A SIMPLE AND PROVABLE METHOD TO ADAPT PRE-TRAINED MODEL ACROSS DOMAINS WITH FEW SAMPLES

Anonymous authors
Paper under double-blind review

ABSTRACT

Adapting the pre-trained model across domains with few samples, known as cross-domain few-shot learning, is a challenging task in statistical machine learning. Most previous efforts focused on training robust and transferable feature representations but rarely explored how to train an accurate few-shot model from a given pre-trained model. In this paper, we are interested in the performance of training a cross-domain few-shot classifier with representations from different layers of a pre-trained model and the impact of reducing the dimensionality of these representations. Based on this, we propose a simple and provable method, *Average Pooling Ensemble Few-shot Learning (APEF)*. We demonstrate the effectiveness of average pooling and ensemble in cross-domain few-shot image classification both theoretically and experimentally. In particular, we provide a theoretical analysis in the PAC-Bayesian framework to illustrate why our method works, and we also empirically evaluate our approach on the challenging CD-FSL benchmark, which shows that our proposed method consistently outperforms all baselines.

1 INTRODUCTION

The availability of large-scale datasets initiates the development of deep learning methods, whose performance can be continuously improved by annotating more samples from the same(source) domain, but they cannot generalize well to a new(target) domain given a few training samples. In contrast, humans can quickly learn new tasks with a few trials by leveraging what they have learned in the past. Therefore, few-shot learning, aiming to learn from a small number of annotated samples, has gained extensive research attention with some creative works proposed. For example, Zhang et al. (2018); Xian et al. (2018) attempted to synthesize samples or features with a generative model to alleviate the data shortage problem. Vinyals et al. (2016); Finn et al. (2017) trained the model in a meta-learning manner so that the model can be quickly adapted to new tasks by learning general information (meta-knowledge) across tasks. However, most previous works still suffer from insufficient generalization capability when there is a large gap across the source and target domains, which is widely encountered by the deployment in real applications (Chen et al., 2019; Guo et al., 2020).

To help investigate this problem, Guo et al. (2020) introduced a challenging benchmark, CD-FSL, which contains images from agricultural, medical, and satellite domains with a wide range of context, color, and perspective variations. Essentially, the pioneer efforts for the CD-FSL problem can be roughly categorized into two directions: how to pre-train the backbone network to obtain more robust and transferable feature representations, and how to fine-tune the given pre-trained model for the subsequent few-shot learning task. Following the first direction, most previous efforts have been contributed by pre-training the backbone network with various techniques (Tseng et al., 2020; Phoo & Hariharan, 2020; Das et al., 2021; Du et al., 2021). Here, we focus on the less explored second direction, which aims to unlock the potential of the

expensive pre-trained models. Without loss of any generality, we use standard feature representations and methods to train the backbone model that will be fine-tuned for the cross-domain few-shot evaluation.

It is widely accepted that we shall fix the pre-trained backbone to retain the previously learned knowledge and retrain a new head for the target few-shot task. However, significant challenges limit the performance and usage of such methods. First, the features extracted from the last layer of the pre-trained model may be irrelevant to the target task. Based on the Information Bottleneck (IB) principle (Tishby & Zaslavsky, 2015; Shwartz-Ziv & Tishby, 2017), the pre-training process can be viewed as finding a compressed map of the input to maximally preserve information about the outputs (labels) of the source domain. Given the independence of the subsequent new task, the last-layer features of the pre-trained model may exhibit great randomness, while useful features related to the new task have been prematurely filtered. Second, although the maximum likelihood estimator (MLE) is statistically consistent and asymptotically normal, with wide usage to guide the fine-tuning process on the target few-shot tasks, MLE is seriously biased for a small number of samples. Recent research has shown that bias is also related to feature dimensions: the larger the dimension, the more serious the bias (Sur & Candès, 2019). To address both challenges, we propose a simple and provable method, *Average Pooling Ensemble Few-shot Learning (APEF)*, where we first perform average pooling on features obtained on different layers of a pre-trained model, then use them independently to train the learners. The final prediction can be achieved by integrating all independent learners.

Our main contribution can be summarized as follows: (1) We introduce a simple yet efficient method combining average pooling and ensemble for cross-domain few-shot evaluation. (2) We provide a theoretical analysis to illustrate why our proposed method works. The proposed theorems may also benefit the further optimization of cross-domain few-shot learning. (3) Our proposed method achieves superior empirical performance on the challenging CD-FSL benchmark (Guo et al., 2020).

2 Motivation from Empirical Observations

To facilitate our presentation, we first formulate the cross-domain few-shot learning problem. Motivated by the IB principle, we compare the performance of classifiers implemented on the features extracted from different layers of the pre-trained model. Finally, some insights are provided by adopting multiple dimensionality reduction methods.

2.1 Problem Formulation

We define a domain as a joint distribution P over input space $\mathcal{X}$ and label space $\mathcal{Y}$. $P_{\mathcal{X}}, P_{\mathcal{Y}}$ represents the marginal distribution of $\mathcal{X}, \mathcal{Y}$ respectively. In cross-domain few-shot setting, we have a source domain $(\mathcal{X}_s, \mathcal{Y}_s) \sim P_s$, and a target domain $(\mathcal{X}_t, \mathcal{Y}_t) \sim P_t$, where $P_{\mathcal{X}_s}$ is significantly different form $P_{\mathcal{X}_t}$, and $\mathcal{Y}_s$ is disjoint from $\mathcal{Y}_t$. The source dataset D_s is sampled from the source domain and used to pre-train or meta-train the model. The target dataset D_t consists of a support set $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N\times K}$ and a query set $Q = \{\boldsymbol{x}_i\}_{i=1}^{N\times H}$ sampled from the target domain. The support set S is used to adapt the pre-trained model, and the query set Q is used to evaluate the performance of the adapted model. Because the support set S contains K data points from N new classes, this configuration is called the N-way K-shot problem.

2.2 Effects of different layers on the performance of cross-domain few-shot learning

To help understand the optimization process DNNs, (Tishby & Zaslavsky, 2015; Shwartz-Ziv & Tishby, 2017) recently propose the Information Bottleneck (IB) principle, which formulates the training process of deep learning as an information-theoretic trade-off between compression and prediction. The IB principle treats the function represented by a neural network as a Markov chain, which successively transforms representations of the original input feature. Each network layer can be quantified by the amount of mutual information

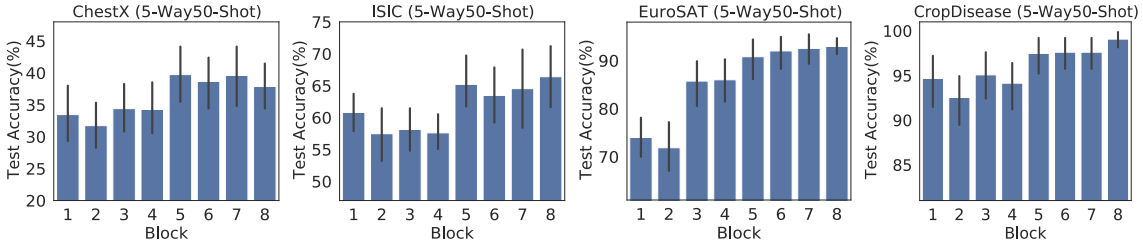

Figure 1: The performance of few-shot classifiers when applied on top of representation from different residual blocks of a ResNet-18 network pre-trained on ImageNet.

between its input variables and output variables. Under this schema, it is believed that by incorporating supervised learning, deep neural networks can capture and effectively represent relevant information from the input variables and maximally preserve the information of the output (label) variables. However, the problem of large domain shift remains challenging. Since the independent downstream tasks may show significant domain shifts, we argue that the relationship between the features from each layer of a pre-trained model and the expected output of the target domain may exhibit a large amount of randomness. To investigate this conjecture, we train multiple classifiers on features extracted from the different residual blocks of the ResNet-18 pre-trained on ImageNet. Figure 1 visualizes the performance of CD-FSL benchmark (Guo et al., 2020) under the 5-Way 50-Shot setting, respectively. It shows that representations from intermediate blocks perform competitively or even better than those of the last block (more results are provided in Appendix D.1). Similar experimental results are also reported by Yosinski et al. (2014); Neyshabur et al. (2020); Adler et al. (2020); Baldock et al. (2021); Abnar et al. (2021).

2.3 The trade-off between dimensionality reduction and performance

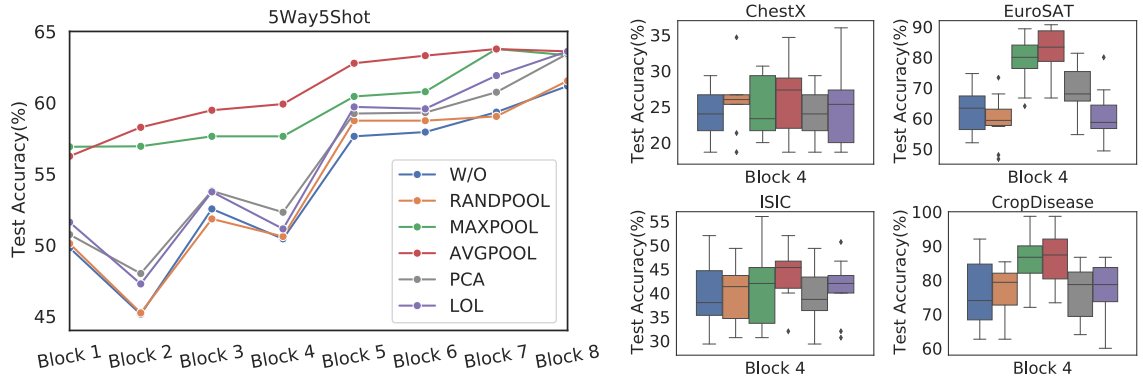

Figure 2: Dimensionality reduction methods produce consistent improvements across different blocks. The experiments are performed on a ResNet-18 network pre-trained on ImageNet, with the few-shot evaluation set to 5-way 5-shot. *Left*: Average test accuracy of four target tasks for each block. *Right*: The four small graphs correspond to the test accuracy of four target tasks on block 4, respectively. (W/O denotes no dimensionality reduction method is used.)

During the above investigation, we found that features extracted from the early layers of the pre-trained model are usually located in high-dimensional spaces. For reliable generalization, the amount of data needed grows exponentially with dimension (Verleysen & François, 2005). This phenomenon is known as the curse of dimensionality. Feature selection and dimensionality reduction methods are widely used to address this problem. Next, we focus on the impact of commonly used dimensionality reduction methods on the few-shot classification problem. More specifically, before training a few-shot classifier, we apply the dimensionality reduction methods to the representations obtained from each residual block to reduce their dimension while

preserving their geometry structure. We selected five classical and effective dimensionality reduction methods, including random pooling (RANDPOOL), maximum pooling (MAXPOOL), average Pooling (AVGPOOL), Principal Component Analysis (PCA), and Linear Optimal Low-Rank Projection (LOL) (Vogelstein et al., 2021). As shown in Figure 2, almost all dimensionality reduction methods improve test accuracy per block, with average pooling achieving the most significant improvement (see appendix D.2 for more results).

3 A PAC-BAYESIAN BASED GENERALIZATION BOUND FOR APEF

Based on the above findings, we are motivated to propose a simple and efficient algorithm, *Average Pooling Ensemble Few-shot Learning (APEF)*. We independently train multiple cross-domain few-shot learners on top of representations from different layers while applying **average pooling** to improve the learners' performance. All learners are integrated through an average **ensemble** model to utilize the pre-trained model's information fully. The complete pseudo-code of our method is outlined in the Algorithm 1.

We further provide a theoretical view based on the PAC-Bayesian framework to demonstrate why our proposed APEF works. To simplify our explanation, given a data distribution P over $\mathcal{X} \times \mathcal{Y}$, we define a predictor $h_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$ with parameter $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, and the expected and empirical loss can be defined as,

$$L\left(\boldsymbol{\theta}\right) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\ell\left(\boldsymbol{\theta}, \boldsymbol{x}, y\right)\right], \qquad \text{and} \qquad \hat{L}(\boldsymbol{\theta}, D) = \frac{1}{n}\sum_{i=1}^{n} \ell\left(\boldsymbol{\theta}, \boldsymbol{x}_i, y_i\right),$$

where $D = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n} \sim P^n$ denotes the *i.i.d.* observation of n elements, $\ell : \boldsymbol{\Theta} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ is an arbitrary loss function (*e.g.* the *cross-entropy* (CE) loss). In the PAC-Bayesian setting, we assume that the predictor $h_{\boldsymbol{\theta}}$ has prior knowledge of the hypothesis space $\boldsymbol{\Theta}$ in the form of a prior distribution π. After the training dataset D is fed to the predictor, the prior is updated to a posterior distribution ρ.

The PAC-Bayesian theory (McAllester, 1999) provides data-dependent generalization guarantees for a model's generalization error $\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})]$ (also known as Gibbs error), given the empirical estimate of $\mathbb{E}_{\rho(\boldsymbol{\theta})}[\hat{L}(\boldsymbol{\theta}, D)]$ and other parameters. The full bound theorem is restated below, derived from the theorems in Germain et al. (2009); Alquier et al. (2016); Masegosa (2020), and we give the proof in appendix A.1 for completeness.

Theorem 3.1 (Germain et al. (2009); Alquier et al. (2016); Masegosa (2020))**.** *Given a data distribution P over $\mathcal{X} \times \mathcal{Y}$, a hypothesis set $\boldsymbol{\Theta}$, a prior distribution π over $\boldsymbol{\Theta}$, for any $\delta \in (0, 1]$, and $\lambda > 0$, with probability at least $1 - \delta$ over samples $D \sim P^n$, we have for all posterior ρ,*

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \leq \mathbb{E}_{\rho(\boldsymbol{\theta})}[\hat{L}(\boldsymbol{\theta}, D)] + \frac{1}{\lambda}\left[D_{\mathrm{KL}}(\rho\|\pi) + \log\frac{1}{\delta} + \Psi_{P,\pi}(\lambda, n)\right],$$

where $\Psi_{P,\pi}(\lambda, n) = \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{\lambda(L(\boldsymbol{\theta}) - \hat{L}(\boldsymbol{\theta}, D))}\right]$.

The above theory provides a well-founded approach. Since the PAC-Bayesian bound applies simultaneously to all posteriors ρ, we can learn the algorithm by choosing an appropriate distribution ρ that minimizes the upper bound on the risk of generalization. In the following content, motivated by previous works (Masegosa, 2020; Ortega et al., 2022), we first derive the relationship between the diversity of our ensemble and generalization error and show that our model achieves a tighter upper bound by implicitly optimizing diversity. Furthermore, we demonstrate the importance of dimensionality reduction in few-shot problems by analyzing the Kullback-Leibler divergence term. Finally, we use a technique inspired by Grønlund et al. (2020) to explain that average pooling is an effective dimensionality reduction method.

3.1 A TIGHTER BOUND BY IMPLICITLY OPTIMIZING DIVERSITY WITH ENSEMBLE

In this paper, we consider an ensemble model consisting of M distinct models $\{\boldsymbol{\theta}_i\}_{i=1}^{M}$, each one defined by a set of parameters $\boldsymbol{\Theta}_i \in \mathbb{R}^{d_i}$. Specifically, we make predictions by simply averaging the predictions of M mod-

els, $\frac{1}{M}\sum_{i=1}^{M} p(y|\boldsymbol{x},\boldsymbol{\theta}_i)$. For a specific data-point $(\boldsymbol{x}, y)$, the CE loss of an individual predictor $h_{\boldsymbol{\theta}_i}$ is defined as $\ell(\boldsymbol{\theta}_i, \boldsymbol{x}, y) = -\log p(y|\boldsymbol{x},\boldsymbol{\theta}_i)$ and the CE loss of an ensemble is $\ell(\boldsymbol{\theta}, \boldsymbol{x}, y) = -\log \frac{1}{M}\sum_{i=1}^{M} p(y|\boldsymbol{x},\boldsymbol{\theta}_i)$. To simplify the notation, we denote $L(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\ell(\boldsymbol{\theta}, \boldsymbol{x}, y)\right]$ as the expected loss of an ensemble.

In the following **Theorem 3.2**, we show that the upper bound on the expected loss $L(\boldsymbol{\theta})$ of an averaging ensemble can be decomposed into the average loss of the individual models and a second-order term, which can be considered as a diversity measure. Meanwhile, we introduce a second-order PAC-Bayesian bound for the averaging ensemble model in **Theorem 3.3**, which also provides generalization guarantees over the performance of the posterior predictive distribution (See Appendix A.2, A.3 for full proof).

Theorem 3.2. *(Second-order Oracle bound) Given a data distribution P, a set of model parameters $\{\boldsymbol{\Theta}_i\}_{i=1}^{M}$, for any distribution $\{\rho_i\}_{i=1}^{M}$ over $\{\boldsymbol{\Theta}_i\}_{i=1}^{M}$ satisfies that,*

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \leq \frac{1}{M}\sum_{i=1}^{M} \mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[L(\boldsymbol{\theta}_i)] - \mathbb{V}(\rho(\boldsymbol{\theta})),$$

where $\boldsymbol{\theta}_i \in \boldsymbol{\Theta}_i, \boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^{M}, \rho(\boldsymbol{\theta}) = \prod_{i=1}^{M} \rho_i(\boldsymbol{\theta}_i)$, and $\mathbb{V}(\rho(\boldsymbol{\theta}))$ is a variance term defined as

$$\mathbb{V}(\rho(\boldsymbol{\theta})) = \mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{2M\max_{\boldsymbol{\theta}} p(y|\boldsymbol{x},\boldsymbol{\theta})^2}\sum_{i=1}^{M}\left(p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x},\boldsymbol{\theta}_k)\right)^2\right].$$

Theorem 3.3. *Given a data distribution P, a set of model parameters $\{\boldsymbol{\Theta}_i\}_{i=1}^{M}$ and associated priors $\{\pi_i\}_{i=1}^{M}$, where π_i is defined over $\boldsymbol{\Theta}_i$, a $\delta \in (0, 1]$, and a real number $c > 0$, with probability at least $1-\delta$ over draws of training data $D \sim P^n$, for all posteriors $\{\rho_i\}_{i=1}^{M}$ over $\{\boldsymbol{\Theta}_i\}_{i=1}^{M}$, simultaneously,*

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \leq \frac{1}{M}\sum_{i=1}^{M}\left(\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i, D)] + \frac{D_{\mathrm{KL}}(\rho_i\|\pi_i)}{cn}\right) - \hat{\mathbb{V}}(\rho(\boldsymbol{\theta}), D) + \frac{\epsilon}{cnM},$$

where $\hat{\mathbb{V}}(\rho(\boldsymbol{\theta}), D)$ is the empirical version of $\mathbb{V}(\rho(\boldsymbol{\theta}))$, $\mathbb{V}(\rho(\boldsymbol{\theta})) = \mathbb{E}_{\rho(\boldsymbol{\theta})}[\mathbb{V}(\boldsymbol{\theta})]$, and $\epsilon(\mathcal{P}, \pi, c, n, \delta) = \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{cn\left(\sum_{i=1}^{M}\left(L(\boldsymbol{\theta}_i) - \hat{L}(\boldsymbol{\theta}_i, D)\right) - M\left(\mathbb{V}(\boldsymbol{\theta}) - \hat{\mathbb{V}}(\boldsymbol{\theta}, D)\right)\right)}\right] + \log\frac{1}{\delta}$.

According to the above theorem, we need to balance how well each model fits the training data, the Kullback-Leibler divergence, and diversity among models to learn the optimal ensemble. Here, the second-order term $\mathbb{V}(\rho(\boldsymbol{\theta}))$ is used to measure the diversity among the predictions of all models following Masegosa (2020); Ortega et al. (2022). It is widely noted that the diversity within an ensemble is a key factor for its superior performance (Geman et al., 1992; Krogh & Vedelsby, 1994; Cunningham & Carney, 2000; Brown et al., 2005). Intuitively, a set of predictors are diverse when their predictions for some samples are inconsistent. Conversely, when all models provide the same predictions, $\mathbb{V}(\rho(\boldsymbol{\theta}))$ is null, with no gain by averaging these sub-models. In consequence, when the ensemble consists of M independent models with different dimensional parameters, the expected diversity $\mathbb{V}(\rho(\boldsymbol{\theta}))$ will be positive, as the following lemma stated:

Lemma 3.4. *If there exists an input sample $\boldsymbol{x} \in P_{\mathcal{X}}$ such that $h_{\boldsymbol{\theta}_i}(\boldsymbol{x}) \neq h_{\boldsymbol{\theta}_j}(\boldsymbol{x})$, we then have that $\mathbb{V}(\rho(\boldsymbol{\theta})) > 0$.*

Our method uses features from different blocks to train M models independently. Although we do not explicitly optimize for the diversity of ensembles, from the above theorem, our optimization process is naturally looking for a tighter upper bound of the expected loss of the averaging ensemble.

3.2 The importance and effectiveness of dimensionality reduction

Since the number of training samples n in the target task is small, the lower-order term $(cn)^{-1}D_{\mathrm{KL}}(\rho_i\|\pi_i)$ in the PAC-Bayesian bound would produce a non-negligible generalization gap that grows with $O(1/n)$. To

further analysis this term, we assume $\boldsymbol{\theta}_i \in \mathbb{R}^{d_i}$, $\pi_i(\boldsymbol{\theta}_i) \sim \mathcal{N}(\mathbf{0}, \sigma^2 \boldsymbol{I})$, and $\rho_i(\boldsymbol{\theta}_i)$ as a Dirac-delta distribution centered around $\boldsymbol{\theta}_i'$ with $\rho_i(\boldsymbol{\theta}_i) = \delta_{\boldsymbol{\theta}_i'}(\boldsymbol{\theta}_i), \forall i \in [M]$. Then, we have

$$\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i, D)] = \mathbb{E}_{\delta_{\boldsymbol{\theta}_i'}(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i, D)] = \int \delta_{\boldsymbol{\theta}_i'} \hat{L}(\boldsymbol{\theta}_i, D)\, d\boldsymbol{\theta}_i = \hat{L}(\boldsymbol{\theta}_i', D),$$

and

$$D_{\mathrm{KL}}(\rho_i \| \pi_i) = \int \delta_{\boldsymbol{\theta}_i'}(\boldsymbol{\theta}_i) \log \frac{\delta_{\boldsymbol{\theta}_i'}(\boldsymbol{\theta}_i)}{\pi(\boldsymbol{\theta}_i)}\, d\boldsymbol{\theta}_i = -\log \pi(\boldsymbol{\theta}_i') = \frac{d_i}{2} \log(2\pi\sigma^2) + \frac{1}{2\sigma^2} \|\theta_i'\|^2.$$

Hence, the upper bound of the Theorem 3.3 can be expressed as

$$\frac{1}{M} \sum_{i=1}^{M} \left(\hat{L}(\boldsymbol{\theta}_i', D) + \frac{1}{2cn\sigma^2} \|\theta_i'\|^2 + \frac{d_i}{2cn} \log(2\pi\sigma^2) \right) - \hat{\mathbb{V}}(\rho(\boldsymbol{\theta}), D) + \frac{\epsilon}{cnL}. \tag{1}$$

The balance between empirical expected loss and Kullback-Leibler divergence can be reformulated as a trade-off between empirical expected loss with a penalty of L2 norm and dimensionality of model parameters. Specifically, for fixed $\pi(\boldsymbol{\theta}), D, c, n$, and δ, minimizing Equation 1 is equivalent to find $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^{M}$ by $\min_{\boldsymbol{\theta}} \sum_{i=1}^{M} \left(\hat{L}(\boldsymbol{\theta}_i, D) + \lambda_1 \|\boldsymbol{\theta}_i\|^2 + \lambda_2 d_i \right) / M$, where d_i is the dimension of $\boldsymbol{\theta}_i$, $\lambda_1, \lambda_2 > 0$ are the hyper-parameters. Since the parameter dimension is not differentiable, we adopt a two-step method based on the idea of the search algorithm. We first perform a dimensionality reduction operation on each layer's features to reduce each classifier's parameter dimension. Each classifier is then independently trained with a loss function with L2 penalty terms.

Dimensionality reduction can also be viewed as restricting the solution space to a linear subspace of the original high dimensional parameter space. In the following, we show that average pooling is a simple and effective method for dimensionality reduction, where the new optimal solution can be as close as possible to the minimum training loss optimized in the original high-dimensional space, especially for image tasks.

Let $f_{\boldsymbol{U}_k} : \mathbb{R}^d \to \mathbb{R}^k$ denote a function of random pooling, where $\boldsymbol{U}_k$ is a projection matrix sampled from $\mathcal{Q}_k$, the projection matrix of average pooling is defined as the expectation of $\boldsymbol{U}_k$ ($\mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})]$) (A detailed definition is provided in Appendix A.4). We use $\boldsymbol{U}_k(\boldsymbol{x})$ instead of $f_{\boldsymbol{U}_k}(\boldsymbol{x})$ for simplicity. The next theorem shows that for any pair $(\boldsymbol{x}, y)$, either sampled from P or a sampled set $S \in P^m$, we can find a $\tilde{\boldsymbol{w}} \in \mathbb{R}^k$ such that the values of $\langle \boldsymbol{x}, \boldsymbol{w} \rangle$ and $\langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}} \rangle$ are not to be too far apart with very high probability.

Theorem 3.5 (Inner Product Preservation)**.** *Let $R, d, k \in \mathbb{N}^+$, and $d > k$. Denote by $\mathcal{X}$ the ball of radius R in $\mathbb{R}^d$, and let P be any data distribution over $\mathcal{X} \times \mathcal{Y}$. For every $\boldsymbol{w} \in \mathbb{R}^d$ with $\|\boldsymbol{w}\|_2 \leq 1$, and $\delta > 0$, there exist a $\tilde{\boldsymbol{w}} \in \mathcal{H}$ satisfying,*

$$\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k} \left[|\langle \boldsymbol{x}, \boldsymbol{w} \rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}} \rangle| \geq \delta \right] \leq 4 \exp(-\frac{\delta^2 k}{8d^2 R^2}) + \frac{(d+k)^2}{\delta^2 k^2} \sum_{j=1}^{k} \mathrm{Var}(\boldsymbol{x}_j),$$

and for every $S \in \mathrm{supp}\,(P^n)$,

$$\Pr_{(\boldsymbol{x},y)\sim S, \boldsymbol{U}_k \sim \mathcal{Q}_k} \left[|\langle \boldsymbol{x}, \boldsymbol{w} \rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}} \rangle| \geq \delta \right] \leq 4 \exp(-\frac{\delta^2 k}{8d^2 R^2}) + \frac{(d+k)^2}{\delta^2 k^2} \sum_{j=1}^{k} \mathrm{Var}(\boldsymbol{x}_j),$$

where $\mathrm{Var}(\boldsymbol{x}_j)$ represents the element-wise variance of batch j of $\boldsymbol{x}$.

The above results show that the probability of inner product preservation is related to the dimension ratio of the pooling operation and the element-variance of the input. More specifically, the smaller the percentage of dimensionality reduction, the more significant the difference between the input elements, then the inner product will be distorted with a high probability. Since images are always piecewise constant (Chan & Vese, 2001), choosing an appropriate pooling kernel size can reduce the variance between input elements. But in general, there exists a trade-off between dimensionality reduction and generalization in the few-shot problem.

4 RELATED WORKS

Few-shot learning (FSL) algorithms can be roughly divided into three categories: generative-based, meta-learning-based, and transfer-learning-based methods. Generative-based methods focus on alleviating data shortages by data augmentation. Most methods implement GANs (Goodfellow et al., 2014) or autoencoder from the source domain and use them to generate samples (Zhang et al., 2018) or features (Xian et al., 2018) for new classes. Meta-learning-based methods aim to quickly adapt to new tasks by learning general information across tasks. It usually includes metric-based and optimization-based methods. Metric-based methods look for suitable learning metrics to judge data similarity for novel classes. Examples of distance metrics include cosine similarity for MatchingNet (Vinyals et al., 2016), Euclidean distance between class feature means for ProtoNet (Snell et al., 2017), CNN-based relational networks for RelationNet (Sung et al., 2018), and linear classification rules for MetaOpt (Lee et al., 2019). Optimization-based methods focus on using prior knowledge to influence the update of model parameters, e.g., MAML (Finn et al., 2017) aims to find a good initialized parameter. Transfer-learning-based methods are based on the core idea of feature reuse and are mainly performed by fine-tuning. The most common practice is to use the pre-trained backbone as a fixed feature extractor, and the obtained high-dimensional feature vectors are used to learn the target task.

Cross-domain few-shot learning (CD-FSL) focuses on the FSL problem with large gaps between source and target domains. Chen et al. (2019); Guo et al. (2020) found that simple fine-tuning methods significantly outperform most meta-learning-based methods when faced with CD-FSL problems. Previous CD-FSL approaches can be roughly categorized into two directions. One is to pre-train a more robust and transferable backbone. For example, FWT (Tseng et al., 2020) introduced a feature-wise transformation layer on top of features to model cross-domain distributions. STARTUP (Phoo & Hariharan, 2020) assumed that many additional unlabeled data from the target domain are available for pre-training. HVM (Du et al., 2021) proposed a hierarchical variational inference framework to optimize and store features at different semantic levels. Another direction focuses on fine-tuning the given pre-trained model for the subsequent few-shot learning task, e.g., CHEF (Adler et al., 2020) applying a fusion of Hebbian learners to increase the importance of low and mid-level features. Additional related works are left to Appendix C.

5 EXPERIMENTS

In this section, we conduct extensive experiments to compare the performance of our proposed method with the state-of-the-art methods on four cross-domain few-shot challenges. Additional results and more details about the datasets, experiment setup, baselines, and model architectures are presented in the Appendix E.

5.1 EXPERIMENTAL SETUP

Datasets We evaluate our algorithm on the CD-FSL benchmark (Guo et al., 2020). The benchmark uses mini-ImageNet (Vinyals et al., 2016), or ImageNet (Deng et al., 2009) as the source domain and evaluates the pre-trained model on four different target domains with only a few labeled data. These four target domains include data from the CropDiseases (Mohanty et al., 2016), EuroSAT (Helber et al., 2019), ISIC2018 (Tschandl et al., 2018; Codella et al., 2019), and ChestX (Wang et al., 2017) datasets, which cover plant disease images, satellite images, dermoscopic images of skin lesions, and X-ray images. More details about the datasets are provided in the Appendix E.1. We evaluate 5-way k-shot classification tasks for $k \in \{5, 20, 50\}$ and report the average accuracy($\%$, top-1) and 95% confidence interval over 600 few-shot episodes following (Guo et al., 2020). Each episode contains randomly sampled 5 classes and k samples per class for adaptation, and 15 query samples per class for evaluation.

Implementation details For a fair comparison, we use the ResNet-10 backbone architecture and adopt the publicly accessed code to pre-train this backbone (Guo et al., 2020). Specifically, during the pre-training stage,

we train the network on the miniImageNet dataset for 400 epochs by the Adam optimizer with a learning rate of 0.001 and a batch size of 16. During the few-shot evaluation phase, we apply AdaptiveAvgPool2d to reduce the dimension of features and train few-shot classifiers by using the official logistic regression (LR) implementation of scikit-learn (Pedregosa et al., 2011). Finally, we apply the average ensemble for all classifiers to predict. For all target tasks, we set the maximum number of iterations of LR to 1000, the output size of AdaptiveAvgPool2d to 1, and the ensemble selects the middle residual block to the last residual block. For example, ResNet-18 has a total of 8 residual blocks. We use the features of block 5 to block 8, reduce their dimensionality, and independently train distinct learners for the future average ensemble.

5.2 Results and Ablation studies

Table 1: Experimental results on four cross-domain few-shot challenges. The average accuracy and 95% confidence interval of 600 runs are reported. †, *, and ⋆ denotes results reported by Guo et al. (2020), Adler et al. (2020) and Du et al. (2021) respectively. The runner-up method is underlined.

	ChestX		ISIC		EuroSAT		CropDiseases	
Method	5-way 5-shot	5-way 20-shot	5-way 5-shot	5-way 20-shot	5-way 5-shot	5-way 20-shot	5-way 5-shot	5-way 20-shot
MatchingNet†	22.40 ± 0.70	23.61 ± 0.86	36.74 ± 0.53	45.72 ± 0.53	64.45 ± 0.63	77.10 ± 0.57	66.39 ± 0.78	76.38 ± 0.67
MatchingNet + FWT†	21.26 ± 0.31	23.23 ± 0.37	30.40 ± 0.48	32.01 ± 0.48	56.04 ± 0.65	63.38 ± 0.69	62.74 ± 0.90	74.90 ± 0.71
MAML†	23.48 ± 0.96	27.53 ± 0.43	40.13 ± 0.58	52.36 ± 0.57	71.70 ± 0.72	81.95 ± 0.55	78.05 ± 0.68	89.75 ± 0.42
ProtoNet†	24.05 ± 1.01	28.21 ± 1.15	39.57 ± 0.57	49.50 ± 0.55	73.29 ± 0.71	82.27 ± 0.57	79.72 ± 0.67	88.15 ± 0.51
ProtoNet + FWT†	23.77 ± 0.42	26.87 ± 0.43	38.87 ± 0.52	43.78 ± 0.47	67.34 ± 0.76	75.74 ± 0.70	72.72 ± 0.70	85.82 ± 0.51
RelationNet†	22.96 ± 0.88	26.63 ± 0.92	39.41 ± 0.58	41.77 ± 0.49	61.31 ± 0.72	74.43 ± 0.66	68.99 ± 0.75	80.45 ± 0.64
RelationNet + FWT†	22.74 ± 0.40	26.75 ± 0.41	35.54 ± 0.55	43.31 ± 0.51	61.16 ± 0.70	69.40 ± 0.64	64.91 ± 0.79	78.43 ± 0.59
MetaOpt†	22.53 ± 0.91	25.53 ± 1.02	36.28 ± 0.50	49.42 ± 0.60	64.44 ± 0.73	79.19 ± 0.62	68.41 ± 0.73	82.89 ± 0.54
Fixed†	25.35 ± 0.96	30.83 ± 1.05	43.56 ± 0.60	52.78 ± 0.58	75.69 ± 0.66	84.13 ± 0.52	87.48 ± 0.58	94.45 ± 0.36
CHEF⋆	24.72 ± 0.14	29.71 ± 0.27	41.26 ± 0.34	54.34 ± 0.34	74.15 ± 0.27	83.31 ± 0.14	86.87 ± 0.27	94.78 ± 0.12
HVM⋆	**27.15 ± 0.45**	30.54 ± 0.47	42.05 ± 0.34	54.97 ± 0.35	74.88 ± 0.45	84.81 ± 0.34	87.65 ± 0.35	95.13 ± 0.35
Ours	26.43 ± 0.44	**32.62 ± 0.45**	**44.02 ± 0.53**	**56.94 ± 0.57**	**81.65 ± 0.65**	**89.34 ± 0.44**	**91.48 ± 0.47**	**96.65 ± 0.27**

Comparison to State-of-the-arts Table 1 shows the performance comparison of our method with other methods on the CD-FSL benchmark. Our proposed method achieves state-of-the-art performance in 7 out of 8 categories. Compared with meta-learning-based methods, the performance of our method is greatly improved in all settings. For example, under the 5-way 5-shot, our method yields an improvement of **9.90%, 9.69%, 11.41%, 14.75%** over the state-of-the-art meta-learning-based method. This result shows that our method can better handle extremely cross-domain few-shot problems. Compared to other non-meta-learning-based methods, our method also shows strong competitiveness. Specifically, under the 5-way 20-shot, our method outperforms the runner-up method by **1.79%, 1.97%, 4.53%, 1.52%** on ChestX, ISIC, EuroSAT and CropDisease, respectively. Furthermore, we can see that the improvement increases as the number of shots on the ChestX and ISIC datasets increases. Results under 5-way 50-shot are provided in Appendix E.4.

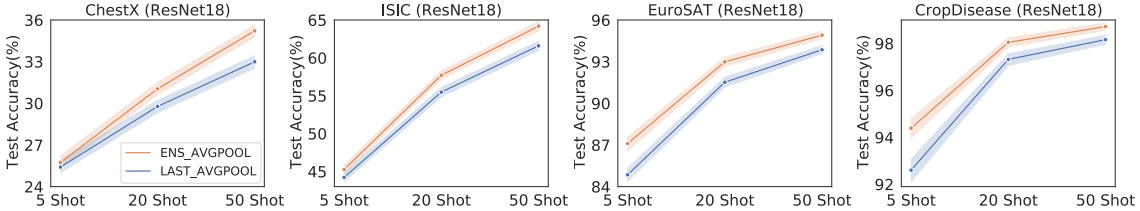

Figure 3: Benefits of the ensemble on four cross-domain challenges. The ensemble is trained on the ImageNet pre-trained ResNet-18 network. Results under different pre-trained networks are provided in Appendix E.5. Better results were obtained under all shots on all datasets using the ensemble.

Benefits of an Ensemble model To show the benefit of the ensemble, we compare our method with a few-shot classifier trained using only the dimensionality-reduced features of the last block. Figure 3 shows that the ensemble model achieves consistent improvement on all cross-domain few-shot classification tasks. Furthermore, we find that on ChestX and ISIC, two tasks with a larger domain shift gap, the advantage of the ensemble model is more pronounced as the number of samples increases. More results with different pre-trained backbones and the benefit of independent training are reported in Appendix E.5, E.6.

Benefits of Average Pooling We further investigate the benefits of using average pooling. The experimental result are provided in Table 2. Our proposed method achieves better results on ChestX, EuroSAT, and CropDiseases. Especially on the EuroSAT and CropDiseases, the improvements over **6.18**% and **4.62**% are obtained, respectively. While the ensemble without average pooling on ISIC shows better performance, it requires more than 20 times the training time (Figure 4). We also conduct an ablation experiment to visualize the effect of reducing the ISIC few-shot task to different dimensions. As shown in Figure 4(c), we find that choosing the appropriate output size of AVGPOOL can significantly improve performance. But how to select the better output size under few-shot problems is non-trivial. We leave it for future exploration.

Table 2: Ablation studies under 5-way 5-shot and the pre-trained ResNet-18 model. Average test accuracy and 95% confidence intervals of 600 runs are reported.

ENS	AVGPOOL	ChestX	ISIC	EuroSAT	CropDiseases
	✓	25.41 ± 0.42	44.22 ± 0.58	84.85 ± 0.49	92.60 ± 0.46
✓		25.60 ± 0.43	**45.45** ± 0.56	80.92 ± 0.57	89.77 ± 0.55
✓	✓	**25.75** ± 0.43	45.26 ± 0.58	**87.10** ± 0.49	**94.39** ± 0.41

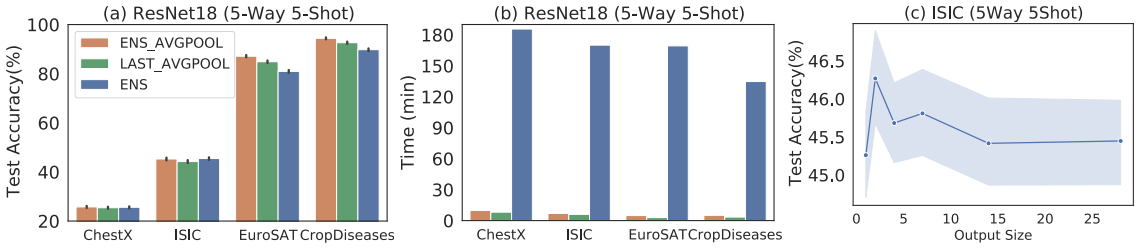

Figure 4: (*Left:*) Average test accuracy and (*Middle:*) Time usage for four cross-domain challenges under 5-way 5-shot and the pre-trained ResNet-18 model. (*Right:*) The effect of different output sizes of AdaptiveAvgPool2d on ISIC few-shot task (see Appendix E.7 for other tasks).

6 Conclusion

This paper explores the performance of a cross-domain few-shot classifier on top of representations from different layers of a pre-trained model and the impact of applying dimensionality reduction to these representations. We find that dimensionality reduction consistently improves for few-shot, while the intermediate layer information of pre-trained models may be more valuable for cross-domain. Based on this finding, we propose a simple and effective method, *Average Pooling Ensemble Few-shot Learning (APEF)*. We apply average pooling on top of representations from different layers of the pre-trained model, use reduced features to train the few-shot classifiers, and finally integrate all classifiers through the average ensemble model for prediction. The evaluation of a challenging benchmark, CD-FSL, with different pre-trained networks and the comparison with some state-of-the-art algorithms show the effectiveness of our method. More importantly, we provide a theoretical analysis under the PAC-Bayesian framework to demonstrate why our method works. We also show that average pooling is an effective dimensionality reduction method for visual recognition tasks. Although we theoretically and empirically demonstrate the advantages of average pooling and ensemble for the cross-domain few-shot problem, whether there exists a better strategy for ensemble and dimensionality reduction remains an unresolved problem and is left for our future exploration.

REFERENCES

Samira Abnar, Mostafa Dehghani, Behnam Neyshabur, and Hanie Sedghi. Exploring the limits of large scale pre-training. *arXiv preprint arXiv:2110.02095*, 2021.

Thomas Adler, Johannes Brandstetter, Michael Widrich, Andreas Mayr, David Kreil, Michael Kopp, Günter Klambauer, and Sepp Hochreiter. Cross-domain few-shot learning by representation fusion. *arXiv preprint arXiv:2010.06498*, 2020.

Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of gibbs posteriors. *The Journal of Machine Learning Research*, 17(1):8374–8414, 2016.

Robert Baldock, Hartmut Maennel, and Behnam Neyshabur. Deep learning through the lens of example difficulty. *Advances in Neural Information Processing Systems*, 34, 2021.

Gavin Brown, Jeremy L Wyatt, Peter Tino, and Yoshua Bengio. Managing diversity in regression ensembles. *Journal of machine learning research*, 6(9), 2005.

Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on image processing*, 10(2):266–277, 2001.

Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *arXiv preprint arXiv:1904.04232*, 2019.

Noel Codella, Veronica Rotemberg, Philipp Tschandl, M Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, et al. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (isic). *arXiv preprint arXiv:1902.03368*, 2019.

Padraig Cunningham and John Carney. Diversity versus quality in classification ensembles based on feature selection. In *European Conference on Machine Learning*, pp. 109–116. Springer, 2000.

Debasmit Das, Sungrack Yun, and Fatih Porikli. Confess: A framework for single source cross-domain few-shot learning. In *International Conference on Learning Representations*, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Yingjun Du, Xiantong Zhen, Ling Shao, and Cees GM Snoek. Hierarchical variational memory for few-shot learning across domains. *arXiv preprint arXiv:2112.08181*, 2021.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58, 1992.

Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. Pac-bayesian learning of linear classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 353–360, 2009.

Saba Ghaffari, Ehsan Saleh, David Forsyth, and Yu-Xiong Wang. On the importance of firth bias reduction in few-shot classification. *arXiv preprint arXiv:2110.02529*, 2021.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Allan Grønlund, Lior Kamma, and Kasper Green Larsen. Near-tight margin-based generalization bounds for support vector machines. In *International Conference on Machine Learning*, pp. 3779–3788. PMLR, 2020.

Yunhui Guo, Noel C Codella, Leonid Karlinsky, James V Codella, John R Smith, Kate Saenko, Tajana Rosing, and Rogerio Feris. A broader study of cross-domain few-shot learning. In *European Conference on Computer Vision*, pp. 124–141. Springer, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4700–4708, 2017.

Ashraful Islam, Chun-Fu Richard Chen, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, and Richard Radke. Dynamic distillation network for cross-domain few-shot recognition with unlabeled data. *Advances in Neural Information Processing Systems*, 34, 2021.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, 7, 1994.

Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10657–10665, 2019.

Xu Luo, Jing Xu, and Zenglin Xu. Channel importance matters in few-shot image classification. In *International Conference on Machine Learning*, pp. 14542–14559. PMLR, 2022.

Andres Masegosa. Learning under model misspecification: Applications to variational and ensemble methods. *Advances in Neural Information Processing Systems*, 33:5479–5491, 2020.

David A McAllester. Some pac-bayesian theorems. *Machine Learning*, 37(3):355–363, 1999.

Sharada P Mohanty, David P Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7:1419, 2016.

Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International Conference on Machine Learning*, pp. 2554–2563. PMLR, 2017.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523, 2020.

Luis A Ortega, Rafael Cabañas, and Andres Masegosa. Diversity and generalization in neural network ensembles. In *International Conference on Artificial Intelligence and Statistics*, pp. 11720–11743. PMLR, 2022.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Cheng Perng Phoo and Bharath Hariharan. Self-training for few-shot transfer across extreme task differences. *arXiv preprint arXiv:2010.07734*, 2020.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018.

Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogerio Feris, Raja Giryes, and Alex Bronstein. Delta-encoder: an effective sample synthesis method for few-shot object recognition. *Advances in neural information processing systems*, 31, 2018.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1199–1208, 2018.

Pragya Sur and Emmanuel J Candès. A modern maximum-likelihood theory for high-dimensional logistic regression. *Proceedings of the National Academy of Sciences*, 116(29):14516–14525, 2019.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 ieee information theory workshop (itw)*, pp. 1–5. IEEE, 2015.

Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018.

Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. *arXiv preprint arXiv:2001.08735*, 2020.

Michel Verleysen and Damien François. The curse of dimensionality in data mining and time series prediction. In *International work-conference on artificial neural networks*, pp. 758–770. Springer, 2005.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.

Joshua T Vogelstein, Eric W Bridgeford, Minh Tang, Da Zheng, Christopher Douville, Randal Burns, and Mauro Maggioni. Supervised dimensionality reduction for big data. *Nature communications*, 12(1):1–9, 2021.

Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2097–2106, 2017.

Yongqin Xian, Tobias Lorenz, Bernt Schiele, and Zeynep Akata. Feature generating networks for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5542–5551, 2018.

Shuo Yang, Lu Liu, and Min Xu. Free lunch for few-shot learning: Distribution calibration. *arXiv preprint arXiv:2101.06395*, 2021.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Jian Zhang, Chenglong Zhao, Bingbing Ni, Minghao Xu, and Xiaokang Yang. Variational few-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1685–1694, 2019.

Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning. *Advances in neural information processing systems*, 31, 2018.

A APPENDIX

This section provides proof of the main theorems and lemmas presented in the paper. We also provide a brief overview of the proof of Theorem 3.1 from Germain et al. (2009); Alquier et al. (2016); Masegosa (2020).

A.1 PROOF OF THEOREM 3.1

Theorem 3.1 (Germain et al. (2009); Alquier et al. (2016); Masegosa (2020))**.** Given a data distribution P over $\mathcal{X} \times \mathcal{Y}$, a hypothesis set $\mathbf{\Theta}$, a prior distribution π over $\mathbf{\Theta}$, for any $\delta \in (0, 1]$, and $\lambda > 0$, with probability at least $1 - \delta$ over samples $D \sim P^n$, we have for all posterior ρ,

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \leq \mathbb{E}_{\rho(\boldsymbol{\theta})}[\hat{L}(\boldsymbol{\theta}, D)] + \frac{1}{\lambda}\left[D_{\mathrm{KL}}(\rho\|\pi) + \log\frac{1}{\delta} + \Psi_{P,\pi}(\lambda, n)\right]$$

where $\Psi_{P,\pi}(\lambda, n) = \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{\lambda(L(\boldsymbol{\theta})-\hat{L}(\boldsymbol{\theta},D))}\right]$.

Proof. The Donsker-Varadhan's change of measure states that for any measurable function $\phi : \mathbf{\Theta} \to \mathbb{R}$, we have

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[\phi(\boldsymbol{\theta})] \leq D_{\mathrm{KL}}(\rho\|\pi) + \log \mathbb{E}_{\pi(\boldsymbol{\theta})}[e^{\phi(\boldsymbol{\theta})}]$$

Thus, with $\phi(\boldsymbol{\theta}) := \lambda\left(L(\boldsymbol{\theta}) - \hat{L}(\boldsymbol{\theta}, D)\right)$, we obtain $\forall\rho$ on $\mathbf{\Theta}$:

$$\begin{aligned}\mathbb{E}_{\rho(\boldsymbol{\theta})}\left[\lambda\left(L(\boldsymbol{\theta}) - \hat{L}(\boldsymbol{\theta}, D)\right)\right] &= \lambda\left(\mathbb{E}_{\rho(\boldsymbol{\theta})}\left[L(\boldsymbol{\theta})\right] - \mathbb{E}_{\rho(\boldsymbol{\theta})}\left[\hat{L}(\boldsymbol{\theta}, D)\right]\right)\\ &\leq D_{\mathrm{KL}}(\rho\|\pi) + \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\left[e^{\lambda\left(L(\boldsymbol{\theta})-\hat{L}(\boldsymbol{\theta},D)\right)}\right]\end{aligned}$$

Next, we apply Markov's inequality on the random variable $\zeta_\pi(D) := \mathbb{E}_{\pi(\boldsymbol{\theta})}\left[e^{\lambda\left(L(\boldsymbol{\theta})-\hat{L}(\boldsymbol{\theta},D)\right)}\right]$:

$$\Pr\left(\zeta_\pi(D) \leq \frac{1}{\delta}\mathbb{E}_{D\sim P^n}\left[\zeta_\pi\left(D\right)\right]\right) \geq 1 - \delta$$

This implies that with probability at least $1 - \delta$ over the choice of $D \sim P^n$, we have $\forall\rho$ on $\mathbf{\Theta}$:

$$\Pr\left(\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \leq \mathbb{E}_{\rho(\boldsymbol{\theta})}[\hat{L}(\boldsymbol{\theta}, D)] + \frac{1}{\lambda}\left[D_{\mathrm{KL}}(\rho\|\pi) + \log\frac{1}{\delta} + \Psi_{P,\pi}(\lambda, n)\right]\right) \geq 1 - \delta,$$

where $\Psi_{P,\pi}(\lambda, n) = \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{\lambda(L(\boldsymbol{\theta})-\hat{L}(\boldsymbol{\theta},D))}\right]$.

A.2 PROOF OF THEOREM 3.2

Theorem 3.2. (Second-order Oracle bound) Given a data distribution P, a set of model parameters $\{\mathbf{\Theta}_i\}_{i=1}^M$, for any distribution $\{\rho_i\}_{i=1}^M$ over $\{\mathbf{\Theta}_i\}_{i=1}^M$ satisfies that,

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \leq \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[L(\boldsymbol{\theta}_i)] - \mathbb{V}(\rho(\boldsymbol{\theta}))$$

where $\boldsymbol{\theta}_i \in \mathbf{\Theta}_i, \boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^M, \rho\left(\boldsymbol{\theta}\right) = \prod_{i=1}^M \rho_i(\boldsymbol{\theta}_i)$, and $\mathbb{V}(\rho(\boldsymbol{\theta}))$ is a variance term defined as

$$\mathbb{V}(\rho(\boldsymbol{\theta})) = \mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{2M\max_{\boldsymbol{\theta}} p(y|\boldsymbol{x},\boldsymbol{\theta})^2}\sum_{i=1}^{M}\left(p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M}p(y|\boldsymbol{x},\boldsymbol{\theta}_k)\right)^2\right].$$

Proof. We first apply Taylor's theorem with a remainder of second order to the logarithm function. That is, given $\log x$ and a fixed value $a > 0$,

$$\log x = \log a + \frac{1}{a}(x-a) - \frac{1}{2\xi^2}(x-a)^2, \xi \in (x, a). \tag{2}$$

Therefore, applying Eq. 2 to $p(y|\boldsymbol{x}, \boldsymbol{\theta}_i)$ centered at $\frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k) > 0$, we have

$$\begin{aligned}\log p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) =& \log \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k) + \frac{1}{\frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)}\left[p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)\right] \\ &- \frac{1}{2\xi_i^2}\left(p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)\right)^2,\end{aligned}$$

where $\xi_i \in \left(p(y|\boldsymbol{x}, \boldsymbol{\theta}_i), \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)\right), \forall i \in [M]$.

Averaging the above equation, we have

$$\frac{1}{M}\sum_{i=1}^{M}\log p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) = \log \frac{1}{M}\sum_{i=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{i=1}^{M}\frac{1}{2\xi_i^2}\left(p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)\right)^2$$

Rearranging terms,

$$\begin{aligned}&-\log \frac{1}{M}\sum_{i=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) \\ =&\frac{1}{M}\sum_{i=1}^{M} -\log p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{i=1}^{M}\frac{1}{2\xi_i^2}\left(p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k))\right)^2 \\ \leq&\frac{1}{M}\sum_{i=1}^{M} -\log p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{2M\max_{\boldsymbol{\theta}} p(y|\boldsymbol{x}, \boldsymbol{\theta})^2}\sum_{i=1}^{M}\left(p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)\right)^2,\end{aligned}$$

where last inequality follows from $\xi_i \leq \max_{\boldsymbol{\theta}_i} p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) \leq \max_{\boldsymbol{\theta}} p(y|\boldsymbol{x}, \boldsymbol{\theta}), \forall \xi_i$.

Finally, the result of the theorem is derived by taking expectation wrt $(\boldsymbol{x}, y) \sim P$ and $\boldsymbol{\theta} \sim \rho(\boldsymbol{\theta})$ on both sides of the above inequality, and rewriting

$$\begin{aligned}&\mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[-\log \frac{1}{M}\sum_{i=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_i)\right] = \mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})], \\ &\mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{M}\sum_{i=1}^{M} -\log p(y|\boldsymbol{x}, \boldsymbol{\theta}_i)\right] = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[-\log p(y|\boldsymbol{x}, \boldsymbol{\theta}_i)\right] \\ &\qquad\qquad = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho(\boldsymbol{\theta})}\left[L(\boldsymbol{\theta}_i)\right] = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}\left[L(\boldsymbol{\theta}_i)\right], \\ &\mathbb{V}(\rho(\boldsymbol{\theta})) = \mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{2M\max_{\boldsymbol{\theta}} p(y|\boldsymbol{x}, \boldsymbol{\theta})^2}\sum_{i=1}^{M}\left(p(y|\boldsymbol{x}, \boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M} p(y|\boldsymbol{x}, \boldsymbol{\theta}_k)\right)^2\right].\end{aligned}$$

A.3 PROOF OF THEOREM 3.3

Before proving Theorem 3.3, we need to introduce the following result,

Lemma A.1. *For any distribution $\{\rho_i\}_{i=1}^M$ over $\{\boldsymbol{\Theta}_i\}_{i=1}^M$, the second-order Jensen bound of Theorem 3.2 bound can be expressed as follows,*

$$\frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[L(\boldsymbol{\theta}_i)] - \mathbb{V}(\rho(\boldsymbol{\theta})) = \mathbb{E}_{\rho(\boldsymbol{\theta})}L_2(\boldsymbol{\theta}),$$

where $\boldsymbol{\theta}_i \in \boldsymbol{\Theta}_i, \boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i=1}^M, \rho(\boldsymbol{\theta}) = \prod_{i=1}^M \rho_i(\boldsymbol{\theta}_i)$, and $L_2(\boldsymbol{\theta})$ is defined as

$$L_2(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{M}\sum_{i=1}^{M}\left(-\log p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{\left(p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M}p(y|\boldsymbol{x},\boldsymbol{\theta}_k)\right)^2}{2\max_{\boldsymbol{\theta}}p(y|\boldsymbol{x},\boldsymbol{\theta})^2}\right)\right]$$

Proof. The result can be directly obtained by the following equation,

$$\begin{aligned}
&\frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[L(\boldsymbol{\theta}_i)] - \mathbb{V}(\rho(\boldsymbol{\theta})) = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[-\log p(y|\boldsymbol{x},\boldsymbol{\theta}_i)\right] - \mathbb{V}(\rho(\boldsymbol{\theta}))\\
=&\mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{M}\sum_{i=1}^{M}-\log p(y|\boldsymbol{x},\boldsymbol{\theta}_i)\right]\\
&-\mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{\sum_{i=1}^{M}\left(p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M}p(y|\boldsymbol{x},\boldsymbol{\theta}_k)\right)^2}{2L\max_{\boldsymbol{\theta}}p(y|\boldsymbol{x},\boldsymbol{\theta})^2}\right]\\
=&\mathbb{E}_{\rho(\boldsymbol{\theta})}\mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{M}\sum_{i=1}^{M}\left(-\log p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{\left(p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M}p(y|\boldsymbol{x},\boldsymbol{\theta}_k)\right)^2}{2\max_{\boldsymbol{\theta}}p(y|\boldsymbol{x},\boldsymbol{\theta})^2}\right)\right].
\end{aligned}$$

We now proceed to prove the Theorem 3.3.

Theorem 3.3. Given a data distribution P, a set of model parameters $\{\boldsymbol{\Theta}_i\}_{i=1}^M$ and associated priors $\{\pi_i\}_{i=1}^M$, where π_i is defined over $\boldsymbol{\Theta}_i$, a $\delta \in (0,1]$, and a real number $c > 0$, with probability at least $1-\delta$ over draws of training data $D \sim P^n$, for all posteriors $\{\rho_i\}_{i=1}^M$ over $\{\boldsymbol{\Theta}_i\}_{i=1}^M$, simultaneously,

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] \le \frac{1}{M}\sum_{i=1}^{M}\left(\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i,D)] + \frac{D_{\mathrm{KL}}(\rho_i\|\pi_i)}{cn}\right) - \hat{\mathbb{V}}(\rho(\boldsymbol{\theta}),D) + \frac{\epsilon}{cnM},$$

where $\epsilon(\mathcal{P},\pi,c,n,\delta) = \log\mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{cn\left(\sum_{i=1}^{M}\left(L(\boldsymbol{\theta}_i)-\hat{L}(\boldsymbol{\theta}_i,D)\right)-M\left(\mathbb{V}(\boldsymbol{\theta})-\hat{\mathbb{V}}(\boldsymbol{\theta},D)\right)\right)}\right] + \log\frac{1}{\delta}$, $\hat{\mathbb{V}}(\rho(\boldsymbol{\theta}),D)$ is the empirical version of $\mathbb{V}(\rho(\boldsymbol{\theta}))$, and $\mathbb{V}(\rho(\boldsymbol{\theta})) = \mathbb{E}_{\rho(\boldsymbol{\theta})}[\mathbb{V}(\boldsymbol{\theta})]$.

Proof. First of all, consider the following tandem loss:

$$L_2(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{x},y)\sim P}\left[\frac{1}{M}\sum_{i=1}^{M}\left(-\log p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{\left(p(y|\boldsymbol{x},\boldsymbol{\theta}_i) - \frac{1}{M}\sum_{k=1}^{M}p(y|\boldsymbol{x},\boldsymbol{\theta}_k)\right)^2}{2\max_{\boldsymbol{\theta}}p(y|\boldsymbol{x},\boldsymbol{\theta})^2}\right)\right]$$

Applying 3.1 to the tandem loss function with prior distribution $\pi(\boldsymbol{\theta}) = \prod_{i=1}^{M} \pi_i(\boldsymbol{\theta}_i)$ described above, we get that for any $cn > 0, \delta \in (0, 1]$, with probability at least $1 - \delta$:

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L_2(\boldsymbol{\theta})] \leq \mathbb{E}_{\rho(\boldsymbol{\theta})}[\hat{L}_2(\boldsymbol{\theta}, D)] + \frac{1}{\lambda}\left(D_{\mathrm{KL}}(\rho(\boldsymbol{\theta})\|\pi(\boldsymbol{\theta})) + \epsilon(\mathcal{P}, \pi, \lambda, n, \delta)\right)$$

where $\epsilon(\mathcal{P}, \pi, \lambda, n, \delta) = \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{\lambda(L_2(\boldsymbol{\theta}) - \hat{L}_2(\boldsymbol{\theta}, D))}\right] + \log\frac{1}{\delta}$.

Next, rewriting

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[L_2(\boldsymbol{\theta})] = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[L(\boldsymbol{\theta}_i)] - \mathbb{V}(\rho(\boldsymbol{\theta})),$$

$$\mathbb{E}_{\rho(\boldsymbol{\theta})}[\hat{L}_2(\boldsymbol{\theta}, D)] = \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i, D)] - \hat{\mathbb{V}}(\rho(\boldsymbol{\theta}), D),$$

$$L_2(\boldsymbol{\theta}) - \hat{L}_2(\boldsymbol{\theta}, D) = \frac{1}{M}\sum_{i=1}^{M}\left(L(\boldsymbol{\theta}_i) - \hat{L}(\boldsymbol{\theta}_i, D)\right) - \left(\mathbb{V}(\boldsymbol{\theta}) - \hat{\mathbb{V}}(\boldsymbol{\theta}, D)\right),$$

and noting that $D_{\mathrm{KL}}(\rho(\boldsymbol{\theta})\|\pi(\boldsymbol{\theta})) = D_{\mathrm{KL}}\left(\prod_{i=1}^{M}\rho_i(\boldsymbol{\theta}_i)\|\prod_{i=1}^{M}\pi_i(\boldsymbol{\theta}_i)\right) = \sum_{i=1}^{M} D_{\mathrm{KL}}(\rho_i\|\pi_i)$.

Finally, applying Theorem 3.2, we can obtain the PAC-Bayes bound of the theorem by reparametrized λ as $\lambda = cnM$.

$$\begin{aligned}
\mathbb{E}_{\rho(\boldsymbol{\theta})}[L(\boldsymbol{\theta})] &\leq \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[L(\boldsymbol{\theta}_i)] - \mathbb{V}(\rho(\boldsymbol{\theta})) \\
&\leq \frac{1}{M}\sum_{i=1}^{M}\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i, D)] - \hat{\mathbb{V}}(\rho(\boldsymbol{\theta}), D) + \frac{1}{cnM}\sum_{i=1}^{M} D_{\mathrm{KL}}(\rho_i\|\pi_i) + \frac{\epsilon}{cnM} \\
&= \frac{1}{M}\sum_{i=1}^{M}\left(\mathbb{E}_{\rho_i(\boldsymbol{\theta}_i)}[\hat{L}(\boldsymbol{\theta}_i, D)] + \frac{D_{\mathrm{KL}}(\rho_i\|\pi_i)}{cn}\right) - \hat{\mathbb{V}}(\rho(\boldsymbol{\theta}), D) + \frac{\epsilon}{cnM}.
\end{aligned}$$

where $\epsilon(\mathcal{P}, \pi, c, n, \delta) = \log \mathbb{E}_{\pi(\boldsymbol{\theta})}\mathbb{E}_{D\sim P^n}\left[e^{cn\left(\sum_{i=1}^{M}\left(L(\boldsymbol{\theta}_i) - \hat{L}(\boldsymbol{\theta}_i, D)\right) - M\left(\mathbb{V}(\boldsymbol{\theta}) - \hat{\mathbb{V}}(\boldsymbol{\theta}, D)\right)\right)}\right] + \log\frac{1}{\delta}$

A.4 Proof of Theorem 3.5

We first introduce the definition of the random pooling process.

Definition A.2 (Batches of vector)**.** Divide $\boldsymbol{x} \in \mathbb{R}^d$ into disjoint k blocks in turn, which can be expressed as $\boldsymbol{x} = [\boldsymbol{x}_1; \cdots; \boldsymbol{x}_k]$, where $\boldsymbol{x}_i \in \mathbb{R}^{\lceil d/k \rceil}$ represents the batch i of $\boldsymbol{x}$. Note that for the case where $\frac{d}{k}$ is not divisible, we can pad 0 to $\boldsymbol{x}_k$.

To get batches of a matrix or tensor, we can first convert them to vectors by flattening, as shown in Figure 5.

Definition A.3 (The Process of Random Pooling)**.** Let $d, k \in \mathbb{N}^+$, and $d > k$. Denote $f_{\boldsymbol{U}_k} : \mathbb{R}^d \to \mathbb{R}^k$ as a function of random pooling. The projection matrix $\boldsymbol{U}_k$ is defined as a vector concatenated by k independent random vectors, denoted as $\boldsymbol{U}_k = [\mathrm{u}_1; \cdots; \mathrm{u}_k]$. Each random vector $\mathrm{u}_i \in \{0, 1\}^{\lceil d/k \rceil}$ is a one-hot random vector representing that each entry will be selected with the same probability $\frac{1}{\lceil d/k \rceil}$. We define the distribution of the projection matrix as $\mathcal{Q}_k$. Therefore, the random pooling process of $\boldsymbol{x}$ can be expressed as $f_{\boldsymbol{U}_k}(\boldsymbol{x}) = [\mathrm{u}_1^T\boldsymbol{x}_1; \cdots; \mathrm{u}_k^T\boldsymbol{x}_k]$, which $\boldsymbol{U}_k \sim \mathcal{Q}_k$, and $\boldsymbol{x}_i \in \mathbb{R}^{\lceil d/k \rceil}$ refers to the batch i of $\boldsymbol{x}$. We use $\boldsymbol{U}_k(\boldsymbol{x})$ instead of $f_{\boldsymbol{U}_k}(\boldsymbol{x})$ for simplify.

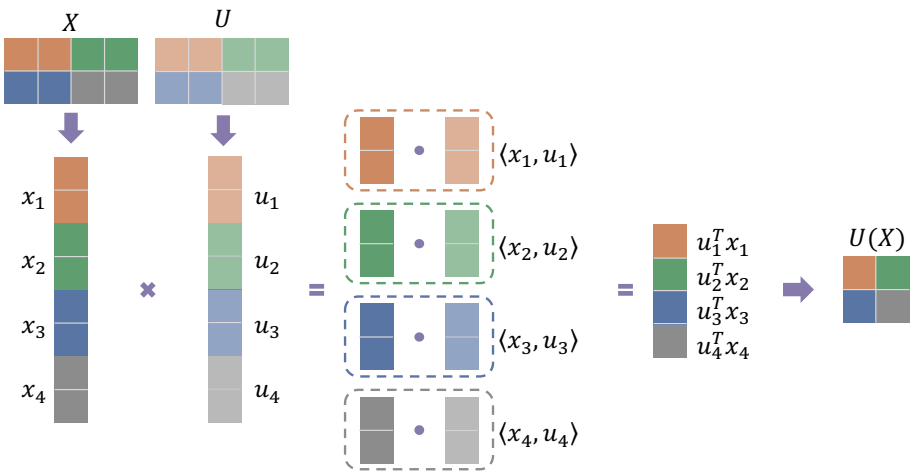

Figure 5: Schematic of pooling process $\boldsymbol{U}(\boldsymbol{x}) : \mathbb{R}^{2\times 4} \to \mathbb{R}^{2\times 2}$. $\boldsymbol{x}, \boldsymbol{U}$ denotes input and projection matrix, respectively. $\boldsymbol{x}_i$ represents the batch i of $\boldsymbol{x}$. The pooling process of $\boldsymbol{x}$ can be expressed as $\boldsymbol{U}(\boldsymbol{x}) = [\mathrm{u}_1^T \boldsymbol{x}_1, \mathrm{u}_2^T \boldsymbol{x}_2; \mathrm{u}_3^T \boldsymbol{x}_3, \mathrm{u}_4^T \boldsymbol{x}_4]$.

Lemma A.4. *Let $\boldsymbol{U}_k$ be the projection matrix of random pooling sampled from $\mathcal{Q}_k$. Then for every $\boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^d$, we have,*

$$\mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[\langle \boldsymbol{U}_k(\boldsymbol{w}), \boldsymbol{U}_k(\boldsymbol{v}) \rangle] = \frac{1}{\lceil d/k \rceil} \langle \boldsymbol{w}, \boldsymbol{v} \rangle,$$

Proof. Since $\boldsymbol{U}_k(\boldsymbol{w}) = [\mathrm{u}_i^T \boldsymbol{w}_i; \cdots; \mathrm{u}_k^T \boldsymbol{w}_k] \in \mathbb{R}^k$, we have

$$\langle \boldsymbol{U}_k(\boldsymbol{w}), \boldsymbol{U}_k(\boldsymbol{v}) \rangle = \sum_{i=1}^{k} (\mathrm{u}_i^T \boldsymbol{w}_i)(\mathrm{u}_i^T \boldsymbol{v}_i)$$

Let $Z_i = (\mathrm{u}_i^T \boldsymbol{w}_i)(\mathrm{u}_i^T \boldsymbol{v}_i)$, for $i \in [k]$. $Z_1, \cdots, Z_k$ are independent random variables based on the independence of $\mathrm{u}_1, \cdots, \mathrm{u}_k$. As $\mathrm{u}_i \in \mathbb{R}^{\lceil d/k \rceil}$ be a one-hot random vector with probability $\frac{1}{\lceil d/k \rceil}$, we have $\mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[\mathrm{u}_i \mathrm{u}_i^T] = \frac{1}{\lceil d/k \rceil} \boldsymbol{I}$. Therefore,

$$\mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[Z_1 + \cdots + Z_k] = \sum_{i=1}^{k} \mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[Z_i] = \sum_{i=1}^{k} \frac{1}{\lceil d/k \rceil} \langle \boldsymbol{w}_i, \boldsymbol{v}_i \rangle = \frac{1}{\lceil d/k \rceil} \langle \boldsymbol{w}, \boldsymbol{v} \rangle.$$

Thus, we obtain

$$\mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[\langle \boldsymbol{U}_k(\boldsymbol{w}), \boldsymbol{U}_k(\boldsymbol{v}) \rangle] = \frac{1}{\lceil d/k \rceil} \langle \boldsymbol{w}, \boldsymbol{v} \rangle.$$

Lemma A.5 (Hoeffding's inequality)**.** *Let $X_1, \cdots, X_n$ be independent random variables such that $a_i \leq X_i \leq b_i$ almost surely. Consider the sum of these random variables, $S_n = X_1 + \cdots + X_n$. Then, for all $t > 0$, we have*

$$\Pr\left(S_n - \mathrm{E}\left[S_n\right] \geq t\right) \leq \exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right)$$

$$\Pr\left(|S_n - \mathrm{E}\left[S_n\right]| \geq t\right) \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right)$$

Theorem 3.5 (Inner Product Preservation)**.** Let $R, d, k \in \mathbb{N}^+$, and $d > k$. Denote by $\mathcal{X}$ the ball of radius R in $\mathbb{R}^d$, and let P be any data distribution over $\mathcal{X} \times \mathcal{Y}$. For every $\boldsymbol{w} \in \mathbb{R}^d$ with $\|\boldsymbol{w}\|_2 \leq 1$, and $\delta > 0$, there exist a $\tilde{\boldsymbol{w}} \in \mathcal{H}$ satisfying,

$$\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}\left[|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle| \geq \delta\right] \leq 4\exp(-\frac{\delta^2 k}{8d^2R^2}) + \frac{(d+k)^2}{\delta^2 k^2}\sum_{j=1}^{k}\mathrm{Var}(\boldsymbol{x}_j),$$

and for every $S \in \mathrm{supp}\,(P^n)$,

$$\Pr_{(\boldsymbol{x},y)\sim S, \boldsymbol{U}_k \sim \mathcal{Q}_k}\left[|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle| \geq \delta\right] \leq 4\exp(-\frac{\delta^2 k}{8d^2R^2}) + \frac{(d+k)^2}{\delta^2 k^2}\sum_{j=1}^{k}\mathrm{Var}(\boldsymbol{x}_j),$$

where $\mathrm{Var}(\boldsymbol{x}_j)$ represents the element-wise variance of batch j of $\boldsymbol{x}$.

Proof. First, by the triangle inequality and the linearity of the dot product, we have

$$\begin{aligned}
&|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle| \\
\leq &|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle| + |\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle| \\
\leq &|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle| + |\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}}\rangle| \\
&+ |\langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}}\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle|.
\end{aligned}$$

Then,

$$\begin{aligned}
&\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}\left[|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle| \geq \delta\right] \\
\leq &\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle| \geq \delta] \\
&+ \Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}}\rangle| \geq \delta] \\
&+ \Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}}\rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}}\rangle| \geq \delta].
\end{aligned} \tag{3}$$

To bound the first probability term, we observe that

$$\mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle] = \lceil\frac{d}{k}\rceil \mathbb{E}_{\boldsymbol{U}_k \sim \mathcal{Q}_k}[\langle \boldsymbol{U}_k(\boldsymbol{x}), \boldsymbol{U}_k(\boldsymbol{w})\rangle] = \langle \boldsymbol{x}, \boldsymbol{w}\rangle, \tag{4}$$

where the last equation follows from Lemma A.4.

Let $Z_j = \lceil\frac{d}{k}\rceil \langle \boldsymbol{U}_k(\boldsymbol{x}_j), \boldsymbol{U}_k(\boldsymbol{w}_j)\rangle$ be the weighted inner product after random pooling of batch j of $\boldsymbol{x}$ and $\boldsymbol{w}$. $Z_1, \cdots, Z_k$ are the independent random variables, and $Z_j \in (-(\frac{d}{k}+1)R, (\frac{d}{k}+1)R), \forall j \in [k]$ follows from the fact that $\Pr_{(\boldsymbol{x},y)\sim P}[\|\boldsymbol{x}\|_2 \leq R] = 1$ and $||\boldsymbol{w}||_2 \leq 1$.

Since $\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle = \sum_{j=1}^{k} Z_j$, applying Hoeffding's inequality and Eq. 4, we have

$$\begin{aligned}
&\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{x}, \boldsymbol{w}\rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil\frac{d}{k}\rceil \boldsymbol{U}_k(\boldsymbol{w})\rangle| \geq \delta] \\
= &\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\mathbb{E}_{\mathcal{Q}_k}[\sum_{j=1}^{k} Z_j] - \sum_{j=1}^{k} Z_j| \geq \delta] \leq 2e^{-\frac{\delta^2 k}{2(d+k)^2 R^2}} \leq 2e^{-\frac{\delta^2 k}{8d^2 R^2}},
\end{aligned} \tag{5}$$

where the last inequality follows from the fact that $k \leq d$.

Next, we bound the second term in 3. Let $\tilde{\boldsymbol{w}} = \lceil \frac{d}{k} \rceil \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{w})]$, we have

$$\mathbb{E}_{\mathcal{Q}_k}[\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil \frac{d}{k} \rceil \boldsymbol{U}_k(\boldsymbol{w}) \rangle - \langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}} \rangle] = \mathbb{E}_{\mathcal{Q}_k}[\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil \frac{d}{k} \rceil \boldsymbol{U}_k(\boldsymbol{w}) - \tilde{\boldsymbol{w}} \rangle] = 0. \tag{6}$$

Furthermore, denote $\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil \frac{d}{k} \rceil \boldsymbol{U}_k(\boldsymbol{w}) - \tilde{\boldsymbol{w}} \rangle = \sum_{j=1}^{k} V_j$, where $V_j = \mathrm{u}_j^T \boldsymbol{x}_j (\lceil \frac{d}{k} \rceil \mathrm{u}_j^T \boldsymbol{w}_j - \mathbf{1}^T \boldsymbol{w}_j), \forall j \in [k]$. Since the random variables $V_1, \cdots, V_k$ are independent, and $V_j \in (-\frac{2dR}{k}, \frac{2dR}{k}), \forall j \in [k]$ follows from the facts that $\Pr_{(\boldsymbol{x},y)\sim P}[\|\boldsymbol{x}\|_2 \leq R] = 1$ and $||\boldsymbol{w}||_2 \leq 1$, we can apply Hoeffding's inequality and Eq. 6 to obtain,

$$\begin{aligned} &\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{U}_k(\boldsymbol{x}), \lceil \frac{d}{k} \rceil \boldsymbol{U}_k(\boldsymbol{w}) - \tilde{\boldsymbol{w}} \rangle| \geq \delta] \\ =&\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\sum_{j=1}^{k} V_j - \mathbb{E}_{\mathcal{Q}_k}[\sum_{j=1}^{k} V_j]| \geq \delta] \leq 2e^{-\frac{\delta^2 k}{8d^2R^2}}. \end{aligned} \tag{7}$$

Finally, we bound the third term in 3.

Denote $\langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}} \rangle = \sum_{j=1}^{k} T_j$, where $T_j = \tilde{w}_j \mathrm{u}_j^T \boldsymbol{x}_j, \forall j \in [k]$. Since $\tilde{\boldsymbol{w}} = \lceil \frac{d}{k} \rceil \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{w})]$, then $\tilde{w}_j = \sum_{i=1}^{\lceil d/k \rceil} w_{ji}$ denotes the element-wise sum of batch j of $\boldsymbol{w}$. For every $j \in [k]$, we have

$$\mathrm{Var}_{\mathcal{Q}_k}(T_j) = \mathbb{E}_{\mathcal{Q}_k}[T_j^2] - (\mathbb{E}_{\mathcal{Q}_k}[T_j])^2 = \tilde{w}_j^2 \left(\mathbb{E}_{\mathcal{Q}_k}[(\mathrm{u}_j^T \boldsymbol{x}_j)^2] - \mathbb{E}_{\mathcal{Q}_k}[(\mathrm{u}_j^T \boldsymbol{x}_j)]^2 \right) = \tilde{w}_j^2 \mathrm{Var}(\boldsymbol{x}_j),$$

where $\mathrm{Var}(\boldsymbol{x}_j)$ represents the element-wise variance of batch j of $\boldsymbol{x}$.

Hence, applying Chebyshev's inequality, we have

$$\begin{aligned} &\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{U}_k(\boldsymbol{x}), \tilde{\boldsymbol{w}} \rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}} \rangle| \geq \delta] \\ =&\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\sum_{j=1}^{k} T_j - \mathbb{E}_{\mathcal{Q}_k}[\sum_{j=1}^{k} T_j]| \geq \delta] \\ \leq&\frac{1}{\delta^2} \mathrm{Var}_{\mathcal{Q}_k}(\sum_{j=1}^{k} T_j) = \frac{1}{\delta^2} \sum_{j=1}^{k} \mathrm{Var}_{\mathcal{Q}_k}(T_j) \quad \text{(Since } T_1, \cdots, T_k \text{ are independent)} \\ =&\frac{1}{\delta^2} \sum_{j=1}^{k} \tilde{w}_j^2 \mathrm{Var}(\boldsymbol{x}_j) < \frac{(d+k)^2}{\delta^2 k^2} \sum_{j=1}^{k} \mathrm{Var}(\boldsymbol{x}_j) \end{aligned} \tag{8}$$

where the second inequality comes from $\tilde{w}_j^2 = (\sum_{i=1}^{\lceil d/k \rceil} w_{ji})^2 < (\lceil d/k \rceil)^2 < (\frac{d}{k} + 1)^2$, $||\boldsymbol{w}||_2 \leq 1$, and the last inequality follows from the fact that $k \leq d$.

Plugging (5), (7) and 8 into (3), we can obtain

$$\Pr_{(\boldsymbol{x},y)\sim P, \boldsymbol{U}_k \sim \mathcal{Q}_k}[|\langle \boldsymbol{x}, \boldsymbol{w} \rangle - \langle \mathbb{E}_{\mathcal{Q}_k}[\boldsymbol{U}_k(\boldsymbol{x})], \tilde{\boldsymbol{w}} \rangle| \geq \delta] \leq 4 \exp(-\frac{\delta^2 k}{8d^2R^2}) + \frac{(d+k)^2}{\delta^2 k^2} \sum_{j=1}^{k} \mathrm{Var}(\boldsymbol{x}_j),$$

where $\mathrm{Var}(\boldsymbol{x}_j)$ represents the element-wise variance of batch j of $\boldsymbol{x}$.

Thus, we have concluded the first part of the lemma.

The proof of the second part is identical, as we did not use any properties of the distribution P other than $\Pr_{(x,y)\sim P}[||\boldsymbol{x}||_2 \leq R] = 1$. For every $S \in \mathrm{supp}(\mathcal{P}^n)$, it holds that $\Pr_{(x,y)\sim S}[||\boldsymbol{x}||_2 \leq R] = 1$, and the result follows.

B PSEUDO-CODE OF OUR METHOD

Algorithm 1 Average Pooling Ensemble Few-shot Learning (APEF)

Require: Support set $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N\times K}$, Query set $Q = \{\boldsymbol{x}_i\}_{i=1}^{N\times M}$, Pre-trained Network $\mathcal{F}$ with L layers, Output size of AdaptiveAvgPool2d d, start layer for ensemble r.
function TRAIN(S)
  for $j = r, \cdots, L$ **do**
    $\boldsymbol{z}_{ij} = \text{AdaptiveAvgPool2d}(d)(\mathcal{F}_j(\boldsymbol{x}_i))$ ▷ Get dimensionality reduction features
    $S_j = \{(\boldsymbol{z}_{ij}, y_i)\}_{i=1}^{N\times K}$ ▷ Generate new support set
    Train a classifier $f_{\boldsymbol{\theta}_j}$ with $\ell_j = \frac{1}{N\times K}\sum_{(\boldsymbol{z}_{ij}, y_i)\sim S_j} -\log P(y_i|\boldsymbol{z}_{ij}, \boldsymbol{\theta}_j)$.
  end for
end function
function INFERENCE(Q)
  $\boldsymbol{z}_{ij} = \text{AdaptiveAvgPool2d}(d)(\mathcal{F}_j(\boldsymbol{x}_i))$ for $j \in [r, \cdots, L]$
  $\hat{y}_i = \sum_{j\in[r,\cdots,L]} f_{\boldsymbol{\theta}_j}(\boldsymbol{z}_{ij})$ ▷ Integrate the predictions of all trained classifiers
end function

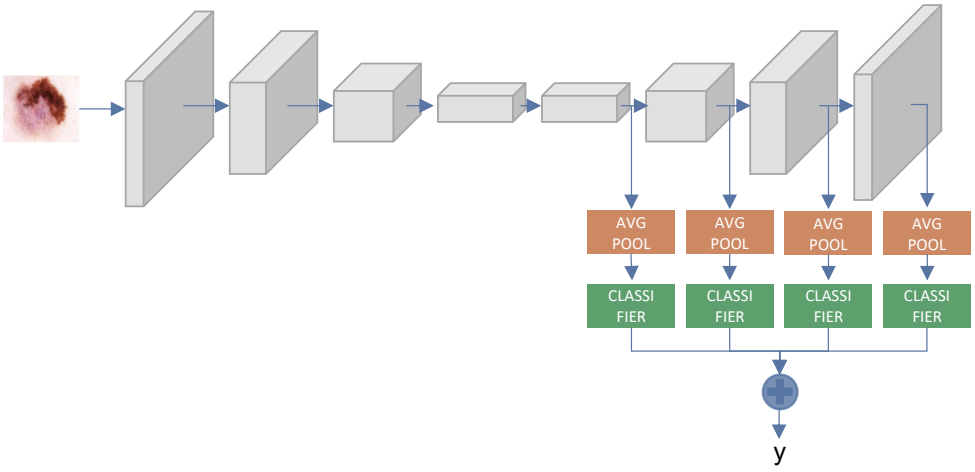

Figure 6: Schematic of APEF inference.

C ADDITIONAL RELATED WORK

C.1 FEW-SHOT LEARNING (FSL)

Given abundant training examples from the source domain, few-shot learning aims to learn to recognize novel classes in the target domain with a limited amount of labeled samples. FSL algorithms can be roughly divided into three categories: generative-based, meta-learning-based, and transfer-learning-based methods.

Generative-based methods focus on learning augmentation to alleviate data shortages. Most methods implement Generative Adversarial Networks (Goodfellow et al., 2014) or autoencoder (Rumelhart et al., 1986) from the source domain and use them to generate samples (Zhang et al., 2018; Schwartz et al., 2018; Yang et al., 2021) or features (Xian et al., 2018; Zhang et al., 2019) for new classes. More specifically, Zhang et al. (2018) and Xian et al. (2018) proposed an adversarial generator to synthetic data, Zhang et al. (2019)

introduced a variational autoencoder to approxiamted the distibution and predict labels based on the estimated statics. Recently, Yang et al. (2021) proposed a method without additional training parameters, which expands samples by a calibration distribution obtained from the statistics of classes with sufficient samples.

Meta-learning-based methods aim to quickly adapt to new tasks by learning general information across tasks. It usually includes metric-based and optimization-based methods. Metric-based methods look for suitable learning metrics or distance functions to judge the similarity of new data classes. Examples of distance metrics include cosine similarity for MatchingNet (Vinyals et al., 2016), Euclidean distance between class feature means for ProtoNet (Snell et al., 2017), CNN-based relational networks for RelationNet (Sung et al., 2018), and linear classification rules for MetaOpt (Lee et al., 2019). Optimization-based methods use prior knowledge to influence the update of model parameters, either by finding a good initialized parameter (Finn et al., 2017; Rusu et al., 2018), or by directly learning an optimizer to output search steps, e.g. Ravi & Larochelle (2016) proposed an LSTM-based meta-learner to replace the stochastic gradient descent optimizer, Munkhdalai & Yu (2017) introduced the weight-update mechanism using external memory.

Transfer-learning-based methods are based on the core idea of feature reuse and are mainly performed through fine-tuning. The most common practice is to use a pre-trained backbone as a fixed feature extractor, and the obtained high-dimensional feature vectors are used to learn the target task. Some works use different classifiers to learn downstream tasks, such as cosine-similarity based classifier (Chen et al., 2019), mean-centroid classifier (Guo et al., 2020). Recent work (Ghaffari et al., 2021) improved the performance of few-shot classifiers from the perspective of firth bias reduction.

C.2 Cross-domain few-shot learning (CD-FSL)

Cross-domain few-shot learning focuses on the FSL problem with large gaps between source and target domains. Previous works (Chen et al., 2019; Guo et al., 2020) found that simple fine-tuning methods significantly outperform most meta-learning-based methods when faced with CD-FSL problems. To help investigate this problem, Guo et al. (2020) also proposed a novel and challenging CD-FSL benchmark, which covers several target domains with different similarities to natural images. Most previous efforts of CDFSL can be roughly categorized into two directions. One is to pre-train a more robust and transferable backbone. For example, Tseng et al. (2020) introduced a feature-wise transformation layer on top of the features to simulate the cross-domain distribution. Phoo & Hariharan (2020) and (Islam et al., 2021) assumed that many unlabeled data from the target domain is available for pre-training and combined contrastive learning and knowledge distillation with adaptation to the target task. HVM (Du et al., 2021) introduces a hierarchical variational inference framework to optimize and store features at different semantic levels. Another direction focuses on fine-tuning the given pre-trained model for the subsequent few-shot learning task, e.g., CHEF (Adler et al., 2020) applying a fusion of Hebbian learners to increase the importance of low and mid-level features. More recently, ConFeSS Das et al. (2021) proposed a framework that combines contrastive learning and feature selection to tackle large domain shifts between the base and novel categories. Luo et al. (2022) introduced a channel-wise feature transformation to alleviate the channel bias problem in few-shot image classification. In contrast, our work focuses on how to quickly adapt a given pre-trained model into a cross-domain with only accessing a few labeled data in the target domain. To the best of our knowledge, we are the first work to propose theoretical guarantees for CD-FSL.

D More Experimental Results of Section 2

D.1 Effects of different layers on the performance of cross-domain few-shot learning

Figure 7 shows 5-shot and 20-shot classifier performance on four challenging datasets adapted to representations of different residual blocks of a ResNet-18 network pre-trained on ImageNet. The result of 50-shot

is shown in Figure 1 of the main text. These experiments report the average accuracy (%, top-1) for ten few-shot episodes. We can see that representations from intermediate blocks may be more transferable than representations from the last block. For example, under 5-way 20-shot, the performance of the ISIC classifier trained on the 5-th block representation is significantly better than that trained on the last block representation.

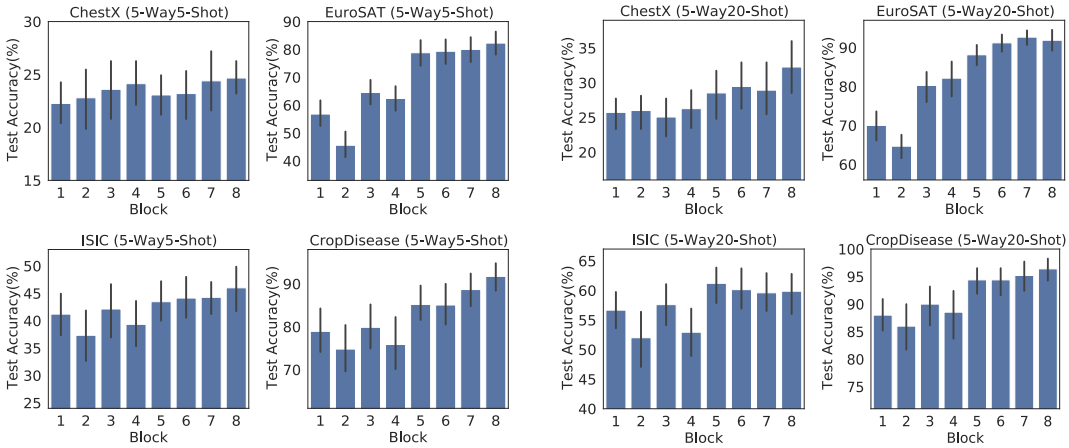

Figure 7: Investigate the performance of few-shot classifiers when applied on top of representation from different residual blocks of a ResNet-18 network pre-trained on ImageNet. The four panels on the *left* show the test accuracy of the 5-Way 5-Shot classifiers, and the four panels on the *right* show the test accuracy of the 5-Way 20-Shot classifiers.

D.2 The trade-off between dimensionality reduction and performance

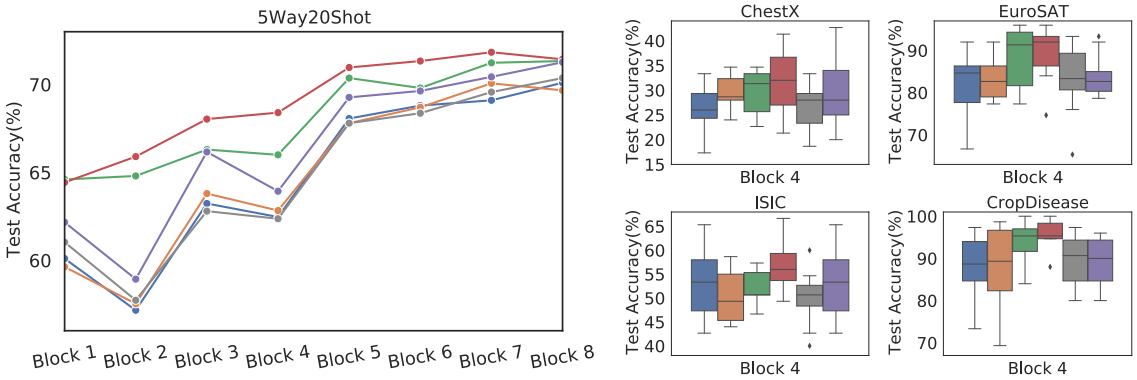

Figure 8: Dimensionality reduction methods produce consistent improvements across different blocks. The experiments are performed on the ResNet-18 network pre-trained on ImageNet, and four target task used for adaptation is set to 5-way 20-shot. *Left*: Average test accuracy of four target tasks for each block. *Right*: The four small graphs correspond to the test accuracy of four target tasks on block 4, respectively. (W/O denotes no dimensionality reduction method is used.)

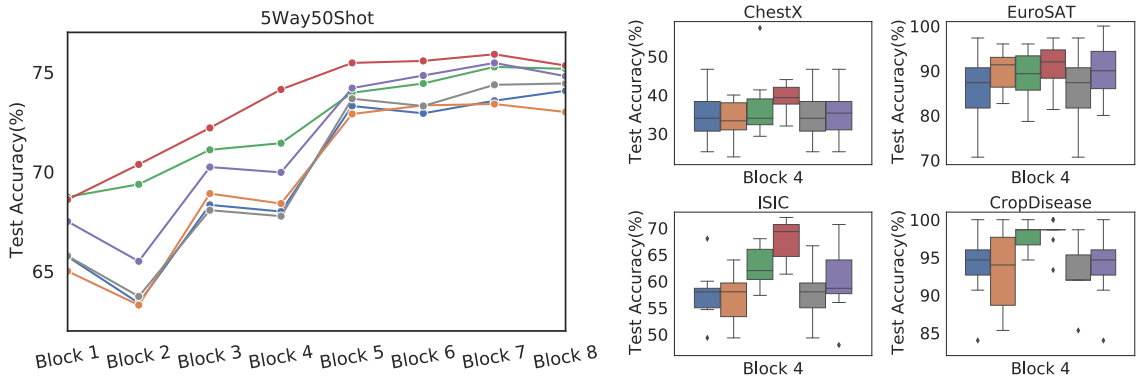

Figure 9: Dimensionality reduction methods produce consistent improvements across different blocks. The experiments are performed on the ResNet-18 network pre-trained on ImageNet, and four target task used for adaptation is set to 5-way 50-shot. *Left*: Average test accuracy of four target tasks for each block. *Right*: The four small graphs correspond to the test accuracy of four target tasks on block 4, respectively. (W/O denotes no dimensionality reduction method is used.)

Figure 8, 9 show the performance of different dimensionality reduction methods under 20-shot and 50-shot. The result of 5-shot is shown in Figure 2 of the main text. We selected five classical and effective dimensionality reduction methods, including random pooling (RANDPOOL), maximum pooling (MAXPOOL), average pooling (AVGPOOL), Principal Component Analysis (PCA), and Linear Optimal Low-Rank Projection (LOL) (Vogelstein et al., 2021). Most of the existing sufficient dimensional reduction (SDR) methods are not well suited for few-shot problems, as they completely break down in the $N \ll d$ regime (N: number of samples, d: sample dimension).

We implement AVGPOOL(MAXPOOL) by applying $\mathrm{AdaptiveAvgPool2d}(\mathrm{AdaptiveMaxPool2d})$. The hyperparameter of output size is set by grid-search ($H_{out} = W_{out}$) listed in Table 3. Note that the dimension of features would be reduced to $C \times H_{out} \times W_{out}$. The implementation of RANDPOOL is similar to that of MAXPOOL, except that the maximum value selection is changed to choose one randomly. PCA is performed by applying the implementation of scikit-learn. Since PCA does not require label information, we use all support and query samples for decomposition. While LOL Vogelstein et al. (2021) is an extension of PCA, it combines the mean and variance of each class, so we only use samples from the support set to train LOL. The number of components to keep is set by grid-search as Table 3. We report the best average accuracy (%, top-1) over 10 few-shot episodes.

Table 3: List of hyperparameters for different dimensionality reduction methods.

Block	Block 1,2	Block 3,4	Block 5,6	Block 7,8
Size of Features (C × H × W)	$64 \times 56 \times 56$	$128 \times 28 \times 28$	$256 \times 14 \times 14$	$512 \times 7 \times 7$
AVG/MAX/RANDPOOL	[56, 28, 14, 7, 4, 2, 1]	[28, 14, 7, 4, 2, 1]	[14, 7, 4, 2, 1]	[7, 4, 2, 1]
PCA	[8, 16, 32, 64, $\cdots$, min (256, (# shot + # query) × # way)]			
LOL	[8, 16, $\cdots$, min (256, # shot × # way)]			

E EXPERIMENTAL DETAILS AND ADDITIONAL RESULTS

E.1 DATASETS

The Cross-Domain Few-Shot Learning (CD-FSL) challenge benchmark we used is proposed by Guo et al. Guo et al. (2020). It uses miniImagenet (Vinyals et al., 2016), or ImageNet (Deng et al., 2009) as the source domain and evaluates the pre-trained model on four different target domains with only a few labeled data. The target domains are chosen based on increasing dissimilarity from the source domain: 1) **CropDiseases** (Mohanty et al., 2016), consisting of leaf images with different plant diseases, 2) **EuroSAT** (Helber et al., 2019), a collection of satellite imagery containing different land use and land cover categories, 3) **ISIC** (Tschandl et al., 2018; Codella et al., 2019), which contains dermoscopic images of skin lesions, and 4) **ChestX** (Wang et al., 2017), which consists of chest X-Ray images of different lung diseases. The above datasets reflect real-world use cases for few-shot learning, as collecting enough examples from these domains is often difficult, expensive, or sometimes impossible. Examples of four datasets are provided in Figure 10. Refer to Guo et al. (2020) for more information.

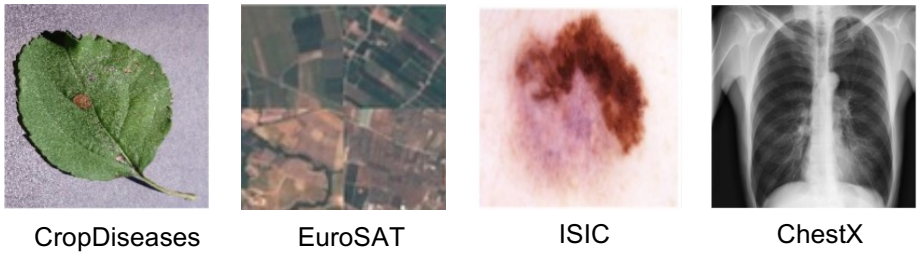

Figure 10: Example of four datasets in the CD-FSL benchmark.

E.2 BASELINES

We compare our method with techniques reported in Guo et al. (2020), which includes most state-of-the-art methods: MatchingNet (Vinyals et al., 2016), MAML (Finn et al., 2017), ProtoNet (Snell et al., 2017), RelationNet (Sung et al., 2018), and MetaOpt (Lee et al., 2019). Moreover, Feature Transform (FWT) (Tseng et al., 2020), as a model-agnostic method, is added to the pre-trained model to simulate the cross-domain setting. We also include Fixed (Guo et al., 2020), which only utilizes the pre-trained model as a fixed feature extractor. In addition, we also compare two relatively new methods, CHEF (Adler et al., 2020) and Hierarchical Variational Memory (HVM) (Du et al., 2021), both of which use multiple layers of features to train cross-domain few-shot tasks.

E.3 PRE-TRAINING MODELS

We use the following four pre-trained models for few-shot evaluation on the CD-FSL benchmark.

1. ResNet-10: We train the ResNet-10 using the publicly accessed code provided in the CD-FSL benchmark. We keep everything the same. Specifically, we train the network on the miniImageNet dataset for 400 epochs by the Adam optimizer with a learning rate of 0.001 and a batch size of 16.
2. ResNet-18 (He et al., 2016): We use the pre-trained ResNet-18 available on PyTorch with ResNet18_Weights.IMAGENET1K_V1.
3. ResNet-50 (He et al., 2016): We use the pre-trained ResNet-50 available on PyTorch with ResNet50_Weights.IMAGENET1K_V2.

4. Wide ResNet-50-2 (Zagoruyko & Komodakis, 2016): We use the pre-trained Wide ResNet-50-2 available on PyTorch with Wide_ResNet50_2_Weights.IMAGENET1K_V2.

5. Vision Transformer (ViT) (Dosovitskiy et al., 2020): We use the pre-trained Vision Transformer (ViT_B_16) available on PyTorch with ViT_B_16_Weights.IMAGENET1K_SWAG_E2E_V1.

6. DenseNet-121 (Huang et al., 2017): We use the pre-trained DenseNet-121 available on PyTorch with $torch.hub.load('pytorch/vision:v0.10.0','densenet121', pretrained = True)$.

E.4 Additional Experimental Results

Table 4 shows the performance comparison of our method with other methods on the CD-FSL benchmark when using the ResNet-10 backbone pre-trained on miniImageNet.

Table 4: Experimental results on four cross-domain few-shot challenges. The average accuracy and 95% confidence interval of 600 runs are reported. †, *, and ⋆ denotes results reported by Guo et al. (2020), Adler et al. (2020) and Du et al. (2021) respectively. The runner-up method is underlined.

	ChestX			**ISIC**		
Method	5-way 5-shot	5-way 20-shot	5-way 50-shot	5-way 5-shot	5-way 20-shot	5-way 50-shot
MatchingNet†	22.40 ± 0.70	23.61 ± 0.86	22.12 ± 0.88	36.74 ± 0.53	45.72 ± 0.53	54.58 ± 0.65
MatchingNet + FWT†	21.26 ± 0.31	23.23 ± 0.37	23.01 ± 0.34	30.40 ± 0.48	32.01 ± 0.48	33.17 ± 0.43
MAML†	23.48 ± 0.96	27.53 ± 0.43	-	40.13 ± 0.58	52.36 ± 0.57	-
ProtoNet†	24.05 ± 1.01	28.21 ± 1.15	29.32 ± 1.12	39.57 ± 0.57	49.50 ± 0.55	51.99 ± 0.52
ProtoNet + FWT†	23.77 ± 0.42	26.87 ± 0.43	30.12 ± 0.46	38.87 ± 0.52	43.78 ± 0.47	49.84 ± 0.51
RelationNet†	22.96 ± 0.88	26.63 ± 0.92	28.45 ± 1.20	39.41 ± 0.58	41.77 ± 0.49	49.32 ± 0.51
RelationNet + FWT†	22.74 ± 0.40	26.75 ± 0.41	27.56 ± 0.40	35.54 ± 0.55	43.31 ± 0.51	46.38 ± 0.53
MetaOpt†	22.53 ± 0.91	25.53 ± 1.02	29.35 ± 0.99	36.28 ± 0.50	49.42 ± 0.60	54.80 ± 0.54
Fixed†	25.35 ± 0.96	30.83 ± 1.05	36.04 ± 0.46	43.56 ± 0.60	52.78 ± 0.58	57.34 ± 0.56
CHEF*	24.72 ± 0.14	29.71 ± 0.27	31.25 ± 0.20	41.26 ± 0.34	54.34 ± 0.34	60.86 ± 0.18
HVM⋆	**27.15 ± 0.45**	30.54 ± 0.47	32.76 ± 0.46	42.05 ± 0.34	54.97 ± 0.35	61.71 ± 0.32
Ours	26.43 ± 0.44	**32.62 ± 0.45**	**37.46 ± 0.50**	**44.02 ± 0.53**	**56.94 ± 0.57**	**64.20 ± 0.55**

	EuroSAT			**CropDiseases**		
Method	5-way 5-shot	5-way 20-shot	5-way 50-shot	5-way 5-shot	5-way 20-shot	5-way 50-shot
MatchingNet†	64.45 ± 0.63	77.10 ± 0.57	54.44 ± 0.67	66.39 ± 0.78	76.38 ± 0.67	58.53 ± 0.73
MatchingNet + FWT†	56.04 ± 0.65	63.38 ± 0.69	62.75 ± 0.76	62.74 ± 0.90	74.90 ± 0.71	75.68 ± 0.78
MAML†	71.70 ± 0.72	81.95 ± 0.55	-	78.05 ± 0.68	89.75 ± 0.42	-
ProtoNet†	73.29 ± 0.71	82.27 ± 0.57	80.48 ± 0.57	79.72 ± 0.67	88.15 ± 0.51	90.81 ± 0.43
ProtoNet + FWT†	67.34 ± 0.76	75.74 ± 0.70	78.64 ± 0.57	72.72 ± 0.70	85.82 ± 0.51	87.17 ± 0.50
RelationNet†	61.31 ± 0.72	74.43 ± 0.66	74.91 ± 0.58	68.99 ± 0.75	80.45 ± 0.64	85.08 ± 0.53
RelationNet + FWT†	61.16 ± 0.70	69.40 ± 0.64	73.84 ± 0.60	64.91 ± 0.79	78.43 ± 0.59	81.14 ± 0.56
MetaOpt†	64.44 ± 0.73	79.19 ± 0.62	83.62 ± 0.58	68.41 ± 0.73	82.89 ± 0.54	91.76 ± 0.38
Fixed†	75.69 ± 0.66	84.13 ± 0.52	86.62 ± 0.47	87.48 ± 0.58	94.45 ± 0.36	96.62 ± 0.25
CHEF*	74.15 ± 0.27	83.31 ± 0.14	86.55 ± 0.15	86.87 ± 0.27	94.78 ± 0.12	96.77 ± 0.08
HVM⋆	74.88 ± 0.45	84.81 ± 0.34	87.16 ± 0.35	87.65 ± 0.35	95.13 ± 0.35	97.83 ± 0.33
Ours	**81.65 ± 0.65**	**89.34 ± 0.44**	**92.07 ± 0.36**	**91.48 ± 0.47**	**96.65 ± 0.27**	**98.07 ± 0.18**

E.5 BENEFITS OF AN ENSEMBLE MODEL

We compare our method with a few-shot classifier trained using only the dimensionality-reduced features of the last block under the pre-trained ResNet-10, ResNet-18, ResNet-50, Wide ResNet-50-2, Vision Transformer (ViT), and DenseNet-121 backbone, respectively. As shown in Figure 12, we find that our method shows more significant improvements under deeper pre-trained backbones. Furthermore, the advantage of our ensemble model is more obvious with the increase of the shot of ChestX and ISIC tasks on all pre-trained backbones. In particular, both tasks have more than 10% improvements on 5-way 50-shot and pre-trained ResNet-50 or Wide ResNet-50-2.

E.6 BENEFITS OF INDEPENDENT TRAINING

Table 5 shows the benefit of training all classifiers independently, where diversity is computed as defined by theorem 3.2. For non-independent training, we sum the losses of all classifiers and then use an optimizer for the ensemble model update. As shown in Table 5, we find that high diversity is consistently positively associated with higher performance across all four challenges. Moreover, independent training also resulted in higher model diversity across all datasets.

Table 5: The benefit of independent training under 5-way 5-shot and the pre-trained ResNet-18 model. Average test accuracy and diversity over 600 runs are reported. (Indep. denotes independent training)

	ChestX		**ISIC**		**EuroSAT**		**CropDiseases**	
Indep.	ACC(%)	Diversity	ACC(%)	Diversity	ACC(%)	Diversity	ACC(%)	Diversity
	25.74 ± 0.42	0.027	44.08 ± 0.56	0.032	85.74 ± 0.50	0.041	92.62 ± 0.44	0.050
✓	**25.75** ± 0.43	0.061	**45.26** ± 0.58	0.056	**87.10** ± 0.49	0.046	**94.39** ± 0.41	0.051

E.7 THE EFFECT OF OUTPUT SIZE OF AVERAGE POOLING

Figure 11 shows the effect of the output size of $\mathrm{AdaptiveAvgPool2d}$ on four cross-domain tasks under 5-way $5-$shot and pre-trained ResNet-18 backbone. The output size d is set to $\{1, 2, 4, 7, 14, 28\}$, which means the feature dimension is reduced to $C \times d \times d$ (C denotes feature's channel size). As shown in Figure 11, we find that the optimal output size of each task is different.

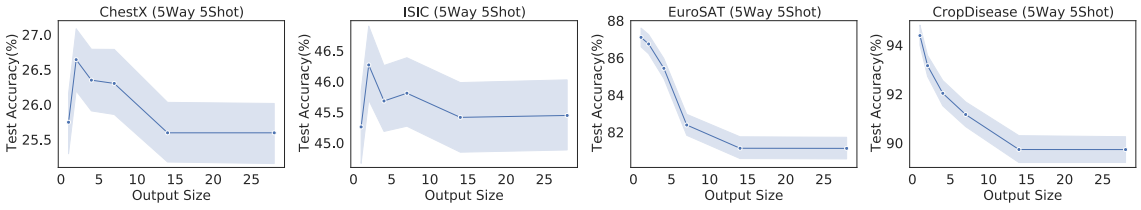

Figure 11: The effect of different output sizes of $\mathrm{AdaptiveAvgPool2d}$ on four cross-domain challenges. Experiments are performed under 5-way 5-shot and the pre-trained ResNet-18 model.

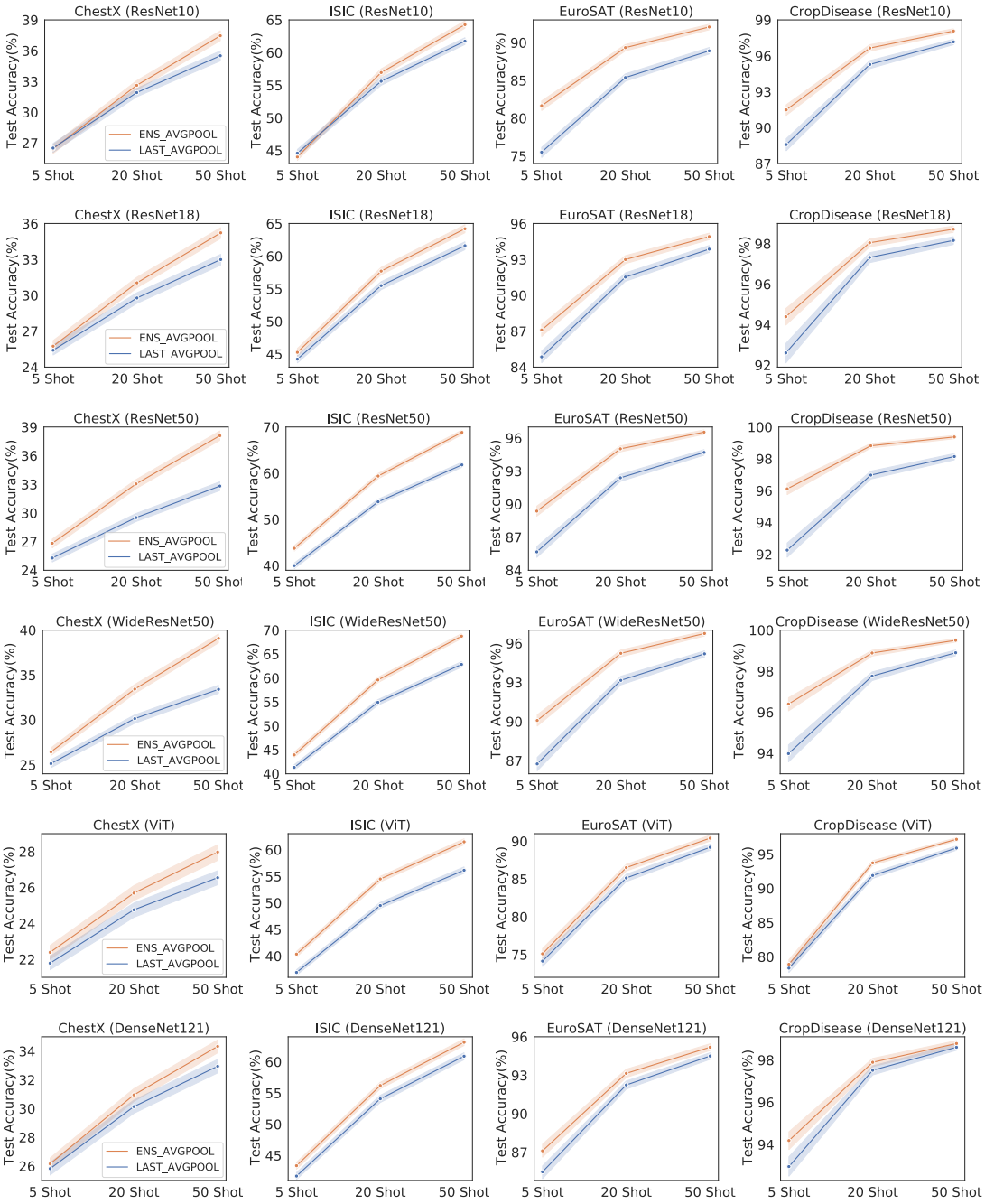

Figure 12: The benefit of the ensemble on four cross-domain challenges. The ensemble is trained on the ImageNet pre-trained ResNet-10, ResNet-18, ResNet-50, Wide ResNet-50-2, Vision Transformer (ViT), and DenseNet-121 backbone, respectively.