Few-Shot Learning for Images

Hongrun Zhou
College of Engineering
Carnegie Mellon University
Pittsburgh, PA 15213
hongrunz@andrew.cmu.edu

Yinghuan Zhang
Dietrich College of Humanities and Social Sciences
Carnegie Mellon University
Pittsburgh, PA 15213
yinghuan@andrew.cmu.edu

Yun Cheng
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
yuncheng@cs.cmu.edu

Yaxin Tan
Heinz College of Information Systems and Public Policy
Carnegie Mellon University
Pittsburgh, PA 15213
yaxint@andrew.cmu.edu

Abstract

Few-Shot Learning (FSL) aims at identifying unseen objects using only a few samples with supervised information available. The limited data in novel classes usually follow a biased distribution, difficult for the model to learn. We proposed **Weighted-distribution Calibration (WC)** to alleviate bias by generating more data from the calibrated distribution of novel classes. We used transferred statistics of all base classes with sufficient data to perform the calibration, weighted by the distance between the novel class and base classes. With a backbone of ResNet-12 and a logistic regression classifier, WC successfully improves the model's performance from a base accuracy of 57.33% to a surprising accuracy of **63.87%**. Our weighted calibration method can be easily combined with any feature extractor and classifier, bringing a boost in Top-1 accuracy in FSL. We further experimented with Vision Transformer (ViT) as a feature extractor. We found that, unfortunately, due to high complexity and insufficient pre-training, ViT does not produce more meaningful features as expected.

1 Introduction

Deep learning models have achieved great success in computer vision tasks such as image recognition, image generation, and segmentation. However, the majority of these models are trained on a large amount of labeled data and are limited when given images without training labels [26] or not enough training examples [22]. In contrast, humans are good at recognizing a new object through a small number of samples. For example, children only need a few pictures in the book to learn what a "zebra" or "rhino" is. Inspired by the rapid learning ability of humans, fields like few-shot [5, 11] and zero-shot [12] emerged to develop models that can classify images with new categories with small or even non-existent training samples. Among the various popular approaches to few-shot learning, e.g. meta-learning, metrics-learning, we chose to focus on the fine-tuning method, for its simplicity and its promising results [14]. [14] achieved test accuracy of 60.63% for *mini*ImageNet and 69.02% for *tiered*ImageNet under $1-$shot $5-$way setting, which outperformed many meta-learning based FSL models with greater complexity.

Many previous approaches work on developing more robust models to learn better feature representations. However, the most critical challenge in FSL is the limitation in the number of training data. Therefore, we are focusing on the property of data itself. Common sense is that the ground truth distribution can be more accurately uncovered with larger dataset. We proposed a

Weighted-distribution Calibration (WC) method which can approximately disclose the ground truth distribution with the limited training data in FSL image classification tasks. WC can leverage the statistics from base classes and calibrate the biased distribution due to limited training data.

Furthermore, another challenge in FSL is that the feature extractor trained on base classes always contains irrelevant information to the objects that need to be classified. Therefore, we introduce Vision Transformer (ViT) as a task-agnostic embedding learner. We expected ViT to pay more attention to target objects, thus decreasing the redundancy in extracted features. Unfortunately, it did not work well in our experiments, possibly, due to its loss of reception field.

2 Related Work

2.1 Problem Statement

There are three datasets in the FSL image classification task: labeled target support set $\mathcal{S}$, unlabeled target query set $\mathcal{Q}$, and a class-disjoint auxiliary set $\mathcal{A}$. $\mathcal{S}$ and $\mathcal{Q}$ share the same label space, whereas $\mathcal{A}$ has a label space disjoint from that of $\mathcal{S}$ and $\mathcal{Q}$. The concept of "*few-shot*" refers to the limited samples supplied by the support set $\mathcal{S}$, where there are C classes but each class only has K (*e.g.*, 1 or 5) labeled samples. This kind of classification task is called C-way K-shot.

2.2 Fine-tuning based methods

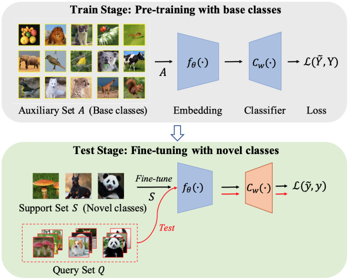
Train Stage: Pre-training with base classes
A
$f_\theta(\cdot)$
$C_w(\cdot)$
$\mathcal{L}(\tilde{Y}, Y)$
Auxiliary Set A (Base classes)
Embedding
Classifier
Loss
Test Stage: Fine-tuning with novel classes
Fine-tune
S
$f_\theta(\cdot)$
$C_w(\cdot)$
$\mathcal{L}(\tilde{y}, y)$
Support Set S (Novel classes)
Test
Query Set Q

Figure 1: Fine-tuning based methods

Fine-tuning based method is one of the most popular FSL methods, consisting of two-stages, *i.e.*, standard pre-training with base classes and fine-tuning with novel classes. In the **training stage**, the whole auxiliary set $\mathcal{A}$ is used to train a classification model, including a feature embedding extractor $f_\theta(\cdot)$ and a N^{base}-class classifier $C_\omega(\cdot)$, using the standard cross-entropy loss $\mathcal{L}^{\text{CE}}$ as below,

$$\Gamma = \underset{\theta,\omega}{\operatorname{argmin}} \sum_{i=1}^{N} \mathcal{L}^{\text{CE}}(C_\omega(f_\theta(X_i)), y) \tag{1}$$

In the **test stage**, we retain the feature embedding extractor $f_\theta(\cdot)$ from the train stage and preserve the parameters. Afterwards, for each specific novel task $\mathcal{T} = \langle \mathcal{S}, \mathcal{Q} \rangle$, a new C-class classifier will be re-learned based on $\mathcal{S}$ every time.

Many fine-tuning based methods are proposed. [2] follows the standard transfer learning procedure of network pre-training and fine-tuning. Baseline [2] adopts a linear layer, *i.e.*, a fully-connected (FC) layer, as the new classifier. Baseline++ [2] replaces the standard inner product (in the FC layer) with a cosine distance between the input feature and weight vector. RFS-simple [2] employs logistic regression instead of the FC layer as the new classifier by first using ℓ_2 normalization for the feature vector. SKD-GEN0 proposed in [18] also use logistic regression as the classifier as same as RFS-simple [2], where the only difference is that additional rotation-based self-supervision is further introduced into the pre-training stage. In addition, both [2] and [18] develop an extended version using knowledge distillation [8]. [16] proposed an embarrassingly simple baseline with random pruning, boosting the performance drastically. The boost is thought to be caused by reduction in redundancy.

2.3 Other methods

Meta-learning based methods *Meta-learning based* methods[6, 7, 18, 1, 19, 17] normally perform a meta-training paradigm [20] on a family of few-shot tasks constructed from the base classes at the **training stage**. Meta-learner is trained in the training stage, enabled to be fast adapted to novel classes in $\mathcal{S}$ at the **test stage**. [6] is one popular representative method, whose core idea is to train a model's initial parameters by involving the second-order gradients, enabling this model to rapidly adapt to a new task with just one or a few gradient steps. [1] is designed from another perspective, by adopting a standard machine learning algorithm such as ridge regression as the base-learner classifier $C_\omega(\cdot)$ in the inner loop. LEO [19] and ANIL [18] follows the same optimization procedure as MAML[6] with some optimization in parameters space.

Metric-learning based methods Different from the two-loop structure of *meta-learning based* methods, *metric-learning based* methods [21, 15, 22, 13, 9] directly compare the similarities/distances between the query set and support set in each meta-like task through one single feed-forward pass through the episodic-training mechanism [24]. At the **training stage**, the standard cross-entropy loss is usually employed to train the entire model. During the **test stage**, the nearest-neighbor classifier (1-NN) can be conveniently used for prediction.

3 Methodology

As we mentioned in Section 2, we use an auxiliary set $\mathcal{A}$ and a support set $\mathcal{S}$ for our train and test stages respectively. We transfer statistics from base classes in $\mathcal{A}$ to effectively generate data from the calibrated distribution of novel classes in $\mathcal{S}$. In the following sections, we describe our adaptive data generation method specifically for 1-shot learning.

3.1 Tukey's Ladder of Powers Transformation

We implement Tukey's Ladder of Powers transformation [23] on features of samples in auxiliary, support set and query set to make the corresponding distribution more Gaussian-like to better leverage statistics transformation. One problem in performing Tukey's Ladder of Powers transformation is that the transformation assumes all positive input data, yet it is possible for our feature extractor to produce negative features. Our solution is to use ReLU to shift all data to the positive domain. Then we perform Tukey's transformation on the shifted data,

$$\tilde{\boldsymbol{x}} = \begin{cases} \boldsymbol{x}^\beta, & \text{if } \beta \neq 0 \\ \log(\boldsymbol{x}), & \text{otherwise} \end{cases} \tag{2}$$

where the β is a hyperparameter to adjust how to correct the distribution. Here we have two hyperparameters β_{base} for base class features transformation and β_{novel} for novel class features transformation.

3.2 Statistics of the Base Classes

Given any pretrained feature extractor, we compute the mean and covariance matrix of the base classes, assuming they follow a Gaussian feature distribution. We first perform Tukey's Ladder of Powers transformation on all feature vectors $\boldsymbol{x}$ and obtain $\tilde{\boldsymbol{x}}$. For each base class i, mean $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$ are computed as follows:

$$\boldsymbol{\mu}_i = \frac{\sum_{\tilde{\boldsymbol{x}}_j \in A_i} \tilde{\boldsymbol{x}}_j}{n_i}, \quad \boldsymbol{\Sigma}_i = \frac{1}{n_i - 1} \sum_{\tilde{\boldsymbol{x}}_j \in A_i} (\tilde{\boldsymbol{x}}_j - \boldsymbol{\mu}_i)(\tilde{\boldsymbol{x}}_j - \boldsymbol{\mu}_i)^T \tag{3}$$

where $A_i = \{\tilde{\boldsymbol{x}} | (\tilde{\boldsymbol{x}}, y) \in \mathcal{A} \wedge y = i\}$.

Algorithm 1 5-way 1-shot testing procedure with weighted calibration and data generation

```
% requires: support and query set features of 5 novel classes $S, Q$
% requires: base class means and covariances $\{\boldsymbol{\mu}_i\}, \{\boldsymbol{\Sigma}_i\}$, $i = 1, 2, \cdots, N_{base}$
% returns: prediction of query set
for $S_c \in S$:
    # $S_c$: support set feature and label of novel class c
    $(\boldsymbol{x}_c, y_c)$ = $S_c$
    $\tilde{\boldsymbol{x}}_c$ = tukey_transform($\boldsymbol{x}_c$)
    $\boldsymbol{D}$ = euclidean_dist($\{\boldsymbol{\mu}_i\}, \tilde{\boldsymbol{x}}$)
    $\boldsymbol{S}$ = softmax($\boldsymbol{D}$)
    $\boldsymbol{\mu}'$,$\boldsymbol{\Sigma}'$ = weighted_calibration($\tilde{\boldsymbol{x}}_c$,$\{\boldsymbol{\mu}_i\}$,$\{\boldsymbol{\Sigma}_i\}$,$\boldsymbol{S}$,k)
    $(\tilde{\boldsymbol{X}}_c^{'}, y_c^{'})$ = multivariate_normal($\boldsymbol{\mu}'$,$\boldsymbol{\Sigma}'$)
$\boldsymbol{X}_{aug}$ = concatenate([$\tilde{\boldsymbol{x}}_c$,$\tilde{\boldsymbol{X}}_c^{'}$])
$\boldsymbol{Y}_{aug}$ = concatenate([$y_c$,$y_c^{'}$])
model = classifier($\boldsymbol{X}_{aug}$,$\boldsymbol{Y}_{aug}$)
return model($Q$)
```

3.3 Calibrated Statistics of the Novel Classes

We first obtain $\tilde{\boldsymbol{x}}$ from $\boldsymbol{x}$ using Tukey's Ladder of Powers as shown in Equation (2).

Then we calibrate the novel class distribution using the transferred statistics from the base classes. The similarity scores are measured by the Euclidean distance. That is, for base class i, we calculate the distance vector $\boldsymbol{D}_i$:

$$\boldsymbol{D}_i = -\|\boldsymbol{\mu}_i - \tilde{\boldsymbol{x}}\|^2, \quad i = 1, \cdots, N_{\text{base}} \tag{4}$$

where $\tilde{\boldsymbol{x}}$ is the feature vector of the support set.

Instead of simply averaging the mean of the top k base classes to calibrate the distribution of the novel class as in [25], we propose a weighted calibration:

$$\begin{aligned} \boldsymbol{\mu}' &= \frac{\sum_{i=1}^{N_{base}} \boldsymbol{\mu}_i S_i + \tilde{\boldsymbol{x}}}{2}, \\ \boldsymbol{\Sigma}' &= \sum_i \boldsymbol{\Sigma}_i S_i + \alpha, \\ S_i &= \text{softmax}(\boldsymbol{D}_i), \quad i = 1, \cdots, N_{\text{base}} \end{aligned} \tag{5}$$

where S_i is the weight for each pair of $(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ normalized by distance such that $\sum_i S_i = 1$. α is a chosen hyperparameter constant that gets added to each element of the estimated covariance matrix, which determines the degree of dispersion of features sampled from the calibrated distributions.

3.4 Leveraging the Calibrated Distribution of the Novel Classes - Data Sampling

We then collect the statistics from the last section and denote the set of results as $\mathbb{S}_y = \{(\boldsymbol{\mu}'_{y,1}, \boldsymbol{\Sigma'}_{y,1}), \cdots, (\boldsymbol{\mu}'_{y,d}, \boldsymbol{\Sigma'}_{y,d})\}$, where $\boldsymbol{\mu}'_y = [\boldsymbol{\mu}'_{y,1}, \cdots, \boldsymbol{\mu}'_{y,d}]$ and $\boldsymbol{\Sigma}'_y = [\boldsymbol{\Sigma'}_{y,1}, \cdots, \boldsymbol{\Sigma'}_{y,d}]$ are calibrated mean and covariance of the features in the support set of class y and d denotes the dimension of feature vector.

With $\mathbb{S}_y$ for class y, we are able to obtain generated data sample:

$$\mathbb{D}_y = \{(\boldsymbol{x}, y) \mid x_i \sim \mathcal{N}(\boldsymbol{\mu'}_{y,i}, \boldsymbol{\Sigma'}_{y,i}), \forall(\boldsymbol{\mu'}_{y,i}, \boldsymbol{\Sigma'}_{y,i}) \in \mathbb{S}_y\} \tag{6}$$

where $\boldsymbol{x} = [x_1, \cdots, x_d]$ denotes the generated feature vector.

We then train the classifier to minimize the following loss function ℓ:

$$\ell = \sum_{(\boldsymbol{x},y)\sim \tilde{S}\cup \mathbb{D}_{y,y\in\mathcal{Y}^{\mathcal{T}}}} -\log \Pr(y|\boldsymbol{x};\theta) \tag{7}$$

where $\mathcal{Y}^{\mathcal{T}}$ is the set of classes for the task $\mathcal{T}$ and $\tilde{S}$ is the support set with transformation.

The whole procedure of $5-$way $1-$shot task test is shown in Algorithm. 1.

4 Experiment

4.1 Setup

Dataset We used the *mini*ImageNet dataset. The *mini*ImageNet dataset was proposed by Vinyals [24]. For few-shot learning evaluation, its complexity is high due to the use of ImageNet images but requires fewer resources and infrastructure than running on the full ImageNet dataset. In total, there are 100 classes with 600 samples of 84×84 color images per class. These 100 classes are divided into 64, 16, and 20 classes respectively for sampling tasks for training, validation, and test. For training, we used a subset of 20130 images in total.

Training, Fine-tuning & Testing For all baseline models in the training stage, we construct a ResNet12 backbone with 64 output classes, which is trained for 50 epochs using an Adam optimizer with a learning rate of $1e-3$ and a batch size of 4. We then perform fine-tuning under the 5-way 1-shot classification setting. For each support set data, we test with 15 query samples. In data generation, we set α to be 0.21 and the β value to 0.5 as in the original paper. In the test stage, we perform a round-test[1], where all 20 classes in the query set are tested for their accuracy, and an average test accuracy is obtained by taking the mean of all 20 results.

Baseline Model We implemented a vanilla fine-tuning baseline using ResNet12 as the backbone and Logistic Regression as the classifier. We chose this baseline for the convenience of quantitatively comparing the effects from our two proposed improvements described in Section 3 above.

4.2 Hyper-Settings and Parameters

Converting features to Positive Domain We consider ReLU and softmax transformations to ensure the positivity of features before the Tukey's Ladder of Powers transformation. We experiment with both transformations and land on ReLU for the better performance. We hypothesize the reason why ReLU outperforms softmax is that it preserves the linearity in the data that contains lots of essential information for the classifier to make correct predictions. On the contrary, the softmax operation skews the data distribution exponentially.

Hyper-parameters for Data Transform We focuse on three parameters in total: α in the weighted calibration (equation 5), β_{novel} and β_{base} in Tukey's Ladder of Powers transformation. We perform a grid search separately on each variable. The default values are as follows: $\beta_{\text{novel}} = 0.5$, β_{base} = None, and $\alpha = 0.21$. When tuning one hyper-parameter, we set the other hyper-parameters to the aforementioned values.

We experiment across β_{novel} and β_{base} ranging from 0.0 to 1.0 with the step size of 0.1, and then α from 0.00 to 1.00 with the step size of 0.01. We conclude that the best value for α, β_{novel} and β_{base} are 0.0, 0.9, and 0.5. The results are shown in table 1.

5 Results

From the data in Table 2, our results are higher than that of our baselines and are comparable to a couple of SOTA models. We produced 63.87% with weighted calibration just on the support set and 62.00% with weighted calibration on the support set plus Tukey's transformation on the base classes. The two approaches yielded 11.4% and 8.1% increase from the baseline. We cannot conclude with existing data that incorporating Tukey's transformation on the base class necessarily leads to better

Table 1: **Results of Hyper-parameter Grid Search** on ResNet-12, round-test[1], and Euclidean distance.

Hyper-parameter	Search Range	Best Value
β_{novel} - correct the distribution	$[0.0, 1.0]$ step 0.1	0.9
β_{base} - correct the distribution	$[0.0, 1.0]$ step 0.1	0.5
α - calibrate covariance matrix	$[0.00, 1.00]$ step 0.01	0.0

[1] *Round-test means we only test each novel class once in the test tasks and each class is guaranteed to be tested in one test task.*

Table 2: **Reproduction and Model Experiment Results** on *mini*ImageNet. Results are reported with the mean accuracy over 10 $5-$way $1-$shot test tasks. Some of the results are reported by LibFewShot for fair comparison with imput image size of 84×84 and task-specific classifier of Logistic Regression.

	Methods	Feature Extractor	Reported Acc (%)	Test Acc (%)
Baselines	Baseline[2][1]	ResNet-12	57.47	57.33
	Baseline++[2][1]	ResNet-12	51.15	
	Distribution Calibration[25][2]	WideResNet	68.57	
	Distribution Calibration[25][2]	ResNet-12		54.13
SOTA	RFS-simple[1]	ResNet-12	61.00	
	RFS-distill[1]	ResNet-12	63.27	
	MTL	ResNet-12	60.20	
	DN4	ResNet-12	54.37	
	CAN	ResNet-12	63.85	
Ours	No Calibration	ResNet-12		55.33
	Weighted Calibration	ResNet-12		**63.87**
	Weighted Calibration (BC[3])	ResNet-12		**62.00**

[1] *Reported by LibFewShot[14], a unified Few-Shot Learning library.*
[2] *Calibrated with transferred statistics from Top-2 base classes.*
[3] *Used Tukey's Ladder of Powers Transformation on base classes' statistics.*

accuracy results as the bump from 62.00% to 63.87% could be attributed to random error, but the bigger jump of 11.4% and 8.1% shows that weighted calibration is helpful for yielding more accurate test results in few-shot learning.

Furthermore, we observe that there is a decrease of test accuracy from the baseline adopting the top-2 transferred statistics approach proposed in [25]. We think this could be attributed to the fact that, in a case where one base class is extremely close to the novel class, averaging across the two closest base class could yield worse calibration. On the other hand, our proposal of using weighted calibration could have gotten better results as it takes into consideration of the relative distance between classes.

In addition, we compare against Top-2 calibration and saw that it did not provide any substaintial increase in test accuracy compared to data generation only. We suspect that the proclaimed performance enhancement brought by Top-2 calibration is mostly attributed to data generation instead of calibration, whereas the weighted calibration provided the substantial improvement.

6 Ablation Study

We identify four stages in our final pipeline – Tukey's transformation on the base classes, Tukey's transformation on the novel classes, weighted calibration, and data generation. In an effort to show the effectiveness of each stage separately, we did an ablation study to compare and contrast the test accuracy contributed by each stage. As shown in Table 3, we see a substantial increase by adding weighted calibration, which further confirms our interpretation earlier in Section 5. There are less substantial increases by adopting Tukey's transformation on the novel classes and the base classes,

Table 3: **Ablation Study** on ResNet-12/Logistic Regression/round-test[1] setting.

Weighted Calibration	Novel Class Transform	Base Class Transform	Data Generation[2]	Test Acc (%)
✓	✓	✓	✓	**60.33**
✓	✓		✓	60.00
✓			✓	59.33
			✓	55.00
				55.33
Top-2 Calibration	✓		✓	55.00

[1] *Round-test means we only test each novel class once in the test tasks and each class is guaranteed to be tested in one test task.*
[2] *If generated, a fixed number of* 750 *new samples following the calibrated distribution are generated.*

but we still observe some increase in test accuracy, which suggests that conforming the data to a more Gaussian distribution is still helpful.

7 Transformer-Based Feature Extractor

7.1 Why introducing a Transformer?

We believe that vision transformers[3] might be effective in differentiating the local and global features certain images possess, therefore we propose a transformer-based task-agnostic embedding learner for Few-Shot Learning. Our embedding learner follows the standard architecture of a vision transformer encoder, as shown in Fig.2. This vision transformer-based embedding learner reshapes the 2D image into a sequence of flattened 2D patches, linearly embeds each of them, adds position embeddings to retain the positional information, and feeds the resulting sequence of vectors to a standard transformer encoder. Such transformer-based embedding learner is only trained on base class data and is not updated in FSL meta tasks.

In the training stage, we train a base image classification model consisting of this vision transformer-based embedding learner $f(\cdot)$ and a classifier $c(\cdot)$. The base model is trained with cross-entropy loss on the auxiliary set $\mathcal{A}$. After a certain period when we are confident that the learner has learned enough to conduct feature extraction, we fix the parameters of the embedding learner. We then extracted the feature embeddings of the auxiliary set $\mathcal{A}$. We used a vision transformer (ViT) feature extractor that used 64 labels, that take in 84×84 images, 6 hidden layers, 12 attention heads. The transformer has a hidden size of 768 and an intermediate size of 1536.

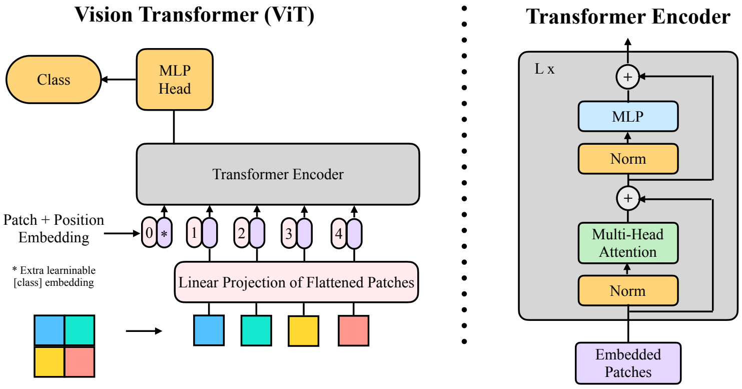

Figure 2: Vision Transformer (ViT) Classifier Overview

Table 4: **Vision Transformer Experiments Results** on *mini*ImageNet.

Methods	Feature Extractor	Test Acc (%)
No Calibration	ResNet-12	55.33
Distribution Calibration[25][2]	ResNet-12	54.13
Weighted Calibration	ResNet-12	**63.87**
No Calibration	ViT	32.67
Distribution Calibration	ViT	31.33
Weighted Calibration	ViT	**33.00**

7.2 Experiments with Vision Transformer as Feature Extractor

Although self-attention-based models are widely shown effective in extracting features from focused objects, the experiment results shown in Table 2 suggest that using ViT did not boost performance in few-shot learning task as expected. There may be several reasons for this. First, transformers generally require large-scale pre-training [10]. This becomes a disadvantage of ViT in the context of few-shot learning due to the challenging low-data problem, which has been mitigated but not fully resolved by our data generation approach. Secondly, transformers generally incur a high compute cost when applied to high-dimensional sequences due to the quadratic complexity of the self-attention mechanism. Therefore, the applicability of ViT is greatly restricted to longer sequences of images. [10]. Moreover, compared to the conventional CNNs, the transformer did not have the advantage in local connectivity and spatial invariance biases inbuilt in the CNN structure. A potential solution is to try to improve the efficiency by including inductive biases and leverage learning the rich visual patterns with local connectivity and spatial invariance, as suggested in [4]. This can be a future line of research.

8 Conclusion

We propose a simple but very effective strategy for calibrating and leveraging the large auxiliary set and the limited data in support set in the few-shot learning task for image classification. Without any training process, our Weighted-distribution Calibration (WC) strategy can be implemented agnostic to the backbone feature extractor and classifier. Our work focus on efficiently and effectively utilizing all possible information when given a specific task. A simple ResNet-12 backbone feature extractor, only pretrained on *mini*ImageNet, combined with a simple Logistic Regression classifier can reach an accuracy of $\mathbf{63.87}\%$ with our WC strategy, which is nearly the state-of-the-art accuracy in FSL.

On the other hand, Vision Transformers did not show effective results in our experiments. As discussed in Section 7.2, unlike CNN which performs well in the presence of small datasets, Vision Transformer itself requires a large amount of dataset to pre-train, and our training set size may not be large enough for the transformer to learn. However, given the aforementioned advantages of Vision Transformer, we believe there are potentials of ViT in few-shot learning. In the future, researches can be done on training transformers with larger datasets to better fit this backbone for few-shot learning.

References

[1] Luca Bertinetto et al. “Meta-learning with differentiable closed-form solvers”. In: *arXiv preprint arXiv:1805.08136* (2018).

[2] Wei-Yu Chen et al. “A closer look at few-shot classification”. In: *arXiv preprint arXiv:1904.04232* (2019).

[3] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: `2010.11929 [cs.CV]`.

[4] Patrick Esser, Robin Rombach, and Björn Ommer. *Taming Transformers for High-Resolution Image Synthesis*. 2020. arXiv: `2012.09841 [cs.CV]`.

[5] Li Fei-Fei, Rob Fergus, and Pietro Perona. “One-shot learning of object categories”. In: *IEEE transactions on pattern analysis and machine intelligence* 28.4 (2006), pp. 594–611.

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. “Model-agnostic meta-learning for fast adaptation of deep networks”. In: *International Conference on Machine Learning* (2017), pp. 1126–1135.

[7] Jonathan Gordon et al. “VERSA: Versatile and efficient few-shot learning”. In: *Advances in Neural Information Processing Systems* (2018), pp. 1–9.

[8] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the knowledge in a neural network”. In: *arXiv preprint arXiv:1503.02531* (2015).

[9] Ruibing Hou et al. “Cross attention network for few-shot classification”. In: *arXiv preprint arXiv:1910.07677* (2019).

[10] Salman H. Khan et al. “Transformers in Vision: A Survey”. In: *arXiv preprint arXiv:2101.01169* (2021).

[11] Brenden Lake et al. “One shot learning of simple visual concepts”. In: *Proceedings of the annual meeting of the cognitive science society*. Vol. 33. 33. 2011.

[12] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. “Attribute-based classification for zero-shot visual object categorization”. In: *IEEE transactions on pattern analysis and machine intelligence* 36.3 (2013), pp. 453–465.

[13] Wenbin Li et al. “Distribution consistency based covariance metric networks for few-shot learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 8642–8649.

[14] Wenbin Li et al. *LibFewShot: A Comprehensive Library for Few-shot Learning*. 2021. arXiv: `2109.04898 [cs.CV]`.

[15] Wenbin Li et al. “Revisiting local descriptor based image-to-class measure for few-shot learning”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7260–7268.

[16] Chen Liu et al. “An Embarrassingly Simple Baseline to One-shot Learning”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2020), pp. 4005–4009.

[17] Aniruddh Raghu et al. “Rapid learning or feature reuse? towards understanding the effectiveness of maml”. In: *arXiv preprint arXiv:1909.09157* (2019).

[18] Jathushan Rajasegaran et al. “Self-supervised knowledge distillation for few-shot learning”. In: *arXiv preprint arXiv:2006.09785* (2020).

[19] Andrei A Rusu et al. “Meta-learning with latent embedding optimization”. In: *arXiv preprint arXiv:1807.05960* (2018).

[20] Adam Santoro et al. “Meta-learning with memory-augmented neural networks”. In: *International conference on machine learning*. PMLR. 2016, pp. 1842–1850.

[21] Jake Snell, Kevin Swersky, and Richard S Zemel. “Prototypical networks for few-shot learning”. In: *arXiv preprint arXiv:1703.05175* (2017).

[22] Flood Sung et al. “Learning to Compare: Relation Network for Few-Shot Learning”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 1199–1208.

[23] John W. Tukey. *Exploratory Data Analysis*. Addison-Wesley, 1977.

[24] Oriol Vinyals et al. “Matching networks for one shot learning”. In: *Advances in neural information processing systems* 29 (2016), pp. 3630–3638.

[25] Shuo Yang, Lu Liu, and Min Xu. “Free Lunch for Few-shot Learning: Distribution Calibration”. In: *International Conference on Learning Representations (ICLR)*. 2021.

[26] Li Zhang, Tao Xiang, and Shaogang Gong. “Learning a deep embedding model for zero-shot learning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2021–2030.