbf{a}_i + \mathbf{b}_q^\top \Gamma_2^{-1} \sum_{i=1}^N y_i^{(1)} \mathbf{b}_i \right). \quad (\text{B.10})$$

Note that composed example $\mathbf{x} = (\mathbf{a}, \mathbf{b})^\top$, $\mathbf{y} = (w_1^\top \mathbf{a}, w_2^\top \mathbf{b})$.

When in-context examples are from a simple task, we have N examples from simple task 1, $\mathcal{S}_1 = \left[\left\{ (\mathbf{a}_i^{(1)}, \mathbf{b}_i^{(1)}), y_i \right\}_{i=1}^N \right]$, and \hat{y}_q has the same form as Equation (B.10), similarly, for task 2.

Suppose $\mathcal{S}_{1 \cup 2}$ contains $2N$ examples from both tasks, where N from task 1 and rest from task 2. We have

$$\hat{y}_q = \frac{1}{2N} \left(\mathbf{a}_q^\top \Gamma_1^{-1} \sum_{i=1}^N y_i^{(1)} \mathbf{a}_i + \mathbf{b}_q^\top \Gamma_2^{-1} \sum_{i=1}^N y_i^{(1)} \mathbf{b}_i \right). \quad (\text{B.11})$$

We finish the proof by checking that Equation (B.10) and Equation (B.11) share the same direction.

□

B.4.3 Proof of Compositional Ability with Model Scale

Here, we provide the proof of our conclusions in Theorem B.3.1 in Section B.3.1 with respect to model performance and model scale. We first introduce a lemma under the $K = 1$ setting.

Accuracy under $K = 1$

When $K = 1$, we can give an upper bound of accuracy by Λ and Γ . Taking into account the optimal solution in Equation (B.2), we have the following accuracy lemma.

Lemma B.4.2. *Consider $K = 1$ and $x_q \sim \mathcal{N}(0, I_d)$. When $N > C$, where C is a constant, we have*

$$\mathbb{E}_{w_\tau, x_1, \dots, x_N} [\text{Acc}_\theta] \leq \text{tr}(\Gamma^{-1} \Lambda).$$

Proof of Lemma B.4.2. Since $K = 1$, the problem reduces to the linear regression problem in ICL. Consider the solution form in Lemma B.4.1, we have

$$\hat{y}_q = x_q^\top \frac{1}{N} \Gamma^{-1} \sum_{i=1}^N \langle w_\tau, x_i \rangle x_i$$

We re-write the form as $\hat{y}_q = x_q^\top \hat{w}$. Following Equation (4.3) in Zhang et al. (2023b), we have:

$$\hat{w} = \frac{1}{N} \Gamma^{-1} \sum_{i=1}^N \langle w_\tau, x_i \rangle x_i.$$

Recall the definition of Acc_θ and $y_{\tau,q}^{(k)} = \text{sgn}(\langle w_\tau, x_{\tau,q} \rangle)$, $\tilde{y}_{\tau,q}^{(k)} = \text{sgn}(\hat{y}_{\tau,q}^{(k)}) = \text{sgn}(\langle \hat{w}, x_{\tau,q} \rangle)$, for any $\alpha > 0$, we have:

$$\mathbb{E}_{w_\tau, x_1, \dots, x_N, x_q} [\text{Acc}_\theta] = \mathbb{P}(\langle x_q, w_\tau \rangle > 0, \langle x_q, \alpha \hat{w} \rangle > 0) + \mathbb{P}(\langle x_q, w_\tau \rangle < 0, \langle x_q, \alpha \hat{w} \rangle < 0).$$

Denote hyperplane orthogonal to w as \mathcal{P}_w and similar to $\mathcal{P}_{\hat{w}}$. Recall that x_q is independent of other samples. We have the expectation conditioned on w_τ, x_1, \dots, x_N is the probability that x_q falls out of the angle between \mathcal{P}_w and $\mathcal{P}_{\hat{w}}$. Denote the angle between w and \hat{w} as $\tilde{\theta}$. As x_q is uniform along each direction (uniform distribution or isotropic Gaussian), then the probability is $1 - \frac{|\tilde{\theta}|}{\pi}$ given w_τ, x_1, \dots, x_N . Then

$\mathbb{E}_{w_\tau, x_1, \dots, x_N} [\text{Acc}_\theta] = \mathbb{E}_{w_\tau, x_1, \dots, x_N} \left[1 - \frac{|\tilde{\theta}|}{\pi} \right]$. Note that

$$\mathbb{E}_{w_\tau, x_1, \dots, x_N} [\cos(\tilde{\theta})] = \left\langle \frac{w_\tau}{\|w_\tau\|_2}, \frac{\hat{w}}{\|\hat{w}\|_2} \right\rangle.$$

As, we can choose α , w.l.o.g, we take $\|w_\tau\| = \|\hat{w}\| = 1$, then we have

$$\mathbb{E}_{w_\tau, x_1, \dots, x_N} [\cos(\tilde{\theta})] = \mathbb{E}_{w_\tau} [\mathbb{E}_{x_1, \dots, x_N} [\langle w_\tau, \hat{w} \rangle | w_\tau]].$$

Given w_τ , we have

$$\begin{aligned} \mathbb{E}[\hat{w}|w_\tau] &= \frac{1}{N} \Gamma^{-1} \sum_{i=1}^N \mathbb{E}[\langle w_\tau, x_i \rangle x_i | w_\tau] \\ &= \frac{1}{N} \Gamma^{-1} \sum_{i=1}^N \Lambda w_\tau \\ &= \Gamma^{-1} \Lambda w_\tau. \end{aligned}$$

Then, we have

$$\begin{aligned} \mathbb{E}_{w_\tau} [\langle \hat{w}, w_\tau \rangle] &= \langle \Gamma^{-1} \Lambda w_\tau^\top, w_\tau \rangle \\ &= \text{tr}(\Gamma^{-1} \Lambda). \end{aligned}$$

Thus, we have

$$\mathbb{E} \cos(\tilde{\theta}) = \text{tr}(\Gamma^{-1} \Lambda) \tag{B.12}$$

$$\mathbb{E} [\text{Acc}_\theta] = \mathbb{E} \left[1 - \frac{|\tilde{\theta}|}{\pi} \right]. \tag{B.13}$$

Note that when $\theta \leq \frac{\pi}{6}$, we have $1 - \frac{|\tilde{\theta}|}{\pi} \leq \cos(\theta)$. Thus, as $N > C$ where C is constant, we have \hat{w} and w_τ are closed and satisfy $\theta \leq \frac{\pi}{6}$. Then we get the statement. \square

Model scale on composite tasks

Here, we present proof for model scale and performance on composite tasks. Recall we consider the rank of W^{*PV} and W^{*KQ} as a measure of the model's scale.

We first introduce a lemma about \mathbf{U} as an optimal full-rank solution.

Lemma B.4.3 (Corollary A.2 in Zhang et al. (2023b)). *The loss function $\tilde{\ell}$ in Lemma B.4.1 satisfies*

$$\min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) = -\frac{1}{2} \text{tr}[\Lambda^2 \Gamma^{-1}],$$

where $\mathbf{U} = c\Gamma^{-1}$, $\mathbf{u} = \frac{1}{c}$ for any non-zero constant c are minimum solution. We also have

$$\tilde{\ell}(\mathbf{U}, \mathbf{u}) - \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) = \frac{1}{2} \left\| \Gamma^{\frac{1}{2}} \left(\mathbf{u} \Lambda^{\frac{1}{2}} \mathbf{U} \Lambda^{\frac{1}{2}} - \Lambda \Gamma^{-1} \right) \right\|_F^2. \quad (\text{B.14})$$

As the scale of the model decreases, the rank of \mathbf{U} also reduces, leading to an optimal reduced rank solution $\tilde{\mathbf{U}}$. Our findings reveal that this reduced rank $\tilde{\mathbf{U}}$ can be viewed as a truncated form of the full-rank solution \mathbf{U} . This implies that smaller-scale models are essentially truncated versions of larger models, maintaining the core structure but with reduced complexity.

Recall Λ is the covariance matrix, we have eigendecomposition $\Lambda = \mathbf{Q}\mathbf{D}\mathbf{Q}^\top$, where \mathbf{Q} is an orthonormal matrix containing eigenvectors of Λ and \mathbf{D} is a sorted diagonal matrix with non-negative entries containing eigenvalues of Λ , denoting as $\mathbf{D} = \text{diag}([\lambda_1, \dots, \lambda_d])$, where $\lambda_1 \geq \dots \geq \lambda_d \geq 0$. We introduce the lemma below.

Lemma B.4.4 (Optimal rank- r solution). *Recall the loss function $\tilde{\ell}$ in (Lemma B.4.1). Let*

$$\mathbf{U}^*, \mathbf{u}^* = \arg \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \text{rank}(\mathbf{U}) \leq r, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}).$$

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

Then, we proof the Lemma B.4.4

Proof of Lemma B.4.4. Note that,

$$\begin{aligned} \arg \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \text{rank}(\mathbf{U}) \leq r, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) &= \arg \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \text{rank}(\mathbf{U}) \leq r, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) - \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) \\ &= \arg \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \text{rank}(\mathbf{U}) \leq r, \mathbf{u} \in \mathbb{R}} \left(\tilde{\ell}(\mathbf{U}, \mathbf{u}) - \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) \right). \end{aligned}$$

Thus, we may consider Equation (B.14) in Lemma B.4.3 only. On the other hand, we have

$$\begin{aligned} \Gamma &= \left(1 + \frac{1}{N}\right) \Lambda + \frac{1}{N} \text{tr}(\Lambda) \mathbf{I}_{d \times d} \\ &= \left(1 + \frac{1}{N}\right) \mathbf{Q} \mathbf{D} \mathbf{Q}^\top + \frac{1}{N} \text{tr}(\mathbf{D}) \mathbf{Q} \mathbf{I}_{d \times d} \mathbf{Q}^\top \\ &= \mathbf{Q} \left(\left(1 + \frac{1}{N}\right) \mathbf{D} + \frac{1}{N} \text{tr}(\mathbf{D}) \mathbf{I}_{d \times d} \right) \mathbf{Q}^\top. \end{aligned}$$

We denote $\mathbf{D}' = \left(1 + \frac{1}{N}\right) \mathbf{D} + \frac{1}{N} \text{tr}(\mathbf{D}) \mathbf{I}_{d \times d}$. 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 $\mathbf{V} = \mathbf{u} \mathbf{Q}^\top \mathbf{U} \mathbf{Q}$. Since Γ and Λ are commutable and the Frobenius norm (F-norm) of a matrix does not change after multiplying it by an orthonormal matrix, we have Equation (B.14) as

$$\begin{aligned} \tilde{\ell}(\mathbf{U}, \mathbf{u}) - \min_{\mathbf{U} \in \mathbb{R}^{d \times d}, \mathbf{u} \in \mathbb{R}} \tilde{\ell}(\mathbf{U}, \mathbf{u}) &= \frac{1}{2} \left\| \Gamma^{\frac{1}{2}} \left(\mathbf{u} \Lambda^{\frac{1}{2}} \mathbf{U} \Lambda^{\frac{1}{2}} - \Lambda \Gamma^{-1} \right) \right\|_{\text{F}}^2 \\ &= \frac{1}{2} \left\| \Gamma^{\frac{1}{2}} \Lambda^{\frac{1}{2}} \left(\mathbf{u} \mathbf{U} - \Gamma^{-1} \right) \Lambda^{\frac{1}{2}} \right\|_{\text{F}}^2 \\ &= \frac{1}{2} \left\| \mathbf{D}'^{\frac{1}{2}} \mathbf{D}^{\frac{1}{2}} \left(\mathbf{V} - \mathbf{D}'^{-1} \right) \mathbf{D}^{\frac{1}{2}} \right\|_{\text{F}}^2. \end{aligned}$$

As $\mathbf{W}^{\mathbf{KQ}}$ is a matrix whose rank is at most r , we have \mathbf{V} is also at most rank r . Then, we denote $\mathbf{V}^* = \arg \min_{\mathbf{V} \in \mathbb{R}^{d \times d}, \text{rank}(\mathbf{V}) \leq r} \left\| \mathbf{D}'^{\frac{1}{2}} \mathbf{D}^{\frac{1}{2}} \left(\mathbf{V} - \mathbf{D}'^{-1} \right) \mathbf{D}^{\frac{1}{2}} \right\|_{\text{F}}^2$. We can see that \mathbf{V}^* is a diagonal matrix. Denote $\mathbf{D}' = \text{diag}([\lambda'_1, \dots, \lambda'_d])$ and $\mathbf{V}^* = \text{diag}([v_1^*, \dots, v_d^*])$.

Then, we have

$$\left\| D'^{\frac{1}{2}} D^{\frac{1}{2}} \left(V - D'^{-1} \right) D^{\frac{1}{2}} \right\|_F^2 \quad (\text{B.15})$$

$$= \sum_{i=1}^d \left(\lambda_i'^{\frac{1}{2}} \lambda_i \left(v_i^* - \frac{1}{\lambda_i'} \right) \right)^2 \quad (\text{B.16})$$

$$= \sum_{i=1}^d \left(\left(1 + \frac{1}{N} \right) \lambda_i + \frac{\text{tr}(\mathbf{D})}{N} \right) \lambda_i^2 \left(v_i^* - \frac{1}{\left(1 + \frac{1}{N} \right) \lambda_i + \frac{\text{tr}(\mathbf{D})}{N}} \right)^2. \quad (\text{B.17})$$

As V^* is the minimum rank r solution, we have that $v_i^* \geq 0$ for any $i \in [d]$ and if $v_i^* > 0$, we have $v_i^* = \frac{1}{\left(1 + \frac{1}{N} \right) \lambda_i + \frac{\text{tr}(\mathbf{D})}{N}}$. Denote

$$g(x) = \left(\left(1 + \frac{1}{N} \right) x + \frac{\text{tr}(\mathbf{D})}{N} \right) x^2 \left(\frac{1}{\left(1 + \frac{1}{N} \right) x + \frac{\text{tr}(\mathbf{D})}{N}} \right)^2 = x^2 \left(\frac{1}{\left(1 + \frac{1}{N} \right) x + \frac{\text{tr}(\mathbf{D})}{N}} \right).$$

It is easy to see that $g(x)$ is an increasing function on $[0, \infty)$. Now, we use contradiction to show that V^* only has non-zero entries in the first r diagonal entries. Suppose $i > r$, such that $v_i^* > 0$, then we must have $j \leq r$ such that $v_j^* = 0$ as V^* is a rank r solution. We find that if we set $v_i^* = 0, v_j^* = \frac{1}{\left(1 + \frac{1}{N} \right) \lambda_j + \frac{\text{tr}(\mathbf{D})}{N}}$ and all other values remain the same, Equation (B.17) will strictly decrease as $g(x)$ is an increasing function on $[0, \infty)$. Thus, here is a contradiction. We finish the proof by $V^* = \mathbf{u} \mathbf{Q}^\top \mathbf{U}^* \mathbf{Q}$. \square

We then ready to prove the Theorem B.3.1 in Section B.3.1, we first re-state it below.

Theorem B.3.1. *Suppose a composite task satisfies confined support. Suppose that we have $(x_1, y_1, \dots, x_N, y_N, x_q)$ as a testing input prompt and the corresponding W where $y_i = W x_i$. As rank r decreases, $\mathbb{E}_{W, x_1, \dots, x_N} [\text{Acc}_\theta]$ will have a smaller upper bound.*

Proof of Theorem B.3.1. We first prove in a simple task setting ($K = 1$), that the accuracy will have such a conclusion. By Lemma B.4.2, consider $x_q \sim \mathcal{N}(0, \mathbf{I}_d)$.

When $N > C$, where C is a constant, we have

$$\mathbb{E}_{w_\tau, x_1, \dots, x_N} [\text{Acc}_\theta] \leq \text{tr}(\Gamma^{-1}\Lambda).$$

Recall Lemma B.4.4. WLOG, we take $c = 1$. We have

$$\begin{aligned} \text{tr}(\Gamma^{-1}\Lambda) &= \text{tr}(QV^*DQ) \\ &= \sum_{i=1}^r \frac{N}{N + 1 + \sum_{j=1}^r \frac{\lambda_j}{\lambda_i}}, \end{aligned}$$

where second equation comes from Lemma B.4.4.

Under the confined support setting, the same conclusion holds since Equation (B.7) in the proof of Theorem 4.4.4.

□

C APPENDIX OF CHAPTER 5

In this appendix, we provide additional implementation details in section C.1. We present detailed results accompanying our experiments in section 5.3, including broader benchmarks and evaluations with other LLM backbones in section C.2. We include further ablations on the design of switches and comparisons with naive sampling strategies in section C.3. We provide additional qualitative results demonstrating latency and content adaptivity in section C.4. We conclude with further discussion on practical deployment considerations in section C.5.

C.1 Further Implementation Details

Probabilistic execution plan sampling. Recall that in our probabilistic model, we define the distribution $p(\mathbf{s}|\{\mathbf{z}^{v^q}\}, l, \Phi)$ via a sampling process. Given the input tokens and a latency budget l , the output of the lightweight scheduler is a logits vector corresponding to the K available switches: $\pi_1, \pi_2, \dots, \pi_K \in \mathbb{R}$, where π_i represents the relative likelihood of selecting the i^{th} switch. The latency budget l allows us to define k , the maximum number of switches allowed to activate. Then, a sampled execution plan from $p(\mathbf{s}|\{\mathbf{z}^{v^q}\}, l, \Phi)$ can be uniquely defined by a subset of k distinct elements from $\{1, 2, \dots, K\}$, corresponding to its activated switches. We sample the execution plan by randomly picking k switches one by one, without replacement, following the logits $\{\pi_i\}_{i=1}^K$. The complete sampling procedure is summarized in Algorithm 3, where $\text{Cat}(\Omega, \{\pi_i : i \in \Omega\})$ denotes the categorical distribution of selecting an element from Ω with probabilities parameterized by $\{\eta_i : i \in \Omega\} = \text{Softmax}(\{\pi_i : i \in \Omega\})$. The process ensures that the sampled execution plan adheres to the input budget while maximizing the utilization.

Differentiable sampling with Gumbel-Softmax. Our designed scheduler is difficult to train as it involves a non-differentiable discrete sampling process, which prevents gradients from backpropagate to the scheduler during training. A common workaround involves using a score function estimator Gu et al. (2015); Wu et al.

Algorithm 3 Sampling $\mathbf{s} \sim p(\mathbf{s}|\{\mathbf{z}^{v|q}\}, l, \Phi)$

Input: Latency budget l , sampling logits $\{\pi_i\}_{i=1}^K$
Output: Sampled binary vector $\mathbf{s} \in \{0, 1\}^K$

Determine number of selections k based on l

Initialize available set of switches $\Omega \leftarrow \{1, 2, \dots, K\}$

Initialize $\mathbf{s} \leftarrow (0, 0, \dots, 0) \in \{0, 1\}^K$
for $i = 1$ to k **do**

 Sample $\omega \sim \text{Cat}(\Omega, \{\pi_i : i \in \Omega\})$

 $\mathbf{s}[\omega] \leftarrow 1$ (activating the chosen switch)

 $\Omega \leftarrow \Omega \setminus \{\omega\}$
end for
return \mathbf{s}

(2018); however, this method often suffers from high variance and slow convergence. Instead, we employ Gumbel-Softmax Jang et al. (2017), a reparameterization trick for sampling from categorical distribution. In our implementation, the Gumbel-Softmax approximates $\omega \sim \text{Cat}(\Omega, \{\pi_i : i \in \Omega\})$ with a continuous random vector $\tilde{\omega}$:

$$\tilde{\omega} = \text{Softmax}([g_i + \log \eta_i]_{i \in \Omega}), \quad (\text{C.1})$$

where each g_i is i.i.d. sample drawn from $\text{Gumbel}(0, 1)$; and $\eta_i = \text{Softmax}(\{\pi_j : j \in \Omega\})[i]$ is the probability of activating the i^{th} switch, computed by the scheduler. Note that $\tilde{\omega}$ is continuous and has a well-defined gradient. To maintain a hard execution plan, we take the one-hot encoding of $\tilde{\omega}$ and apply the straight-through estimator (see Jang et al. (2017) for more details).

Training details. Training details were discussed in section 5.3 of the main paper. Here we show training curve of AdaLLaVA-L-7B with LLaVA 1.5 in fig. C.1.

C.2 Detailed Results

Full results on LLaVA 1.5. We report the full set of results on LLaVA 1.5, LLaVA-PruMerge, LLaVA-PruMerge+ and FastV in tables C.2 to C.4, as a complement to tables 5.1 to 5.3. All experiments follow the same setting as described in section 5.3.1.

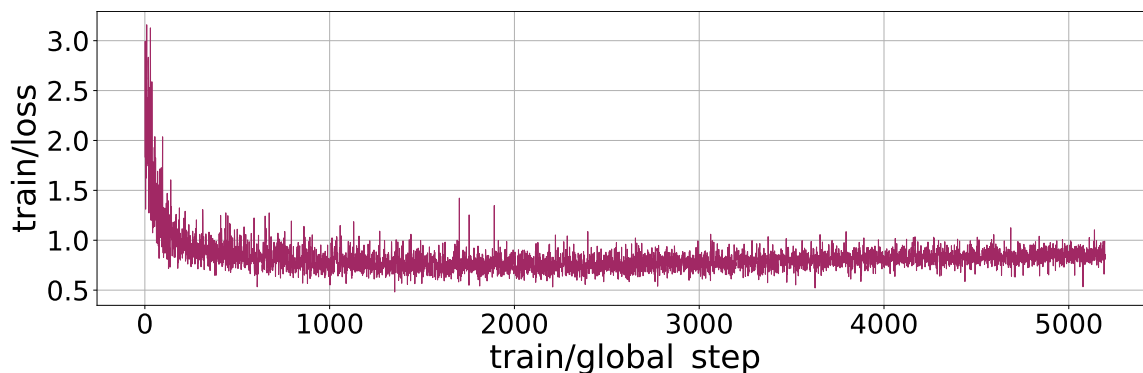


Figure C.1: Training logs

| Model | GQA | SEED-Bench | MM-Vet | LLaVa-WILD | VizWiz |
|--------------------|------|------------|--------|------------|--------|
| LLaVA-1.5-7B | 62.0 | 58.6 | 31.1 | 65.4 | 50.0 |
| AdaLLaVA-L-7B-100% | 61.5 | 60.5 | 30.7 | 64.2 | 54.3 |
| AdaLLaVA-L-7B-85% | 61.3 | 60.2 | 30.0 | 62.1 | 51.5 |
| AdaLLaVA-L-7B-60% | 58.7 | 59.8 | 23.9 | 46.3 | 44.8 |

Table C.1: Results on broader benchmarks.

These results confirm that our AdaLLaVA framework successfully adapts to LLaVA 1.5 across different backbone sizes, and can be further combined with recent token selection methods (PruMerge, PruMerge+ and FastV) to further enhance efficiency. We maintain comparable performance while improving efficiency across multiple benchmarks. Additionally, our analysis reveals how performance varies under different latency constraints, demonstrating our framework’s ability to trade between accuracy and latency.

Broader benchmarks. We extend our AdaLLaVA-L framework on broader benchmarks reported in Liu et al. (2024a), namely GQA Hudson and Manning (2019), SEED-Bench Li et al. (2023a), MM-Vet Yu et al. (2023), LLaVa-WILD Liu et al. (2023), and VizWiz Gurari et al. (2018) (see table C.1). The model shows comparable performance and adaptive ability under different latency budget. The results demonstrate the strong generalization of AdaLLaVA to a wide range of benchmarks.

AdaLLaVA-L maintains comparable performance under full computational

budgets. With reduced compute budgets, AdaLLaVA-L shows minimal performance degradation: an average accuracy drop of only 0.7% at 85% budget. Notably, AdaLLaVA-L shows 1.5% average performance boost at full compute budget.

Generalization across MLLMs. We demonstrate that AdaLLaVA can generalize to other MLLMs beyond LLaVA. We consider Mipha-3B Zhu et al. (2024b), a lightweight MLLM built on Phi-2.7B Javaheripi et al. (2023). Specifically, we apply AdaLLaVA-L on Mipha-3B, following its training strategy Zhu et al. (2024b), and report the results on a comprehensive MLLM benchmark (MME), shown in fig. C.2. We see that AdaLLaVA-L maintains comparable performance under full computational budgets. With reduced compute budgets, AdaLLaVA-L shows minimal performance degradation: an average accuracy drop of only 3.4% at 85% budget and 6.1% at 60% budget. These results have similar trend to those with LLaVA-1.5 in fig. 5.1.

C.3 Additional Ablation Studies

We now conduct ablation study, exploring different design choices. We explore the performance of different designs of tunable switches, namely AdaLLaVA-L and AdaLLaVA-H (detailed in section 5.2.5). All results are reported with LLaVA 1.5-7B Model on VQAv2 dataset benchmark.

Number & granularity of switches. We here conduct ablation studies to examine how the number and granularity of switches affect performance. fig. C.3 (**Left**) compares switches for the last 16 layers (used in section 5.3) versus 24 layers in AdaLLaVA-L. While 24 switches enable finer FLOPs control, they significantly reduce model performance. The 16-switch configuration provides better accuracy while maintaining efficient adaptability. fig. C.3 (**Right**) evaluates attention sampling group sizes in AdaLLaVA-H, focusing on operations within the last 16 layers. While both 4-head and 8-head (used in section 5.3) configurations show comparable performance-latency tradeoffs, the 4-head version enables more granular latency control.

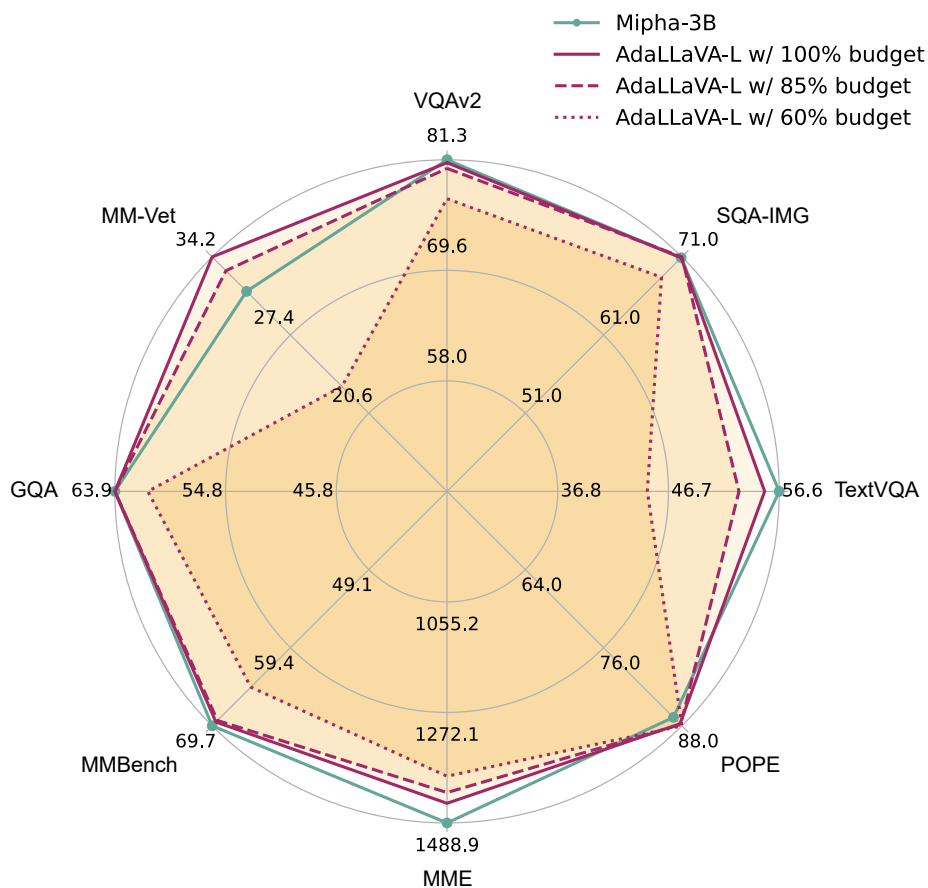


Figure C.2: **Relative performance of applying AdaLLaVA-L to Mipha-3B under various latency budget.** The center of the radar corresponds to 60% performance of the base Mipha-3B.

Design of the switches L vs H. We also explore the performance of design of tunable switches, particularly AdaLLaVA-L versus AdaLLaVA-H. Both methods allow adaptivity to latency requirements without significant modification to the pretrained LLM, while AdaLLaVA-H offers better flexibility to latency input.

As shown in fig. C.3, from FLOPs ranging from 5T to 8T, AdaLLaVA-H-8-head shows slightly better performance overall, reaching approximately 76% on VQA v2 Accuracy compared to AdaLLaVA-L-16-switch which peaks around 75%. Moreover, AdaLLaVA-H demonstrates finer-grained control over the accuracy-latency trade-off. This is evident from the smoother curve of AdaLLaVA-H, which can be attributed to its head/neuron-level switches providing more granular control

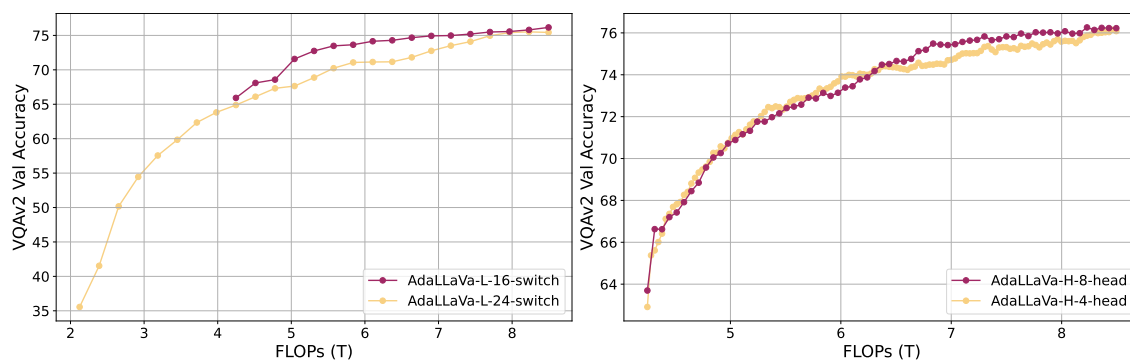


Figure C.3: Ablation studies on switch design choices.

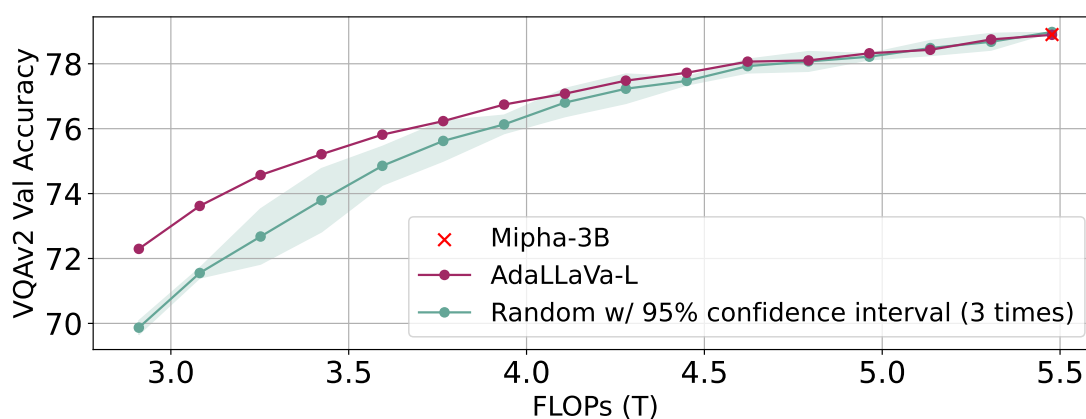


Figure C.4: Comparison to random sampling

over computational resources compared to the layer-level switches. This flexibility allows AdaLLaVA-H to accommodate a wider range of latency budgets.

Comparison with naive sampling strategies. We compare the performance of our AdaLLaVA-L versus random uniform sampling, where we disable the scheduler during training, showing in fig. C.4. Both methods are built on Mipha-3B and fine-tuned using the same procedure. Random sampling is worse than AdaLLaVA and has high variance in results (shaded area).

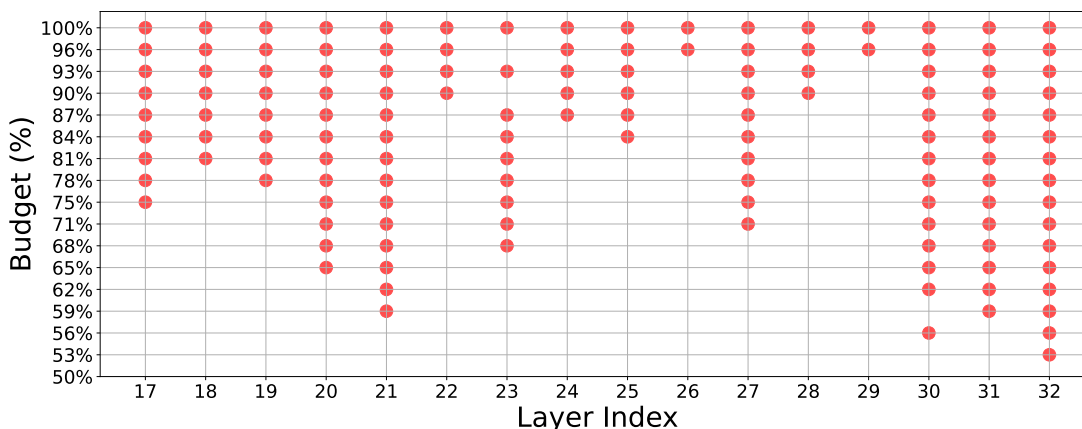


Figure C.5: Visualization of execution plan

C.4 Additional Results on Adaptivity

We provide further results to demonstrate AdaLLaVA’s latency and content adaptivity.

Model Response under different latency. Here we show additional results on model response given same image-text input under different latency budget, similar to fig. 5.1. As shown in tables C.5 to C.7, given an image-query pair and latency constraint, AdaLLaVA learns to generate appropriate responses while adapting to varying computational budgets.

Visualization of execution plans with different latency. We report execution plans of the same input with varying budgets in fig. C.5. As budget decreases, the scheduler prioritizes keeping the last 3 layers over others. We have included outputs for the same input with varying budgets in tables C.5 to C.7.

Visualization for latency token attention. We provide additional results on content awareness by showing the key-query attention scores of the latency token and the input visual tokens with different text questions, similar to fig. 5.4.

Figures C.6 and C.7 further demonstrate the model’s content-aware behavior. In the father-child scene image, attention spans the entire street for scene description but focuses centrally for query asking for activity. For Happy Plaza image, attention targets the storefront sign for location queries but shifts to promotional areas for

query about special offers. In the restaurant scene, attention distributes across interior elements when identifying location type but concentrates on the woman’s clothing for attire questions. This consistently shows model adjusts its attention based on the query.

Visualization for latency token across layers. We plot the evolution of the latency token from layers 12 to 16 using the same example in fig. 5.4 of the main paper (see figure below). As seen in fig. C.8, the latency token progressively gathers key information from the input visual tokens for scheduling.



Figure C.8: Evolution of latency token across layers in AdaLLaVA-L on 7b model.

C.5 Further Discussion

FLOPs, latency, and cross-device portability. In our work, compute budgets are expressed as percentages of a base model’s FLOPs. Percentages can be translated into absolute FLOP targets, given base model’s architecture and input size, from which total FLOPs can be estimated. We choose FLOPs over runtime latency, as it abstracts away hardware / software specific variations . We acknowledge that the relationship between FLOPs and latency vary across devices, depending on model architecture, hardware and software. We leave cross-device portability to future work.

| Method | LLM | Budget (%) | FLOPs (T) | Prefill time (ms) |
|--------------------------------------|------------|------------|-----------|-------------------|
| BLIP-2 Li et al. (2023b) | Vicuna-13B | 100 | - | - |
| InstructBLIP Dai et al. (2023) | Vicuna-7B | 100 | - | - |
| InstructBLIP Dai et al. (2023) | Vicuna-13B | 100 | - | - |
| Shikra Chen et al. (2023) | Vicuna-13B | 100 | - | - |
| IDEFICS-9B Laurençon (2023) | LLaMA-7B | 100 | - | - |
| IDEFICS-80B Laurençon (2023) | LLaMA-65B | 100 | - | - |
| Qwen-VL Bai et al. (2023) | Qwen-7B | 100 | - | - |
| Qwen-VL-Chat Bai et al. (2023) | Qwen-7B | 100 | - | - |
| LLaVA-1.5 Liu et al. (2024a) | Vicuna-7B | 100 | 8.6 | 81 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 8.6 | 81 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 7.2 | 69 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 5.1 | 49 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 8.6 | 81 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 7.2 | 69 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 5.1 | 49 |
| LLaVA-1.5 | Vicuna-13B | 100 | 16.7 | 157 |
| w/ AdaLLaVA-L | Vicuna-13B | 100 | 16.7 | 157 |
| w/ AdaLLaVA-L | Vicuna-13B | 85 | 14.2 | 133 |
| w/ AdaLLaVA-L | Vicuna-13B | 60 | 10.0 | 94 |
| w/ AdaLLaVA-H | Vicuna-13B | 100 | 16.7 | 157 |
| w/ AdaLLaVA-H | Vicuna-13B | 85 | 14.2 | 133 |
| w/ AdaLLaVA-H | Vicuna-13B | 60 | 10.0 | 94 |
| Prumerge Shang et al. (2024) | Vicuna-7B | 100 | 1.4 | 16 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 1.4 | 16 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 1.2 | 14 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 0.8 | 10 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 1.4 | 16 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 1.2 | 14 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 0.8 | 10 |
| Prumerge+ Shang et al. (2024) | Vicuna-7B | 100 | 3.0 | 29 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 3.0 | 29 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 2.6 | 24 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 1.8 | 17 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 3.0 | 29 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 2.6 | 24 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 1.8 | 17 |
| FastV (K=2,R=0.5) Chen et al. (2024) | Vicuna-7B | 100 | 4.9 | 47 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 4.9 | 47 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 4.2 | 40 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 3.0 | 29 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 4.9 | 47 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 4.2 | 40 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 3.0 | 29 |

Table C.2: **Efficiency metrics of MLLMs.** Budget (%): input latency requirement. AdaLLaVA-L: switches on selecting different transformer blocks. AdaLLaVA-H: switches on selecting different attention heads and MLP activations.

| Method | LLM | Budget (%) | VQA ^{v2} | SQA ^I | VQA ^T |
|--------------------------------------|------------|------------|---------------------|-------------------|---------------------|
| | | | Goyal et al. (2017) | Lu et al. (2022a) | Singh et al. (2019) |
| BLIP-2 Li et al. (2023b) | Vicuna-13B | 100 | 41.0 | 61 | 42.5 |
| InstructBLIP Dai et al. (2023) | Vicuna-7B | 100 | - | 60.5 | 50.1 |
| InstructBLIP Dai et al. (2023) | Vicuna-13B | 100 | - | 63.1 | 50.7 |
| Shikra Chen et al. (2023) | Vicuna-13B | 100 | 77.4 | - | - |
| IDEFICS-9B Laurençon (2023) | LLaMA-7B | 100 | 50.9 | - | 25.9 |
| IDEFICS-80B Laurençon (2023) | LLaMA-65B | 100 | 60.0 | - | 30.9 |
| Qwen-VL Bai et al. (2023) | Qwen-7B | 100 | 78.8 | 67.1 | 63.8 |
| Qwen-VL-Chat Bai et al. (2023) | Qwen-7B | 100 | 78.2 | 68.2 | 61.5 |
| LLaVA-1.5 Liu et al. (2024a) | Vicuna-7B | 100 | 78.5 | 66.8 | 58.2 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 78.4 | 67.8 | 57.0 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 77.1 | 67.4 | 54.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 75.0 | 66.9 | 47.7 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 77.9 | 68.5 | 57.1 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 76.8 | 68.2 | 55.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 74.2 | 68.1 | 48.7 |
| LLaVA-1.5 | Vicuna-13B | 100 | 80.0 | 71.6 | 61.3 |
| w/ AdaLLaVA-L | Vicuna-13B | 100 | 79.7 | 72.4 | 59.9 |
| w/ AdaLLaVA-L | Vicuna-13B | 85 | 79.1 | 72.4 | 58.0 |
| w/ AdaLLaVA-L | Vicuna-13B | 60 | 77.4 | 71.8 | 54.3 |
| w/ AdaLLaVA-H | Vicuna-13B | 100 | 80.0 | 72.6 | 59.9 |
| w/ AdaLLaVA-H | Vicuna-13B | 85 | 78.9 | 72.3 | 59.0 |
| w/ AdaLLaVA-H | Vicuna-13B | 60 | 76.4 | 71.3 | 53.3 |
| Prumerge Shang et al. (2024) | Vicuna-7B | 100 | 72.0 | 68.5 | 56.0 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 71.0 | 69.1 | 54.1 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 69.7 | 68.6 | 52.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 67.8 | 68.7 | 44.7 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 70.4 | 67.9 | 54.4 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 69.2 | 67.2 | 52.3 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 66.8 | 68.1 | 45.9 |
| Prumerge+ Shang et al. (2024) | Vicuna-7B | 100 | 76.8 | 68.3 | 57.1 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 76.3 | 68.3 | 55.8 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 75.3 | 68.5 | 52.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 73.0 | 67.7 | 47.4 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 76.0 | 67.9 | 56.0 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 75.0 | 68.1 | 54.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 72.2 | 67.6 | 47.2 |
| FastV (K=2,R=0.5) Chen et al. (2024) | Vicuna-7B | 100 | 77.7 | 68.7 | 58.1 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 77.8 | 67.7 | 57.0 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 76.9 | 67.8 | 54.4 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 74.5 | 67.0 | 47.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 77.4 | 68.4 | 57.0 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 76.6 | 67.7 | 54.8 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 73.9 | 68.3 | 48.7 |

Table C.3: **VQA benchmark results.** VQA^{v2}: VQAv2 set. SQA^I: ScienceQA set. VQA^T: TextVQA set.

| Method | LLM | Budget (%) | POPE | MME | MMBench |
|--------------------------------------|------------|------------|-------------------|------------------|--------------------|
| | | | Li et al. (2023c) | Fu et al. (2023) | Liu et al. (2025a) |
| BLIP-2 Li et al. (2023b) | Vicuna-13B | 100 | 85.3 | 1293.8 | - |
| InstructBLIP Dai et al. (2023) | Vicuna-7B | 100 | - | - | 36 |
| InstructBLIP Dai et al. (2023) | Vicuna-13B | 100 | 78.9 | 1212.8 | - |
| Shikra Chen et al. (2023) | Vicuna-13B | 100 | - | - | 58.8 |
| IDEFICS-9B Laurençon (2023) | LLaMA-7B | 100 | - | - | 48.2 |
| IDEFICS-80B Laurençon (2023) | LLaMA-65B | 100 | - | - | 54.5 |
| Qwen-VL Bai et al. (2023) | Qwen-7B | 100 | - | - | 38.2 |
| Qwen-VL-Chat Bai et al. (2023) | Qwen-7B | 100 | - | 1487.5 | 60.6 |
| LLaVA-1.5 Liu et al. (2024a) | Vicuna-7B | 100 | 85.9 | 1510.7 | 64.3 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 85.9 | 1521.0 | 63.7 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 86.4 | 1487.2 | 63.7 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 86.1 | 1463.8 | 63.8 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 86.9 | 1471.1 | 64.1 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 86.7 | 1494.9 | 64.3 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 85.0 | 1489.6 | 64.8 |
| LLaVA-1.5 | Vicuna-13B | 100 | 85.9 | 1531.3 | 67.7 |
| w/ AdaLLaVA-L | Vicuna-13B | 100 | 86.9 | 1559.3 | 69.2 |
| w/ AdaLLaVA-L | Vicuna-13B | 85 | 86.2 | 1563.9 | 68.9 |
| w/ AdaLLaVA-L | Vicuna-13B | 60 | 87.3 | 1552.6 | 68.6 |
| w/ AdaLLaVA-H | Vicuna-13B | 100 | 87.3 | 1531.9 | 67.4 |
| w/ AdaLLaVA-H | Vicuna-13B | 85 | 86.1 | 1554.5 | 67.0 |
| w/ AdaLLaVA-H | Vicuna-13B | 60 | 85.0 | 1529.5 | 66.9 |
| Prumerge Shang et al. (2024) | Vicuna-7B | 100 | 76.3 | 1350.3 | 60.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 74.2 | 1312.6 | 58.4 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 75.6 | 1313.3 | 59.1 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 75.8 | 1332.5 | 57.0 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 77.2 | 1311.4 | 60.1 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 75.5 | 1309.7 | 60.7 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 76.4 | 1289.3 | 58.7 |
| Prumerge+ Shang et al. (2024) | Vicuna-7B | 100 | 84.0 | 1462.4 | 64.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 85.1 | 1455.5 | 61.9 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 85.7 | 1429.5 | 62.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 85.6 | 1450.9 | 61.3 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 86.6 | 1503.2 | 63.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 86.4 | 1511.8 | 63.6 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 86.4 | 1458.0 | 63.6 |
| FastV (K=2,R=0.5) Chen et al. (2024) | Vicuna-7B | 100 | 82.5 | 1516.2 | 64.3 |
| w/ AdaLLaVA-L | Vicuna-7B | 100 | 82.8 | 1494.3 | 63.5 |
| w/ AdaLLaVA-L | Vicuna-7B | 85 | 83.3 | 1478.1 | 63.7 |
| w/ AdaLLaVA-L | Vicuna-7B | 60 | 83.8 | 1463.1 | 63.2 |
| w/ AdaLLaVA-H | Vicuna-7B | 100 | 84.3 | 1484.2 | 63.8 |
| w/ AdaLLaVA-H | Vicuna-7B | 85 | 83.9 | 1520.5 | 63.9 |
| w/ AdaLLaVA-H | Vicuna-7B | 60 | 82.4 | 1452.8 | 65.3 |

Table C.4: **Hallucination and general benchmark results.** POPE evaluates object hallucination. MME and MMBench assess general multimodal understanding.

Visual input example:



| | |
|-----------------|---|
| User | Describe this image. |
| LLaVA 1.5-7B | The image features an older man sitting at a table, surrounded by various items. He is using a cell phone, possibly checking messages or browsing the internet... |
| AdaLLaVA-L-50% | <p>The image depicts a man sitting at a table with a large number of papers and books on it. He is using a cell phone, possibly checking or communicating with someone. The man is wearing a tie, which adds to the professional setting.</p> <p>The table is surrounded by several chairs, with one close to the man and others further away. A clock can be seen on the left side of the table, and a handbag is placed on the right side. The scene appears to be a workspace or a study area, with the man working on his tasks and using the cell phone.</p> |
| AdaLLaVA-L-75% | <p>The image features a man sitting at a table, surrounded by various items. He is wearing a suit and tie, and appears to be using a cell phone. The table is set with a clock, a cup, and a book. There are also two chairs in the scene, one near the man and another further away. In the background, there are two more books placed on the ground, and a handbag is located near the edge of the scene. The man seems to be engaged in a task or a conversation, as evidenced by his use of the cell phone.</p> |
| AdaLLaVA-L-100% | <p>The image features an older man sitting at a table, working on a laptop. He is surrounded by various items, including a cup, a cell phone, and a book. The table is adorned with a red and black tablecloth, and there are several signs and posters on the wall behind him.</p> <p>In the room, there are two chairs, one near the man and another further away. A clock is also visible on the wall, indicating the time. The scene appears to be a workspace or a study area, with the man focused on his tasks.</p> |

Table C.5: **AdaLLaVA-L on LLaVA 1.5-7B model (Example 1)**, generating appropriate responses while adapting to varying computational budgets.

Visual input example:



| | |
|-----------------|--|
| User | Describe this image. |
| LLaVA 1.5-7B | The image features a wooden block with a set of wooden letters arranged in a way that spells out the word "press stop." ... |
| AdaLLaVA-L-50% | The image features a wooden board with a series of wooden chicks on it. The wooden chicks are placed in a row, with each one occupying a different position on the board. The arrangement of the chicks is a close-up of the word "press" and the word "stap." |
| AdaLLaVA-L-75% | The image features a wooden word puzzle made up of four wooden letters, each with a number on them. The letters are arranged in a way that reads "Press Stop." The letters are placed on a white background, creating a visually appealing and playful display. The arrangement of the letters and numbers adds a unique and creative touch to the word puzzle. |
| AdaLLaVA-L-100% | The image features a wooden tray with a set of wooden letters arranged on it. The letters spell out the word "Press Stop," with each letter positioned next to the other. The arrangement of the letters creates a visually appealing and organized display. The wooden tray itself is placed on a white surface, which further emphasizes the contrast between the wooden letters and the background. |

Table C.6: **AdaLLaVA-L on LLaVA 1.5-7B model (Example 2)**, continued from Table C.5.

Visual input example:



| | |
|-----------------|--|
| User | Describe this image. |
| LLaVA 1.5-7B | The image features a woman walking down a street, holding a bouquet of flowers in her hand... |
| AdaLLaVA-L-50% | <p>The image captures a young woman standing in a busy street, holding a bouquet of flowers. She appears to be smiling, possibly indicating that she is enjoying the moment. The street is filled with various vehicles, including several motorcycles and a motorcycle parked close to the woman.</p> <p>There are numerous people in the scene, some standing closer to the woman, while others are further away. Some of the people are closer to the left side of the image, while others are further back, creating a sense of a busy city street.</p> |
| AdaLLaVA-L-75% | <p>The image depicts a woman walking down a street, holding a bouquet of flowers in her hand. She is wearing a dress and appears to be enjoying her time outdoors. The street is lined with various vehicles, including motorcycles and a truck, parked along the side.</p> <p>There are several people in the scene, some of whom are walking or standing near the vehicles. A handbag can be seen resting on the ground, possibly belonging to one of the individuals. The overall atmosphere of the scene is lively and bustling, with people going about their daily activities.</p> |
| AdaLLaVA-L-100% | <p>The image depicts a woman walking down a street, holding a bouquet of flowers in her hand. She is wearing a flowered dress and appears to be enjoying her time. The street is lined with various vehicles, including several motorcycles parked on both sides of the road.</p> <p>There are also a few people walking around, some of whom are carrying handbags. The scene captures a lively atmosphere with people going about their daily activities.</p> |

Table C.7: **AdaLLaVA-L on LLaVA 1.5-7B model (Example 3)**, continued from Table C.6.



Question: Describe the scene?

Answer: The scene depicts a man walking down a sidewalk with a young boy on a bicycle. The man is holding the boy's hand, ensuring his safety as they walk together. There are several other people in the background, possibly going about their daily activities. The setting appears to be a city street, with a mix of pedestrians and urban elements.

Question: What are they doing?

Answer: The man and the child are walking together on a sidewalk, with the man pushing a bike with a child on it.



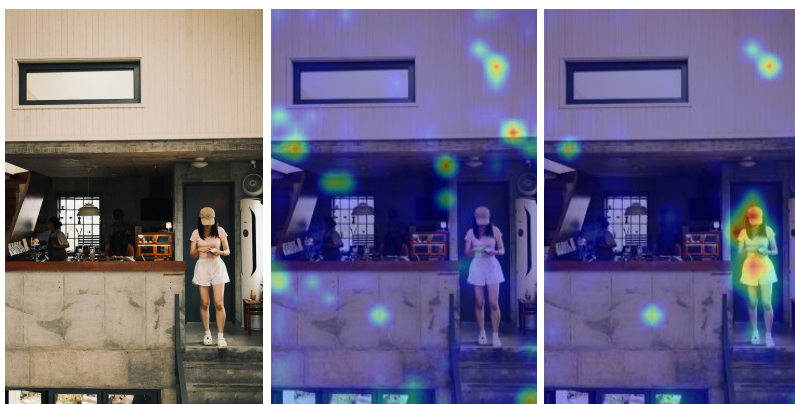
Question: What is the name of the place?

Answer: The name of the place is Happy Plaza.

Question: Are there any special promotion?

Answer: Yes, there is a special promotion at the Happy Plaza shopping center. The store is advertising a buy one, get one free deal, which is likely to attract customers and encourage them to shop at the location.

Figure C.6: The key-query attention scores between latency token and visual tokens. The latency input is 1.0 in these examples.



Question: Where is this place?

What is the woman wearing?

Answer: This place is a restaurant, as evidenced by the presence of a dining table, chairs, and a counter. The woman is standing on a staircase, possibly near the entrance or exit of the restaurant.

The woman is wearing a pink shirt and white shorts.

Figure C.7: The key-query attention scores between latency token and visual tokens. The latency input is 1.0 in these examples.

REFERENCES

Akyürek, 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*.

Alajrami, Ahmed, Katerina Margatina, and Nikolaos Aletras. 2023. Understanding the role of input token characters in language models: How does information loss affect performance? In *The 2023 conference on empirical methods in natural language processing*.

Alayrac, Jean-Baptiste, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems* 35:23716–23736.

An, Shengnan, Zeqi Lin, Bei Chen, Qiang Fu, Nanning Zheng, and Jian-Guang Lou. 2023a. Does deep learning learn to abstract? a systematic probing framework. In *The eleventh international conference on learning representations*.

An, Shengnan, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Jian-Guang Lou, and Dongmei Zhang. 2023b. How do in-context examples affect compositional generalization? *arXiv preprint arXiv:2305.04835*.

Anthropic. 2024. The claude 3 model family: Opus, sonnet, haiku. https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf.

———. 2025. The system card: Claude opus 4 & claude sonnet 4. <https://www-cdn.anthropic.com/07b2a3f9902ee19fe39a36ca638e5ae987bc64dd.pdf>.

Arora, Sanjeev, and Anirudh Goyal. 2023. A theory for emergence of complex skills in language models. *arXiv preprint arXiv:2307.15936*.

Arora, Sanjeev, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. 2019. A theoretical analysis of contrastive unsupervised representation learning. In *36th international conference on machine learning, icml 2019*. International Machine Learning Society (IMLS).

Bai, Jinze, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*.

Bengio, Emmanuel, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup. 2015. Conditional computation in neural networks for faster models. *arXiv preprint arXiv:1511.06297*.

Berglund, Lukas, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2023. The reversal curse: Llms trained on "a is b" fail to learn "b is a". *arXiv preprint arXiv:2309.12288*.

Black, Sid, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow. Tech. Rep., Zenodo. If you use this software, please cite it using these metadata.

Blei, David M, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning research*.

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*.

Bowman, Samuel R., Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. Association for Computational Linguistics.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in neural information processing systems*, ed. H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, vol. 33, 1877–1901. Curran Associates, Inc.

Cai, Ruisi, Saurav Muralidharan, Greg Heinrich, Hongxu Yin, Zhangyang Wang, Jan Kautz, and Pavlo Molchanov. 2024. Flextron: Many-in-one flexible large language model. In *Forty-first international conference on machine learning*.

Cao, Jianjian, Peng Ye, Shengze Li, Chong Yu, Yansong Tang, Jiwen Lu, and Tao Chen. 2024. Madtp: Multimodal alignment-guided dynamic token pruning for accelerating vision-language transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 15710–15719.

Chen, Keqin, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. 2023. Shikra: Unleashing multimodal llm’s referential dialogue magic. *arXiv preprint arXiv:2306.15195*.

Chen, Liang, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. 2024. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European conference on computer vision*, 19–35. Springer.

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR.

- Chen, Xinlei, Saining Xie, and Kaiming He. 2021a. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.
- Chen, Yanda, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2022. Meta-learning via language model in-context tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, ed. Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, 719–730. Dublin, Ireland: Association for Computational Linguistics.
- Chen, Yinbo, Zhuang Liu, Huijuan Xu, Trevor Darrell, and Xiaolong Wang. 2021b. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.
- Chhikara, Prateek, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*.
- Child, Rewon, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.
- 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*.
- Chung, Hyung Won, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research* 25(70):1–53.
- Dai, Wenliang, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale Fung, and Steven Hoi. 2023. InstructBLIP: Towards general-purpose vision-language models with instruction tuning. In *Thirty-seventh conference on neural information processing systems*.

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.

Dong, Qingxiu, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Baobao Chang, Xu Sun, Lei Li, and Zhifang Sui. 2024. A survey on in-context learning. In *Proceedings of the 2024 conference on empirical methods in natural language processing*, ed. Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, 1107–1128. Miami, Florida, USA: Association for Computational Linguistics.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xi-aohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*.

Du, Nan, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, 5547–5569. PMLR.

Du, Simon Shaolei, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. 2021. Few-shot learning via learning the representation, provably. In *9th international conference on learning representations, ICLR 2021, virtual event, austria, may 3-7, 2021*. OpenReview.net.

Dziri, Nouha, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. 2023. Faith and fate: Limits of transformers on compositionality. In *Thirty-seventh conference on neural information processing systems*.

Edge, Darren, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva N. Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *ArXiv* abs/2404.16130.

Figurnov, Michael, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. 2017. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1039–1048.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*. PMLR.

Fu, Chaoyou, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, et al. 2023. Mme: A comprehensive evaluation benchmark for multimodal large language models. *arXiv preprint arXiv:2306.13394*.

Galanti, Tomer, András György, and Marcus Hutter. 2022. Generalization bounds for transfer learning with pretrained classifiers. *arXiv preprint arXiv:2212.12532*.

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*, ed. Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, 6894–6910. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

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* 35:30583–30598.

- Garg, Siddhant, and Yingyu Liang. 2020. Functional regularization for representation learning: A unified theoretical perspective. *Advances in Neural Information Processing Systems*.
- Ge, Weifeng, and Yizhou Yu. 2017. Borrowing treasures from the wealthy: Deep transfer learning through selective joint fine-tuning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.
- Geng, Xinyang, and Hao Liu. 2023. Openllama: An open reproduction of llama.
- Goyal, Sachin, Ananya Kumar, Sankalp Garg, Zico Kolter, and Aditi Raghunathan. 2023. Finetune like you pretrain: Improved finetuning of zero-shot vision models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.
- Goyal, Yash, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6904–6913.
- Grill, Jean-Bastien, Florian Strub, Florent Alché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent—a new approach to self-supervised learning. *Advances in neural information processing systems*.
- Grubb, Alex, and Drew Bagnell. 2012. Speedboost: Anytime prediction with uniform near-optimality. In *Artificial intelligence and statistics*, 458–466. PMLR.
- Gu, Jiuxiang, Chenyang Li, Yingyu Liang, Zhenmei Shi, Zhao Song, and Tianyi Zhou. 2024a. Fourier circuits in neural networks: Unlocking the potential of large language models in mathematical reasoning and modular arithmetic. *arXiv preprint arXiv:2402.09469*.

- Gu, Jiuxiang, Yingyu Liang, Heshan Liu, Zhenmei Shi, Zhao Song, and Junze Yin. 2024b. Conv-basis: A new paradigm for efficient attention inference and gradient computation in transformers. *arXiv preprint arXiv:2405.05219*.
- Gu, Jiuxiang, Yingyu Liang, Zhenmei Shi, Zhao Song, and Chiwun Yang. 2024c. Toward infinite-long prefix in transformer. *arXiv preprint arXiv:2406.14036*.
- Gu, Jiuxiang, Yingyu Liang, Zhenmei Shi, Zhao Song, and Yufa Zhou. 2024d. Tensor attention training: Provably efficient learning of higher-order transformers. *arXiv preprint arXiv:2405.16411*.
- Gu, Shixiang, Sergey Levine, Ilya Sutskever, and Andriy Mnih. 2015. Muprop: Unbiased backpropagation for stochastic neural networks. *arXiv preprint arXiv:1511.05176*.
- Guo, Zirui, Xubin Ren, Lingrui Xu, Jiahao Zhang, and Chao Huang. 2025a. Rag-anything: All-in-one rag framework. 2510.12323.
- Guo, Zirui, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025b. LightRAG: Simple and fast retrieval-augmented generation. In *Findings of the association for computational linguistics: Emnlp 2025*, ed. Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, 10746–10761. Suzhou, China: Association for Computational Linguistics.
- Gurari, Danna, Qing Li, Abigale J Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P Bigham. 2018. Vizwiz grand challenge: Answering visual questions from blind people. In *Proceedings of the ieee conference on computer vision and pattern recognition*, 3608–3617.
- Han, Yizeng, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. 2021. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44(11):7436–7456.

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*.

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*.

He, Kaiming, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

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*.

Hoffmann, Jordan, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Holtzman, Ari, Peter West, Vered Shwartz, Yejin Choi, and Luke Zettlemoyer. 2021. Surface form competition: Why the highest probability answer isn't always right. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, 7038–7051.

Hongjin, SU, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2023. Selective annotation makes language models better few-shot learners. In *The eleventh international conference on learning representations*.

Hospedales, Timothy, Antreas Antoniou, Paul Micaelli, and Amos Storkey. 2021. Meta-learning in neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

- Hu, Edward J, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022a. LoRA: Low-rank adaptation of large language models. In *International conference on learning representations*.
- Hu, Hanzhang, Debadeepta Dey, Martial Hebert, and J Andrew Bagnell. 2019. Learning anytime predictions in neural networks via adaptive loss balancing. In *Proceedings of the aaai conference on artificial intelligence*, vol. 33, 3812–3821.
- Hu, Jerry Yao-Chieh, Maojiang Su, En-Jui Kuo, Zhao Song, and Han Liu. 2024. Computational limits of low-rank adaptation (lora) for transformer-based models. *arXiv preprint arXiv:2406.03136*.
- Hu, Minqing, and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth acm sigkdd international conference on knowledge discovery and data mining*.
- Hu, Shell Xu, Da Li, Jan Stühmer, Minyoung Kim, and Timothy M Hospedales. 2022b. Pushing the limits of simple pipelines for few-shot learning: External data and fine-tuning make a difference. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*.
- Hu, Zhiqiang, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. LLM-adapters: An adapter family for parameter-efficient fine-tuning 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, 5254–5276. Singapore: Association for Computational Linguistics.
- Huang, Haiyang, Newsha Ardalani, Anna Sun, Liu Ke, Shruti Bhosale, Hsien-Hsin S. Lee, Carole-Jean Wu, and Benjamin Lee. 2024. Toward efficient inference for mixture of experts. In *The thirty-eighth annual conference on neural information processing systems*.
- Huang, Jie, and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

- Huang, Weiran, Mingyang Yi, Xuyang Zhao, and Zihao Jiang. 2023. Towards the generalization of contrastive self-supervised learning. In *The eleventh international conference on learning representations*.
- Hudson, Drew A, and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6700–6709.
- Jang, Eric, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with gumbel-softmax. In *International conference on learning representations*.
- Javaheripi, Mojan, Sébastien Bubeck, Marah Abdin, Jyoti Aneja, Sebastien Bubeck, Caio César Teodoro Mendes, Weizhu Chen, Allie Del Giorno, Ronen Eldan, Sivakanth Gopi, et al. 2023. Phi-2: The surprising power of small language models. *Microsoft Research Blog*.
- Jia, Chao, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. 2021. Scaling up visual and vision-language representation learning with noisy text supervision. In *International conference on machine learning*. PMLR.
- Jie, Zequn, Peng Sun, Xin Li, Jiashi Feng, and Wei Liu. 2019. Anytime recognition with routing convolutional networks. *IEEE transactions on pattern analysis and machine intelligence* 43(6):1875–1886.
- Kaplan, Jared, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.
- Karayev, Sergey, Mario Fritz, and Trevor Darrell. 2014. Anytime recognition of objects and scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 572–579.

Khaki, Samir, and Konstantinos N Plataniotis. 2024. The need for speed: Pruning transformers with one recipe. In *The twelfth international conference on learning representations*.

Kim, Najoung, and Tal Linzen. 2020. COGS: A compositional generalization challenge based on semantic interpretation. In *Proceedings of the 2020 conference on empirical methods in natural language processing (emnlp)*, ed. Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, 9087–9105. Online: Association for Computational Linguistics.

Kumar, Ananya, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. 2022. Fine-tuning can distort pretrained features and underperform out-of-distribution. In *The tenth international conference on learning representations, ICLR 2022, virtual event, april 25-29, 2022*. OpenReview.net.

Lake, Brenden M, Ruslan Salakhutdinov, and Joshua B Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science*.

Latif, Siddique, Moazzam Shoukat, Fahad Shamshad, Muhammad Usama, Yi Ren, Heriberto Cuayáhuitl, Wenwu Wang, Xulong Zhang, Roberto Togneri, Erik Cambria, et al. 2023. Sparks of large audio models: A survey and outlook. *arXiv preprint arXiv:2308.12792*.

Laurençon, Hugo. 2023. Introducing idefics: An open reproduction of state-of-the-art visual language model. <https://huggingface.co/blog/idefics>.

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.

Levy, Itay, Ben Bogin, and Jonathan Berant. 2022. Diverse demonstrations improve in-context compositional generalization. *arXiv preprint arXiv:2212.06800*.

Li, Bo, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. 2024. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Li, Bohao, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. 2023a. Seed-bench: Benchmarking multimodal LLMs with generative comprehension. *arXiv preprint arXiv:2307.16125*.

Li, Hengduo, Zuxuan Wu, Abhinav Shrivastava, and Larry S Davis. 2021. 2d or not 2d? adaptive 3d convolution selection for efficient video recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 6155–6164.

Li, Junnan, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023b. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, 19730–19742. PMLR.

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, Yifan, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2023c. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*.

Liang, Yingyu, Heshan Liu, Zhenmei Shi, Zhao Song, Zhuoyan Xu, Jiale Zhao, and Zhen Zhuang. 2025. Conv-basis: A new paradigm for efficient attention inference and gradient computation in transformers. In *The 2025 conference on empirical methods in natural language processing*.

Lin, Bin, Zhenyu Tang, Yang Ye, Jiayi Cui, Bin Zhu, Peng Jin, Jinfa Huang, Junwu Zhang, Yatian Pang, Munan Ning, et al. 2024. Moe-llava: Mixture of experts for large vision-language models. *arXiv preprint arXiv:2401.15947*.

Liu, Chen, Yanwei Fu, Chengming Xu, Siqian Yang, Jilin Li, Chengjie Wang, and Li Zhang. 2021. Learning a few-shot embedding model with contrastive learning. In *Proceedings of the aaai conference on artificial intelligence*.

Liu, Haotian, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024a. Improved baselines with visual instruction tuning. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, 26296–26306.

Liu, Haotian, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024b. Llava-next: Improved reasoning, ocr, and world knowledge. <https://llava-vl.github.io/blog/2024-01-30-llava-next/>.

Liu, Haotian, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems* 36:34892–34916.

Liu, Jiachang, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2022. What makes good in-context examples for GPT-3? In *Proceedings of deep learning inside out (deelio 2022): The 3rd workshop on knowledge extraction and integration for deep learning architectures*, ed. Eneko Agirre, Marianna Apidianaki, and Ivan Vulić, 100–114. Dublin, Ireland and Online: Association for Computational Linguistics.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Liu, Yuan, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, et al. 2025a. Mmbench: Is your multi-modal model an all-around player? In *European conference on computer vision*, 216–233. Springer.

Liu, Zidong, Zhuoyan Xu, Zhenmei Shi, and Yingyu Liang. 2025b. Can language models compose skills in-context? *arXiv preprint arXiv:2510.22993*.

Logeswaran, Lajanugen, and Honglak Lee. 2018. An efficient framework for learning sentence representations. In *6th international conference on learning representations, ICLR 2018, vancouver, bc, canada, april 30 - may 3, 2018, conference track proceedings*. OpenReview.net.

Lu, Pan, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022a. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in Neural Information Processing Systems* 35:2507–2521.

Lu, Yao, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022b. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: Long papers)*, ed. Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, 8086–8098. Dublin, Ireland: Association for Computational Linguistics.

Luo, Haozheng, Jiahao Yu, Wenxin Zhang, Jialong Li, Jerry Yao-Chieh Hu, Xinyu Xing, and Han Liu. 2024. Decoupled alignment for robust plug-and-play adaptation. *arXiv preprint arXiv:2406.01514*.

Madasu, Avinash, and Shashank Srivastava. 2022. What do large language models learn beyond language? In *Findings of the association for computational linguistics: Emnlp 2022*, ed. Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, 6940–6953. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Maddison, Chris J., Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *International conference on learning representations*.

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*.

Maurer, Andreas. 2016. A vector-contraction inequality for rademacher complexities. In *International conference on algorithmic learning theory*. Springer.

Meng, Lingchen, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. 2022. Adavit: Adaptive vision transformers for efficient image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12309–12318.

Meng, Yue, Chung-Ching Lin, Rameswar Panda, Prasanna Sattigeri, Leonid Karlinsky, Aude Oliva, Kate Saenko, and Rogerio Feris. 2020. Ar-net: Adaptive frame resolution for efficient action recognition. In *Computer vision—eccv 2020: 16th european conference, glasgow, uk, august 23–28, 2020, proceedings, part vii* 16, 86–104. Springer.

Meta. 2024. Introducing meta llama 3: The most capable openly available llm to date. <https://ai.meta.com/blog/meta-llama-3/>.

Min, Sewon, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022a. MetaICL: Learning to learn in context. In *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies*, ed. Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz, 2791–2809. Seattle, United States: Association for Computational Linguistics.

Min, Sewon, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022b. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 conference on empirical methods in natural language processing*. Association for Computational Linguistics.

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*.

Mohri, Mehryar, Afshin Rostamizadeh, and Ameet Talwalkar. 2018. *Foundations of machine learning*. MIT press.

Muennighoff, Niklas, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M. Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. In *Proceedings of the 61st annual meeting of the association for computational linguistics (acl)*.

Murty, Shikhar, Tatsunori B Hashimoto, and Christopher D Manning. 2021. DrecA: A general task augmentation strategy for few-shot natural language inference. In *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies*.

Ni, Jianmo, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the association for computational linguistics: Acl 2022*.

Olshausen, B., and D. Field. 1997. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*.

Oord, Aaron van den, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

OpenAI. 2022a. Gpt-4v(ision) system card. https://cdn.openai.com/papers/GPTV_System_Card.pdf. Accessed: 2023-09-25.

———. 2022b. Introducing ChatGPT. <https://openai.com/blog/chatgpt>. Accessed: 2022-11-30.

———. 2023. GPT-4 technical report. *arXiv preprint arxiv:2303.08774*.

———. 2025. Introducing deep research. <https://openai.com/index/introducing-deep-research/>. Accessed: 2025-2-2.

Oquab, Maxime, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. 2024. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*. Featured Certification.

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*.

Packer, Charles, Vivian Fang, Shishir G. Patil, Kevin Lin, Sarah Wooders, and Joseph Gonzalez. 2023. Memgpt: Towards llms as operating systems. *ArXiv abs/2310.08560*.

Pan, Bowen, Rameswar Panda, Yifan Jiang, Zhangyang Wang, Rogerio Feris, and Aude Oliva. 2021. Ia-red2: Interpretability-aware redundancy reduction for vision transformers. *Advances in Neural Information Processing Systems* 34:24898–24911.

Pang, Bo, and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd annual meeting of the association for computational linguistics*.

———. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACL'05)*, ed. Kevin Knight, Hwee Tou Ng, and Kemal Oflazer, 115–124. Ann Arbor, Michigan: Association for Computational Linguistics.

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 international conference on computer vision*.

Perez, Ethan, Douwe Kiela, and Kyunghyun Cho. 2021. True few-shot learning with language models. *Advances in neural information processing systems*.

Pham, Phu, Wentian Zhao, Kun Wan, Yu-Jhe Li, Zeliang Zhang, Daniel Miranda, Ajinkya Kale, and Chenliang Xu. 2024. Quadratic is not what you need for multi-modal large language models. *arXiv preprint arXiv:2410.06169*.

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*.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*. PMLR.

Radford, Alec, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. *OpenAI blog*.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1(8):9.

Raghu, Aniruddh, Maithra Raghu, Samy Bengio, and Oriol Vinyals. 2020. Rapid learning or feature reuse? towards understanding the effectiveness of maml. In *International conference on learning representations*.

Rao, Yongming, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. 2021. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *Advances in neural information processing systems*, ed. A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan.

Raposo, David, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam Santoro. 2024. Mixture-of-depths: Dynamically allocating compute in transformer-based language models. *arXiv preprint arXiv:2404.02258*.

Rasmussen, Preston, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*.

Ren, Mengye, Sachin Ravi, Eleni Triantafillou, Jake Snell, Kevin Swersky, Josh B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. 2018. Meta-learning for semi-supervised few-shot classification. In *International conference on learning representations*.

Roberts, Nicholas, Xintong Li, Dyah Adila, Sonia Crompt, Tzu-Heng Huang, Jitian Zhao, and Frederic Sala. 2023. Geometry-aware adaptation for pretrained models. *arXiv preprint arXiv:2307.12226*.

Rotem, Daniel, Michael Hassid, Jonathan Mamou, and Roy Schwartz. 2023. Finding the sweet spot: Analysis and improvement of adaptive inference in low resource settings. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, 14836–14851.

Rubin, Ohad, Jonathan Herzig, and Jonathan Berant. 2022. Learning to retrieve prompts for in-context learning. In *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 2655–2671.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International journal of computer vision*.

Sanh, Victor, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *International conference on learning representations*.

Shang, Yuzhang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. 2024. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. *arXiv preprint arXiv:2403.15388*.

Shi, Zhenmei, Jiefeng Chen, Kunyang Li, Jayaram Raghuram, Xi Wu, Yingyu Liang, and Somesh Jha. 2023a. The trade-off between universality and label efficiency of representations from contrastive learning. In *International conference on learning representations*.

Shi, Zhenmei, Yifei Ming, Ying Fan, Frederic Sala, and Yingyu Liang. 2023b. Domain generalization via nuclear norm regularization. In *Conference on parsimony and learning (proceedings track)*.

———. 2024a. Domain generalization via nuclear norm regularization. In *Conference on parsimony and learning*, 179–201. PMLR.

Shi, Zhenmei, Junyi Wei, and Yingyu Liang. 2022. A theoretical analysis on feature learning in neural networks: Emergence from inputs and advantage over fixed features. In *International conference on learning representations*.

———. 2023c. Provable guarantees for neural networks via gradient feature learning. In *Thirty-seventh conference on neural information processing systems*.

Shi, Zhenmei, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. 2023d. Why larger language models do in-context learning differently? In *R0-fomo: robustness of few-shot and zero-shot learning in large foundation models*.

———. 2024b. Why larger language models do in-context learning differently? In *Forty-first international conference on machine learning*.

Shu, Fangxun, Yue Liao, Le Zhuo, Chenning Xu, Guanghao Zhang, Haonan Shi, Long Chen, Tao Zhong, Wanggui He, Siming Fu, et al. 2024. Llava-mod: Making llava tiny via moe knowledge distillation. *arXiv preprint arXiv:2408.15881*.

Singh, Amanpreet, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards vqa models that can

read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8317–8326.

Snell, Jake, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. *Advances in neural information processing systems*.

Socher, Richard, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*.

Song, Haoyu, Li Dong, Weinan Zhang, Ting Liu, and Furu Wei. 2022. Clip models are few-shot learners: Empirical studies on vqa and visual entailment. In *Proceedings of the 60th annual meeting of the association for computational linguistics*.

Song, Jiajun, Zhuoyan Xu, and Yiqiao Zhong. 2025. Out-of-distribution generalization via composition: a lens through induction heads in transformers. *Proceedings of the National Academy of Sciences* 122(6):e2417182122.

Song, Yixin, Zeyu Mi, Haotong Xie, and Haibo Chen. 2024. Powerinfer: Fast large language model serving with a consumer-grade gpu. In *Proceedings of the ACM SIGOPS 30th symposium on operating systems principles*, 590–606.

Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1):1929–1958.

Sun, Yiyou, Zhenmei Shi, and Yixuan Li. 2023a. A graph-theoretic framework for understanding open-world semi-supervised learning. In *Thirty-seventh conference on neural information processing systems*.

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.

Team, Tongyi DeepResearch, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, et al. 2025. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*.

Tian, Yonglong, Dilip Krishnan, and Phillip Isola. 2020a. Contrastive multiview coding. In *European conference on computer vision*. Springer.

Tian, Yonglong, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. 2020b. Rethinking few-shot image classification: a good embedding is all you need? In *European conference on computer vision*. Springer.

Tosh, Christopher, Akshay Krishnamurthy, and Daniel Hsu. 2021. Contrastive learning, multi-view redundancy, and linear models. In *Algorithmic learning theory*. 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*.

Triantafillou, Eleni, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. 2020. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International conference on learning representations*.

Tripuraneni, Nilesch, Chi Jin, and Michael Jordan. 2021. Provable meta-learning of linear representations. In *International conference on machine learning*. PMLR.

Tripuraneni, Nilesh, Michael Jordan, and Chi Jin. 2020. On the theory of transfer learning: The importance of task diversity. *Advances in Neural Information Processing Systems*.

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.

Vinje, William E, and Jack L Gallant. 2000. Sparse coding and decorrelation in primary visual cortex during natural vision. *Science*.

Vinyals, Oriol, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. *Advances in neural information processing systems*.

Von Oswald, Johannes, Eyvind Niklasson, Ettore Randazzo, Joao Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *Proceedings of the 40th international conference on machine learning*, ed. Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, vol. 202 of *Proceedings of Machine Learning Research*, 35151–35174. PMLR.

Voorhees, Ellen M, and Dawn M Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd annual international acm sigir conference on research and development in information retrieval*.

Vu, Tu, Minh-Thang Luong, Quoc Le, Grady Simon, and Mohit Iyyer. 2021. Strata: Self-training with task augmentation for better few-shot learning. In *Proceedings of the 2021 conference on empirical methods in natural language processing*.

Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018a. 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, and Neel Joshi. 2024a. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. *arXiv preprint arXiv:2406.14852*.

Wang, Peng, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Wang, Tongzhou, and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*. PMLR.

Wang, Xin, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. 2018b. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the european conference on computer vision (eccv)*, 409–424.

Wang, Xinyi, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. 2023a. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. In *Thirty-seventh conference on neural information processing systems*.

Wang, Yaqing, Quanming Yao, James T Kwok, and Lionel M Ni. 2020. Generalizing from a few examples: A survey on few-shot learning. *ACM computing surveys*.

Wang, Yifei, Qi Zhang, Yisen Wang, Jiansheng Yang, and Zhouchen Lin. 2022. Chaos is a ladder: A new theoretical understanding of contrastive learning via augmentation overlap. In *International conference on learning representations*.

Wang, Yulin, Rui Huang, Shiji Song, Zeyi Huang, and Gao Huang. 2021. Not all images are worth 16x16 words: Dynamic transformers for efficient image recognition. In *Advances in neural information processing systems*, ed. A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan.

Wang, Zhen, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. 2023b. Multitask prompt tuning enables parameter-efficient transfer learning. In *The eleventh international conference on learning representations*.

Waswani, A, N Shazeer, N Parmar, J Uszkoreit, L Jones, A Gomez, L Kaiser, and I Polosukhin. 2017. Attention is all you need. In *Nips*.

Wei, Colin, Kendrick Shen, Yining Chen, and Tengyu Ma. 2021. Theoretical analysis of self-training with deep networks on unlabeled data. In *International conference on learning representations*.

Wei, Haoran, Lingyu Kong, Jinyue Chen, Liang Zhao, Zheng Ge, En Yu, Jianjian Sun, Chunrui Han, and Xiangyu Zhang. 2024. Small language model meets with reinforced vision vocabulary. *arXiv preprint arXiv:2401.12503*.

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. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. 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, and Quoc V Le. 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*. PMLR.

Wiebe, Janyce, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*.

Wortsman, Mitchell, Gabriel Ilharco, Jong Wook Kim, Mike Li, Simon Kornblith, Rebecca Roelofs, Raphael Gontijo Lopes, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, et al. 2022. Robust fine-tuning of zero-shot models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Wu, Zuxuan, Tushar Nagarajan, Abhishek Kumar, Steven Rennie, Larry S Davis, Kristen Grauman, and Rogerio Feris. 2018. Blockdrop: Dynamic inference paths in residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8817–8826.

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*.

Xie, Sang Michael, Shibani Santurkar, Tengyu Ma, and Percy Liang. 2023. Data selection for language models via importance resampling. *arXiv preprint arXiv:2302.03169*.

Xu, Zhixiang, Kilian Q Weinberger, and Olivier Chapelle. 2012. The greedy miser: learning under test-time budgets. In *Proceedings of the 29th international conference on machine learning*, 1299–1306.

Xu, Zhuoyan, Khoi Duc Nguyen, Preeti Mukherjee, Saurabh Bagchi, Somali Chatterji, Yingyu Liang, and Yin Li. 2025. Learning to inference adaptively for multimodal large language models. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, 3552–3563.

Xu, Zhuoyan, Khoi Duc Nguyen, Preeti Mukherjee, Somali Chaterji, Yingyu Liang, and Yin Li. 2024a. Adainf: Adaptive inference for resource-constrained foundation models. In *Workshop on efficient systems for foundation models ii @ icml2024*.

Xu, Zhuoyan, Zhenmei Shi, and Yingyu Liang. 2024b. 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. 2024c. Towards few-shot adaptation of foundation models via multitask finetuning. In *The twelfth international conference on learning representations*.

Yang, Zhanyuan, Jinghua Wang, and Yingying Zhu. 2022. Few-shot classification with contrastive learning. In *European conference on computer vision*. Springer.

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, Yuan, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. 2024. Minicpm-v: A GPT-4V level MLLM on your phone. *arXiv preprint arXiv:2408.01800*.

Ye, Jiacheng, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023. Compositional exemplars for in-context learning. In *Proceedings of the 40th international conference on machine learning*, ed. Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, vol. 202 of *Proceedings of Machine Learning Research*, 39818–39833. PMLR.

Yu, Weihao, Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Zicheng Liu, Xinchao Wang, and Lijuan Wang. 2023. Mm-vet: Evaluating large multimodal models for integrated capabilities. *arXiv preprint arXiv:2308.02490*.

Yuan, Zhengqing, Zhaoxu Li, Weiran Huang, Yanfang Ye, and Lichao Sun. 2023. Tinygpt-v: Efficient multimodal large language model via small backbones. *arXiv preprint arXiv:2312.16862*.

Yuan, Zhihang, Yuzhang Shang, Yang Zhou, Zhen Dong, Zhe Zhou, Chenhao Xue, Bingzhe Wu, Zhikai Li, Qingyi Gu, Yong Jae Lee, et al. 2024. Llm inference unveiled: Survey and roofline model insights. *arXiv preprint arXiv:2402.16363*.

Zeng, Yuchen, and Kangwook Lee. 2024. The expressive power of low-rank adaptation. In *The twelfth international conference on learning representations*.

Zhang, Renrui, Jiaming Han, Aojun Zhou, Xiangfei Hu, Shilin Yan, Pan Lu, Hongsheng Li, Peng Gao, and Yu Qiao. 2023a. 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. 2023b. Trained transformers learn linear models in-context. *arXiv preprint arXiv:2306.09927*.

Zhang, Tianyi, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020. Revisiting few-sample BERT fine-tuning. In *International conference on learning representations*.

Zhao, Yulai, Jianshu Chen, and Simon Du. 2023. Blessing of class diversity in pre-training. In *International conference on artificial intelligence and statistics*. PMLR.

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*, 12697–12706. PMLR.

Zhong, Ruiqi, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections.

In *Findings of the association for computational linguistics: Emnlp 2021*. Association for Computational Linguistics.

Zhou, Baichuan, Ying Hu, Xi Weng, Junlong Jia, Jie Luo, Xien Liu, Ji Wu, and Lei Huang. 2024. Tinyllava: A framework of small-scale large multimodal models. *arXiv preprint arXiv:2402.14289*.

Zhou, Kaiyang, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022a. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

———. 2022b. Learning to prompt for vision-language models. *International Journal of Computer Vision*.

Zhu, Deyao, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2024a. MiniGPT-4: Enhancing vision-language understanding with advanced large language models. In *The twelfth international conference on learning representations*.

Zhu, Minjie, Yichen Zhu, Xin Liu, Ning Liu, Zhiyuan Xu, Chaomin Shen, Yaxin Peng, Zhicai Ou, Feifei Feng, and Jian Tang. 2024b. A comprehensive overhaul of multimodal assistant with small language models. *arXiv e-prints arXiv-2403*.

Zhu, Yichen, Minjie Zhu, Ning Liu, Zhicai Ou, Xiaofeng Mou, and Jian Tang. 2024c. Llava- ϕ : Efficient multi-modal assistant with small language model. *arXiv preprint arXiv:2401.02330*.

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*. PMLR.