Few-Shot Learning: Contributions to Deep Learning With Limited Data

by

Zhongjie Yu

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

Doctor of Philosophy

(Statistics)

at the

UNIVERSITY OF WISCONSIN–MADISON

2021

Date of final oral examination: 04/22/2021

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

Sebastian Raschka, Assistant Professor, Statistics
Yazhen Wang, Professor, Statistics
Karl Rohe, Associate Professor, Statistics
Anthony Gitter, Assistant Professor, Biostatistics & Medical Informatics
Diego Hernando, Assistant Professor, Radiology

ACKNOWLEDGMENTS

First of all, I would like to express my sincere appreciation to my advisor Professor Sebastian Raschka, for the continuous support of my Ph.D. studies. He always encouraged me on both the academic and mentality perspective and gave the opportunity to pursue my research interests. His patient and valuable guidance including coding and methodology opened the path for my research and motivated me a lot throughout my Ph.D. life.

Besides, I would like to greatly thank my committee members Professor Yazhen Wang, Professor Karl Rohe, Professor Anthony Gitter, and Professor Diego Hernando for their time, interest, and insightful discussions about different aspects of my thesis.

Furthermore, I want to give special thanks to Lin Chen. He gave me many opportunities to develop skills both in academia and industry. I am deeply influenced by his enthusiasm and insight into many interesting projects. The collaboration with him was an invaluable life experience.

My thanks also go to the Department of Statistics at University of Wisconsin-Madison for providing high-quality education and a diversified environment. There is always something to learn from every member in the department.

Last but not least, I am grateful to my family for their care and encouragement from the time when I set foot on the foreign land five years ago. Ph.D life is not easy and I cannot imagine this accomplishment would be possible without their support.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Humans are capable of learning new concepts from small numbers of examples. In contrast, deep learning models usually lack the ability to extract reliable predictive rules from limited data scenarios when attempting to classify new examples. The successful application of deep learning to many visual recognition tasks relies heavily on the availability of a large amount of labeled data which is usually expensive to obtain. This challenging scenario is commonly known as few-shot learning. Few-shot learning has garnered increased attention in recent years due to its significance for many real-world problems. Most of research on few-shot learning is focused on image classification. This dissertation describes four research work we made: 1) the Looking-back method, which utilizes lower-level information to construct additional graphs for label propagation and improve the meta-learner performance in few-shot learning; 2) TransMatch, a new transfer-learning framework for semi-supervised few-shot learning to fully utilize the auxiliary information from labeled base-class data and unlabeled novel-class data; 3) TOL and DCP, simple post-hoc methods achieve state-of-the-art performance in both supervised and semi-supervised few-shot learning; 4) a new proposed problem: few-shot video object detection and novel analysis towards this problem. Our extensive experiments on several popular few-shot learning benchmark datasets and our designed few-shot video object detection datasets demonstrate the efficacy of these contributions.

1 OVERALL INTRODUCTION

Deep learning (DL) is already ubiquitous in our daily lives, including image-based object detection (Liu et al., 2020b), face recognition (Wani et al., 2020), and medical imaging and healthcare (Wang et al., 2020a). While DL is outperforming traditional machine learning methods in these aforementioned application areas (Raschka et al., 2020), a major downside of DL is that it requires large amounts of data to achieve good performance (Goodfellow et al., 2016). The straightforward adoption of deep learning methods with a limited amount of labeled data usually leads to the overfitting issue. However, obtaining a large amount of labeled data is usually very expensive and time-consuming to collect.

It is well-known that humans have the ability to learn from a single or very few of labeled samples. This motivates recent research efforts on learning a novel concept from a single or few examples, hence the name *few-shot learning* (FSL). FSL is a subfield of DL that focuses on training DL models under scarce data regimes, thereby opening possibilities for applying DL to new problem areas where the amount of labeled data is limited.

In FSL settings, datasets are comprised of large numbers of categories (i.e., class labels), but only a few examples per class are available. Figure 1.1 gives an example of a bird classification task. The main objective of FSL is the design of methods that achieve good generalization performance from

the limited number of examples per category. The overarching concept of FSL is very general and applies to different data modalities and tasks like image classification (Wang et al., 2020c), object detection (Kang et al., 2019), and text classification (Bao et al., 2019). However, most FSL research is focused on image classification so that we will use the terms *examples* and *images* (in a supervised learning context) interchangeably.

Most FSL methods use an episodic training strategy known as meta-learning (Vinyals et al., 2016), where a meta-learner is trained on (classification) tasks with the goal to learn to perform well on new, unseen tasks. Many of the most recent FSL methods are based on episodic meta-learning, such as prototypical networks (Snell et al., 2017), relation networks (Sung et al., 2018), model agnostic meta-learning (Finn et al., 2017), and LSTM-based meta-learning (Ravi and Larochelle, 2017). Another successful approach to FSL is the use of transfer learning, where models are trained on large datasets and then appropriately transferred to smaller datasets that contain the novel target classes; examples include weight imprinting (Qi et al., 2018), dynamic few-shot object recognition with attention modules (Gidaris and Komodakis, 2018), and few-shot image classification by predicting parameters from activation values (Qiao et al., 2018).

This dissertation proposes several new approaches to improve FSL systems. Starting with classical meta-learning methods for FSL, we first address the question whether obtaining additional lower-level information embedded in the feature extracting network can improve few-shot image

classification (Yu and Raschka, 2020).

In addition to leveraging lower-level information, using abundant information from unlabeled images (also known as semi-supervised FSL) provides another angle for improving FSL. In particular, we focus on a transfer-learning framework to effectively utilize unlabeled data (Yu et al., 2020).

Most FSL methods rely on complicated setups that are not easy to extend and apply in practice. To make FSL more accessible and widely applicable, we study how to achieve state-of-the-art performance in both supervised FSL and semi-supervised FSL by developing simple and convenient post-hoc methods.

At last, FSL is a field that is largely focused on image classification as the standard setting. We are interested to extend FSL to other fields and investigate a new problem: few-shot learning for video object detection (Yu et al., 2021).

This thesis can be summarized as follows:

- Chapter 1 briefly summarizes the motivation of few-shot learning and reviews the progress this new deep learning paradigm made in the past few years.

- Chapter 2 presents our research efforts to address how to use additional lower-level information: *Looking back to lower-level information in few-shot learning.*

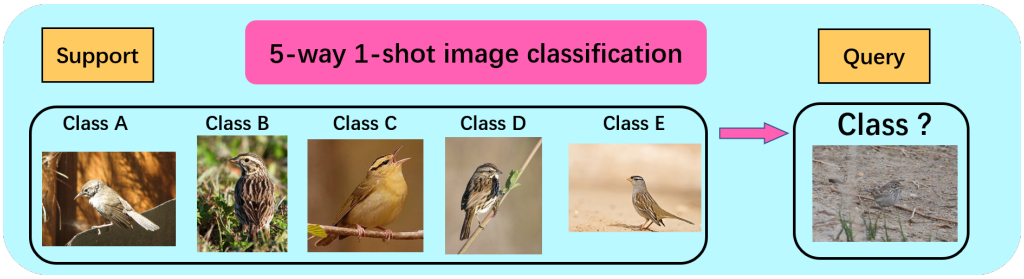

Figure 1.1: An example of 5-way 1-shot bird image classification task from CUB-200-2011 dataset. Given one support image for each bird class, the goal is to identify the correct bird class for the query image.

- Chapter 3 describes our contributions towards incorporating unlabeled data: *TransMatch: a transfer-learning scheme for semi-supervised few-shot learning*.

- Chapter 4 illustrates that our simple methods can achieve state-of-the-art performance in both supervised FSL and semi-supervised FSL: *Simple post-hoc work can improve supervised and semi-supervised few-shot learning*

- Chapter 5 shows our development of few-shot learning for video object detection, which is motivated by the rapid growth of online video content: *Few-shot learning for video object detection in a transfer-learning scheme*.

- Chapter 6 summarizes this dissertation and points out further future directions for few-shot learning research.

2 LOOKING BACK TO LOWER-LEVEL INFORMATION IN FEW-SHOT LEARNING

2.1 Introduction

Apart from recent developments in FSL, many researchers have recently proposed methods for implementing graph neural networks (GNNs) to extend deep learning approaches for graph-structured data. In this context, graphs are used as data structures for modeling the relationships (edges) between data instances (nodes), which was first proposed via the graph neural network model (Scarselli et al., 2009) and extended via graph convolutional networks (Duvenaud et al., 2015), semi-supervised graph convolutional networks (Kipf and Welling, 2017), graph attention networks (Velickovic et al., 2018), and message passing neural networks (Gilmer et al., 2017). Since FSL methods are centered around modeling relationships between the examples in the support and query datasets, GNNs have also gained a growing interest in FSL research, including approaches aggregating node information from densely connected support and query image graphs (Garcia and Bruna, 2017), transductive inference (Liu et al., 2019), and edge-labeling (Kim et al., 2019). GNNs can be computationally prohibitive on large datasets. However, we shall note that one of the significant characteristics of FSL is that datasets for meta-training and meta-testing contain only "few" examples per class, such that the computational cost of

graph construction becomes small in FSL.

Previous research has shown that FSL can be improved by incorporating additional information. For instance, unlabeled data, which is used in conventional (Ren et al., 2018) self-trained (Li et al., 2019b) and transfer learning-based (Yu et al., 2020) semi-supervised FSL, could improve the predictive performance of FSL models. Also, FSL benefits from the inclusion of additional modalities (e.g., textual information describing the images to be classified), which was demonstrated via an adaptive cross-model approach enhancing metric-based FSL (Xing et al., 2019) as well as cross-modal FSL utilizing latent features from aligned autoencoders (Schonfeld et al., 2019). While the aforementioned works showed that additional *external* information benefits FSL, we raise the question of whether additional *internal* information can be useful as well.

While the incorporation of additional information can be beneficial, the utilization of additional *internal* information is not very common in FSL research, and only two recent research papers explored this approach, i.e., Li et al.'s deep nearest neighbor neural network (Li et al., 2019a) and the dense classification network by Lifchitz et al. (2019). In these works, the researchers expanded the feature embeddings (the low-dimensional representation) of the data inputs (i.e., images), extracted from the last layer in the neural network, to higher-dimensional embeddings. These higher-dimensional embeddings were split into several smaller vectors, such that multiple embedding vectors correspond to the same image. In the DN4

model proposed Li et al. (2019a), the last layer's feature embeddings were expanded to form many local descriptors. The dense classification network by Lifchitz et al. (2019) expanded the feature embeddings to three separate vectors that are used for computing the cross-entropy loss during training.

When it comes to utilizing additional internal information, both DN4 (Li et al., 2019a) and the dense classification network (Lifchitz et al., 2019) only considered the last layer's information. In contrast to existing work on FSL, we consider additional information that is hidden in the earlier layers of the neural network. More specifically, the extra information hidden in the network considered in this work is comprised of the feature embeddings that can be obtained from layers before the last layer. We propose using a graph structure to integrate this lower-level information into the neural network, since graph structures are well-suited for modeling relationships in data.

We refer to the FSL method proposed in this work as *Looking-Back*, because unlike DN4 (Li et al., 2019a) and the dense classification network (Lifchitz et al., 2019), this method fully utilizes previous layers' feature embeddings (i.e., lower-level information) rather than focusing on the final layer's feature embeddings alone. During training, the lower-level information is expected to help the meta-learner to absorb more information overall. Although this lower-level information may not be as useful as the embedding vectors obtained from the last layer, we believe it has a positive impact on the meta-learner. To show this, we adopt the

widely used Conv-64F (Li et al., 2019a) in few-shot learning as a backbone, and construct graphs for label propagation, following the Transductive Propagation Network (TPN) (Liu et al., 2019), to capture lower-level information.

Besides the feature embeddings of the last layer, the previous layers' feature embeddings are also used for computing the pair-wise similarities between the inputs, based on relational network modules (Liu et al., 2019). In the Looking-Back method, three groups of pair-wise similarity measures are computed. The similarity scores between all support and query images in one episode amount to three separate graph Laplacians, which are used for iterative label propagation, to generate three separate cross-entropy losses. As the experimental results indicate, the losses from lower-level features are used during meta-training to enhance the performance of the meta-learner. After meta-training, we adopt the last layer's feature embeddings for testing on new tasks (i.e., images with class labels that are not seen during training) in a transductive fashion. As the experimental results reveal, the resulting FSL models have a better predictive performance on new, unseen tasks compared to models generated by meta-learners that don't utilize lower-level information.

The contributions of this work can be summarized as follows:

1. We propose a novel meta-learning method, Looking-Back, that utilizes lower-level information from hidden layers, different from existing methods that only use feature embedding of the last layer during

meta-training.

2. We employ a graph neural network for our Looking-back, which fully utilizes the advantage of graph structures for few-shot learning to absorb the lower-level information in the hidden layers of the neural network.

3. We evaluate our proposed Looking-Back method on two popular FSL datasets, *mini*ImageNet and *tiered*ImageNet, and achieve new state-of-the-art results, providing supporting evidence that using lower-level information could result in better meta-learners in FSL tasks.

2.2 Related Work

In this section, we discuss the recent developments in FSL with a focus on methods related to our work. FSL, based on meta-learning, typically uses episodic training strategies. In each episode, the meta-learner is trained on a meta-task, which can be thought of as an image classification task. During training, these tasks are drawn randomly from the training dataset across the episodes. During the model evaluation, tasks are chosen from a separate test dataset, which consists of images from novel classes that are not contained in the training dataset.

In N-way-k-shot FSL, when a meta-learner is trained on several tasks

sampled from the training dataset, each training task is subdivided into a support set and a query set. Each task consists of N unique class labels, and the support set consists of k labeled images per class. Utilizing the support set, the model learns to predict the image labels in the query set. After training, the meta-learner is then evaluated on new tasks sampled from the test set. Similar to the training tasks, each new task consists of N unique class labels with k images (in the support set) each. However, to assess how well the meta-learner performs on new tasks, the classes in the test dataset are not overlapping with the classes in the training set.

Based on the general FSL meta-learning framework described above, we can divide meta-learning approaches further into metric-, optimization-, and graph-based meta-learning, which we discuss in the following subsections.

2.2.1 Metric-based Meta-learning

Metric-based methods are primarily focused on learning feature embeddings that enable similarity comparisons between support and query images. The Prototypical Network (Snell et al., 2017) used a Euclidean distance measure to compare the feature embeddings of the query images with centroids of the support images in different classes. The Relation Network (Sung et al., 2018) constructed an additional network to compute the similarity score between images directly, instead of using the Euclidean distance measure on the images' feature embeddings similar to

the Prototypical Network. DN4 (Li et al., 2019a) used a cosine similarity measure on multiple local descriptors, obtained by expanding the feature embeddings of the last layer to higher dimensions, to find the most similar images via nearest neighbor search.

2.2.2 Optimization-based Meta-learning

Optimization-based methods are focused on parameter optimization and how to rapidly learn knowledge from limited training images that can be adapted to novel images. The model agnostic meta-learning framework (MAML) (Finn et al., 2017) learned a general model that can be efficiently fine-tuned to perform well on other tasks using conventional gradient descent-based optimization. While MAML used second-order partial derivatives to train the general model before task-specific fine-tuning, Reptile (Nichol et al., 2018) was a first-order approximation of MAML that simplified the training procedure and boosted computational performance. Ravi and Larochelle (Ravi and Larochelle, 2017) introduced a related yet different approach to optimization-based meta-learning. They proposed the use of an LSTM to model the sequence corresponding to the sequential optimization of the model parameters across different tasks.

2.2.3 Graph-based Meta-learning

Graph-based meta-learning uses graph structures to model the relationship between query and support images based on relative similarity measures, where each labeled and unlabeled image represents a node in the graph. There are very few treatments of graph-based methods for FSL in the literature; however, the topic has recently gained more attention in the FSL research community.

In 2017, Garcia and Bruna (2017) proposed the use of a GNN for aggregating node information in an iterative fashion via a message-passing model, where the support and query images are densely connected in the graph. The edge-labeling graph neural network (EGNN) modified this approach, using edge- rather than node-label information, combined with inter-cluster dissimilarity and intra-cluster similarity measures (Kim et al., 2019). Like GNN, the Transductive Propagation Network (TPN) considered the graph nodes for representing the feature embeddings of the images (Liu et al., 2019). However, instead of performing inductive inference (that is, predicting test images one by one), TPN used transductive inference to predict the labels of the entire test set at once, which alleviated the low-data problem in FSL and achieved state-of-the-art performance (Liu et al., 2019).

The Looking-Back method we propose in this work (Figure 3.2) uses the same graph construction approach as TPN (Liu et al., 2019). However, Looking-Back incorporates the feature embeddings from hidden layers in

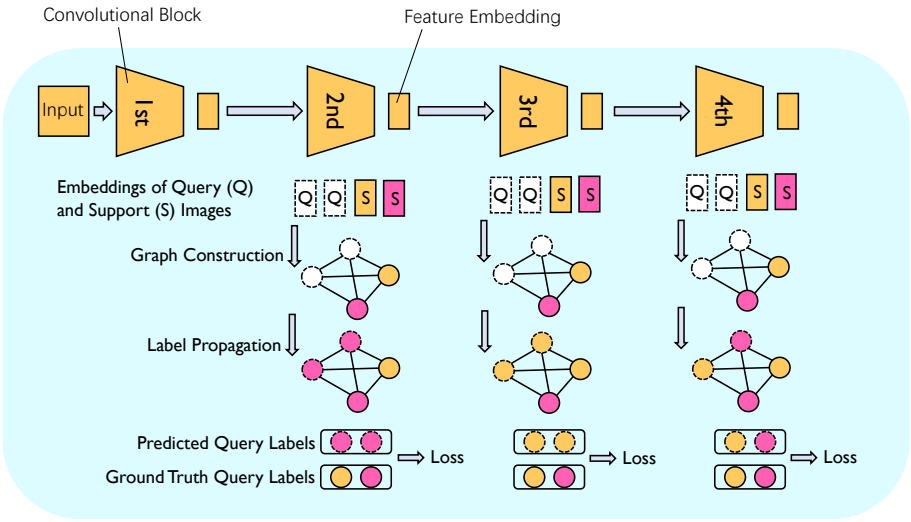

Figure 2.1: A conceptual overview of the proposed Looking-Back method.

the graph construction procedure as well. We shall note that the simultaneous training with graphs built on lower-level information could also be seen as a particular case of multi-task learning or incremental learning, which was mentioned in (Mallya and Lazebnik, 2018) but is rarely adopted in FSL.

2.3 Proposed Method

In this section, we introduce our proposed Looking-Back approach utilizing lower-level information to enhance the predictive performance of FSL models.

2.3.1 Problem Definition

The goal of FSL is to train predictive models that learn from and perform well on classification tasks, given only a few labeled examples per class. For instance, N-way K-shot classification can be understood as a classification task with N unique classes, where K labeled examples per class are provided for supervised learning.

In an N-way K-shot setting, the dataset for a given task is divided into a support set $\mathbf{S} = \{(\mathcal{X}_s, \mathcal{Y}_s)\}$ and a query set $\mathbf{Q} = \{(\mathcal{X}_q, \mathcal{Y}_q)\}$. $\mathbf{S}$ consists of $N \times K$ examples $\mathcal{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N\times K})$ and the corresponding class labels $\mathcal{Y} = (y_1, y_2, \ldots, y_{N\times K})$. The goal is to utilize $\mathbf{S}$ to predict the class labels $(y'_1, y'_2, \ldots, y'_{Q\times K})$ for the $Q \times K$ examples in $\mathbf{Q}$, $(\mathbf{x}'_1, \mathbf{x}'_2, \ldots, \mathbf{x}'_{Q\times K})$.

Given a large training dataset $\mathcal{D}_{base}$, with base classes $\mathcal{C}_{base}$, FSL meta-learning approaches sample many different N-way K-shot classification tasks $\mathcal{T} = \{T_1, T_2, ..., T_m\}$ randomly from $\mathcal{D}_{base}$, to train the meta-learner for m episodes. After training, the meta-learner is given a novel N-way K-shot classification task T_{novel}, such that the N classes do not overlap with the base classes in $\mathcal{D}_{base}$ encountered during training. The dataset corresponding to T_{novel} is split into support and query sets, and the meta-learner uses the $N \times K$ labeled examples in the support set to classify the $Q \times K$ examples in the query set. The classes in T_{novel} are all novel classes $\mathcal{C}_{novel}$ which are disjoint with $\mathcal{C}_{base}$.

A successful FSL meta-learner learns from the training tasks $T_1, T_2, ..., T_m$ how to efficiently utilize the few labeled examples in the support set of

a novel task $\mathsf{T}_{\text{novel}}$ so that the resulting model is able to predict the class labels in the unlabeled query set with good generalization performance.

Considering the general problem definition of FSL and meta-learning given above, the examples in the query set can be used in a transductive manner as suggested by (Liu et al., 2019). I.e., instead of classifying the query examples one at a time, the whole query set can be propagated into the network all at once, which improves the predictive performance compared to classifying each query example independently (Liu et al., 2019).

2.3.2 Feature Extractor Module

The two predominant types of neural network backbone architectures used in FSL research are ResNet-12 (Mishra et al., 2018; Oreshkin et al., 2018; Lee et al., 2019; Sun et al., 2019) and Conv-64F (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018; Garcia and Bruna, 2017; Li et al., 2019a; Liu et al., 2019). In this work, we adopt Conv-64F since it is easier to experiment with. However, we shall note that our proposed method is architecture-agnostic and can be implemented for other types of feedforward neural networks.

Conv-64F contains four convolutional blocks where every block is constructed by one convolutional layer with 64 filters of size 3×3, followed by a batch normalization layer, ReLU activation, and a 2×2 max-pooling layer. Both the convolutional layers and the max-pooling layers have a stride of 1.

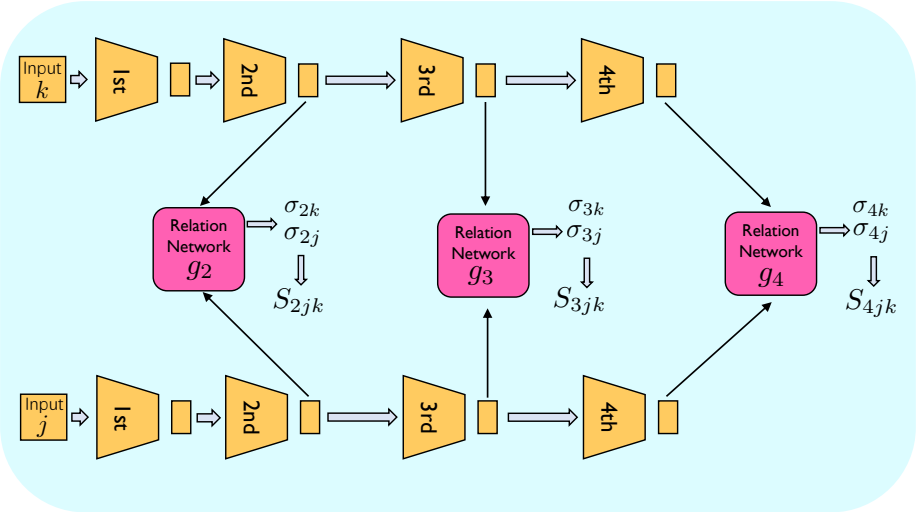

Figure 2.2: Computing the similarity between a pair of images, inputs j and k, based on the feature embeddings produced by the 2nd, 3rd, and 4th convolutional block (layer). The similarity values computed by the relation network modules are then used to construct multiple graphs for label propagation.

Besides extracting feature embeddings from the last layer of the last convolutional block, the proposed Looking-Back also extracts the embeddings from the last layer of the second and third convolutional block. These three feature embeddings are then used in the graph-based label propagation, as illustrated in Figure 3.2. The dimensions of the feature embeddings extracted by the three convolutional blocks are $64\times21\times21$, $64\times10\times10$, and $64\times5\times5$, respectively. Here, the number of channels, 64, is determined by the Conv-64F architecture, whereas the channel heights and widths are a consequence of the input image dimensions given the Conv-64F architecture.

2.3.3 Graph Construction Module

In the original work of TPN (Liu et al., 2019), the authors proposed a pairwise similarity function that used an example-wise length-scale parameter. Adopting this mechanism, for the output of i-th convolutional block, we compute the similarity of two images (j, k) via

$$S_{ijk} = \exp\left(-\frac{1}{2}\left\|\frac{f_i(\mathbf{x}_{ij})}{\sigma_{ij}} - \frac{f_i(\mathbf{x}_{ik})}{\sigma_{ik}}\right\|_2^2\right), \tag{2.1}$$

which measures the distance between the two feature embeddings. Here σ_{ij} is a scale parameter for the feature embedding computed by a relation network module, which is described in the next paragraph. As illustrated in Figure 2.2, we use a separate relation network for the second, third, and fourth convolutional block, since the dimensions and information contents of the respective feature embeddings differ.

The overall architecture of the relation network module, which computes σ_{ij} and σ_{ik}, is similar to the architecture used by Li et al. (2019a). Each relation network module consists of two convolutional blocks, followed by two fully-connected layers. Each convolutional block is composed of a 3×3 convolutional layer with a stride of 1, a batch normalization layer, ReLU activation, and a 2×2 max-pooling layer with a stride of 1.

In the Looking-Back model, we compute multiple symmetric normal-

ized graph Laplacians (Chung and Graham, 1997) via

$$L_i = D_i^{-1/2} S_i D_i^{-1/2}, \tag{2.2}$$

where D_i is the diagonal matrix whose d-th diagonal element is the sum of the d-th row of the S_i. Here, we only keep m-max values from every row in S_i to construct a m-nearest neighbor graph for each layer during episodic training to improve computational efficiency as suggested by Liu et al. (2019).

2.3.4 Classification Loss

After constructing different graphs for multiple layers as explained in Section 2.3.3, label propagation (Zhou et al., 2004) is used to compute the prediction (i.e., class-membership) scores for the query images (Liu et al., 2019).

Let $P^{(0)}$ be an initial score matrix. For a given image $\langle \mathbf{x}_j, y_j \rangle$ in the support set,

$$P_{jl}^{(0)} = \begin{cases} 0 & \text{if } y_j \neq l, \\ 1 & \text{if } y_j = l. \end{cases} \tag{2.3}$$

The label propagation process is an iterative process

$$P^{(t+1)} = \alpha L_i P^{(t)} + (1 - \alpha) P^{(0)}, \tag{2.4}$$

where $P^{(t)}$ is the predicted label at time step t. The predicted scores P_i^* for an input image's feature embedding from the i-th convolutional block are computed via

$$P_i^* = (I - \alpha L_i)^{-1} P^{(0)}, \tag{2.5}$$

where I is the identity matrix, L_i is the normalized graph Laplacian of that feature embedding from the i-th convolutional block, and α is a hyperparameter controlling propagation rate.

After computing the prediction scores, we obtain class-membership probability scores for the feature embeddings from the i-th convolutional block by applying a softmax function as follows:

$$p(\hat{y}_{ij} = k|\mathbf{x}_{ij}) = \frac{\exp(P^*_{i,jk})}{\sum_{k=1}^{N} \exp(P^*_{i,jk})}, \tag{2.6}$$

where $\hat{y}_{ij}$ is the predicted class label for feature embedding of the j-th input image from the i-th convolutional block, and $P^*_{i,jk}$ is the predicted score at the k-th position.

The total loss term is the combination of cross-entropy loss for different layers' features:

$$\text{Loss} = -\sum_{i} \sum_{j=1}^{N \times K+Q} \sum_{k=1}^{N} w_i I(y_{ij} = k) \log(p(\hat{y}_{ij} = k|\mathbf{x}_{ij})), \tag{2.7}$$

where w_i is a relative weight for the cross-entropy loss term of the feature embeddings from the i-th convolutional block and is a hyperparameter

during the episodic training.

The feature embeddings from the second ($i = 2$) and third ($i = 3$) convolutional block containing lower-level information are only used during training to improve the feature extractor module (Section 2.3.2). In both the validation and test stage, the class labels $\hat{\mathbf{y}} = \arg\max_k p(\hat{\mathbf{y}}_i = k|\mathbf{x}_i)$ are obtained from the prediction on feature embeddings of the last convolutional block only, that is, the fourth convolutional block, $i = 4$.

2.4 Experiments

In this section, we evaluate the proposed Looking-Back method on two popular FSL benchmark datasets, i.e., *mini*ImageNet (Ravi and Larochelle, 2017) and *tiered*ImageNet (Ren et al., 2018), and compare with other state-of-the-art FSL methods.

2.4.1 Datasets

***mini*ImageNet.** The *mini*ImageNet dataset is widely used for comparing different few-shot learning methods (Ravi and Larochelle, 2017). It is a small subset of ImageNet (Russakovsky et al., 2015) that consists of 100 classes with 600 examples per class. For our experiments, we split the dataset into 64 classes for training, 16 classes for validation, and 20 classes for testing following (Ravi and Larochelle, 2017).

***tiered*ImageNet.** Similar to *mini*ImageNet, the *tiered*ImageNet dataset is a small, simplified version of ImageNet proposed by Ren et al. (2018). Different from *mini*ImageNet, *tiered*ImageNet has a hierarchical or *tiered* structure consisting of 34 larger classes, where each larger class contains 10 to 30 smaller classes (i.e., related subcategories). *tiered*ImageNet contains 608 smaller classes and 779,165 images in total. We split the dataset as described in (Ren et al., 2018), resulting in a training set consisting of 20 larger classes, a validation set consisting of 6 larger classes, and the test set consisting of 8 larger classes. The advantage of splitting the dataset based on the larger classes, as opposed to splitting into the subclasses, is that this approach creates a clearer distinction between training, test, and validation sets.

2.4.2 Implementation Details

As mentioned before, we adopted the Conv-64F architecture (Section 2.3.2) as the backbone for our model. During training, we used the three layers' feature embeddings as shown in Figure 3.2 and Figure 2.2. For label propagation, we chose the same hyperparameters as described in (Liu et al., 2019), setting α (the propagation coefficient, Eq. 2.4 and 2.5) to 0.99 and m (the per-row max values of the graph Laplacians) to 20. Moreover, we gave equal weighting to the individual loss terms when computing the total loss Eq. 2.7, that is, setting w_2, w_3, and w_4 to 1.

During the episodic training, each episode was a 5-way K-shot task

with 15-query images in each task, mimicking the testing scenario. We used the Adam optimizer (Kingma and Ba, 2014) to train the model and set the initial learning rate to 0.001. For *mini*ImageNet, the learning rate was decayed by a multiplicative factor of 0.8 every 5,000 episodes. The same multiplicative factor was used for decaying the learning rate when training on *tiered*ImageNet, but it was decayed more frequently, every 2,000 epochs, due to the larger size and complexity of *tiered*ImageNet.

To evaluate the model on the test set, we randomly sampled 600 5-way K-shot tasks from an independent test set with $K = 1$ and $K = 5$, respectively. In both scenarios, $K = 1$ and $K = 5$, there were 15 query samples in each class (that is, 75 query examples in total), which were used to compute the prediction accuracy for a given task or episode. To compute the overall prediction accuracy of a given model, we randomly sampled the test set 600 times and calculated the accuracy by averaging the prediction accuracy across these 600 episodes.

2.4.3 Results and Discussion

Overall performance. In this section, we compare our proposed Looking-Back method to other state-of-the-art FSL methods. All neural network implementations are based on a Conv-64F backbone architecture for feature extraction as described in Section 2.3.2. Following the established conventions, we consider both 5-way 1-shot and 5-way 5-shot settings for the performance comparisons, using the two common FSL benchmark datasets

Table 2.1: Accuracy (in %) on *mini*ImageNet with 95% confidence interval. Best results are shown in bold.

Method	Extract. Net.	1-shot	5-shot
Matching Net (2016)	Conv-64	43.56 ± 0.84	55.31 ± 0.73
Prototypical Net (2017)	Conv-64	49.42 ± 0.78	68.20 ± 0.66
Relation Net (2018)	Conv-64	50.44 ± 0.82	65.32 ± 0.70
Reptile (2018)	Conv-64	49.97 ± 0.32	65.99 ± 0.58
GNN (2017)	Conv-64	49.02 ± 0.98	63.50 ± 0.84
MAML (2017)	Conv-64	48.70 ± 1.84	63.11 ± 0.92
TPN (2019)	Conv-64	53.75 ± 0.86	69.43 ± 0.67
Looking-Back	Conv-64	$\mathbf{55.91 \pm 0.86}$	$\mathbf{70.99 \pm 0.68}$

Table 2.2: Accuracy (in %) on *tiered*ImageNet with 95% confidence interval. Best results are shown in bold.

Method	Extract. Net.	1-shot	5-shot
Prototypical Net (2017)	Conv-64	53.31 ± 0.89	72.69 ± 0.74
Relation Net (2018)	Conv-64	54.48 ± 0.93	71.31 ± 0.78
Reptile (2018)	Conv-64	52.36 ± 0.23	71.03 ± 0.22
MAML (2017)	Conv-64	51.67 ± 1.81	70.30 ± 1.75
TPN (2019)	Conv-64	57.53 ± 0.96	72.85 ± 0.74
Looking-Back	Conv-64	$\mathbf{58.97 \pm 0.97}$	$\mathbf{73.59 \pm 0.74}$

*mini*ImageNet and *tiered*ImageNet as described in Section 2.4.1. The accuracy is computed as the average of 600 test episodes (as described in Section 2.4.2) with a 95% confidence interval. As the results for *mini*ImageNet (Table 3.1) and *tiered*ImageNet (Table 2.2) indicate, our proposed Looking-Back method achieves state-of-the-art results on both datasets, in both the 5-way 1-shot and 5-way 5-shot scenarios.

Comparing Looking-Back and TPN training in a "Higher Shot" setting. The performance comparisons between Looking-Back and TPN (Liu

Table 2.3: Accuracy (in %) after training with higher shots. Here, the system evaluated on a 1-shot test set was trained in a 5-shot setting, and the system evaluated on a 5-shot test was trained in a 10-shot setting. Best results are shown in bold.

Dataset	Method	1-shot	5-shot
*mini*ImageNet	TPN	55.51 ± 0.86	69.86 ± 0.65
	Looking-Back	$\mathbf{56.49 \pm 0.83}$	$\mathbf{70.47 \pm 0.66}$
*tiered*ImageNet	TPN	59.91 ± 0.94	73.30 ± 0.75
	Looking-Back	$\mathbf{61.19 \pm 0.92}$	$\mathbf{73.78 \pm 0.74}$

et al., 2019) (Table 3.1 and 2.2) provides supportive evidence that utilizing lower-level information, which is contained in previous layers' feature embeddings, improves the predictive performance by a substantial amount by our Looking-back. In this section, we investigate whether the lower-level information can also enhance the performance in a "Higher Shot" setting.

In FSL, it is common to use support sets of similar size during meta-training and testing. However, some researchers found that using larger support sets during meta-training (i.e., increasing the number of "shots") can improve the predictive performance of FSL systems based on evaluation on the same (i.e., smaller shot) test sets (Snell et al., 2017; Li et al., 2019a). Similar observations have been made in the original TPN paper (Liu et al., 2019), where the authors described that increasing the number of examples in the support sets during meta-training (referred to as "Higher Shot") can improve the predictive accuracy during testing. However, using a larger number of shots during meta-training than testing does not always improve the predictive performance, and it is still an open

area of research (Cao et al., 2019).

Although "Higher Shot" training is not the focus of this work, we conducted experiments with higher shots and report the results in Table 2.3, adopting the procedure described in the original TPN paper (Liu et al., 2019) to enable fair comparisons. The results in Table 2.3 indicate that Looking-Back utilizing lower-level information outperforms TPN in a "Higher Shot" setting as well.

Table 2.4 summarizes the performance gain of Looking-Back over TPN for the regular meta-training scenario (same number of shots in the training and test tasks, Table 3.1 and 2.2) and meta-training with higher shots (Table 2.3). From Table 2.4, we can observe that on both datasets, the improvement of *same* versus *higher* shot meta-training in 1-shot settings is more significant than in 5-shot settings. We argue that when more support images are available (higher shot), the role of utilizing lower-level information becomes less important. The main rationale behind using previous layers' feature embeddings is to use additional lower-level information when information from the final layer's feature embedding is scarce. Intuitively, the role of using lower-level information degrades if a meta-learner can utilize a larger number of examples in the support set.

Influence of higher-shot training on Looking-Back. As indicated by the results in Table 2.4 and hypothesized in the previous section, our Looking-Back method could be more useful when the data is more scarce. This is likely because the more information is available during training (i.e.,

Table 2.4: Performance gain (in % points) of Looking-Back vs TPN on the test sets when using lower-level information in same-shot (Same) and higher-shot (Higher) training.

Training approach	Dataset	1-shot	5-shot
Same	*mini*ImageNet	2.16	1.56
	*tiered*ImageNet	1.44	0.74
Higher	*mini*ImageNet	0.98	0.61
	*tiered*ImageNet	1.28	0.48

Table 2.5: Performance gain (in % points) of Looking-Back when trained with higher shots compared to training with same shots.

Dataset	1-shot	5-shot
*mini*ImageNet	0.58	-0.52
*tiered*ImageNet	2.22	0.19

the support sets consist of additional examples in higher-shot settings), the more negligible the information from earlier layers becomes as supportive information.

In a 1-shot setting, we were still able to observe that the lower-level information used by Looking-Back models benefits the model performance when training in the higher shots setting, as summarized in Table 2.5. However, in the presence of a larger number of images, using lower-level information during training results in more limited improvements (5-shot test setting on *tiered*ImageNet) or may have a small detrimental impact (5-shot test settings on *mini*ImageNet) as shown in Table 2.5. This finding provides further evidence that the lower-level information has a more beneficial effect when the data is more scarce.

Why only using the last layer's information during inference. Both

Table 2.6: Different layers' prediction accuracy (in %) on 5-way tasks after label propagation with same-shot training.

Dataset	Setting	2nd layer	3rd layer	4th layer
*mini*ImageNet	1-shot	42.24 ± 0.76	50.87 ± 0.81	55.91 ± 0.86
	5-shot	58.10 ± 0.72	67.07 ± 0.69	70.99 ± 0.68
*tiered*ImageNet	1-shot	46.25 ± 0.87	54.70 ± 0.93	58.97 ± 0.97
	5-shot	61.12 ± 0.75	69.94 ± 0.74	73.59 ± 0.74

DN4 (Li et al., 2019a) and the dense classification network (Lifchitz et al., 2019) use the entire expanded feature embeddings of the last layer during training as well as inference. One of the main reasons we only use the feature embeddings of the last layer during inference is that the lower-level information from previous layers is used to augment the graph construction during training but does not have equal relevance for the prediction task during inference. In contrast to Looking-Back, in both DN4 and the dense classification network, the additional information of the expanded feature embeddings are on the same footing.

To test our hypothesis that the feature embeddings of the last layer bear the most relevance for the prediction task, we compared the prediction accuracy of Looking-Back when using different layers for the class label prediction. As indicated by the results in Table 2.6, the prediction accuracy of the 4th (last) layer is higher than the prediction accuracy of the 3rd layer, and the accuracy of the 3rd layer is higher than the accuracy of the 2nd layer, supporting the hypothesis that the last layer contains the most useful information.

2.5 Conclusion

In this work, we propose a new approach to FSL that captures additional information inside the feature extracting network to improve prediction performance. In particular, the proposed Looking-Back method employs a graphical structure to utilize the lower-level information from previous layers' feature embeddings, which differs from existing methods that only focus on expansions of the last layer's feature embeddings. Experiments on two popular FSL datasets provide evidence that the utilization of lower-level information in FSL improves the performance of FSL meta-learners.

3 TRANSMATCH: A TRANSFER-LEARNING SCHEME FOR SEMI-SUPERVISED FEW-SHOT LEARNING

3.1 Introduction

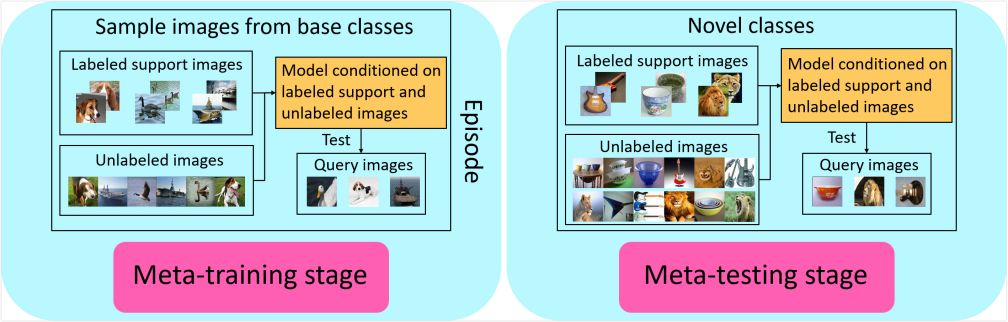

Figure 3.1: An overview of meta-learning based semi-supervised few-shot classification framework. Unlabeled images are required during training to allow the meta-learner learn how to leverage unlabeled images for classification.

We believe the sufficient and proper utilization of the extra information is crucial to the success of applying few-shot learning. Such extra information can exist in various forms, while in this work, we focus on leveraging the extra information from the labeled base-class data and unlabeled novel-class data. These two types of information are usually easy to obtain. Many existing large-scale datasets for visual recognition tasks can be used for pre-training a model which can be later transferred to a new task. Meanwhile, it is also relatively easy to acquire a large amount of unlabeled data for a new task. Thus, a new paradigm called semi-supervised

few-shot learning arises recently.

A representative work for semi-supervised few-shot learning (Ren et al., 2018) employed the meta-learning framework and enhanced the prototypical networks (Snell et al., 2017) to use unlabeled data. In each episode during meta-training, the unlabeled data for base classes was included to mimic the test scenario where the unlabeled data for novel classes would be available. Liu et al. (2019) proposed transductive propagation to incorporate the popular label propagation method to utilize the unlabeled data in episodic training. These works demonstrated that considering the unlabeled data helped to improve the accuracy of few-shot classification under the meta-learning framework.

In this section, we propose a new framework for semi-supervised few-shot learning to fully utilize the auxiliary information from labeled base-class data and unlabeled novel-class data. The flowchart of our proposed framework is showed in Fig. 3.2, which consists of three components. We first train a model using the large amount of labeled data from the base classes, encoding the knowledge from base-class data into the pre-trained model. Then this pre-trained model is adopted as a feature extractor to generate the feature embeddings of the labeled few-shot examples from the novel classes, which can be directly used to imprint classifier weights for the novel classes or as the initialization of classifier weights for further fine-tuning, following the transfer-learning framework (Qi et al., 2018). Different from meta-learning, unlabeled images are no longer needed

during pre-training on base classes, and could be directly utilized upon this imprinted classifier with state-of-the-art semi-supervised method such as MixMatch (Berthelot et al., 2019). To the best of our knowledge, this is the first work of semi-supervised few-shot learning under the transfer-learning framework in contrast to the meta-learning framework.

In summary, the contributions of our work are:

1. We propose a new transfer-learning framework for semi-supervised few-shot learning, which can fully utilize the auxiliary information from labeled base-class data and unlabeled novel-class data.

2. We develop a new method called *TransMatch* under the proposed framework. TransMatch integrates the advantages of transfer-learning based few-shot learning methods and semi-supervised learning methods, and is different from the previous work on meta-learning based methods.

3. We conduct extensive experiments on two popular benchmark datasets for few-shot learning to demonstrate that our method can effectively leverage unlabeled data in few-shot learning and achieve new state-of-the-art results.

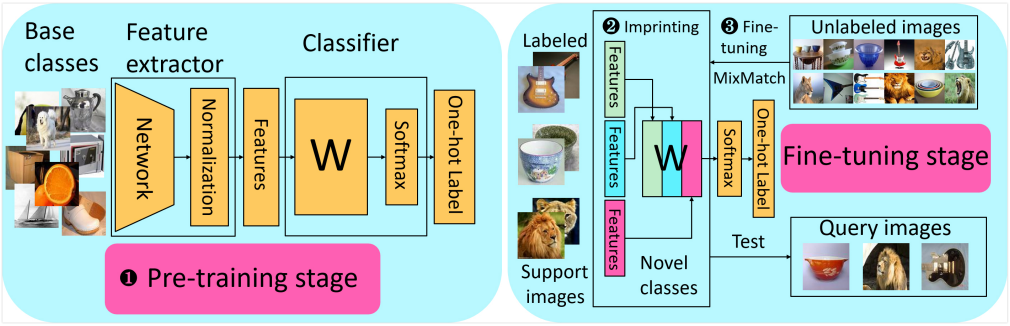

Figure 3.2: Our proposed framework of transfer-learning scheme for semi-supervised few-shot learning. We first pre-train a classifier from base-class images. Then use it as a feature extractor to initialize the weights for novel-class classifier. Finally, we further fine-tune the novel-class classifier with unlabeled images by semi-supervised learning method MixMatch.

3.2 Related Work

In this section, we review the related work to our proposed transfer-learning based semi-supervised few-shot learning framework.

3.2.1 Few-Shot Learning

Few-shot learning has attracted increasing attention in recent years. Existing work can be roughly categorized into (i) meta-learning methods, and (ii) transfer-learning methods.

Meta-learning based method: Meta-learning based few-shot learning, also known as *learning to learn*, aims to learn a paradigm that can be adapted to recognize novel classes with only few-shot training examples. Meta-learning based methods usually consist of two stages: 1) meta-training; and 2) meta-testing. In the meta-training stage, a sequence of episodes

are randomly sampled from the examples of base classes where each episode contains K support examples and Q query examples from N classes, denoted as an N-way K-shot episode. In this way, the meta-training stage can mimic the few-shot testing stage where only a few examples per class are available. The meta-learning based methods can be further divided into two categories: a) metric-based methods; and b) optimization-based methods.

a) Metric-based methods have been proposed in many work (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018; Oreshkin et al., 2018; Li et al., 2019a). These methods mainly focus on learning a good metric to measure the distance or similarity among support images and query images. For example, prototypical networks (Snell et al., 2017) calculated the distance of the prototype representations of each class between supports and queries. Relation Net (Sung et al., 2018) implemented a network to measure the relation similarities between the supports and queries.

b) Optimization-based methods aim to design an optimization algorithm that can adapt the information during meta-training stage to the meta-testing stage. MAML (Finn et al., 2017) learned an optimization method that can follow the fast gradient direction to rapidly learn the classifier for novel classes. LEO (Rusu et al., 2019) decoupled the gradient-based adaptation process with high-dimensional parameters to few-shot scenarios. However, meta-learning based method needs to construct a sequence of episodes during meta-training, and requires the model to train

from scratch in order to adapt the episodic learning process to meta-testing stage. This introduces complexity into the training. While our method can use the simpler conventional pre-training, and adapt the pre-trained model to novel classes at ease.

Transfer-learning based methods: Transfer-learning based methods are different from meta-learning based methods, as they do not use the episodic training strategy. Instead, such methods can use conventional techniques to pre-train a model on the large amount of data from the base classes. The pre-trained model is then adapted to the few-shot learning task of recognizing novel classes. Qi et al. (2018) proposed to imprint the classifier weights of novel classes by the mean vectors of the feature embeddings of few-shot examples. Qiao et al. (2018) learned a mapping function from the activations (*i.e.*, feature embeddings) of novel class examples to classifier weights. Gidaris and Komodakis (2018) proposed an attention module to dynamically predict the classifier weights for novel classes. Chen et al. (2019) showed such transfer-learning based methods can achieve competitive performance as meta-learning based methods. Our proposed framework shares a similar idea with (Qi et al., 2018) by pre-training a feature extractor and uses it to extract features for few-shot examples from novel classes which are used to imprint classifiers weights.

3.2.2 Semi-Supervised Learning

Semi-supervised learning focuses on developing algorithms to learn from unlabeled and labeled data. Existing work can be roughly categorized into (i) consistency regularization methods, and (ii) entropy minimization methods.

Consistency regularization methods: Consistency regularization methods mainly focus on adding noise and augmentation to images without changing their label distribution. Π-Model (Laine and Aila, 2017) added an loss term to regularize the model by stochastic augmentation. Mean Teacher (Tarvainen and Valpola, 2017) improved Π-Model by using the exponential moving average of parameters. Virtual Adversarial Training (VAT) (Miyato et al., 2018) regularized the model by adding local perturbation on unlabeled data.

Entropy minimization methods: This family of methods focuses on giving low entropy for unlabeled data. It is initially proposed by Grandvalet and Bengio (2005) which minimized conditional entropy of unlabeled data. Pseudo-Label (Lee, 2013) minimized the entropy directly by predicting the labels for unlabeled data and used this in cross-entropy, showing its good performance.

MixMatch (Berthelot et al., 2019) united different kinds of consistency regularization and entropy minimization methods and achieved state-of-the-art performance by a large margin comparing with all the previous methods. It is a holistic method in semi-supervised learning and we would

introduce briefly in Section 3.3.3. Due to its good performance, we adopt MixMatch in our framework, and we also compared with using other mainstream semi-supervised learning methods in the experiments. Semi-supervised learning methods are usually compared on small datasets (Oliver et al., 2018; Berthelot et al., 2019; Miyato et al., 2018) where there is a small amount of labeled data. But the number of labeled images in typical semi-supervised learning is still greater than few-shot learning. The techniques for semi-supervised method may not be directly used for few-shot setting, which is also demonstrated in our experiments that naively applying MixMatch to few-shot learning may lead to poor performance especially in 1-shot and 2-shot.

3.2.3 Semi-Supervised Few-Shot Learning

When there are few-shot examples for novel classes, it is straightforward to utilize extra unlabeled data to improve the learning. This leads to the family of semi-supervised few-shot learning methods (SSFSL). There are very few works in this direction. Ren et al. (2018) extended prototypical networks to incorporate unlabeled data by producing prototypes for the unlabeled data. Liu et al. (2019) constructed a graph between labeled and unlabeled data and utilize label propagation to obtain the labels of unlabeled data. Li et al. (2019b) applied self-training by adding the confident prediction of unlabeled to the labeled training set in each round of optimization.

However, all existing semi-supervised few-shot learning methods are meta-learning based methods as in Fig. 3.1. As showed in (Chen et al., 2019), transfer-learning based method can achieve competitive performance compared with meta-learning based methods. This motivates our work. We need to emphasize that meta-learning based methods have shown their success to utilize unlabeled data by integrating unlabeled data in episodic training. However, this episodic training strategy is different from typical semi-supervised learning and it is not appropriate to combine them together directly. The techniques of leveraging unlabeled data in existing SSFSL methods are not state-of-the-art in semi-supervised areas and the more powerful and holistic methods like MixMatch would be difficult to integrate in meta-learning framework. Meanwhile, directly applying semi-supervised methods to utilize unlabeled data during test may lead to bad performance due to the extreme small number of labeled data.

3.3 The Proposed Framework

In this section, we introduce our proposed transfer-learning framework for semi-supervised few-shot learning. The flowchart is illustrated in Fig. 3.2, which contains three modules: 1) pre-training a feature extractor on base-class data; 2) use the feature extractor to extract features from novel-class data and imprint novel-class classifier weights; and 3) further fine-tuning

the model by semi-supervised learning method. Before elaborating the details of each module, let us first introduce our problem definition.

Problem definition: We have a large-scale dataset $\mathcal{D}_{base}$ containing many-shot labeled examples from each base class in $\mathcal{C}_{base}$ and a small-scale dataset $\mathcal{D}_{novel}$ of only few-shot labeled examples and many-shot unlabeled examples from each novel class in $\mathcal{C}_{novel}$, where $\mathcal{C}_{novel}$ is disjoint from $\mathcal{C}_{base}$. The task of semi-supervised few-shot learning is to learn a robust classifier using both the few-shot labeled examples and many-shot unlabeled examples in $\mathcal{D}_{novel}$ with the examples in $\mathcal{D}_{base}$ as auxiliary data. Usually in a conventional few-shot learning task, a small support set of N classes with K images per class is sampled from $\mathcal{D}_{novel}$, leading to the N-way-K-shot problem. In semi-supervised few-shot learning, additional U unlabeled images are sampled from each of the N novel classes or distractor classes.

3.3.1 Part I: Pre-train Feature Extractor

The first module of our framework, as showed in the left part of Fig. 3.2, is a pre-training module, which relies on the many-shot examples from base classes, $\mathcal{D}_{base}$, to train a base model which encodes as much as possible the information of $\mathcal{D}_{base}$ and can be used in the later stage of few-shot learning as prior information, similar to human intelligence. This is different from conventional meta-learning based few-shot learning as showed in Fig. 3.1, where an episodic training strategy is employed for base classes as well to

mimic the few-shot scenario in the testing phase.

3.3.2 Part II: Classifier Weight Imprinting

The weight imprinting method was proposed by Qi et al. (2018), and has achieved impressive performance in the few-shot learning task as a representative of transfer-learning based few-shot learning method. Specifically, it directly sets the classifier weights by the mean feature vectors of the N-way-K-shot examples, where features are obtained by the model from the pre-training stage. For convenience, we denote the classifier on large scale base classes as $f(\mathbf{x}) = f^{base}(f^e(\mathbf{x}))$, where $\mathbf{x}$ is an input example, $f^e(\cdot)$ is the feature extractor and $f^{base}(\cdot)$ is the classifier. We have $f^e(\mathbf{x}) \in \mathcal{R}^d$ and $f^{base}(\cdot) \in \mathcal{R}^{|\mathcal{C}_{base}|}$.

Given the N-way-K-shot examples from novel classes and let us denote them as $\mathcal{D}_{novel} = \{\mathbf{x}_k^c|_{k=1...K,\ c=1...N}\}$ with $\mathbf{x}_k^c$ as the k-th example in c-th class. We can use the feature extractor learned on base classes to extract features for the N-way-K-shot examples, denoted as $f^e(\mathbf{x}_k^c)$. Meanwhile, let us write the classifier for novel classes as $f^{novel}(\mathbf{x}) = \mathbf{W}'\mathbf{x}$, where $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_N] \in \mathcal{R}^{d\times N}$. Note that we omit the bias for simplicity. By normalizing the weight $\mathbf{w}_c$ and the feature vector $\mathbf{x}$ onto a unit ball, the aforementioned equation can be further simplified as

$$f^{novel}(\mathbf{x}) = [\cos(\theta(\mathbf{w}_1, \mathbf{x}), ..., \cos(\theta(\mathbf{w}_N, \mathbf{x}))]', \tag{3.1}$$

where $\theta(\mathbf{w}_i, \mathbf{x})$ denotes the angle between $\mathbf{w}_i$ and $\mathbf{x}$, and the classification for a given example $\mathbf{x}$ is based on computing the cosine similarity between every $\mathbf{w}_k$ and $\mathbf{x}$, and predict the label of $\mathbf{x}$ based on maximum similarity score.

In this sense, there is a duality between $\mathbf{w}_i$ and $\mathbf{x}$. Based on this observation, weight imprinting uses the mean feature vectors of the few-shot examples to imprint $\mathbf{w}_c$, *i.e.*, by setting

$$\mathbf{w}_c = \frac{1}{K} \sum_{k=1}^{K} f^e(\mathbf{x}_k^c). \tag{3.2}$$

The classification of an given example $\mathbf{x}$ can be also deemed as computing the mean of the similarities between $\mathbf{x}$ and all K-shot examples.

By imprinting the classifier weights with mean feature vectors of the few-shot examples, it provides a better initialization of classifier weights to reduce the intra-class variations of features and benefits fine-tuning the new classifier for novel classes. Experimental results show that it can achieve good performance even without fine-tuning.

3.3.3 Part III: Semi-Supervised Fine-tuning

After we get the classifier which fully absorbs the information from base classes with a better initialization by imprinting, we fine-tune this classifier during test when there is unlabeled data. This fine-tuning process is the same as semi-supervised training. Any semi-supervised learning can

be applied, and in this work we employed MixMatch (Berthelot et al., 2019) not only because of its excellent performance in the semi-supervised learning task, but also because it is a holistic method to leverage unlabeled data in semi-supervised learning area.

MixMatch combines multiple existing improvements from state-of-the-art semi-supervised learning methods which is discussed in Section 3.2.2. In our setting, we denote $\mathcal{L} = \{(\mathbf{x}_i, p_i)\}_{i=1}^{B}$ as a mini-batch of B labeled examples with p_i as the label, and $\mathcal{U} = \{\mathbf{x}_u\}_{u=1}^{U}$ as a mini-batch of U unlabeled examples. The imprinted classifier from Part II can be used to obtain estimated labels for the examples in $\mathcal{U}$, *i.e.*, $f^{novel}(\mathbf{x}_u)$. We will omit the superscript ^{novel} for the ease of illustration when there is no confusion. For robustness, we augment each example M times to get M versions of each unlabeled data, *i.e.*, $\{\mathbf{x}_{u,1}, ..., \mathbf{x}_{u,M}\}$, and use the mean prediction as the label estimation: $\bar{p}_u = \frac{1}{M}\sum_{i=1}^{M} f(\mathbf{x}_{u,i})$. The sharpen operation is used to enhance to prediction as $p_u = \bar{p}_u^{\frac{1}{T}} / \sum_{j=1}^{N} (\bar{p}_u)_j^{\frac{1}{T}}$, we set $T = 0.5$ in the experiments. The same data augmentation is also applied to labeled examples in $\mathcal{L}$. Following Berthelot et al. (2019), we concatenate $\mathcal{L}$ and $\mathcal{U}$ and shuffle the examples, *i.e.*, $\mathcal{W} = \text{Shuffle}(\text{Concat}(\mathcal{L}, \mathcal{U}))$, and then split this set into two new sets:

$$
\begin{aligned}
\mathcal{X}_1' &= \{\text{MixUp}\,(\mathcal{L}_i, \mathcal{W}_i) \qquad |i \in 1, \ldots, |\mathcal{L}|\}, \\
\mathcal{X}_2' &= \left\{\text{MixUp}\left(\mathcal{U}_i, \mathcal{W}_{i+|\mathcal{L}|}\right) \; |i \in 1, \ldots, |\mathcal{U}|\right\},
\end{aligned}
$$

where MixUp is defined as

$$\begin{aligned}&\text{MixUp}\left((\mathbf{x}_1, p_1), (\mathbf{x}_2, p_2)\right)\\&\quad = ((\lambda'\mathbf{x}_1 + (1-\lambda')\mathbf{x}_2), (\lambda' p_1 + (1-\lambda')p_2))\end{aligned} \tag{3.3}$$

with $\lambda' = \max(\lambda, 1-\lambda)$. The parameter λ is randomly generated from a beta distribution Beta(α, α). The objective function to minimize is defined as

$$\ell = \ell_1 + \gamma\ell_2, \tag{3.4}$$

where

$$\ell_1 = -\frac{1}{|\mathcal{X}_1'|}\sum_{(\mathbf{x},p)\in\mathcal{X}_1'} p\log(f(\mathbf{x})), \tag{3.5}$$

is cross-entropy loss, and

$$\ell_2 = \frac{1}{N|\mathcal{X}_2'|}\sum_{(\mathbf{x},p)\in\mathcal{X}_2'} \|p - f(\mathbf{x})\|_2^2. \tag{3.6}$$

is consistency regularization loss in (Sajjadi et al., 2016). The details of our algorithm is summarized in Algorithm 1.

3.4 Experiments

In this section, we evaluate our proposed TransMatch and compare with state-of-the-art few-shot learning methods on two popular benchmark

Algorithm 1 Algorithm for our proposed TransMatch

Input: An auxiliary dataset $\mathcal{D}_{base}$ with examples from $\mathcal{C}_{base}$, N-way-K-shot dataset $\mathcal{D}_l$=$\{\mathbf{x}_{nk}, p|n = 1, \cdots, N; k = 1, \cdots, K\}$ with $p \in \mathcal{C}_{novel}$, and $\mathcal{D}_u = \{\mathbf{x}_u|u = 1, \cdots, U\}$

Output: N-way-K-shot classifier f^{novel} for novel classes in $\mathcal{C}_{novel}$

1: Pre-train a base network on all examples in $\mathcal{D}_{base}$ and denote it as $f^{base}(f^e(\mathbf{x}))$;
2: Apply the feature extractor $f^e(\mathbf{x})$ to imprint the novel classifier f^{novel} based on $\mathcal{D}_l$;
3: Apply semi-supervised learning method, MixMatch, to update the novel classifier f^{novel} with both $\mathcal{D}_l$ and $\mathcal{D}_u$;

datasets for few-shot learning, including miniImageNet and CUB-200-2011.

3.4.1 Experiments on miniImageNet

Dataset configuration: The miniImageNet dataset was originally proposed by Vinyals et al. (2016). It has been widely used for evaluating few-shot learning methods. It consists of 60,000 color images from 100 classes with 600 examples per class, which is a simplified version of ILSVRC 2015 (Russakovsky et al., 2015). We follow the split given by Ravi and Larochelle (2017) consisting of 64 base classes, 16 validation classes and 20 novel classes. We randomly select K (*resp.* U) examples from each novel class as the few-shot labeled (*unlabeled*) examples, and Q images from the rest as the test examples. In the experiments, we set $N = 5$, $K = \{1, 5\}$, $Q = 15$ and study the effect of using different values of U. We repeat the test experiments 600 times and report the mean accuracy with the 95%

confidence interval.

Compared methods: The miniImageNet dataset has been widely used for evaluating the performance of few-shot learning methods, and is a good benchmark to compare state-of-the-art methods. In particular, we compare with several conventional few-shot learning methods, as well as state-of-the-art semi-supervised few-shot learning methods including the semi-supervised extension to Prototypical Networks by Ren et al. (2018) (Soft k-Means, Soft k-Means+Cluster, Masked Soft k-Means), and TPN-semi by Liu et al. (2019). We also re-implement Soft k-Means, Soft k-Means+Cluster, Masked Soft k-Means with the same backbone (*i.e.*, WRN-28-10) as our method for fair comparison. As the area of semi-supervised few-shot learning has not been explored much yet, we also conduct extensive experiments to evaluate the performance of utilizing unlabeled data by our TransMatch under different few-shot settings.

Implementation details: Following the work (Qiao et al., 2018) for transfer-learning based method on miniImageNet, we use the wide residual network (*i.e.*, WRN-28-10) (Zagoruyko and Komodakis, 2016) as the backbone for our base model f^{base}. We train it from scratch using the examples from the base classes. In particular, we first train a WRN-28-10 classification network on all examples from the 80 base and validation classes. We then replace the last layer of this network by a 256-d fully connected layer, followed by a L2 normalization layer and a 80-d classifier. We set the batch size to 128, and set learning-rate to 0.01 for the last two layers

and 0.001 for all other layers. We reduce the learning rate by 0.1 every 10 epochs and train for a total of 28 epochs.

The base classifier f^{base} is used as the feature extractor to generate feature vectors for the few-shot examples from novel classes. We use the few-shot labeled examples to fine-tune the base classifier to novel classes. We also augment each labeled image for 10 times by random transformation and use the mean features to imprint the weights for novel classifier. We use a batch size of 16, and set 64 batches as an epoch[1]. We set weight decay to 0.04, learning rate to 0.001, and use SGD optimizer with a momentum of 0.9. For the fine-tuning stage, we set the parameters of MixMatch as follows. We set M (the times for augmentation) to 2, T (the temperature for the label distribution) to 0.5, γ (the weight for regularization term) to 5, α (the parameter in Beta distribution) to 0.75. Meanwhile we use an exponential moving average for model parameters when guessing labels. For 5-way-1-shot scenario, we fine-tune for 10 epochs when there are 20 or 50 unlabeled images, and 20 epochs when there are 100 or 200 unlabeled images. For 5-way-5-shot scenario, we fine-tune for 20 epochs when there are 20 and 50 unlabeled images, and 25 epochs when there are 100 and 200 unlabeled images. All the test results are based on 600 random experiments.

[1]We duplicate the labeled images dataset to make it larger, so that each batch may contain the same image multiple times.

Method	Type	1-shot	5-shot
Prototypical Net (Snell et al., 2017)	Meta, Metric	49.42±0.78	68.20±0.66
TADAM (Oreshkin et al., 2018)	Meta, Metric	58.50±0.30	76.70±0.30
MAML (Finn et al., 2017)	Meta, Optimization	48.70±1.84	63.11±0.92
SNAIL (Mishra et al., 2018)	Meta, Optimization	55.71±0.99	68.88±0.92
Activation Net (Qiao et al., 2018)	Transfer-learning	59.60±0.41	73.74±0.19
Imprinting (Qi et al., 2018)	Transfer-learning	58.68±0.81	76.06±0.59
Soft k-Means (Ren et al., 2018)	Semi, Meta-learning	50.09±0.45	64.59±0.28
Soft k-Means+Cluster (Ren et al., 2018)	Semi, Meta-learning	49.03±0.24	63.08±0.18
Masked Soft k-Means (Ren et al., 2018)	Semi, Meta-learning	50.41±0.31	64.39±0.24
TPN-semi (Liu et al., 2019)	Semi, Meta-learning	52.78±0.27	66.42±0.21
Soft k-Means (Re-implement with WRN-28-10)	Semi, Meta-learning	51.88±0.93	67.31±0.70
Soft k-Means+Cluster (Re-implement with WRN-28-10)	Semi, Meta-learning	50.47±0.86	64.14±0.65
Masked Soft k-Means (Re-implement with WRN-28-10)	Semi, Meta-learning	52.35±0.89	67.67±0.65
TransMatch (100 unlabeled images per class)	Semi, Transfer-learning	**63.02±1.07**	81.19±0.59
TransMatch (200 unlabeled images per class)	Semi, Transfer-learning	62.93±1.11	**82.24±0.59**

Table 3.1: Accuracy (in %) on miniImageNet with 95% confidence interval. Best results are in bold.

Method	# unlabeled	1-shot	5-shot
Imprinting	—	58.68±0.81	76.06±0.59
Imprinting+FT	0	55.60±0.77	74.17±0.60
TransMatch	20	58.43±0.93	76.43±0.61
TransMatch	50	61.21±1.03	79.30±0.59
TransMatch	100	**63.02±1.07**	81.19±0.59
TransMatch	200	62.93±1.11	**82.24±0.59**

Table 3.2: Accuracy (in %) with different number of unlabeled images on miniImageNet. Best results are in bold.

Results on miniImageNet: The results are summarized in Table 3.1. It is not surprising that our method outperforms conventional few-shot learning methods without using unlabeled by a large margin, as showed in the top portion of Table 3.1. Our method also outperforms state-of-the-art semi-supervised few-shot learning methods, which can be observed from the middle portion of Table 3.1. These results clearly show the superiority of our TransMatch as its effective utilization of information from unlabeled data.

Influence of unlabeled examples: In Table 3.2, we report the results using different numbers of unlabeled images. Note that Imprinting+FT stands for fine-tuning the imprinted classifier without unlabeled data. It is obvious that our TransMatch could achieve better performance with more unlabeled images. We also observe that the results begin to saturate after 100 unlabeled images for 1-shot setting. In general, the results show that our TransMatch can effecively utilize the unlabeled data.

Ablation study: We conduct an ablation study of our method without

# shot	Method	Accuracy	Gain
1-shot	w/ Pseudo-Label	57.01 ± 1.13	+6.01
	w/ MixMatch	63.02 ± 1.07	
2-shot	w/ Pseudo-Label	70.07 ± 0.96	+2.29
	w/ MixMatch	72.36 ± 0.88	
3-shot	w/ Pseudo-Label	76.01 ± 0.81	+1.40
	w/ MixMatch	77.41 ± 0.76	
4-shot	w/ Pseudo-Label	78.35 ± 0.73	+1.39
	w/ MixMatch	79.74 ± 0.65	
5-shot	w/ Pseudo-Label	80.00 ± 0.66	+1.19
	w/ MixMatch	81.19 ± 0.59	

Table 3.3: Comparison of our method using different semi-supervised learning methods (*i.e.*, Pseudo-Label and MixMatch) in our framework both with 100 unlabeled images for 5-way classification on miniImageNet.

Imprinting or MixMatch. Without Imprinting, our method reduces to semi-supervised learning method, *i.e.*, MixMatch (Note here the feature extractor is still already trained from base classes) and without MixMatch, our method reduces to Imprinting. The results are showed in Fig. 3.3. It is clear that both MixMatch and Imprinting are worse than our TransMatch. The inferior performance of MixMatch to our TransMatch clearly shows that directly applying MixMatch to the few-shot setting cannot lead to good performance especially in 1-shot and 2-shot setting. This is due to the lack of labeled data, which makes it hard to fine-tune the classifier during test when there is unlabeled data. However, our proposed TransMatch can obtain a good initialization by incorporating weight imprinting module.

We also observe a larger gain by our TransMatch over MixMatch when using a smaller number of shots. The gain showed in Fig. 3.3 is

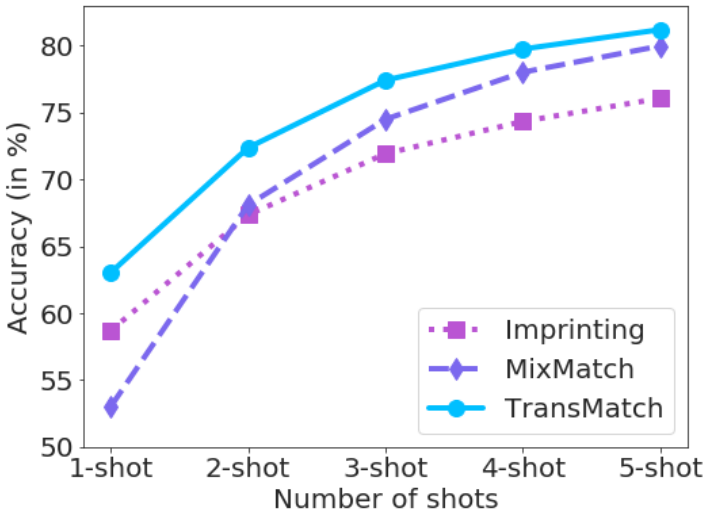

Figure 3.3: Comparison of Imprinting, MixMatch and our TransMatch both with 100 unlabeled images for 5-way classification with different number of shots on miniImageNet.

$\{11.02, 4.28, 2.92, 1.73, 1.22\}$ in $\{1, 2, 3, 4, 5\}$-shot setting. This is reasonable and worth attention as fewer shots means fewer labeled examples, which makes fine-tuning more difficult. Therefore, the importance of weight imprinting to give the classifier good initial weights becomes more evident.

Comparing different semi-supervised learning methods: In addition to MixMatch (Berthelot et al., 2019), in this section, we also compare with other semi-supervised learning methods (*i.e.*, Pseudo-Label (Lee, 2013)) in order to understand the influence the semi-supervised learning module. The results, showed in Table 3.3, are consistent with our observations when using MixMatch as semi-supervised learning module. Since Pseudo-Label is worse than MixMatch, the overall performance of our method using

Distractor	Method	1-shot	5-shot
—	Imprinting	58.68 ± 0.81	76.06 ± 0.59
1-class	MixMatch	50.14 ± 1.06	79.32 ± 0.63
	TransMatch	$\mathbf{62.32 \pm 1.04}$	$\mathbf{80.28 \pm 0.62}$
2-class	MixMatch	50.68 ± 1.15	78.07 ± 0.69
	TransMatch	$\mathbf{60.41 \pm 1.02}$	$\mathbf{79.48 \pm 0.64}$
3-class	MixMatch	49.48 ± 1.16	77.48 ± 0.66
	TransMatch	$\mathbf{59.32 \pm 1.10}$	$\mathbf{79.29 \pm 0.62}$

Table 3.4: Accuracy (in %) of MixMatch and our TransMatch with 100 unlabeled images from $\{1, 2, 3\}$ *distractor* classes on miniImageNet. Note that Imprinting does not use any unlabeled image.

Pseudo-Label is also worse than using MixMatch.

Influence of distractor classes: In typical semi-supervised learning, unlabeled images come from the same classes for the labeled images. This may not reflect realistic situations in real-world application. So we also study the influence of distractor classes, and report the results of Imprinting, MixMatch, and our TransMatch when there are unlabeled images from various distractor classes. In our experiments, distractor classes are randomly chosen from the remaining classes which are disjoint with the novel classes during test. The results are showed in Table 3.4. We can observe that all the results for MixMatch degrade due to the distractor classes, while our TransMatch still outperforms Imprinting in all cases.

3.4.2 Experiments on CUB-200-2011

Dataset configuration: The CUB-200-2011 dataset (CUB) is originally proposed by Wah et al. (2011) and contains 200 fine-grained classes of

birds with 11,788 images in total (about 30 images per class for support images and 30 images per class for query images). We strictly follow the setup in (Qi et al., 2018) to ensure a fair comparison. In particular, we use the standard train/test split provided by the dataset, and treat the first 100 classes as the base classes $\mathcal{C}_{base}$ and the remaining 100 classes as the novel classes $\mathcal{C}_{novel}$. Therefore, we have $N = 100$. We use all the training examples from the base classes for large scale pre-training to obtain the base model f^{base} and use the few-shot examples from the novel classes to train f^{novel}. In the experiment, we set K to $\{1, 2, 5, 10, 20\}$ and use the rest images $\{29, 28, 25, 20, 10\}$ as unlabeled images for support images. All the remaining 30 images are still used for query images.

Implementation details: We are interested in performance of our Trans-Match on the 100 novel classes, *i.e.*, the *transfer-learning* setting in (Qi et al., 2018). In order to ensure fair comparison, we follow Qi et al. (2018) and use Inception_v1 as our network backbone. We set the dimension of the fully connected embedding layer to 256, followed by an L2 normalization. We resize the input images to 256×256 and then randomly crop to 224×224. During the large scale pre-training stage, we set the initial learning rate to 0.001 and a $10\times$ multiplier for the embedding layer and classification layer. We reduce the learning rate by 0.1 after every 30 epochs, and train the model for a total of 90 epochs. During the fine-tuning stage, we set the number of batches to 64 for each epoch with a batch size of 64. By default, we set the weight decay to 0.0001, use a learning rate of 0.001, and

Model K=	1	2	5	10	20
Imprinting	26.08	34.13	43.34	48.91	52.94
Imprinting+FT	26.59	34.33	49.39	61.65	70.07
MixMatch	22.93	30.24	56.41	67.13	73.00
TransMatch	**28.02**	**38.05**	**59.83**	**68.60**	**74.61**

Table 3.5: Accuracy (in %) comparison on CUB-200-2011. Best results are in bold.

train the model for 100 epochs. For the extreme case of 1-shot and 2-shot settings (100-way), we set the weight decay to 0.04, the learning rate to 0.0001 and early stopping at 10 epochs in order to avoid overfitting.

Results on CUB-200-2011: We follow Qi et al. (2018) to report the results of their proposed Imprinting, and Imprinting+FT. Then we evaluate the performance of our proposed TransMatch using different numbers of shots and unlabeled images. We compare TransMatch with Imprinting and MaxMatch in Table 3.5, and the results show our proposed TransMatch achieves the best result which demonstrates its effectiveness in utilizing auxiliary labeled base-class data and unlabeled novel-class data. Table 3.6 shows the results of our TransMatch using different numbers of unlabeled images, and we can observe that better performance can be achieved with more unlabeled data. These results are similar to the results on miniImageNet dataset.

Model	# unlabeled	5-shot	10-shot
Imprinting (Qi et al., 2018)	—	43.34	48.91
Imprinting+FT (Qi et al., 2018)	0	49.39	61.65
TransMatch	5	52.90	63.79
TransMatch	10	54.78	66.21
TransMatch	15	56.86	67.71
TransMatch	20	59.25	68.60

Table 3.6: Accuracy (in %) comparison using different numbers of unlabeled images on CUB-200-2011.

3.5 Conclusion

While almost all existing semi-supervised few-shot learning methods are based on the meta-learning framework, we propose a new transfer-learning framework for semi-supervised few-shot learning to effectively explore the information from labeled base-class data and unlabeled novel-class data. We develop a new method under the proposed framework by incorporating the state-of-the-art semi-supervised and few-shot learning methods, leading to a new method called TransMatch. Extensive experiments on two popular few-shot learning datasets show that our proposed TransMatch achieves the state-of-the-art results, which demonstrate its effectiveness in utilizing both the labeled base-class data and unlabeled novel-class data.

4 SIMPLE POST-HOC WORK CAN IMPROVE SUPERVISED AND SEMI-SUPERVISED FEW-SHOT LEARNING

4.1 Introduction

Although few-shot learning (FSL) has seen noticeable progress in recent years. However, many methods require complicated setups and extensive computational resources. This is exacerbated in semi-supervised few-shot learning, where the goal is to utilize additional unlabeled data.

For supervised FSL, most of work focuses on adopting meta-learning framework where the feature extractor needs to train from scratch on base-class data to adapt for novel-class data, which causes inconvenience in practice. Only a small number of methods consider training a feature extractor on base-class data using a conventional supervised approach and consider how to adapt the regular feature extractor for novel-class data. Recently, a feature-level based method called Distribution Calibration (Yang et al., 2021) achieved new state-of-the-art results by generating simulated features for novel-class data, demonstrating the effectiveness of this group of methods.

Semi-supervised FSL is another emerging topic that gained momentum in recent years. Methods that utilize unlabeled data for FSL in a semi-supervised context include designing additional modules or use graph structures in a meta-learning setting to incorporate the unlabeled data (Ren

et al., 2018; Li et al., 2019b; Liu et al., 2019; Kim et al., 2019; Yang et al., 2020). This shows that utilizing unlabeled data in FSL usually relies on meta-learning based modules. However, the recent TransMatch (Yu et al., 2020) method showed that it can efficiently employ unlabeled data in a transfer-learning approach by weight imprinting. However, TransMatch still requires fine-tuning the backbone to absorb unlabeled data, which is computationally expensive.

To design a simple and practical approach to FSL, we formulate post-hoc FSL with the following two conditions: (1) the feature extractor is pre-trained on base-class data in a regular supervised fashion, and (2) the feature extractor is not fine-tuned using novel-class samples. Furthermore, we raise the question, *can we achieve state-of-the-art performance in supervised and semi-supervised FSL in this simple way?*

Currently, none of the aforementioned semi-supervised FSL methods meet the two conditions for post-hoc FSL defined above, which are very important for the application of few-shot data in practice. Firstly, many state-of-the-art models for computer vision are trained as regular classifiers. Re-training a model designed for few-shot data is complicated and computationally expensive. Secondly, generating features from a fixed feature extractor – for example, a computer vision model that has been pre-trained on a large dataset and is publicly available – is cheap but further fine-tuning is expensive and sometimes not feasible, which limits the adoption of FSL in many application areas such as biomedical research

and medical imaging. We define FSL based on regularly pre-trained feature extractors that do not require fine-tuning as *post-hoc* FSL. We believe post-hoc FSL is simple, fast, useful in practice.

In this paper, for supervised FSL, we propose a simple post-hoc method called TOL (Transformation of features with One-vs.-rest Logistic regression). Specifically, we transform the N-way classification problem into a multiple binary classification problem, and the baseline multi-class logistic regression model is replaced with a one-vs.-rest (OvR) logistic regression model. Then, the extracted features are transformed by power transformation and standardization to improve the performance further. Our simple TOL method does not require any complicated adaptation for few-shot data, and it shows a substantial performance improvement compared to baseline models, achieving state-of-the-art results for supervised FSL.

For semi-supervised FSL, we develop a method called DCP (Distribution Calibration for the Post-selection of unlabeled data). Here, we first generate pseudo-labels for unlabeled data using a trained classifier based on the Distribution Calibration method (Yang et al., 2021) designed for supervised FSL. Then, we select the unlabeled data with a sharp prediction distribution (we call it post-selection). By adopting Distribution Calibration again for unlabeled data, we compute nearest base-class statistics for extracting features based on these unlabeled samples. Finally, the generated features, which are based on the selected unlabeled data and the original labeled data, are used to train the final classifier. Our simple

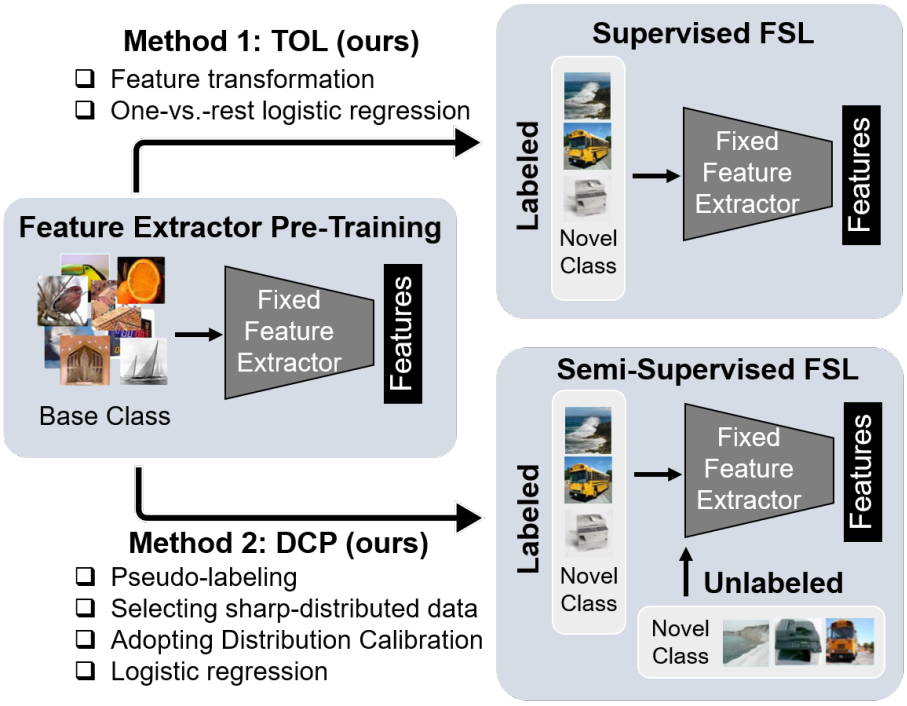

Figure 4.1: Illustration of post-hoc FSL with a regular-trained fixed feature extractor trained on base classes and our TOL and DCP methods applied to novel classes.

post-hoc method achieves new state-of-the-art results in semi-supervised FSL. To the best of our knowledge, we are among the first to develop a post-hoc method for semi-supervised FSL.

Figure 4.1 summarizes our methods and our contributions are four-fold:

1. We define the concept of post-hoc FSL, which is based on pre-trained feature extractors that do not require fine-tuning for novel-class data. It is simple and useful for practical applications of FSL.

2. We propose the TOL method for supervised post-hoc FSL that recasts the FSL multi-class setting as a multiple binary classification problem with transformed features.

3. We develop the DCP method for semi-supervised post-hoc FSL, which adopts Distribution Calibration (Yang et al., 2021) with a post-selection scheme to leverage unlabeled data effectively. We further show post-selection alone is not sufficient for achieving good performance in a semi-supervised FSL setting, and Distribution Calibration is an essential component in our method.

4. Extensive experiments on two popular benchmark datasets demonstrate the simplicity and effectiveness of our proposed post-hoc methods, TOL and DCP for supervised and semi-supervised FSL.

4.2 Related Work

4.2.1 Supervised Few-Shot Learning

Meta-learning methods are the most influential methods for FSL. Episodic training strategies are used for training the feature extractor to mimic the test task scenario where few-shot images are provided. Meta-learning can be grouped into several subcategories. Metric-based methods focus on learning similarities between support and query images (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018; Li et al., 2019a). Optimization-

based methods aim to develop optimization algorithms to quickly adapt parameters from base classes to novel classes (Finn et al., 2017; Nichol et al., 2018; Rusu et al., 2019; Lee et al., 2019). Graph-based methods utilize graph structures to learn the relationship between supports and queries (Garcia and Bruna, 2017; Liu et al., 2019; Kim et al., 2019; Yang et al., 2020). These meta-learning methods are designed for few-shot tasks, where the feature extractor is not trained for conventional image understanding.

While transfer-learning based methods constitute a small subset of FSL, these methods have recently gained more momentum due to the simplicity compared to meta-learning. Unlike meta-learning, transfer-learning based FSL methods train the feature extractor on base-class data in a conventional supervised fashion. The pre-trained models are then directly adapted to novel-class data (Qi et al., 2018; Qiao et al., 2018; Wang et al., 2019; Mangla et al., 2020; Chen et al., 2019; Yang et al., 2021). Some recent works also extend to a transductive setting by utilizing the features of queries (Guneet S. Dhillon, 2020; Jinlu Liu and Qin, 2020). Most transfer-learning methods do not add specific modules for few-shot tasks. Consequently, a traditional pre-trained image classification model can be used without modification in those few-shot settings. The recently proposed Distribution Calibration (Yang et al., 2021) uses nearest base-class features to generate samples for novel-class features, which resulted in an astonishingly good performance. Our work shows that a simple and fast transformation of the classification problem and features can also achieve

state-of-the-art results.

4.2.2 Semi-supervised Few-shot Learning

Semi-supervised FSL aims to utilize unlabeled data for FSL (Ren et al., 2018). However, traditional semi-supervised methods are hard to adapt to few-shot settings (Miyato et al., 2018; Berthelot et al., 2019; Tarvainen and Valpola, 2017). Hence, Ren et al. (2018) developed a masked soft K-means method for unlabeled samples based on ProtoNet. Another method, LST (Li et al., 2019b), used a self-training strategy to adapt to cherry-picked unlabeled samples. While graph-based methods (Liu et al., 2019; Yang et al., 2020; Kim et al., 2019) can incorporate unlabeled data naturally into the graph structure, these methods are based on meta-learning, which is not compatible with our desire to develop a simple post-hoc method for utilizing unlabeled data.

TransMatch (Yu et al., 2020) utilizes a transfer-learning framework and is successful in combining it with a rather complex semi-supervised method called MixMatch (Berthelot et al., 2019), which can be directly employed for unlabeled data. However, the entire backbone architecture needs fine-tuning when using MixMatch, which is very compute-intensive compared to our simple post-hoc method for semi-supervised FSL. Although there are several post-hoc methods for supervised FSL, no notable work has used a simple post-hoc method for unlabeled data, indicating the gap between *regular* semi-supervised learning and semi-supervised

FSL. In this paper, we propose a method based on pseudo-labeling (Lee, 2013) and the recent Distribution Calibration method (Yang et al., 2021) where we build supportive samples from post-selected unlabeled data. Unlabeled data can be leveraged well by this simple post-hoc method. We also show that post-selection alone is not the panacea for semi-supervised post-hoc FSL and requires more thoughtful design choices such as using Distribution Calibration.

4.3 Problem Definition

Suppose $\mathcal{D}_{base}$ represents a relatively large dataset that contains many-shot labeled samples from base classes $\mathcal{C}_{base}$. Furthermore, $\mathcal{D}_{novel}$ represents a relatively small dataset containing few-shot labeled samples from novel classes $\mathcal{C}_{novel}$, where $\mathcal{C}_{novel} \bigcap \mathcal{C}_{base} = \emptyset$. For supervised FSL, a labeled support dataset of N classes from $\mathcal{C}_{novel}$ with K images per class is sampled from $\mathcal{D}_{novel}$. The task is to learn a classifier for these N classes based on this support dataset, which is denoted as a N-way K-shot problem. For semi-supervised FSL, an extra number of unlabeled images U per class are sampled from these N classes and provide additional information for training the classifier.

The setting of our practitioner-friendly and cost-effective post-hoc method is defined as follows:

1. A general feature extractor is trained on $\mathcal{D}_{base}$ for image classification

in a conventional fashion, such that it lends itself to a "quick & easy" adoption for FSL in practice.

2. After training on $\mathcal{D}_{base}$, the feature extractor remains fixed when it is applied to novel-class images in $\mathcal{D}_{novel}$. The rationale behind this design decision is that fine-tuning a complex neural network is computationally expensive, while training a classifier on the extracted features is simple and cheap.

The feature extractor outlined above produces feature vectors from images in $\mathcal{D}_{base}$ and $\mathcal{D}_{novel}$ wherein the post-hoc setting these features remain fixed. Regarding the first condition of the post-hoc setting outlined above, the feature extractor is trained as wide residual network (*i.e.*, WRN-28-10) (Zagoruyko and Komodakis, 2016) in a conventional way for image understanding as used in (Mangla et al., 2020) and the Distribution Calibration method by Yang et al. (2021). The model is trained on all base-class images, and a self-supervised technique, predicting image rotations (Spyros Gidaris, 2018), is employed for generating better features. Note that the self-supervised technique is not specifically designed for FSL. Since the feature extractor needs to remain fixed and no further fine-tuning is allowed in the post-hoc setting, we only keep the extracted features (represented as a 640-dimensional vector) for all base-class and all novel-class images. All of the following methods are built based on the extracted features following Yang et al. (2021).

4.4 Our Post-Hoc Method for Labeled Data

In this section, we first recap the recent Distribution Calibration (Yang et al., 2021) method, which is used for comparison with our supervised method and employed in our semi-supervised setting which will be covered in Section 4.5. Then, we introduce our proposed simple post-hoc method TOL, which combines feature transformation with one-vs.-rest logistic regression. Note that since Distribution Calibration uses logistic regression as a baseline due to its simplicity, our post-processing work for labeled data is also based on logistic regression to allow for a fair comparison. TOL contains two modules that can be summarized as follows: (1) transform the regular multi-class (multinomial) logistic regression to one-vs.-rest binary logistic regression; (2) apply a power transformation and standardization to the extracted features.

4.4.1 Regular Multi-Class Logistic Regression

In regular logistic regression for multi-class data, for each few-shot dataset containing N-way K-shot labeled samples $\mathcal{D}_l$, the prediction $\hat{y}$ of each sample $\mathbf{x}$ is calculated via the softmax function: $\hat{y} = \frac{\exp(\mathbf{a})}{\sum_{j=1}^{N}\exp(\mathbf{a}_j)}$, where $\mathbf{a} = \mathbf{W}^T\mathbf{x} + \mathbf{b}$ and $\hat{y}, \mathbf{a} \in \mathcal{R}^N$. The loss is computed as the multi-category cross-entropy loss:

$$\mathcal{L} = \sum_{\mathbf{x},y\in\mathcal{D}_l}\sum_{j=1}^{N} -y_j \log(\hat{y}_j), \tag{4.1}$$

where y is the true one-hot label corresponding to $\mathbf{x}$. The multinomial model is used as a performance baseline.

4.4.2 Recap of Distribution Calibration

Distribution Calibration (Yang et al., 2021) aims to generate samples from nearest base-class features for the labeled few-shot support data, which establishes a bridge between few-shot learning and many-shot learning. Specifically, for $\mathcal{D}_{base}$, $\boldsymbol{\mu}_i$ denotes the mean of the extracted features corresponding to the i-th base class:

$$\boldsymbol{\mu}_i = \frac{\sum_{j=1}^{s_i} \mathbf{x}_j}{s_i}. \tag{4.2}$$

The covariance matrix $\boldsymbol{\Sigma}_i$ for the i-th base class is defined as follows:

$$\boldsymbol{\Sigma}_i = \frac{1}{s_i - 1} \sum_{j=1}^{s_i} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T, \tag{4.3}$$

where $\mathbf{x}_j$ is the j-th sample in the i-th base class and s_i the total number of samples in the i-th base class.

In a N-way K-shot dataset $\mathcal{D}_l$ on novel classes $\mathcal{C}_{novel}$, the features for each sample are first transformed to make it more normal-distributed via

$$\tilde{\mathbf{x}} = \mathbf{x}^{\lambda}. \tag{4.4}$$

After that, k nearest base classes $\mathcal{C}_{near}$ with respect to $\tilde{\mathbf{x}}$ are selected by

the distance $||\tilde{\mathbf{x}} - \boldsymbol{\mu}_i||^2$.

A calibrated mean and a calibrated covariance matrix are computed based on the novel-class sample and the corresponding k-nearest base classes as follows:

$$\boldsymbol{\mu}^* = \frac{\sum_{i \in \mathcal{C}_{near}} \boldsymbol{\mu}_i + \tilde{\mathbf{x}}}{k+1}, \tag{4.5}$$

$$\boldsymbol{\Sigma}^* = \frac{\sum_{i \in \mathcal{C}_{near}} \boldsymbol{\Sigma}_i}{k} + \alpha, \tag{4.6}$$

where k and α are both hyperparameters. The total number of $\frac{S}{K}$ 'similar' samples are sampled from the following Normal distribution: $\mathcal{N}(\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*)$, where S is a hyperparameter for the desired number of generated samples in one class.

Then, for each sample $\mathbf{x}$, there is a corresponding generated dataset $\mathcal{D}_{\mathbf{x}}$. The whole generated dataset for all samples in $\mathcal{D}_l$ is $\mathcal{D}_{DC} = \bigcup_{\mathbf{x} \in \mathcal{D}_l} \mathcal{D}_{\mathbf{x}}$. $\mathcal{D}_{DC}$ is then used together with the original few-shot samples to train the classifier.

4.4.3 Our Proposed TOL

Our simple TOL method relies on the transformation of features and one-vs.-rest (OvR) logistic regression, which we describe in this section.

First, we consider transforming the few-shot multi-class problem to a binary classification problem, where multiple classifiers are built in an OvR (Bishop, 2006) fashion. In the N-way K-shot task $\mathcal{D}_l$, for j-th novel

class, we consider a binary logistic regression based on all the samples from class j (positive samples) and all samples in the remaining classes (negative samples). Then the probability of one given sample's feature $\mathbf{x}$ belonging to the j-th class is calculated as

$$p_j = \text{Sigmoid}(\mathbf{W}_j^T \mathbf{x} + \mathbf{b}_j). \tag{4.7}$$

We build N binary logistic regression for all the novel classes in this task and compute the membership probability for each class as $\{p_1, p_2, \ldots, p_N\}$. Then the final prediction of each class $\hat{y}$ is normalized across all p_j:

$$\hat{y} = \frac{p_j}{\sum_{j=1}^{N} p_j}. \tag{4.8}$$

When building multiple classifiers, we use the optimization algorithm Liblinear (Fan et al., 2008), which is a coordinate descent algorithm designed for an OvR setting.

In addition to the transformation of the classification problem, we also consider the transformation of the extracted features that serve as input to the OvR model. First, a power transformation is applied to the features (Eq. 4.4). Next, we standardize the features. Considering the features from the last step of power transformation $\tilde{\mathbf{x}} = (x_1, \ldots, x_{640})$, the standardized features are computed by $\frac{\tilde{x}-\nu}{\sigma}$, where $\nu = \frac{\sum_{i=1}^{640} x_i}{640}$. Then, the standard deviation σ of the feature vector is computed as $\sqrt{\frac{\sum_{i=1}^{640}(x_i-\nu)^2}{639}}$. The transformed data is used for training the aforementioned OvR regression

model.

4.4.4 Experiments

We show that using an OvR logistic classifier, we can achieve high performance in a rather simple way: without requiring expensive data processing procedures, we simply feed novel data to the classifier, which is then able to produce predictions with high accuracy. This is a phenomenon that is largely ignored in FSL literature.

We compare our proposed TOL with other state-of-the-art methods on two popular few-shot datasets: miniImageNet (Vinyals et al., 2016) and CUB-200-2011 (Wah et al., 2011).

miniImageNet is a simplified version of ILSVRC 2015 (Russakovsky et al., 2015) that is widely used in FSL. It contains 60,000 color images of size 84×84 from 100 classes with 600 images per class. Following the common practice, we use the same split of 64 base classes, 16 validation classes, and 20 novel classes as described in (Ravi and Larochelle, 2017).

CUB-200-2011 (Wah et al., 2011) is a fine-grained dataset for few-shot bird classification. It contains 11,788 color images of size 84×84 from 200 classes. We use the same split of 100 base classes, 50 validation classes, and 50 novel classes as (Hilliard et al., 2018).

Implementation details. We use the same features for all base-class images and novel-class images to enable a fair comparison with the official

Table 4.1: Accuracy (in %) on miniImageNet and CUB with 95% confidence interval on supervised FSL. Best results are highlighted in bold.

Method	*mini*ImageNet		CUB	
	1-shot	5-shot	1-shot	5-shot
MAML (2017)	48.70±1.84	55.31±0.73	55.92±0.95	72.09±0.76
ProtoNet (2017)	49.42±0.78	68.20±0.66	51.31±0.91	70.77±0.69
Matching Net (2016)	43.56±0.84	55.31±0.73	61.16±0.89	72.86±0.70
RelationNet (2018)	50.44±0.82	65.32±0.70	62.45±0.98	76.11±0.69
DN4 (2019a)	51.24±0.74	71.02±0.64	53.15±0.84	81.90±0.60
Imprinting (2018)	58.68±0.81	76.06±0.59	–	–
LEO (2019)	61.76±0.08	77.59±0.12	–	–
MetaOptNet (2019)	62.64±0.64	78.63±0.46	–	–
Baseline++ (2019)	51.87±0.77	75.68±0.63	67.02±0.90	83.58±0.54
Neg-Cosine (2020a)	63.85±0.81	81.57±0.56	72.66±0.85	89.40±0.43
SimpleShot (2019)	64.29±0.20	81.50±0.14	70.28 ± –	86.37± –
TOL (ours)	**68.30±0.81**	**83.47±0.54**	**80.61±0.79**	**91.26±0.44**

implementation of Distribution Calibration (Yang et al., 2021). The power transformation coefficient λ in TOL is set to 0.75, and the coefficient for generated samples S it set to 750. Each random test experiment contains N-way K-shot labeled images from novel classes with Q query images per class, where $N = 5$, $K = \{1, 5\}$, and $Q = 15$. Following common practice (Snell et al., 2017), we report the mean accuracy and the 95% confidence interval from 600 random experiments.

Main Results.

Table 4.1 shows the comparison of our proposed simple TOL with other state-of-the-art methods on the two popular benchmark datasets. We observe that our methods outperform others by a large margin, demon-

strating the effectiveness of our simple method. Meanwhile, the running time for 600 experiments for DC on miniImageNet (1-shot) is 773 seconds, whereas the running time for TOL is 11 seconds.

Ablation Study.

We conduct an ablation study for our method since it contains both the transformation of the classification problem and the transformation of the features. The results are reported in Table 4.2. We also compare TOL with the popular cosine classifier (Chen et al., 2019; Mangla et al., 2020), which is reported by Mangla et al. (2020) using the same feature extractor we used for TOL. We observe that replacing multi-class logistic regression with multiple binary logistic regression results in a remarkable accuracy improvement of 6.75% (*resp.* 2.03%) for 1-shot (*resp.* 5-shot) on miniImageNet. Similarly, we can observe an improvement of 6.39% (*resp.* 0.66%) accuracy for 1-shot (*resp.* 5-shot) on CUB.

When adding simple feature transformation, the results are further improved, especially on miniImageNet. We see the combination of the two types of transformation may contribute differently to the FSL performance, but they have both positive effects in general.

The recently proposed state-of-the-art Distribution Calibration (Yang et al., 2021) method also falls into the category of post-hoc work. However, it works in a more complicated way as it requires the information from base classes to generate features for the logistic regression classifier. Our simple

Table 4.2: Ablation study for TOL on miniImageNet and CUB. Results are shown as accuracy in %. Best results and the results which fall into the best results' 95% confidence intervals are shown in bold. Our OvR regression outperforms the baseline multi-class regression and the popular cosine classifier by a large margin. Feature transformation further improves the results.

Method	*mini*ImageNet		CUB	
	1-shot	5-shot	1-shot	5-shot
Cosine classifier (2020)	63.90±0.18	81.03±0.11	77.61±0.86	89.32±0.46
Multi-class logistic	59.42±0.87	81.44±0.57	73.81±0.89	90.44±0.43
OvR logistic	66.17±0.78	83.47±0.53	**80.20±0.80**	91.10±0.44
+ Power transformation	66.72±0.79	**84.50±0.52**	**80.57±0.79**	**91.62±0.43**
+ Standardization (TOL)	**68.30±0.79**	83.47±0.54	**80.61±0.79**	**91.26±0.44**
Distri. Calibration (2021)	**68.26±0.81**	83.42±0.55	**80.62±0.82**	90.81±0.45

method shows comparable performance with Distribution Calibration in a 1-shot setting and even outperforms it in a 5-shot setting. Note that the results of Distribution Calibration reported in Table 4.2 are based on the official code that we ran on the same 600 experiments for a fair comparison.

Influence of the Number of Shots.

We compare the performance of baseline multi-class and OvR logistic regression on different shots without feature transformation on both the miniImageNet and CUB dataset as shown in Figure 4.2. The results show that the performance gain from using OvR regression over multi-class regression increases when there are fewer shots. When there are many shots, their performance is similar. This demonstrates that the transformation of the classifier is particularly useful for the few-shot setting.

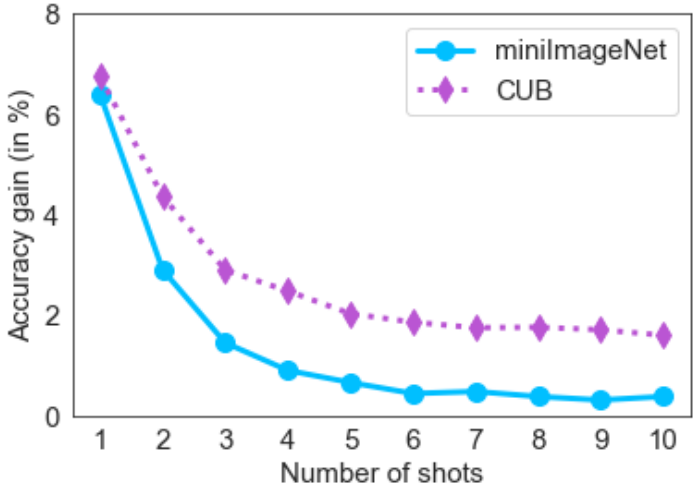

Figure 4.2: Improvement of OvR logistic regression over multi-class logistic regression for different shots on miniImageNet and CUB. We clearly see this OvR regression performs better especially when there are fewer shots.

Table 4.3: Ablation study for different power-transformation coefficients for TOL on miniImageNet. Results are shown as accuracy in %.

Power	0.65	0.75	0.85	0.95
TOL	68.15±0.78	68.30±0.78	68.27±0.78	68.11±0.78

Influence of the Power-Transformation

We follow a similar step of (Yang et al., 2021) to show the impact of different power-transformation coefficients on our TOL. The results are shown in Table 4.3. It shows our methods are not sensitive to the coefficients.

4.5 Our Post-Hoc Method for Unlabeled Data

In this section, we first point out why post-hoc work is challenging for semi-supervised FSL. Then, we introduce a simple post-selection strategy. In essence, we employ Distribution Calibration for generating the post-selection of unlabeled data (called DCP). The newly generated data improves performance in a semi-supervised few-shot setting by a large margin. However, we then show this post-selection strategy itself is still not the panacea for semi-supervised few-shot problems due to its few-data nature.

4.5.1 Utilizing Unlabeled Data

When utilizing unlabeled data, most of the existing work relies on meta-learning, where the whole network needs to be updated for unlabeled data and trained from scratch on base-class images as mentioned previously. For methods based on transfer learning, where the feature extractor does not need to train from scratch on base-class images, the extractor still needs to be fine-tuned on novel-class unlabeled data, increasing the computational cost by a large margin like TransMatch (Yu et al., 2020). Meanwhile, we should note that many state-of-the-art semi-supervised methods rely on the image data itself and employ image consistency-based methods (Berthelot et al., 2019; Miyato et al., 2018) that are not suitable for post-hoc work for fixed features.

4.5.2 Post-selection

As pseudo-labeling appears to be a reasonable strategy for our post-hoc work, we consider selecting data corresponding to highly confident predictions for re-training the classifier. We call this method *post-selection*. This technique shares some similarities with LST (Li et al., 2019b). However, the cherry-picking procedure in LST needs to be learned during meta-training, where the feature extractor needs to be trained from scratch. In contrast, our post-selection method is consistent with the simple post-hoc framework we consider in this work.

Specifically, in our proposed post-selection method, consider a semi-supervised few-shot task $\mathcal{T}_{semi}$. Besides the labeled N-way K-shot dataset $\mathcal{D}_l = \{(\mathbf{x}_{nk}, y_n) | n = 1, \cdots, N; k = 1, \cdots, K\}$ with $y_n \in \mathcal{C}_{novel}$, we are also given an unlabeled dataset $\mathcal{D}_u = \{\mathbf{x}_{uk} | u = 1, \cdots, U; k = 1, \cdots, K\}$. After training the classifier (for Distribution Calibration, TOL, or multi-class logistic) for $\mathcal{D}_l$, we apply the classifier to $\mathcal{D}_u$ to obtain the predictions of each $\mathbf{x}_{uk}$, denoted as $\hat{y}_{nk} \in \mathcal{R}^N$. In addition, we apply a temperature T (Goodfellow et al., 2016) to make the prediction distribution sharper when necessary:

$$\tilde{y}_{nk} = \hat{y}_{nk}^{1/T} / \sum_{i=1}^{N} (\hat{y}_{nk})_i^{1/T}. \tag{4.9}$$

We select the $\mathbf{x}_{uk}$, whose prediction has a value greater than a threshold H, as a hyperparameter, such that the post-selected unlabeled dataset $\mathcal{D}_{pick}$

is constructed as

$$\mathcal{D}_{pick} = \{(\mathbf{x}_p, \hat{y}_p) | \max_{i=1}^{N} (\hat{y}_p)_i > H\}. \tag{4.10}$$

Then $\mathcal{D}_{pick} \bigcup \mathcal{D}_l$ are used for retraining the classifier.

4.5.3 Our Proposed DCP

We propose using Distribution Calibration (Section 4.4.2) for post-selected unlabeled data, which we refer to as DCP. After obtaining $\mathcal{D}_{pick}$, the data is first transformed by a power transformation in (Eq. 4.4). Then we use the post-selected unlabeled data to find the k nearest base classes and obtain a calibrated mean $\boldsymbol{\mu}_p^*$ as well as a calibrated covariance matrix $\boldsymbol{\Sigma}_p^*$. This is similar to Distribution Calibration for labeled data stated in Eq. 4.5 and Eq. 4.6.

Suppose that for the post-selected unlabeled samples, the number of predictions for each class is denoted as $\{K_1, K_2, \ldots, K_N\}$. Then a generated dataset $\mathcal{D}_{x_p}$ for $\mathbf{x}_p$ with j-th novel class prediction of size $\frac{S}{K_j}$ is sampled from the normal distribution $\mathcal{N}(\boldsymbol{\mu}_p^*, \boldsymbol{\Sigma}_p^*)$ where S is a hyperparameter as mentioned in Section 4.4.2. Note that if $K_j = 0$, then $\mathcal{D}_{x_p} = \emptyset$.

The complete dataset generated from post-selected unlabeled data is denoted as

$$\mathcal{D}_{u,DC} = \bigcup_{\mathbf{x}_p \in \mathcal{D}_{pick}} \mathcal{D}_{x_p}. \tag{4.11}$$

Then, both $\mathcal{D}_{u,DC}$, $\mathcal{D}_l$ and $\mathcal{D}_{DC}$ (mentioned in Section 4.4.2) are used for re-training a multi-class logistic regression model.

4.5.4 Experiments

To evaluate the proposed methods for semi-supervised FSL, we use both miniImageNet and CUB. For DCP, we set the post-selection threshold H to 0.9, temperature T to 1.0, power-transformation coefficient λ to 0.5, nearest number k to 0.2, calibration coefficient α to 0.21 (following (Yang et al., 2021)), and S to 750. When post-selection is combined with TOL or multi-class logistic regression, we change the temperature T to 0.005 and set the threshold to 0.7 since both methods return a flatter distribution for unlabeled data compared to DCP. Each random test experiment contains N-way K-shot labeled images, U unlabeled images, and Q query images per class from the selected N novel classes, where $N = 5$, $K = \{1, 5\}$, $Q = 15$. We report the average accuracy and the 95% confidence interval from 600 random experiments.

Main Results.

Table 4.4 summarizes the performance of our proposed DCP on mini-ImageNet. In a 1-shot setting, our method exceeds the accuracy of the second-place method LST by 5.11%. Similarly, our method exceeds the second-place TransMatch by a large margin (6.45%) in a 5-shot setting. These results show that even though our method has a substantially lower

Table 4.4: Accuracy (in %) on miniImageNet with 95% confidence interval. Best results are highlighted in bold. As a simple post-hoc method, DCP exceeds existing popular non post-hoc semi-supervised FSL methods a lot.

Method	Type	1-shot	5-shot
Soft k-Means (2018)	Non post-hoc	50.09 ± 0.45	64.59 ± 0.28
Soft k-Means+Cluster (2018)	Non post-hoc	49.03 ± 0.24	63.08 ± 0.18
Masked Soft k-Means (2018)	Non post-hoc	50.41 ± 0.31	64.39 ± 0.24
TPN-semi (2019)	Non post-hoc	52.78 ± 0.27	66.42 ± 0.21
TransMatch (2020)	Non post-hoc	63.02 ± 1.07	81.19 ± 0.59
LST (2019b)	Non post-hoc	70.10 ± 1.90	78.70 ± 0.80
TPN with MTL (2019b)	Non post-hoc	$62.70 \pm$ –	$74.2 \pm$ –
DCP (Ours)	Post-hoc	$\mathbf{75.21 \pm 0.90}$	$\mathbf{87.64 \pm 0.49}$

computational complexity, it has a substantially better predictive performance.

Influence of Unlabeled Images.

To investigate the influence of the number of unlabeled data, we report the results from using different numbers of unlabeled images during testing for {15, 30, 50, 75, 100} on miniImageNet and {5, 10, 15, 20} CUB in Table 4.5. (Note that there are no such abundant images per class on CUB dataset so that the available unlabeled data is limited.) We can observe a clear pattern where the number of images is positively correlated with our method's performance while reaching saturation at around 100 unlabeled samples on miniImageNet. This clearly shows that generating 'synthetic' labeled samples by post-selecting unlabeled data is effective in utilizing abundant unlabeled data.

Table 4.5: Accuracy (in %) on miniImageNet and CUB with different number of unlabeled data with 95% confidence interval. Best results are highlighted in bold. This shows our DCP leverages unlabeled data very well.

Dataset	Method	#Unlabeled	1-shot	5-shot
miniImageNet	DC	0	67.64 ± 0.79	83.15 ± 0.54
	DCP	15	70.79 ± 0.92	85.71 ± 0.55
	DCP	30	72.70 ± 0.92	86.70 ± 0.52
	DCP	50	74.22 ± 0.88	87.40 ± 0.49
	DCP	75	$\mathbf{75.21 \pm 0.90}$	$\mathbf{87.64 \pm 0.48}$
	DCP	100	74.84 ± 0.83	87.44 ± 0.48
CUB-200-2011	DC	0	80.62 ± 0.79	90.81 ± 0.45
	DCP	5	80.83 ± 0.87	91.36 ± 0.43
	DCP	10	82.52 ± 0.87	91.85 ± 0.39
	DCP	15	82.96 ± 0.94	92.11 ± 0.41
	DCP	20	$\mathbf{83.43 \pm 0.86}$	$\mathbf{92.22 \pm 0.42}$

Table 4.6: Ablation study of combining post-selection with TOL or multi-class logistic regression on miniImageNet. The result show these methods cannot utilize unlabeled data well. The usage of unlabeled data even harms the performance in most cases.

Method	#Unlabeled	1-shot	5-shot
TOL	0	$\mathbf{68.30 \pm 0.79}$	$\mathbf{83.47 \pm 0.54}$
+post-selection	15	60.87 ± 1.00	77.68 ± 0.85
+post-selection	30	65.20 ± 0.99	80.56 ± 0.77
+post-selection	50	68.01 ± 0.99	82.86 ± 0.73
+post-selection	75	69.41 ± 0.94	83.50 ± 0.66
multi-class logistic	0	$\mathbf{59.42 \pm 0.87}$	$\mathbf{81.44 \pm 0.57}$
+post-selection	15	55.18 ± 1.01	80.61 ± 0.66
+post-selection	30	57.60 ± 0.99	81.73 ± 0.60
+post-selection	50	59.13 ± 0.97	82.57 ± 0.58
+post-selection	75	59.75 ± 0.97	82.87 ± 0.58

4.5.5 Is Post-selection the Panacea for Semi-Supervised FSL?

Post-selection does not require any further training of the feature extractor and does not require a model designed for unlabeled data in a meta-learning fashion. It would be interesting to see whether the use of post-selection alone can close the performance gap between semi-supervised learning and semi-supervised FSL.

To investigate this, we combined post-selection with our proposed TOL method or multi-class logistic regression; the results are shown in Table 4.6.

Even though TOL shows state-of-the-art performance and is comparable with Distribution Calibration in a supervised setting, simply adding post-selection to TOL is not working well. On the contrary, the post-selected unlabeled data is even harmful to the classifier. For example, the performance decreases 7.43% (*resp.* 5.79%) on 1-shot (*resp.* 5-shot) for TOL and decreases 4.24% (*resp.* 0.83%) on 1-shot (*resp.* 5-shot) for multi-class logistic regression. This underlines the complex challenges in few-shot semi-supervised learning. We also observe a similar phenomenon when combining multi-class logistic regression with post-selection. This, from another angle, demonstrates the significance of our proposed DCP in this area because the simple post-selection procedure itself is not the panacea for semi-supervised FSL.

Our results show that simple post-selection alone is insufficient for improving semi-supervised FSL. One possible explanation is that unlabeled data relies more on base-class information to provide more confident evidence for classification. Although we can achieve significant improvement by TOL for labeled data, unlabeled data still needs more confident support (generated samples from nearest base classes) to benefit the classifier.

4.6 Conclusion

Our paper focuses on a unique perspective for few-shot learning (FSL): How can we push the state-of-the-art performance in both supervised and semi-supervised FSL without fine-tuning a regularly trained feature-extractor for novel classes? We refer to the use of this simple technique as post-hoc work which is an easy and computationally effective way to solve few-shot problems in practice. For labeled data, we develop a post-hoc method, called TOL, based on binary classification and feature transformation. For unlabeled data, we propose a post-hoc method DCP based on post-selection and Distribution Calibration. Both TOL and DCP achieve state-of-the-art results in supervised and semi-supervised FSL, respectively, demonstrating the simplicity and efficacy of our methods.

5 FEW-SHOT LEARNING FOR VIDEO OBJECT DETECTION IN A TRANSFER-LEARNING SCHEME

5.1 Introduction

With the popularity of cameras in surveillance systems and mobile phones, as well as the mass adoption of social media content sharing platforms, there are more and more video content generated every day. Therefore, the need for developing algorithms to detect objects in videos grows rapidly in computer vision. Although it is possible to train a robust video object detector with sufficient labeled videos, powerful deep neural networks and abundant computational resources, collecting such a large number of videos with bounding box annotations is costly.

Humans can learn new concepts easily with only a few examples. Despite that deep learning has been successfully applied to many real-world applications, it usually suffers from the overfitting problem when there are only a few samples for new concepts. Few-shot learning, which tries to learn a robust model from only a few samples of a new concept, has thus attracted great attention recently (Qi et al., 2018; Vinyals et al., 2016; Finn et al., 2017; Snell et al., 2017; Rusu et al., 2019; Liu et al., 2019; Li et al., 2019a; Nichol et al., 2018; Kang et al., 2019; Karlinsky et al., 2019; Fan et al., 2020; Perez-Rua et al., 2020; Wang et al., 2020b; Cao et al., 2020; Zhu and Yang, 2018; Yu et al., 2020).

Most existing few-shot learning methods focus on either image classification (Qi et al., 2018; Finn et al., 2017; Qiao et al., 2018; Vinyals et al., 2016; Snell et al., 2017; Rusu et al., 2019; Liu et al., 2019; Li et al., 2019a; Nichol et al., 2018) or video classification (Cao et al., 2020; Zhu and Yang, 2018). While some recent few-shot learning methods (Kang et al., 2019; Karlinsky et al., 2019; Fan et al., 2020; Perez-Rua et al., 2020; Wang et al., 2020b) have investigated object detection, all of them focus on object detection in static images instead of videos. Different from static images, videos contain abundant spatial and temporal information of objects. Therefore, it becomes more imperative to design a model for video object detection given a few videos of novel-class objects. This poses a new problem of *few-shot learning for video object detection*.

Since videos contain more information than static images, using the spatial-temporal information in videos is critical to achieving good performance. Since it is computationally prohibitive to build episodes by representing a video by its frames, the meta-learning based few-shot learning methods designed for the image classification problem cannot be directly used to solve the few-shot video object detection problem. While some techniques target few-shot video classification (Cao et al., 2020; Zhu and Yang, 2018), it is different from video object detection, as video classification only aims to classify the entire video as one of the classes. In contrast, video object detection needs to detect both the presence and spatial-temporal location of an object in all frames in a video. This increases

difficulty exponentially in terms of both computing and realization. The recent finding (Wang et al., 2020b) that transfer-learning based methods can achieve good results on image object detection opens a possible path for solving the problem of few-shot video object detection.

In this paper, we study the new problem of *few-shot learning for video object detection*. In particular, given a few video clips of novel-class objects, we would like to build a robust object detector for novel-class objects. Realizing there is no prior work studying this, we curate a new benchmark dataset derived from the popular ImageNet-VID dataset (Russakovsky et al., 2015) for this problem. We design two types of base datasets: a strong dataset and a weak dataset to investigate the influence of the strength of the feature extractor for the video object detector. We employ the transfer-learning framework. Specifically, we first train a video object detector on the whole base dataset, which can aggregate the temporal information from other frames in the entire video based on the state-of-the-art method MEGA (Chen et al., 2020). After that, we fine-tune the cosine classifier and bounding box regressor in the RPN head. Based on different fine-tuning strategies, we consider two methods: Joint and Freeze. By analyzing the performance of the two methods on the curated benchmark datasets, we reveal the *insufficiency* and *overfitting* problems. In order to solve this issue, a simple but effective method called Thaw is naturally developed by our analysis to balance the insufficiency and overfitting problems effectively. Based on the evaluation of the curated benchmark dataset, we demonstrate

the effectiveness of our proposed method.

Our contributions are summarized as follows:

1) We propose a new paradigm of few-shot learning for video object detection. Specifically, we study how an object detector can be learned from a few videos of new concepts where abundant temporal information is available while maintaining good performance for existing classes.

2) We curate a dataset derived from the popular ImageNet-VID dataset (Russakovsky et al., 2015) for investigating this new problem. A strong base dataset and a weak base dataset are designed for further scenario analysis.

3) We propose a transfer-learning framework for solving this problem and investigate two methods under the framework: Joint and Freeze. We reveal two issues: *insufficiency* and *overfitting* based on a novel quantitative analysis.

4) We propose a simple method called Thaw to trade off the *insufficiency* and *overfitting* problems revealed by our analysis. Extensive experiments demonstrate that our proposed Thaw naturally motivated by our novel analysis can help the video object detector efficiently learn new concepts from a few novel-class videos and achieve promising performance.

5.2 Related Work

5.2.1 Few-Shot Learning

Few-shot learning methods can be categorized into meta-learning based methods and transfer-learning methods.

Meta-learning aims to learn a paradigm based on the base-class data with an episodic training strategy so that it can generalize to new tasks with only a few novel examples. Metric-based meta-learning methods learn a good distance metric from the few-shot examples of base classes (Vinyals et al., 2016; Snell et al., 2017; Sung et al., 2018; Li et al., 2019a). For example, Prototypical Network (Snell et al., 2017) measures the distance from queries and the prototypes of each class. DN4 (Li et al., 2019a) explores the local descriptors to measure similarities. Optimization-based meta-learning methods like MAML (Finn et al., 2017), LEO (Rusu et al., 2019) and Reptile (Nichol et al., 2018) aim to find a good optimization direction that converges faster to the optimal solution with fewer gradient steps. However, most meta-learning based methods are designed for image classification. It is challenging to directly extend them to the few-shot video object detection scenario in this paper.

Transfer-learning based methods focus on how to train a good base model from the large amount of base-class data and then adapt the model to novel classes with only a few-shot of samples (Qi et al., 2018; Qiao et al., 2018; Chen et al., 2019; Yu et al., 2020). Unlike meta-learning based methods,

the base model is trained using traditional methods and a new classifier is built with a frozen feature extractor. Cosine-similarity is usually used to build the new classifier given novel-class samples (Qi et al., 2018; Chen et al., 2019; Qiao et al., 2018). Chen et al. (2019) systematically analyzed the performance of building a cosine classifier when freezing the feature extractor and compared with popular meta-learning methods, demonstrating the effectiveness of transfer-learning based methods for few-shot learning. Again, most transfer-learning based methods are for image classification and have not been applied to the few-shot video object detection problem in this paper.

5.2.2 Object Detection

Object detection is a fundamental problem in computer vision and has been studied for decades with significant progress made in recent years due to the advancement of deep learning. Nowadays, CNN-based object detection methods have become the mainstream and can be divided into two main categories: 1) One-stage detection; and 2) two-stage detection. One-stage detection methods, such as YOLO (Redmon et al., 2016) and SSD (Liu et al., 2016), do not require region proposals and can predict the bounding box directly. Two-stage detection methods require region proposals and generally show better performance than one-stage detection methods. The representative methods include RCNN (Girshick et al., 2014), Fast-RCNN (Girshick, 2015), Faster-RCNN (Ren et al., 2015), and

Mask R-CNN (He et al., 2017). Recently, anchor-free methods have also been proposed, which can directly regress a bounding box from the location in the feature map (Zhou et al., 2019; Law and Deng, 2018; Tian et al., 2019). However, its application to few-shot learning is not sufficiently explored yet.

5.2.3 Few-Shot Image Object Detection

Few-shot image object detection is is still a developing area of research, with only a handful of notable works (Yan et al., 2019; Kang et al., 2019; Karlinsky et al., 2019; Fan et al., 2020; Perez-Rua et al., 2020; Wang et al., 2020b). Most of them are based on meta-learning. Meta-RCNN (Yan et al., 2019) proposed the meta-learning process for RoI features in Faster-RCNN (Ren et al., 2015). Feature Reweighting (Kang et al., 2019) developed a reweighting module and map features with corresponding classes in YOLOv2 (Redmon and Farhadi, 2017). Fan et al. (2020) proposed Attention RPN and Multi-Relation Head for matching classes. Recently, Wang et al. (2020b) investigated the transfer-learning based method by first training a Faster-RCNN model on the base-class data, and then freezing the feature extractor and fine-tuning a cosine classifier and a regressor on a balanced dataset of base-class and novel-class data. This method has achieved promising performance compared to previous meta-learning methods. However, none of the existing works considered few-shot video object detection, which is the focus of this paper.

5.2.4 Video Object Detection

Object detection in videos is a challenging task due to the high variation across the videos. The objects in videos may be blurry and change pose and status. Meanwhile, the moving background and unstable lighting conditions pose challenges to object detection. On the other hand, the temporal information in videos can provide more information than static images. These aspects have remained under-explored in the past several years but have recently started to attract more attention after the ImageNet-VID dataset (Russakovsky et al., 2015) was released. Most recent works (*e.g.*, RDN (Deng et al., 2019), FGFA (Zhu et al., 2017) and STSN (Bertasius et al., 2018)) focus on utilizing nearby frames in a short range to gather local information. Meanwhile, other works aim to use global information from frames in a wider range like SELSA (Wu et al., 2019). MEGA (Chen et al., 2020) was recently proposed to effectively combine the local and global information with a memory-enhanced module so that one frame can access more content from nearby and far-away frames, resulting in state-of-the-art performance.

5.3 Few-Shot Video Object Detection

In this section, we introduce the definition of *few-shot video object detection* and dataset construction process.

5.3.1 Problem Definition

Let us define two sets of classes: base classes $\mathcal{C}_{base}$ and novel classes $\mathcal{C}_{novel}$, where $|\mathcal{C}_{base}| = M$, $|\mathcal{C}_{novel}| = N$ with $\mathcal{C}_{novel} \cap \mathcal{C}_{novel} = \emptyset$. Assume there is a large base dataset consisting of many videos per class $\mathcal{V}_{base} = \{V^i_{base} | i = 1, \dots\}$, where $V^i_{base} = \{(I^i_1, Y^i_1), (I^i_2, Y^i_2), \dots\}$ is the i-th video in the base dataset. I^i_t is the t-th frame in the i-th video, and $Y^i_t = \{(c^i_{t,1}, B^i_{t,1}), (c^i_{t,2}, B^i_{t,2}), \dots\}$ with $c^i_{t,b}$, $B^i_{t,b}$ being the class and the bounding box coordinates for b-th object in t-th frame of the i-th video, respectively. Note that Y^i_t could be $\emptyset$ and $c^i_{t,b} \in \mathcal{C}_{base}$. A video object detection model can be built on $\mathcal{V}_{base}$.

After building the model from the base dataset, we would like to adapt the base model to novel classes given only a few videos per novel class. It is natural to define *shot* for image classification and image object detection because one image or one bounding box has only one class label. In order to have an appropriate definition of *shot* for video object detection, we define two types of videos as follows.

Clean videos: Videos contain only one class of objects. Each frame may contain more than one object of that class.

Perfect videos: Videos contain only one class of objects and each frame contains only one object from that class. It is a subset of clean videos.

We denote one perfect video as one-shot. Also, we expect a good detector to perform well on both base and novel classes. Motivated by these aspects, we denote an (N+M)-way K-shot balanced dataset with both

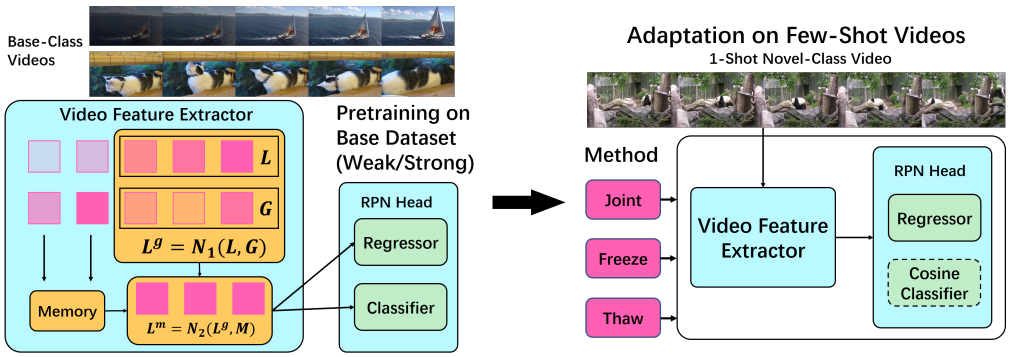

Figure 5.1: The proposed framework of few-shot learning for video object detection. A video object detector is pretrained on the base dataset by aggregating local and global information from different frames in the videos, and then adapted to novel classes based on few-shot novel-class video dataset. During the adaptation, the cosine classifier is used in the detection head and the model is fine-tuned by three methods: Joint, Freeze, and our developed Thaw.

Table 5.1: The statistics of different types of datasets.

Dataset Type	Total Number of Frames
All training videos	57,834 key frames
Perfect training videos	31,530 key frames
All validation videos	176,126 frames
Balanced validation videos	23,624 frames

novel and base classes from perfect videos as $\mathcal{V}_{balance} = \{V^i_{balance} | i = 1, \ldots, (N+M) \times K\}$, where $V^i_{balance} = \{(I^i_1, Y^i_1), (I^i_2, Y^i_2), \ldots\}$, $Y^i_j = (c^i_{t,1}, B^i_{t,1})$ and $c^i_{t,1} \doteq c^i \in \mathcal{C}_{novel} \cup \mathcal{C}_{base}$ for all t. The model will be further trained on this balanced dataset.

5.3.2 Dataset Construction

Since we are studying a new few-shot learning video object detection problem, we construct a new dataset from ImageNet VID dataset (Russakovsky et al., 2015) widely used for video object detection. It consists of 30 categories with 3862 training videos and 555 validation videos. We split them to 25 base classes and 5 novel classes in each novel-base split. Specifically, we create four different types of datasets as follows:

Strong base dataset: It consists of all the videos from base-class objects. Meanwhile, like existing work for video object detection (Chen et al., 2020; Shvets et al., 2019), we additionally include images from the image object detection dataset DET (Russakovsky et al., 2015), leading to a *strong base dataset* for learning strong feature extractors of the base classes.

Weak base dataset: It contains only the perfect videos from base classes, and thus is a subset of strong base dataset.

Remarks: We need to clarify that it is important to investigate the impact from different types of base datasets for real-life applications since the available amount of data for base classes could vary in different scenarios. This has been overlooked in few-shot learning for image classification. Recently, Yue et al. (2020) investigated the different performances between strong and weak backbones used in few-shot image classification. This work shares similarities with our design for strong and weak base datasets.

Balanced few-shot dataset: It is used for few-shot adaptation, and is randomly sampled from the perfect videos in the ImageNet VID training set

for both novel and base class objects. We sample K videos per class with $K = 1, 2, 3$ as the shot number.

Balanced validation dataset: It is the subset of the original clean validation videos and is used to evaluate the few-shot video object detector. Noting that the minimum number of clean validation videos among all classes is 3, we randomly sample 3 clean videos per class to construct this dataset. In this way, we can alleviate the impact from different numbers of videos in different classes.

5.4 The Proposed Framework

In this section, we introduce our proposed framework for few-shot video object detection, which is illustrated in Figure 5.1. First, a video object detector is pretrained on base dataset. Second, a cosine classifier is used to replace the classifier in the detection head and fine-tuned on the balanced few-shot video dataset. More details are provided in the subsequent sections.

5.4.1 Part I: Pretraining the Video Object Detector

In the first stage, we train a video object detector on the base dataset where many videos per class are provided. In particular, given the base dataset $\mathcal{V}_{base}$, we first construct a video object detector to fully utilize the abundant video information by employing the state-of-the-art method

MEGA (Chen et al., 2020). Any video detector could be used here, and we use MEGA as it can efficiently combine both local and global information throughout the video with its memory enhanced module.

We rewrite the video as a set of consecutive frames $\mathcal{V} = \{I_t\}_{t=1}^{T}$. And $B_t = \{b_t^i\}$ denotes the set of proposals generated by RPN in each frame I_t. Following Chen et al. (2020), the local pool for proposals for a key frame I_k is the proposals in the nearby frames, *i.e.*,

$$\mathcal{L} = \{B_t\}_{t=k-T_l}^{k+T_l}. \tag{5.1}$$

We omit the index for $\mathcal{L}$ for simplicity. For the global pool, the ordered frames are randomly reordered such that the index set $\{1, 2, \ldots, T\}$ is mapped to a new shuffled set $\{S_1, S_2, \ldots, S_T\}$. The global pool is constructed by

$$\mathcal{G} = \{B_{S_t}\}_{t=k}^{k+T_g-1}. \tag{5.2}$$

A function $\mathcal{N}_1$ is used to aggregate the global features from $\mathcal{G}$ to $\mathcal{L}$ to produce a new pool $\mathcal{L}^g$, where $\mathcal{N}_1$ consists of stacked location-free relation modules [1].

After obtaining the global aggregated pool $\mathcal{L}^g$, it is then aggregated with a long range memory pool $\mathcal{M}$ by another function $\mathcal{N}_2$, which is composed of stacked location-based relation modules proposed by Chen et al. (2020) to generate an enhanced pool $\mathcal{L}^m$. The aggregation process is given

[1] For simplicity, we did not elaborate this in this paper.

by

$$\mathcal{L}^{g} = \mathcal{N}_1(\mathcal{L}, \mathcal{G}), \tag{5.3}$$

$$\mathcal{L}^{m} = \mathcal{N}_2(\mathcal{L}^{g}, \mathcal{M}). \tag{5.4}$$

The memory pool $\mathcal{M}$ is initialized as an empty set and is updated throughout all frames in one video. When finishing the detection on I_k, the features in $\mathcal{L}^{m}$ are added to $\mathcal{M}$, which will be used for the next key frame I_{k+1}. This recurrent process increases the efficiency of combining features for each key frame. Moreover, one frame can benefit from its subsequent frames without forgetting.

We denote the feature extractor as $f^{e}(\cdot)$, which is extracted by ROI-Align and full-connected layer (He et al., 2017). For each key frame I_k, the final features for all proposals is denoted as

$$f^{e}(I_k) = \{f^{e}(I_k)_1, f^{e}(I_k)_2, ...\}. \tag{5.5}$$

And $f^{e}(I_k)$ is used in the RPN head for classification and bounding box regression. This process is used for all the key frames in a video.

5.4.2 Part II: Adaptation on Few-Shot Videos

After obtaining the pretrained video object detector, we adapt the model based on the balanced few-shot video dataset including novel-class and

base-class objects.

Modification on the RPN Head

Since the cosine classifier is shown to be effective for few-shot image classification problem in (Chen et al., 2019), and more suitable to decorrelate the feature space for different classes (Wang et al., 2017), we adopt a cosine classifier for the few-shot fine-tuning stage. Specifically, a weight matrix $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_{N+M}] \in \mathcal{R}^{d\times(N+M)}$ is fine-tuned where $\mathbf{w}_i$ represents the prototype of i-th class in $\mathcal{C}_{novel} \cup \mathcal{C}_{base}$. Then, the cosine similarity with the different classes for a given proposal $\mathbf{x} \coloneqq f^e(I_t)_l$ is written as:

$$S(\mathbf{W}, \mathbf{x}) = [\cos(\theta(\mathbf{w}_1, \mathbf{x}), ..., \cos(\theta(\mathbf{w}_{N+M}, \mathbf{x}))]'. \tag{5.6}$$

Because cosine similarity ranges between -1 and 1, the softmax function is unable to predict the correct class via the one-hot encoding regime for class labels, causing a discrepancy between the one-hot and real distribution. In order to solve this issue, a scaling factor is usually applied on softmax for better convergence as used in (Qi et al., 2018; Chen et al., 2019; Wang et al., 2017). With that, the probability for i-th class can be represented as:

$$\frac{\exp(\sigma S(\mathbf{W}, \mathbf{x})_i)}{\sum_c \exp(\sigma S(\mathbf{W}, \mathbf{x})_c)}, \tag{5.7}$$

where σ is the scale factor.

The structure of the regressor remains the same as in the pretraining stage but is appended $4 \times$ N dimension (*i.e.*, coordinates of bounding boxes) to account for the N-way novel-class videos. The weights of cosine classifier and regressor for novel-class videos are randomly initialized.

Adaptation Strategies

With the new RPN head, we use multiple strategies (including the newly proposed **Thaw** to be described in Section 6) to adapt the pretrained model when given few-shot videos.

Joint: All the weights from the feature extractor and detection head are fine-tuned jointly on the *balanced few-shot dataset*. This joint fine-tuning is not designed for few-shot learning and usually suffers from overiftting, because the feature extractor can be easily impacted by the few-shot samples. Therefore, it is seldom used in few-shot image classification and usually serves as a low-performing baseline for few-shot image object detection (Yan et al., 2019; Kang et al., 2019).

Freeze: In this method, the feature extractor is frozen and only the detection head is fine-tuned. Freezing feature extractor method is particularly suitable for few-shot learning and widely used in image classification (Chen et al., 2019). The recent work in (Wang et al., 2020b) also shows its superiority in overcoming overfitting and achieves new state-of-the-art performance in few-shot image object detection.

5.5 Preliminary Experiments

In this section, we conduct experiments on the designed dataset under our proposed framework. We evaluate the different adaptation strategies of **Joint** and **Freeze** in different settings. We analyze the results and discover the *insufficiency* and *overfitting* problems in few-shot video object detection.

5.5.1 Implementation Details

Video object detector network: ResNet-101 (He et al., 2016) is used as the backbone. RPN head is applied to *conv4* block in ResNet where the anchors have 4 scales and 3 aspect ratios. 300 bounding box proposals per frame are created during training and validation with the default IoU threshold set to 0.7. Next, RoI-Align and a fully-connected layer are employed after *conv5* block to extract RoI pooled features, followed by the classifier and regressor. For the video detector, we set the local pool and global pool range T_l and T_g to 12 and 10, respectively. The hyperparameters in $\mathcal{N}_1$ and $\mathcal{N}_2$ modules are the same as in (Chen et al., 2020). For the cosine classifier, we set the scale factor σ to 15.

Training: All models are trained on 4 Tesla V100 GPUs, with each GPU holding one set of frames. During pretraining, the initial learning rate is set to 0.001 and drops to 0.0001 after $80,000$ iterations. We train the model for a total of $120,000$ iterations. During fine-tuning on the balanced few-shot dataset, the classifier and regressor are fine-tuned for $4,000$ iterations

in all settings. We set the learning rate during fine-tuning to 0.001. We create three random splits (denoted as A, B, C) from novel-base classes, which is a common practice in few-shot image object detection. During training the detector on the base dataset and fine-tuning it on the balanced few-shot dataset, 15 frames are evenly-spaced selected from the whole videos. For videos with fewer than 15 frames, all frames are selected. The experiments for each few-shot video setting are repeated 5 times. Each time the balanced few-shot dataset is selected **randomly** but kept the **same** for different methods to ensure a fair comparison. The dataset statistics are shown in Table 5.1. The code for the algorithm and dataset will be released.

Inference: During inference on the validation set, we set the NMS threshold to 0.5 IoU and use mean average precision at 0.5 IoU (mAP50) as the evaluation metric. The balanced validation dataset remains the same for different few-shot settings to reduce randomness. The reported mAP50 in each setting is the average of 5 random experiments.

Base Dataset	Class	Method / Split	1-shot			2-shot			3-shot		
			A	B	C	A	B	C	A	B	C
Weak	Novel	Joint (better)	**22.11**	21.88	**9.76**	**32.57**	**32.29**	**23.28**	**36.5**	39.84	**29.86**
		Freeze	18.93	**24.05**	8.09	25.07	31.60	13.88	28.89	**40.53**	13.73
	Base	Joint	56.41	59.23	56.30	58.41	61.77	58.86	60.12	63.75	60.26
		Freeze (better)	**60.98**	**63.97**	**60.95**	**61.15**	**64.12**	**60.83**	**61.32**	**65.02**	**61.10**
Strong	Novel	Joint	21.69	31.10	20.06	39.37	45.94	34.74	44.56	51.43	**43.33**
		Freeze (better)	**41.85**	**40.69**	**31.71**	**50.14**	**47.81**	**42.66**	**53.15**	**52.41**	43.08
	Base	Joint	72.08	76.09	75.80	73.87	77.06	76.97	75.96	78.66	77.25
		Freeze (better)	**82.79**	**84.75**	**86.45**	**83.00**	**84.69**	**86.53**	**83.14**	**85.00**	**86.43**

Table 5.2: Novel-class and base-class mAP50 (in %) on the validation videos when the base dataset is **weak** or **strong**. Better results are in bold. For novel-class performance, Freeze is **better** than Joint on **strong** base dataset but **worse** than Joint on **weak** base dataset, *opposite* to the research findings on images. For base-class performance, Freeze performs consistently better than Joint.

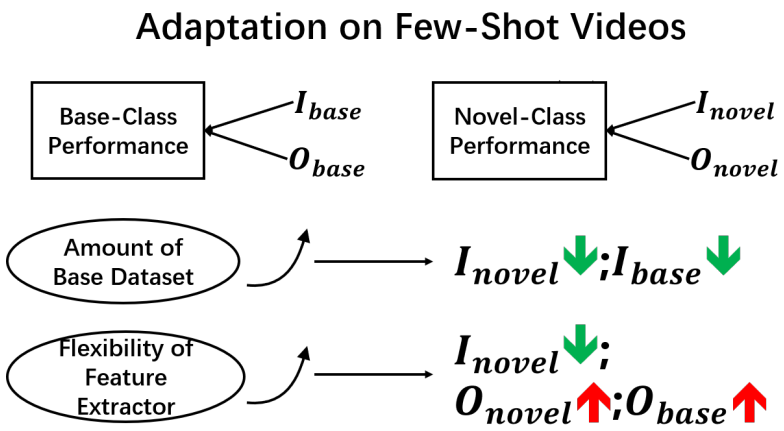

Figure 5.2: Illustration of the insufficiency and overfitting problems. I represents the insufficiency problem and O represents the overfitting problem. Strong and weak base datasets indicate the amount of base dataset. Freeze and Joint indicate the flexibility of the feature extractor. The green down arrows indicate the problem is alleviated and the red up arrows indicate the problem is aggravated.

5.5.2 Preliminary Results

The experiment results for weak and strong base datasets are shown in Table 5.2.

Base-class performance: From the results on both base datasets, Freeze is consistently better than Joint on the base-class objects. This is not surprising as freezing the feature extractor can alleviate the overfitting problem.

Novel-class performance: When the base dataset is strong, Freeze is better than Joint on novel-class objects in almost all cases but one. This is consistent with the prior work in few-shot image classification and image object detection, as freezing the feature extractor can prevent overfitting.

However, the pattern shown on the weak base dataset is opposite to strong base dataset (see Table 5.2). When the base dataset is weak, Joint generally performs better than Freeze, which is the *opposite* to the the research findings on images. An explanation is that it is usually easier to obtain sufficient information from the base dataset to build a sufficient feature extractor for the single image situation. Therefore, the overfitting problem will dominate for novel-class images and Freeze could work well in this situation. However, the complicated structure and abundant information in videos may not be sufficiently learned from the base dataset and thus learning good features for novel-class objects cannot be guaranteed.

5.5.3 Insufficiency vs. Overfitting

The preliminary results from the previous section reveal the insufficiency and overfitting problems for the novel and base classes.

Insufficiency problem corresponds to the situation that the features learned from the base dataset may not be sufficient for building detectors on novel-class objects.

Overfitting problem corresponds to the situation that some good features learned from the base dataset may be distorted by the few-shot videos during fine-tuning when the feature extractor is unfrozen.

We summarize the analysis in Figure 5.2. It is clear that a strong base dataset can alleviate the base-class insufficiency problem. Meanwhile, a strong base dataset can provide a better feature extractor, which improves

the capability to extract better features for novel classes. Therefore, it can alleviate the insufficiency problem for novel classes. On the other hand, freezing the feature extractor (Freeze) could largely solve the overfitting problem both for base and novel classes when fine-tuning on the few-shot videos. However, it does not allow the feature extractor to encode possible novel information from novel classes. Therefore, unfreezing the feature extractor (Joint) would alleviate the insufficiency problem for novel classes since the feature extracted for base classes in the base training stage may not be sufficient to describe novel-class objects. Since the base-class objects in few-shot videos do not provide any further useful information for the base classes (they come from the base-class objects used for pretraining the feature extractor), unfreezing the feature extractor (Joint) does not help reduce the base-class insufficiency problem.

Therefore, in terms of novel-class performance, when the base dataset is **weak**, there is a significant novel-class insufficiency problem. Although Joint aggravates the novel-class overfitting problem, it largely alleviates novel-class insufficiency problem, such that its performance exceeds Freeze. When the base dataset is **strong**, the novel-class insufficiency problem becomes negligible, such that Freeze's performance is better than Joint's in this case.

On the other hand, in terms of base-class performance, unfreezing the feature extractor could only increase the base-class overfitting problem and could not reduce the base-class insufficiency problem, such that Joint's

Table 5.3: Novel-class and base-class mAP50 (in %) on the validation videos, averaged from all novel-base splits. Best results are in bold. * indicates results are similar.

Class	Method / Shot	Weak Base Dataset				Strong Base Dataset			
		1-shot	2-shot	3-shot	Rank	1-shot	2-shot	3-shot	Rank
Novel	Joint	17.92	29.38	35.40	2	24.28	40.02	46.44	3
	Freeze	17.02	23.52	27.72	3	**38.08**	46.87	49.55	1 *
	Thaw (Ours)	**20.05**	**32.13**	**37.32**	1	36.73	**48.71**	**51.38**	1 *
Base	Joint	57.31	59.68	61.37	3	74.66	75.97	77.29	3
	Freeze	**61.97**	**62.03**	**62.48**	1	**84.66**	**84.74**	**84.86**	1
	Thaw (Ours)	60.13	60.33	61.52	2	80.79	78.81	78.94	2

performance is always worse than Freeze no matter the base dataset is strong or weak.

5.6 Improved Method and Experiment

In this section, we further propose a simple but effective method called Thaw to balance the tradeoff of Joint and Freeze during adaptation and demonstrate the rationality of our analysis. We conduct further experiments to illustrate the insufficiency and overfitting problems.

5.6.1 Improved Method: Thaw

As discussed in the previous section, there is a tradeoff between insufficiency and overfitting problems caused by unfreezing the feature extractor. To this end, we still first freeze the feature extractor and fine-tune on the detection head. This would give us a good detection head and prevent the overfitting problem. After convergence, we further unfreeze the feature

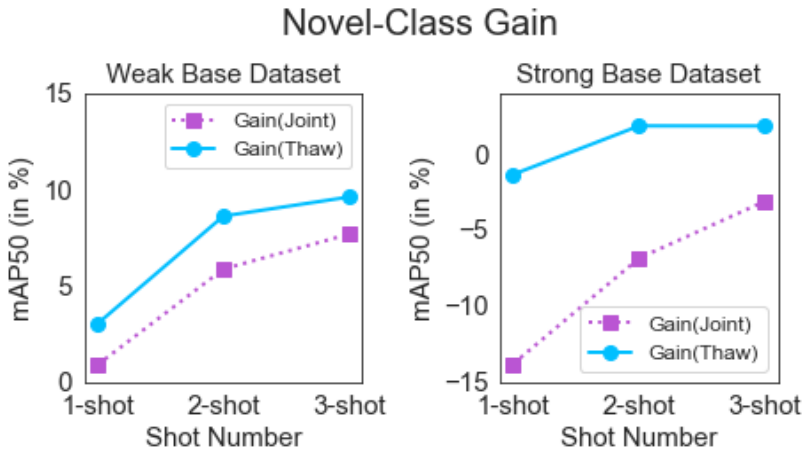

Figure 5.3: Novel-class mAP50 (in %) improvement of Joint and Thaw over Freeze. The gain is from the component of unfreezing the feature extractor. Clearly, the gain is increasing with more shots, and our Thaw improves over Joint by a large margin.

extractor and fine-tune all the weights jointly to let the feature extractor learn extra information to reduce novel-class insufficiency problem. The detection head can be regarded as being initialized with better weights compared with Joint. We call this improved process *Thaw*:

$$\text{Freeze } \mathrm{f}^{e}(\cdot) \xrightarrow{\text{Thaw}} \text{Unfreeze } \mathrm{f}^{e}(\cdot). \tag{5.8}$$

During the experiments, after the detection head is fine-tuned by Freeze, the entire model is further trained for 500 iterations for 1 shot and 2000 iterations for 2 and 3 shots.

5.6.2 Results

The experimental results are shown in Table 5.3. When the base dataset is **weak**, it is apparent that our proposed Thaw outperforms Freeze and Joint by a large margin for novel classes. As described in the previous section, Joint outperforms Freeze substantially in this case (see Table 5.2). Our proposed Thaw is even better, which demonstrates that it can balance the overfitting and insufficiency problems.

When the base dataset is **strong**, Thaw performs comparably to Freeze. We should note that Joint performs poorly in this case as also mentioned in Table 5.2. Our simple method improves Joint by a large margin.

For the base-class performance, our Thaw also significantly outperforms Joint, demonstrating this simple technique significantly alleviates the base-class overfitting problem. Note that the unfreezing part in Thaw can aggravate base-class overfitting problem, so it is expected that Freeze performs the best on the base classes (mentioned in section 5.3).

Number of shots vs. insufficiency/overfitting: When the feature extractor is unfrozen, more shots during few-shot adaptation can simultaneously alleviate novel-class insufficiency and overfitting problems. To better illustrate this phenomenon, we compare the improvement of Joint and Thaw over Freeze on novel classes. The novel-class gain of Joint or Thaw over Freeze is calculated as

$$\mathtt{Gain}_{\mathtt{Joint/Thaw}} = \mathtt{mAP50}_{\mathtt{Joint/Thaw}} - \mathtt{mAP50}_{\mathtt{Freeze}}. \tag{5.9}$$

Figure 5.4: Examples of few-shot learning for video object detection of *giant panda* (weak base dataset). Blue (*resp.* red) bounding boxes denote correct (*resp.* incorrect) detection. Note that *red panda* is a *different* animal. The 1st row shows the 1-shot novel-class video used in few-shot adaptation. The 2nd, 3rd, and 4th rows show the detection results in the validation videos by Joint, Freeze and Thaw, respectively.

Table 5.4: Ablation study on two types of classifiers on strong base dataset and novel-base split A.

Method	Classifier / Shot	Novel-Class mAP50		
		1	2	3
Thaw	Fully-connected	36.91	44.62	52.86
	Cosine	**37.81**	**50.55**	**56.15**

The gain can be seen as the contribution of unfreezing feature extractor. The results are shown in Figure 5.3. We can clearly see the trend that the novel-class gain increases with the increase of the number of shots.

Table 5.5: Ablation study on the influence of temporal information from the video on the strong base dataset and novel-base split A. Note that *1-shot* video is similar to *15-shot* images. The image-based Freeze (the first row) could be regarded as the implementation of the state-of-the-art few-shot image object detection method (Wang et al., 2020b) on 15-shot video images.

Method	Format / Shot	Novel-Class mAP50		
		1	2	3
Freeze	Image-based	19.73	32.24	37.56
	Video-based	**41.85**	**50.14**	**56.00**

5.6.3 Ablation Study

Influence of classifier: We conduct the ablation study of choosing different classifiers in detection head on novel-base split A. Table 5.4 shows the results of these two classifiers on 1-, 2-, and 3-shot settings with Thaw. We can see that the cosine classifier outperforms the fully-connected classifier in our few-shot video object detection.

Influence of video temporal information: Without using the temporal information in the video, the video object detector degenerates to an image object detector. In this case, given few-shot videos, all the key frames are used separately for fine-tuning the image object detector. Thus 1-shot in our video object detection is equivalent to 15-shot in image object detection. To study this further, we conduct an ablation study using novel-base split A. We use Freeze here since image-based Freeze could be considered as the state-of-the-art transfer-learning based few-shot image object detection method (Wang et al., 2020b). The results in Table 5.5

clearly demonstrate the importance of using video information rather than single image information and the meaning of few-shot learning for video object detection.

Visualization of few-shot video object detection: An example of 1-shot video object detection results for different methods are shown in Figure 5.4. The blue and red bounding boxes denote the correct and wrong detection, respectively. From the results, we can see that Joint suffers from the overfitting problem as indicated by the red boxes. For Freeze, there is a mixture of correct and wrong detections. For Thaw, there are more correct bounding boxes compared with Joint and Freeze, highlighting its effectiveness.

5.7 Conclusion

We study a new problem of few-shot learning for video object detection. Specifically, we define the problem, construct a new dataset, and propose a transfer-learning framework for solving this problem. Insufficiency and overfitting problems are revealed from extensive experiments on our designed weak and strong base datasets by two methods (Joint and Freeze) under the proposed framework. Finally, a simple yet effective method called Thaw is naturally developed to validate our analysis and trade off the observed insufficiency and overfitting problems. Our work leads to significantly improved novel-class performance on the weak base dataset

and competitive novel-class performance on the strong base dataset, while maintaining high base-class performance in few-shot video object detection.

6 FUTURE DIRECTION

Few-shot learning is a novel subfield of machine and deep learning that aims to close the gap between human and artificial intelligence, focusing on how to learn new concepts quickly. Most of few-shot learning work focuses on image classification. This focus is motivated by the fact that using established standard benchmark datasets allow for easier and fairer comparisons between methods. However, FSL can be extended to other problem domains as well.

The research presented in this thesis begins with a focus on improving classical meta-learning methods. Then, methods to extend to semi-supervised few-shot learning are proposed. After that, simple techniques for both supervised and semi-supervised setting are presented, which result in uncomplicated few-shot learning methods with astonishingly good performance. Lastly, this thesis presents an investigation of the novel area of few-shot video object detection.

Given the rapid development of few-shot learning research, possible future directions can be summarized as follows:

- Focusing on standard few-shot learning, it will become more challenging to develop novel meta-learning structures. However, recent work focusing on the relationship between novel-class and base-class features (Yang et al., 2021; Wang et al., 2020d) presents a new avenue and interesting topic of future studies. Considering fixed features

obtained from pre-trained feature extractors, new approaches can be inspired by the field of high-dimensional statistics.

- Considering the extension of few-shot learning to other areas, there is prior work on few-shot video classification (Zhu and Yang, 2018), segmentation (Nguyen and Todorovic, 2019), object detection (Perez-Rua et al., 2020), and sentiment classification (Geng et al., 2019). These works are focused on "classification" since the core of few-shot learning is learning "new" concepts. In biology and chemistry, the few-shot learning concept is used more loosely (Ma et al., 2021; Altae-Tran et al., 2017), where the task is to perform binary classification (broadly, focusing on two classes: "same" or "different") in different domains. In these settings, meta-learning remains most suitable. However, there are still opportunities to extend transfer-learning based few-shot learning to biology and chemistry studies when there are suitable problem setups developed. Meanwhile, we are also hoping that more standard benchmark datasets and methods emerge in areas other than image-classification, which could promote the growth of few-shot learning in these domains.

REFERENCES

Altae-Tran, Han, Bharath Ramsundar, Aneesh S Pappu, and Vijay Pande. 2017. Low data drug discovery with one-shot learning. *ACS central science* 3(4):283–293.

Bao, Yujia, Menghua Wu, Shiyu Chang, and Regina Barzilay. 2019. Few-shot text classification with distributional signatures. In *ICLR*.

Bertasius, Gedas, Lorenzo Torresani, and Jianbo Shi. 2018. Object detection in video with spatiotemporal sampling networks. In *ECCV*, 331–346.

Berthelot, David, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. 2019. MixMatch: A holistic approach to semi-supervised learning. In *NIPS*, 5050–5060.

Bishop, Christopher M. 2006. *Pattern recognition and machine learning*. Springer.

Cao, Kaidi, Jingwei Ji, Zhangjie Cao, Chien-Yi Chang, and Juan Carlos Niebles. 2020. Few-shot video classification via temporal alignment. In *CVPR*, 10618–10627.

Cao, Tianshi, Marc Law, and Sanja Fidler. 2019. A theoretical analysis of the number of shots in few-shot learning. *arXiv preprint arXiv:1909.11722*.

Chen, Wei-Yu, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. 2019. A closer look at few-shot classification. In *ICLR*.

Chen, Yihong, Yue Cao, Han Hu, and Liwei Wang. 2020. Memory enhanced global-local aggregation for video object detection. In *CVPR*, 10337–10346.

Chung, Fan RK, and Fan Chung Graham. 1997. *Spectral graph theory*. American Mathematical Soc.

Deng, Jiajun, Yingwei Pan, Ting Yao, Wengang Zhou, Houqiang Li, and Tao Mei. 2019. Relation distillation networks for video object detection. In *ICCV*, 7023–7032.

Duvenaud, David K, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, 2224–2232.

Fan, Qi, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. 2020. Few-shot object detection with attention-rpn and multi-relation detector. In *CVPR*, 4013–4022.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *the Journal of Machine Learning Research* 9:1871–1874.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

Garcia, Victor, and Joan Bruna. 2017. Few-Shot Learning with Graph Neural Networks. In *ICLR*.

Geng, Ruiying, Binhua Li, Yongbin Li, Xiaodan Zhu, Ping Jian, and Jian Sun. 2019. Induction networks for few-shot text classification. *arXiv preprint arXiv:1902.10482*.

Gidaris, Spyros, and Nikos Komodakis. 2018. Dynamic few-shot visual learning without forgetting. In *CVPR*, 4367–4375.

Gilmer, Justin, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *ICLR*, 1263–1272.

Girshick, Ross. 2015. Fast R-CNN. In *ICCV*, 1440–1448.

Girshick, Ross, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 580–587.

Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*, vol. 1. MIT Press Cambridge.

Grandvalet, Yves, and Yoshua Bengio. 2005. Semi-supervised learning by entropy minimization. In *NIPS*, 529–536.

Guneet S. Dhillon, Avinash Ravichandran Stefano Soatto, Pratik Chaudhari. 2020. A baseline for few-shot image classification. In *ICLR*.

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask R-CNN. In *ICCV*, 2961–2969.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Hilliard, Nathan, Lawrence Phillips, Scott Howland, Artëm Yankov, Courtney D Corley, and Nathan O Hodas. 2018. Few-shot learning with metric-agnostic conditional embeddings. *arXiv preprint arXiv:1802.04376*.

Jinlu Liu, Liang Song, and Yongqiang Qin. 2020. Prototype rectification for few-shot learning. In *ECCV*.

Kang, Bingyi, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. 2019. Few-shot object detection via feature reweighting. In *ICCV*.

Karlinsky, Leonid, Joseph Shtok, Sivan Harary, Eli Schwartz, Amit Aides, Rogerio Feris, Raja Giryes, and Alex M Bronstein. 2019. RepMet: Representative-based metric learning for classification and few-shot object detection. In *CVPR*, 5197–5206.

Kim, Jongmin, Taesup Kim, Sungwoong Kim, and Chang D Yoo. 2019. Edge-labeling graph neural network for few-shot learning. In *CVPR*, 11–20.

Kingma, Diederik P, and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *ArXiv* abs/1412.6980.

Kipf, Thomas, and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ArXiv* abs/1609.02907.

Laine, Samuli, and Timo Aila. 2017. Temporal ensembling for semi-supervised learning. In *ICLR*.

Law, Hei, and Jia Deng. 2018. CornerNet: Detecting objects as paired keypoints. In *ECCV*.

Lee, Dong-Hyun. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, icml*.

Lee, Kwonjoon, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. 2019. Meta-learning with differentiable convex optimization. In *CVPR*, 10657–10665.

Li, Wenbin, Lei Wang, Jinglin Xu, Jing Huo, Yang Gao, and Jiebo Luo. 2019a. Revisiting local descriptor based image-to-class measure for few-shot learning. In *CVPR*, 7260–7268.

Li, Xinzhe, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. 2019b. Learning to self-train for semi-supervised few-shot classification. In *NIPS*, 10276–10286.

Lifchitz, Yann, Yannis Avrithis, Sylvaine Picard, and Andrei Bursuc. 2019. Dense classification and implanting for few-shot learning. *CVPR* 9250–9259.

Liu, Bin, Yue Cao, Yutong Lin, Qi Li, Zheng Zhang, Mingsheng Long, and Han Hu. 2020a. Negative margin matters: Understanding margin in few-shot classification. In *ECCV*, 438–455. Springer.

Liu, Li, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. 2020b. Deep learning for generic object detection: A survey. *IJCV* 128(2):261–318.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. SSD: Single shot multibox detector. In *ECCV*, 21–37.

Liu, Yanbin, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. 2019. Learning to propagate labels: transductive propogation network for few-shot learning. In *ICLR*.

Ma, Jianzhu, Samson H Fong, Yunan Luo, Christopher J Bakkenist, John Paul Shen, Soufiane Mourragui, Lodewyk FA Wessels, Marc Hafner, Roded Sharan, Jian Peng, et al. 2021. Few-shot learning creates predictive models of drug response that translate from high-throughput screens to individual patients. *Nature Cancer* 2(2):233–244.

Mallya, Arun, and Svetlana Lazebnik. 2018. PackNet: Adding multiple tasks to a single network by iterative pruning. *CVPR* 7765–7773.

Mangla, Puneet, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. 2020. Charting the right manifold: Manifold mixup for few-shot learning. In *WACV*.

Mishra, Nikhil, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. 2018. A simple neural attentive meta-learner. In *ICLR*.

Miyato, Takeru, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41(8):1979–1993.

Nguyen, Khoi, and Sinisa Todorovic. 2019. Feature weighting and boosting for few-shot segmentation. In *ICCV*, 622–631.

Nichol, Alex, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Oliver, Avital, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. 2018. Realistic evaluation of deep semi-supervised learning algorithms. In *NIPS*, 3235–3246.

Oreshkin, Boris, Pau Rodríguez López, and Alexandre Lacoste. 2018. TADAM: Task dependent adaptive metric for improved few-shot learning. In *NIPS*, 721–731.

Perez-Rua, Juan-Manuel, Xiatian Zhu, Timothy M Hospedales, and Tao Xiang. 2020. Incremental few-shot object detection. In *CVPR*, 13846–13855.

Qi, Hang, Matthew Brown, and David G. Lowe. 2018. Low-shot learning with imprinted weights. In *CVPR*.

Qiao, Siyuan, Chenxi Liu, Wei Shen, and Alan L Yuille. 2018. Few-shot image recognition by predicting parameters from activations. In *CVPR*.

Raschka, Sebastian, Joshua Patterson, and Corey Nolet. 2020. Machine learning in python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *Information* 11(4): 193.

Ravi, Sachin, and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *ICLR*.

Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *CVPR*, 779–788.

Redmon, Joseph, and Ali Farhadi. 2017. YOLO9000: better, faster, stronger. In *CVPR*, 7263–7271.

Ren, Mengye, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. 2018. Meta-learning for semi-supervised few-shot classification. In *ICLR*.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 91–99.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *IJCV* 115(3):211–252.

Rusu, Andrei A, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. 2019. Meta-learning with latent embedding optimization. In *ICLR*.

Sajjadi, Mehdi, Mehran Javanmardi, and Tolga Tasdizen. 2016. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, 1163–1171.

Scarselli, Franco, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2009. The graph neural network model. *IEEE Transactions on Neural Networks* 20:61–80.

Schonfeld, Edgar, Sayna Ebrahimi, Samarth Sinha, Trevor Darrell, and Zeynep Akata. 2019. Generalized zero-and few-shot learning via aligned variational autoencoders. In *CVPR*, 8247–8255.

Shvets, Mykhailo, Wei Liu, and Alexander C. Berg. 2019. Leveraging long-range temporal relationships between proposals for video object detection. In *ICCV*.

Snell, Jake, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NIPS*, 4077–4087.

Spyros Gidaris, Nikos Komodakis, Praveer Singh. 2018. Unsupervised representation learning by predicting image rotations. In *ICLR*.

Sun, Qianru, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. 2019. Meta-transfer learning for few-shot learning. In *CVPR*, 403–412.

Sung, Flood, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *CVPR*, 1199–1208.

Tarvainen, Antti, and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, 1195–1204.

Tian, Zhi, Chunhua Shen, Hao Chen, and Tong He. 2019. FCOS: Fully convolutional one-stage object detection. In *ICCV*.

Velickovic, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. *ArXiv* abs/1710.10903.

Vinyals, Oriol, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. 2016. Matching networks for one shot learning. In *NIPS*, 3630–3638.

Wah, C., S. Branson, P. Welinder, P. Perona, and S. Belongie. 2011. The Caltech-UCSD Birds-200-2011 Dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology.

Wang, Feng, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. 2017. Normface: l2 hypersphere embedding for face verification. In *Acm international conference on multimedia*, 1041–1049.

Wang, Weibin, Dong Liang, Qingqing Chen, Yutaro Iwamoto, Xian-Hua Han, Qiaowei Zhang, Hongjie Hu, Lanfen Lin, and Yen-Wei Chen. 2020a. Medical image classification using deep learning. In *Deep learning in healthcare*, 33–51. Springer.

Wang, Xin, Thomas E. Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. 2020b. Frustratingly simple few-shot object detection. In *ICML*.

Wang, Yan, Wei-Lun Chao, Kilian Q Weinberger, and Laurens van der Maaten. 2019. SimpleShot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*.

Wang, Yaqing, Quanming Yao, James T Kwok, and Lionel M Ni. 2020c. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)* 53(3):1–34.

Wang, Yikai, Chengming Xu, Chen Liu, Li Zhang, and Yanwei Fu. 2020d. Instance credibility inference for few-shot learning. In *CVPR*.

Wani, M Arif, Farooq Ahmad Bhat, Saduf Afzal, and Asif Iqbal Khan. 2020. Supervised deep learning in face recognition. In *Advances in deep learning*, 95–110. Springer.

Wu, Haiping, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. 2019. Sequence level semantics aggregation for video object detection. In *ICCV*, 9217–9225.

Xing, Chen, Negar Rostamzadeh, Boris Oreshkin, and Pedro OO Pinheiro. 2019. Adaptive cross-modal few-shot learning. In *NIPS*, 4848–4858.

Yan, Xiaopeng, Ziliang Chen, Anni Xu, Xiaoxi Wang, Xiaodan Liang, and Liang Lin. 2019. Meta R-CNN: Towards general solver for instance-level low-shot learning. In *ICCV*, 9577–9586.

Yang, Ling, Liangliang Li, Zilun Zhang, Xinyu Zhou, Erjin Zhou, and Yu Liu. 2020. Dpgn: Distribution propagation graph network for few-shot learning. In *CVPR*, 13390–13399.

Yang, Shuo, Lu Liu, and Min Xu. 2021. Free lunch for few-shot learning: Distribution calibration. In *ICLR*.

Yu, Zhongjie, Lin Chen, Zhongwei Cheng, and Jiebo Luo. 2020. TransMatch: A transfer-learning scheme for semi-supervised few-shot learning. In *CVPR*.

Yu, Zhongjie, and Sebastian Raschka. 2020. Looking back to lower-level information in few-shot learning. *Information* 11(7):345.

Yu, Zhongjie, Gaoang Wang, Lin Chen, Sebastian Raschka, and Jiebo Luo. 2021. Few-shot learning for video object detection in a transfer-learning scheme. *arXiv preprint arXiv:2103.14724*.

Yue, Zhongqi, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. 2020. Interventional few-shot learning.

Zagoruyko, Sergey, and Nikos Komodakis. 2016. Wide residual networks. In *British machine vision conference*.

Zhou, Dengyong, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *NIPS*, 321–328.

Zhou, Xingyi, Dequan Wang, and Philipp Krähenbühl. 2019. Objects as points. *arXiv preprint arXiv:1904.07850*.

Zhu, Linchao, and Yi Yang. 2018. Compound memory networks for few-shot video classification. In *ECCV*, 751–766.

Zhu, Xizhou, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. 2017. Flow-guided feature aggregation for video object detection. In *ICCV*, 408–417.