# Adaptive manifold for imbalanced transductive few-shot learning

Michalis Lazarou[1] Yannis Avrithis[2] Tania Stathaki[1]
Imperial College London[1]
Institute of Advanced Research on Artificial Intelligence (IARAI)[2]

## Abstract

*Transductive few-shot learning algorithms have showed substantially superior performance over their inductive counterparts by leveraging the unlabeled queries at inference. However, the vast majority of transductive methods are evaluated on perfectly class-balanced benchmarks. It has been shown that they undergo remarkable drop in performance under a more realistic, imbalanced setting.*

*To this end, we propose a novel algorithm to address imbalanced transductive few-shot learning, named* Adaptive Manifold*. Our algorithm exploits the underlying manifold of the labeled examples and unlabeled queries by using manifold similarity to predict the class probability distribution of every query. It is parameterized by one centroid per class and a set of manifold parameters that determine the manifold. All parameters are optimized by minimizing a loss function that can be tuned towards class-balanced or imbalanced distributions. The manifold similarity shows substantial improvement over Euclidean distance, especially in the 1-shot setting.*

*Our algorithm outperforms all other state of the art methods in three benchmark datasets, namely* mini*ImageNet,* tiered*ImageNet and CUB, and two different backbones, namely ResNet-18 and WideResNet-28-10. In certain cases, our algorithm outperforms the previous state of the art by as much as* 4.2%*. The publicly available source code can be found in* https://github.com/MichalisLazarou/AM

## 1. Introduction

Despite the success of deep learning models for visual recognition, their reliance on large labeled datasets still remains a fundamental limitation. This is because obtaining large labeled datasets requires significant human labour to manually annotate images which is expensive and time-consuming.

The *few-shot learning* paradigm has attracted significant interest because it investigates how to make deep learning models acquire knowledge from limited labeled data [9, 36, 40]. Different methodologies have been proposed to address few-shot learning such as *meta-learning* [9, 15, 36], *transfer learning* [21,25,38] and *synthetic data generation* [18,19,24]. The vast majority of these methods focus on the *inductive* setting, where the assumption is that at inference, every query example is classified independently of the others.

Recent studies explored the *transductive* few-shot learning setting, where all query examples can be exploited together at inference time, showing remarkable improvement in performance [12, 17, 30, 41, 46]. Some approaches exploit all query examples at the same time by utilizing the data manifold through label propagation [17] and embedding propagation [33]. Other approaches utilize the available query examples to improve the class centroids by optimizing specialized loss functions [2], using soft K-means [12] or by minimizing the cross-class and intra-class variance [22].

While the query set of transductive few-shot learning benchmarks is unlabeled, it is still curated in the sense that the tasks are perfectly class-balanced. Several state of the art methods exploit this assumption and use class balancing approaches to improve their performance [2,12,17,46]. However, it has been argued that this is not a realistic setting [39]. As a way to address this flaw, the latter study introduced a new *imbalanced transductive few-shot learning* setting, comparing numerous state of the art methods under a fair setting and showing that their performance drops dramatically.

In this work, focusing on this imbalanced transductive setting [39], we introduce a new algorithm, called *Adaptive Manifold* (AM), that exploits the complementary merits of class centroid and data manifold approaches. In particular, we hypothesize that class centroid approaches will benefit from exploiting the data manifold to obtain more representative centroids. As illustrated in Figure 1, we initialize the class centroids from the labeled support examples and we propagate the labels along the data manifold, using a $k$-nearest neighbour graph [14]. We iteratively update both the class centroids and the manifold-specific parameters by minimizing the loss function proposed in [39]. Our algorithm achieves new state of the art performance in the imbalanced transductive few-shot learning setting.

In summary, we make the following contributions:

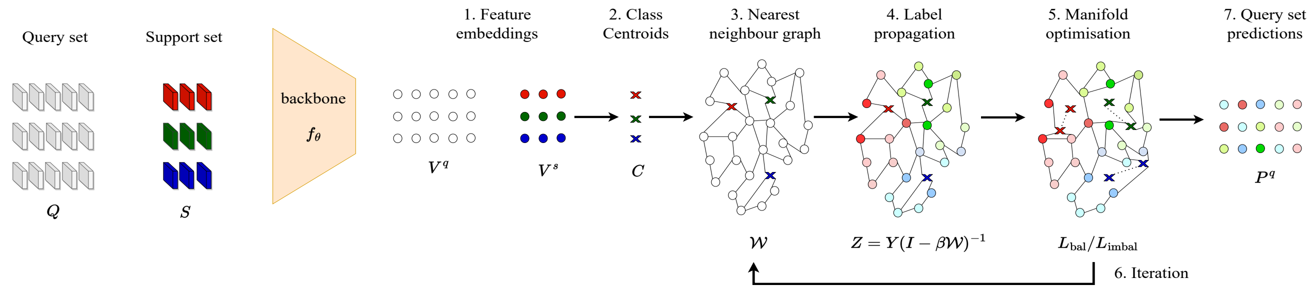



Figure 1. *Overview of our method.* 1) Given a support set, $S$, and a query set, $Q$, we extract features $V^s$ and $V^q$ using the pre-trained backbone $f_\theta$. 2) We calculate class centroids, $C$, using (1). 3) We calculate the $k$-nearest neighbour graph using (3), (4), (5) and (6). 4) We perform label propagation using (8). 5) We optimize the manifold parameters, $\Phi$, using (14) or (17). 6) We iterate the procedure from graph construction for $r$ steps. 7) We predict pseudo-labels using (18).

- We are the first, to the best of our knowledge, to optimize the class centroids along with the manifold-specific parameters using manifold class similarity.
- We achieve new state of the art performance on the imbalanced transductive few-shot setting under multiple datasets and networks, outperforming by as much as $4.2\%$ the previous state of the art in the 1-shot setting.
- Our method can also perform on par or even outperform many state of the art methods in the standard balanced transductive few-shot setting.

## 2. Related work

### 2.1. Few-shot learning

Learning from limited data is a long-standing problem [8]. A large number of the current methods focus on the *meta-learning* paradigm. These can be grouped into three directions: model-based [11, 26, 27, 35], optimization based [9, 28, 31, 34] and metric-based [15, 36, 37, 40]. *Model-based* methods utilize specialized networks such as memory augmented networks [35] and meta-networks [27] to aid the meta-learning process. *Optimization-based* methods focus on learning a robust model initialization, through gradient-based solutions [9, 28], closed-form solutions [1] or an LSTM [32]. *Metric-based* approaches operate in the embedding space and compare an individual query to all examples of a class [40] or to the class centroid [36]. Learning the similarity function has also been proposed [37].

Recent works [5, 38] have shown that the transfer learning paradigm can outperform meta-learning methods. *Transfer learning* methods decouple the training from the inference stage and aim at learning powerful representations through the use of well-designed pre-training regimes to train the backbone network. This often involves training with auxiliary loss functions along with the standard cross entropy loss, such as *knowledge-distillation* [38], mixup-based *data augmentation* [25] and *self-supervision*, such as predicting rotations [10] and contrastive learning [42].

Another way to address the data deficiency is to augment the support set with synthetic data. *Synthetic data* can be generated either in the image space or in the feature space, by using a *hallucinator* trained on the base classes. The hallucinator can be trained using common generative models, such as *generative adversarial networks* (GANs) [19, 23, 44] and *variational autoencoders* (VAEs) [24]. Hallucinators have also been specifically designed for the few-shot learning paradigm [4, 6, 18, 43].

### 2.2. Transductive few-shot learning

Transductive few-shot learning studies the case where all queries are available at inference time and can be exploited to improve predictions. Several methods exploit the data manifold by using label propagation [17, 20], embedding propagation [33] or by exploiting Riemannian geometry through the use of the oblique manifold [30]. Another direction is to use both labeled and unlabelled examples to refine the class centroids. For example, one may use soft $k$-means to iteratively update the class centroids [12], rectify prototypes by minimizing the inter-class and intra-class variance [22], or iteratively adapt the class centroids by minimizing a modified mutual information loss between query features and their label predictions [2]. It has also been proposed to iteratively select the most confident pseudo-labeled queries, for example by interpreting this problem as label denoising [17] or by calculating the credibility of each pseudo-label [41].

Concurrently to our work, ProtoLP [47] has been proposed to find and update class centroids using label propagation. However, this method focuses mainly on the few-shot balanced setting while ours is effective in both balanced and imbalanced settings. Furthermore, our AM adapts manifold-specific parameters, while ProtoLP does not. Each method uses a different way to update parameters: We minimize the modified mutual information loss [39], while ProtoLP uses soft $k$-means. Our AM outperforms ProtoLP in both imbalanced and balanced transductive few-shot learning, as it can be seen in section 4.

### 2.3. Class balancing

The commonly used transductive few-shot learning benchmarks use perfectly class-balanced tasks [2]. Several methods exploit this bias by encouraging class-balanced predictions over queries, thereby improving their performance. One way is to optimize the query probability matrix, $P$, to have specific row and column sums $\mathbf{p}$ and $\mathbf{q}$ respectively by using the Sinkhorn-Knopp algorithm [12, 17]. The row sum $\mathbf{p}$ amounts to the probability distribution of every query, while the column sum $\mathbf{q}$ corresponds to the total number of queries per class. Another way is to maximize the entropy of the marginal distribution of predicted labels over queries, thus encouraging it to follow a uniform distribution [2].

However, the authors of [39] argue that using perfectly class-balanced tasks is unrealistic. They propose a more realistic imbalanced setting and protocol, benchmarking the performance of several methods. They also introduce a relaxed version of [2] based on $\alpha$-divergence, which can effectively address class-imbalanced tasks. This imbalanced setting is also addressed by adaptive dimensionality reduction and clustering based on Variational Bayesian inference [13].

## 3. Method

### 3.1. Problem formulation

**Representation learning** We assume access to a *base dataset* $D_{\text{base}} = \{(x_i, \mathbf{y}_i)\}_{i=1}^{I}$ of $I$ images, where each image $x_i$ has a one-hot encoded label $\mathbf{y}_i$ corresponding to a class from a set of *base classes* $C_{\text{base}}$. Denoting by $\mathcal{X}$ the image space, we assume access to a network $f_\theta : \mathcal{X} \to \mathbb{R}^d$ that has been trained on $D_{\text{base}}$, which maps an image $x \in \mathcal{X}$ to an embedding $f_\theta(x) \in \mathbb{R}^d$.

**Inference** We assume access to a *novel dataset* $D_{\text{novel}}$ consisting of images with corresponding one-hot encoded labels from a set $C_{\text{novel}}$ of *novel classes*, where $C_{\text{novel}} \cap C_{\text{base}} = \emptyset$. We sample $N$-way $K$-shot tasks, each consisting of a labeled support set, $S = \{(x_i^s, \mathbf{y}_i^s)\}_{i=1}^{L}$, where each image $x_i^s$ has a corresponding one-hot encoded label $\mathbf{y}_i^s = (y_{ji}^s)_{j=1}^{N} \in \{0,1\}^N$ over $C_{\text{novel}}$, with $N$ novel classes in total and $K$ examples per class, such that the number of examples in $S$ is $L = |S| = NK$. We focus on the transductive setting, therefore a task also contains an unlabeled *query set* $Q = \{x_i^q\}_{i=1}^{M}$ sampled from the same $N$ classes as the support set $S$ where the number of examples in $Q$ is $M = |Q|$.

**Feature extraction** Given a novel task, we embed all images in $S$ and $Q$ using $f_\theta$ and apply a feature pre-processing function $\eta : \mathbb{R}^d \to \mathbb{R}^d$, to be discussed in section 4. Let $V^s = (\mathbf{v}_1^s \cdots \mathbf{v}_L^s)$ be the $d \times L$ matrix containing the embeddings of $S$, where $\mathbf{v}_i^s = \eta(f_\theta(x_i^s)) \in \mathbb{R}^d$. Similarly, let $V^q = (\mathbf{v}_1^q \cdots \mathbf{v}_M^q)$ be the $d \times M$ matrix containing the embeddings of $Q$, where $\mathbf{v}_i^q = \eta(f_\theta(x_i^q)) \in \mathbb{R}^d$. We also represent $V^s, V^q$ as sets $\mathcal{V}^s = \{\mathbf{v}_i^s\}_{i=1}^{L}$, $\mathcal{V}^q = \{\mathbf{v}_i^q\}_{i=1}^{M}$. Both sets remain fixed in our method.

### 3.2. Class centroids

Following [36], we define a class centroid $\mathbf{c}_j \in \mathbb{R}^d$ in the embedding space for each class $j$ in the support set $S$. The centroids are learnable variables but initialized by standard class prototypes [36]. That is, the centroid $\mathbf{c}_j$ of class $j$ is initialized by the mean

$$\mathbf{c}_j = \frac{1}{K} \sum_{\mathbf{v}_i^s \in \mathcal{V}^s} y_{ji}^s \mathbf{v}_i^s \tag{1}$$

of support embeddings of class $j$. Let $C = (\mathbf{c}_1 \cdots \mathbf{c}_N)$ be the $d \times N$ matrix containing the learnable centroids of all $N$ support classes. We also represent $C$ as a set $\mathcal{C} = \{\mathbf{c}_j\}_{j=1}^{N}$.

### 3.3. Nearest neighbour graph

We collect centroids, support and query embeddings in a single $d \times T$ matrix

$$V = (\mathbf{v}_1 \cdots \mathbf{v}_T) = (C \;\; V^s \;\; V^q), \tag{2}$$

where $T = N + L + M$. We also represent $V$ as a set $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^{T}$. Following [14, 17], we construct a $k$-nearest neighbour graph of $\mathcal{V}$. We define edges between distinct nearest neighbours in $\mathcal{V}$ that are both not centroids:

$$E = \{(\mathbf{v}_i, \mathbf{v}_j) \in \mathcal{V}^2 \setminus \mathcal{C}^2 : \mathbf{v}_i \in \text{NN}_k(\mathbf{v}_j)\}, \tag{3}$$

where $\text{NN}_k(\mathbf{v})$ is the set of $k$-nearest neighbours of $\mathbf{v}$ in $\mathcal{V}$, excluding $\mathbf{v}$. Given $E$, we define the $T \times T$ *affinity matrix* $A = (a_{ij})$ as

$$a_{ij} = \begin{cases} \exp\left(-\frac{\|\mathbf{v}_i - \mathbf{v}_j\|^2}{g_{ij}\sigma^2}\right), & \text{if } (\mathbf{v}_i, \mathbf{v}_j) \in E \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where $g_{ij}$ is a learnable pairwise scaling factor for every pair $(\mathbf{v}_i, \mathbf{v}_j)$, collectively represented by $T \times T$ matrix $G = (g_{ij})$, and $\sigma^2$ is a global scaling factor set equal to the standard deviation of $\|\mathbf{v}_i - \mathbf{v}_j\|^2$ for $(\mathbf{v}_i, \mathbf{v}_j) \in \mathcal{V}^2$ as in [33]. We symmetrize $A$ into the $T \times T$ *adjacency matrix*, $W = \frac{1}{2}(A + A^\top)$. We calculate $W_B$ which is a scaled version of $W$ defined as:

$$W_B = W \circ B \tag{5}$$

where $B \in [0,1)^{T \times T}$ is a learnable $T \times T$ matrix and $\circ$ is the Hadamard product. We normalize $W_B$ by

$$\mathcal{W} = D^{-1/2} W_B D^{-1/2}, \tag{6}$$

where $D = \text{diag}(W_B \mathbf{1}_T)$ is the $T \times T$ degree matrix of $W_B$.

### 3.4. Label Propagation

**Labels** Following [45], we define the $N \times T$ *label matrix*

$$Y = (Y^c \;\; Y^s \;\; Y^q) = (I_N \;\; \mathbf{0}_{N\times L} \;\; \mathbf{0}_{N\times M}). \tag{7}$$

That is, $Y$ has one row per class and one column per example, which is an one-hot encoded label for every class centroid in $\mathcal{C}$ and a zero vector for both support embeddings in $\mathcal{V}^s$ and query embeddings in $\mathcal{V}^q$.

**Label propagation** Given the graph represented by $\mathcal{W}$ and the label matrix $Y$, label propagation amounts to

$$Z = Y(I - \beta\mathcal{W})^{-1}, \tag{8}$$

where $\beta \in [0, 1)$ is a scalar hyperparameter that is referred to as $\alpha$ in the standard label propagation [45].

**Predicted probabilities** The resulting $N \times T$ matrix $Z = (\mathbf{z}_1 \cdots \mathbf{z}_T)$ is called *manifold class similarity* matrix, in the sense that column $\mathbf{z}_i \in \mathbb{R}^N$ expresses how similar embedding vector $\mathbf{v}_i$ is to each of the $N$ support classes. By taking softmax over columns

$$\mathbf{p}_i = \frac{\exp(\tau \mathbf{z}_i)}{\sum_{j=1}^N \exp(\tau z_{ji})}, \tag{9}$$

with $\tau > 0$ being a positive scalar hyperparameter, we define the $N \times T$ *probability matrix*

$$P = (\mathbf{p}_1 \;\cdots\; \mathbf{p}_T) = (P^c \;\; P^s \;\; P^q). \tag{10}$$

Matrix $P$ expresses the predicted probability distributions over the support classes. If $P = (p_{ji})$, element $p_{ji}$ expresses the predicted probability of class $j$ for example $i$. Similarly for class centroids $P^c = (p^c_{ji}) \in \mathbb{R}^{N\times N}$, support examples $P^s = (p^s_{ji}) \in \mathbb{R}^{N\times L}$ and queries $P^q = (p^q_{ji}) \in \mathbb{R}^{N\times M}$.

### 3.5. Loss function: Class balancing or not

The set of all learnable parameters is $\Phi = \{C, G, B\}$ is optimized jointly using a modified mutual information loss [2, 39]. We distinguish between class-balanced and imbalanced tasks.

#### 3.5.1 Class-balanced tasks

Following [2], we optimize parameters $\Phi$ using three loss terms. The first is the standard average cross-entropy over the labeled support examples:

$$L_{\text{CE}}(P^s) = -\frac{1}{L}\sum_{i=1}^{L}\sum_{j=1}^{N} y^s_{ji}\log(p^s_{ji}). \tag{11}$$

The second is the average, over queries, entropy of predicted class probability distributions per query

$$\overline{\mathcal{H}}(P^q) = -\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N} p^q_{ji}\log(p^q_{ji}). \tag{12}$$

This term aims at minimizing the uncertainty of the predicted probability distribution of every query, hence encouraging confident predictions. The third term is

$$-\mathcal{H}(\bar{\mathbf{p}}^q) = \sum_{j=1}^{N} \bar{p}^q_j \log(\bar{p}^q_j), \tag{13}$$

where $\bar{p}^q_j = \frac{1}{M}\sum_{i=1}^M p^q_{ji}$ and $\bar{\mathbf{p}}^q = (\bar{p}^q_j)_{j=1}^N = P^q \mathbf{1}_M \in \mathbb{R}^N$ is a vector representing the average predicted probability distribution of set $Q$. By maximizing its entropy, this term aims at maximizing its uncertainty, encouraging it to be uniform, hence balancing over classes.

The complete loss function to be minimized w.r.t. $\Phi$ is

$$L_{\text{bal}} = \lambda_3 L_{\text{CE}}(P^s) + \lambda_2\overline{\mathcal{H}}(P^q) - \lambda_1\mathcal{H}(\bar{\mathbf{p}}^q), \tag{14}$$

where $\lambda_1, \lambda_2, \lambda_3$ are scalar hyperparameters.

#### 3.5.2 Imbalanced tasks

By encouraging the average predicted probability distribution to be uniform, the third term (13) is strongly biased towards class-balanced tasks. To make the loss more tolerant to imbalanced distributions, a relaxed version has been proposed based on the $\alpha$-divergence [39]. In particular, the second (12) and third term (13) become respectively

$$\overline{\mathcal{H}}_\alpha(P^q) = -\frac{1}{\alpha - 1}\frac{1}{M}\sum_{i=1}^{M}\sum_{j=1}^{N}(p^q_{ji})^\alpha \tag{15}$$

$$-\mathcal{H}_\alpha(\bar{\mathbf{p}}^q) = \frac{1}{\alpha - 1}\sum_{j=1}^{N}(\bar{p}^q_j)^\alpha \tag{16}$$

In this case, the complete loss function (14) to be minimized with respect to $\Phi$ is modified as

$$L_{\text{imbal}} = \lambda_3 L_{\text{CE}}(P^s) + \lambda_2\overline{\mathcal{H}}_\alpha(P^q) - \lambda_1\mathcal{H}_\alpha(\bar{\mathbf{p}}^q). \tag{17}$$

### 3.6. Manifold parameter optimization

In contrast to [2] and [39], rather than only optimizing the class centroids, we optimize the entire set of manifold parameters $\Phi$, which includes the class centroids $C$ as well as the manifold-specific parameters $G$ (4) and $B$ (5). We update $\Phi$ by minimizing (14) or (17) through any gradient-based optimization algorithm with learning rate $\epsilon$. The entire procedure from graph construction in subsection 3.3 to manifold parameter optimization in subsection 3.6 is iterated for $r$ steps. Algorithm 1 summarizes the complete optimization procedure of our method.

### 3.7. Transductive Inference

Upon convergence of the optimization of manifold parameters $\Phi$, we obtain the final query probability matrix $P^q$ (10)

**Algorithm 1:** Adaptive Manifold (AM).

**input** : Pre-trained backbone $f_\theta$
**input** : labeled support set $S$ with $|S| = L$
**input** : unlabeled query set $Q$ with $|Q| = M$

$(V^s, V^q) \leftarrow (f_\theta(S), f_\theta(Q))$
$C \leftarrow \text{CENTROIDS}(V^s)$ ▷ class centroids (1)
$\mathcal{V} \leftarrow \{C, V^s, V^q\}$
$(G, B) \leftarrow \text{INITIALIZE}()$
$\Phi \leftarrow \{C, G, B\}$
**for** *r steps* **do**
  $A \leftarrow \text{AFFINITY}(\mathcal{V}; G, k)$ ▷ affinity matrix (4)
  $W \leftarrow \frac{1}{2}(A + A^T)$ ▷ symmetric adjacency matrix
  $W_B \leftarrow W \circ B$ ▷ scaled adjacency matrix (5)
  $\mathcal{W} \leftarrow D^{-1/2} W_B D^{-1/2}$ ▷ adjacency matrix (6)
  $Y \leftarrow (I_N \ \mathbf{0}_{N\times L} \ \mathbf{0}_{N\times M})$ ▷ label matrix (7)
  $Z \leftarrow Y(I - \beta\mathcal{W})^{-1}$ ▷ label propagation (8)
  $P \leftarrow \text{SOFTMAX}(Z)$ ▷ class probabilities (9)
  $L_{\text{bal}}/L_{\text{imbal}} \leftarrow \text{LOSS}(P; \Phi)$ ▷ loss function (14) or (17)
  $\Phi \leftarrow \text{UPDATE}(\Phi; L_{\text{bal}}/L_{\text{imbal}})$ ▷ update $\Phi$
**return** $P^q$

and for each query $x_i^q \in Q$, we predict the *pseudo-label*

$$\hat{y}_i^q = \arg\max_j p_{ji}^q \tag{18}$$

corresponding to the maximum element of the $i$-th column of matrix $P^q$.

# 4. Experiments

## 4.1. Setup

**Datasets** In the state of the art comparisons regarding the imbalanced setting we experiment with the following datasets, *mini*ImageNet [40], *tiered*ImageNet [3] and CUB [5]. In the state of the art comparisons regarding the balanced setting, we also experiment on CIFAR-FS [5, 16].

**Backbones** We use the two pre-trained backbones from the publicly available code [39], namely ResNet-18 and WideResNet-28-10 (WRN-28-10). All backbones are trained using standard cross entropy loss on $D_{\text{base}}$ for 90 epochs with learning rate 0.1, divided by 10 at epochs 45 and 66. Color jittering, random cropping and random horizontal flipping augmentations are used during training. We also carry out experiments using the publicly available code and the pre-trained WRN-28-10 backbones provided by [17].

**Tasks** Unless otherwise stated, we consider $N$-way, $K$-shot tasks with $N = 5$ randomly sampled classes from $C_{\text{novel}}$ and $K \in \{1, 5\}$ random labeled examples for the support set $S$. The query set $Q$ contains $M = 75$ query examples in total. In the balanced setting, there are $\frac{M}{N} = \frac{75}{5} = 15$ queries per class. In the imbalanced setting, the total number of queries remains $M = 75$. Following [39], we sample imbalanced tasks by modeling the proportion of examples from each class in $Q$ as a vector $\boldsymbol{\pi} = (\pi_1, \dots, \pi_N)$ sampled from a symmetric Dirichlet distribution $\text{Dir}(\gamma)$ with parameter $\gamma = 2$. We follow [39] and [17], performing 10000 and 1000 tasks respectively when using the code and settings of each work.

**Implementation details** Our implementation is in Pytorch [29]. We carry out experiments for balanced and imbalanced transductive few-shot learning using the publicly available code provided by [39][1]. For additional experiments in the balanced setting, we use the publicly available code provided from [17][2]. We used Adam optimizer for the manifold parameter optimization in subsection 3.6.

**Hyperparameters** Following [39], we keep the same values of hyper-parameters $\tau$, $\lambda_1$, $\lambda_2$, $\lambda_3$, $\epsilon$ and $r$. We set $\epsilon = 0.0001$, $r = 1000$, $\tau = 15$ (9). In the imbalanced setting we set $\lambda_1 = \lambda_2 = \lambda_3 = 1$, while in the balanced setting we set $\lambda_1 = \lambda_3 = 1$ and $\lambda_2 = 10$. Regarding hyper-parameter $\alpha$ (15),(16) we ablate it in section subsection 4.2 and set $\alpha = 2$ for 1-shot and $\alpha = 5$ for 5-shot for all experiments unless stated otherwise. For label propagation, we set $k = 20$ (3) for 1-shot and $k = 10$ for 5-shot; we initialize $G = J_T$ (4) and $B = J_T$ (5) where $J_T$ is the $T \times T$ all-ones matrix; we initialize $\beta = 0.8$ (8) for 1-shot and $\beta = 0.9$ for 5-shot. We optimized $k$, $\beta$ and the initialization of $G$ and $B$ on the *mini*ImageNet validation set using ResNet-18. To avoid hyperparameter overfitting, *all hyper-parameters are kept fixed across all datasets and backbones*.

**Baselines** In the imbalanced setting, we carry out experiments using the publicly available source code provided by $\alpha$-TIM [39]. We compare against all available methods implemented in the code as well as an imbalanced variant of ProtoLP [47], which we obtain from the official code of ProtoLP [47][3] by removing Sinkhorn-Knopp balancing. We do not compare against [13] since there is no publicly available source code.

In the balanced setting, we compare against all methods provided in the official code of [39] as well as ProtoLP [47]. Further comparisons are made against [12, 17] using the reported results from [17]. For fair comparisons we use the publicly available source code from [17] to reproduce results for EASE+SIAMESE [46][4], ProtoLP [47], TIM and AM. It should be noted that [47] uses data augmentations to improve their performance. In order to compare all algorithms fairly we do not use data augmentations in our comparisons.

**Feature pre-processing** We experiment with two commonly used feature pre-processing methods, denoted as $\eta$ in subsection 3.1, namely $\ell_2$-normalization and the method used in [12, 17], which we refer to as PLC. $\ell_2$-normalization

[1] https://github.com/oveilleux/Realistic_Transductive_Few_Shot
[2] https://github.com/MichalisLazarou/iLPC
[3] https://github.com/allenhaozhu/protoLP
[4] https://github.com/allenhaozhu/EASE

Table 1. *Ablation study of algorithmic components* of both balanced and imbalanced versions of our method AM on *mini*ImageNet. $\text{NN}_k$: $k$-nearest neighbour graph; otherwise, complete graph. $C$: learnable class centroids. $G$: learnable pairwise scaling factors $G$ (4). $B$: learnable adjacency matrix $B$ (5). PLC: feature pre-processing as defined in subsection 4.1.

| Components | | | | | Imbalanced | | | | Balanced | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | ResNet-18 | | WRN-28-10 | | ResNet-18 | | WRN-28-10 | |
| $\text{NN}_k$ | $C$ | $G$ | $B$ | PLC | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot | 1-shot | 5-shot |
| | | | | | $60.21_{\pm 0.27}$ | $74.24_{\pm 0.21}$ | $63.34_{\pm 0.27}$ | $76.19_{\pm 0.21}$ | $59.09_{\pm 0.21}$ | $71.54_{\pm 0.19}$ | $62.38_{\pm 0.21}$ | $73.46_{\pm 0.19}$ |
| ✓ | | | | | $63.95_{\pm 0.27}$ | $81.15_{\pm 0.17}$ | $67.14_{\pm 0.27}$ | $83.40_{\pm 0.16}$ | $63.82_{\pm 0.22}$ | $80.47_{\pm 0.15}$ | $67.22_{\pm 0.21}$ | $82.58_{\pm 0.16}$ |
| ✓ | ✓ | | | | $68.57_{\pm 0.28}$ | $82.69_{\pm 0.16}$ | $71.22_{\pm 0.26}$ | $84.74_{\pm 0.16}$ | $73.43_{\pm 0.23}$ | $84.37_{\pm 0.14}$ | $75.94_{\pm 0.22}$ | $86.55_{\pm 0.13}$ |
| ✓ | ✓ | ✓ | | | $70.16_{\pm 0.29}$ | $82.62_{\pm 0.17}$ | $72.89_{\pm 0.28}$ | $84.89_{\pm 0.16}$ | $75.59_{\pm 0.27}$ | $84.80_{\pm 0.15}$ | $78.72_{\pm 0.25}$ | $87.11_{\pm 0.13}$ |
| ✓ | ✓ | | ✓ | | $69.11_{\pm 0.29}$ | $82.97_{\pm 0.16}$ | $71.64_{\pm 0.28}$ | $85.16_{\pm 0.15}$ | $74.85_{\pm 0.25}$ | $84.66_{\pm 0.14}$ | $77.70_{\pm 0.23}$ | $86.91_{\pm 0.13}$ |
| ✓ | ✓ | ✓ | ✓ | | **70.24**$_{\pm 0.29}$ | $82.71_{\pm 0.17}$ | **73.22**$_{\pm 0.29}$ | $85.00_{\pm 0.16}$ | $76.06_{\pm 0.28}$ | $84.82_{\pm 0.15}$ | $79.37_{\pm 0.26}$ | $87.12_{\pm 0.13}$ |
| ✓ | ✓ | ✓ | ✓ | ✓ | $69.97_{\pm 0.29}$ | **83.31**$_{\pm 0.17}$ | $71.98_{\pm 0.29}$ | **85.66**$_{\pm 0.15}$ | **77.35**$_{\pm 0.27}$ | **85.47**$_{\pm 0.14}$ | **80.99**$_{\pm 0.26}$ | **87.86**$_{\pm 0.13}$ |

is defined as $\frac{\mathbf{v}}{||\mathbf{v}||_2}$ for $\mathbf{v} \in V$. PLC, standing for *power transform, $\ell_2$-normalization, centering*, performs elementwise power transform $\mathbf{v}^{\frac{1}{2}}$ for $\mathbf{v} \in V$, followed by $\ell_2$-normalization and centering, subtracting the mean over $V$.

In the balanced and imbalanced settings respectively, we refer to our method as AM, $\alpha$-AM when using $\ell_2$-normalization and as $\text{AM}_{\text{PLC}}$, $\alpha\text{-AM}_{\text{PLC}}$ when using PLC pre-processing. TIM [2] and $\alpha$-TIM [39] use only $\ell_2$-normalization originally. For fair comparison, we apply PLC pre-processing on TIM and $\alpha$-TIM, referring to them as $\text{TIM}_{\text{PLC}}$ and $\alpha\text{-TIM}_{\text{PLC}}$ in the balanced and imbalanced settings respectively.

**Reporting results** In every table we denote the best performing results with bold regardless the pre-processing method used. Nevertheless, since our work is influenced by [2] and [39], we also compare with these two methods under the same feature pre-processing settings. In Table 2, Table 3, Table 4 and Table 5, we use the code by [39], reporting the mean accuracy over 10000 tasks [39]. In Table 6, we use the code by [17], reporting the mean accuracy and 95% confidence interval over 1000 tasks. In the ablation study in Table 1 we use the code by [39], however, since we ablate our own method, we report both the mean accuracy and the 95% confidence interval.

### 4.2. Ablation study

**Ablation of hyper-parameter $\alpha$** Figure 2 ablates $\alpha$-AM and $\alpha$-TIM with respect to $\alpha$. It is evident that the value of $\alpha$ has a lot more effect in the 1-shot setting. Furthermore, $\alpha$ behaves similarly for both $\alpha$-AM and $\alpha$-TIM. Nevertheless, in the majority of the cases $\alpha$-AM outperforms $\alpha$-TIM. Since the optimal value of $\alpha$ for $\alpha$-AM is 2 and 5 for the 1-shot and 5-shot settings respectively, we choose these values in our imbalanced experiments unless stated otherwise.

**Algorithmic components** We ablate all components of our method under both the imbalanced (17) and balanced (14) settings in Table 1. As it can be seen from the first and second

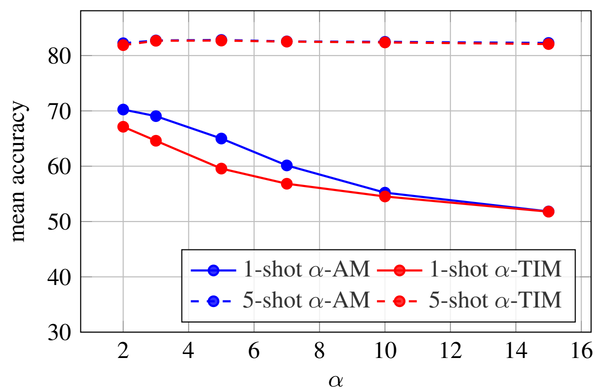



Figure 2. *Effect of parameter* $\alpha$ on $\alpha$-AM and $\alpha$-TIM, on 1-shot and 5-shot *mini*ImageNet using ResNet-18.

rows, using a $k$-nearest neighbour graph gives significant performance improvement over using a dense graph. Adapting the centroids, $C$, brings further substantial improvement. Adapting the centroids, $C$, along with either $G$ or $B$ brings further performance improvement. Adapting both manifold parameters $G$ and $B$ along with $C$ provides better performance than just adapting either $G$ or $B$ in most experiments, especially in the balanced setting. Using PLC pre-processing yields further performance improvement except in the 1-shot imbalanced setting.

### 4.3. Comparison with state of the art

**Imbalanced transductive few-shot learning** Table 2 and Table 3 show that our method achieves new state of the art performance using both ResNet-18 and WRN-28-10 on all three datasets and both 1-shot and 5-shot settings. Impressively, we improve the 1-shot state of the art performance in all cases significantly, by as much as 4.2% on CUB with ResNet-18. Even though we outperform $\alpha$-TIM without PLC pre-processing in every experiment, PLC brings further improvement in 5-shot, while is not being beneficial in 1-shot. Interestingly, PLC pre-processing does not have the same

Table 2. *Imbalanced transductive inference* on *mini*ImageNet and *tiered*ImageNet. Results as reported by [39]. *: Results were reproduced using the official code provided by [39]. † Our reproduction of the imbalanced ProtoLP using the official code.

| METHOD | *mini*IMAGENET | | *tiered*IMAGENET | |
|---|---|---|---|---|
| | 1-shot | 5-shot | 1-shot | 5-shot |
| RESNET-18 | | | | |
| Entropy-min [7] | 58.50 | 74.80 | 61.20 | 75.50 |
| LR+ICI [41] | 58.70 | 73.50 | 74.60 | 85.10 |
| PT-MAP [12] | 60.10 | 67.10 | 64.10 | 70.00 |
| LaplacianShot [48] | 65.40 | 81.60 | 72.30 | 85.70 |
| BD-CSPN [22] | 67.00 | 80.20 | 74.10 | 84.80 |
| ProtoLP [47]† | 65.42 | 78.48 | 71.12 | 82.51 |
| TIM [2] | 67.30 | 79.80 | 74.10 | 84.10 |
| $\alpha$-TIM [39] | 67.40 | 82.50 | 74.40 | 86.60 |
| $\alpha$-TIM$_{\text{PLC}}$* [39] | 63.38 | 82.80 | 70.17 | 86.82 |
| $\alpha$-AM | **70.24** | 82.71 | **77.28** | 86.97 |
| $\alpha$-AM$_{\text{PLC}}$ | 69.97 | **83.31** | 76.44 | **87.19** |
| WRN-28-10 | | | | |
| Entropy-min [7] | 60.40 | 76.20 | 62.90 | 77.30 |
| PT-MAP [12] | 60.60 | 66.80 | 65.10 | 71.00 |
| LaplacianShot [48] | 68.10 | 83.20 | 73.50 | 86.80 |
| BD-CSPN [22] | 70.40 | 82.30 | 75.40 | 85.90 |
| ProtoLP [47]† | 68.90 | 80.97 | 72.92 | 83.95 |
| TIM [2] | 69.80 | 81.60 | 75.80 | 85.40 |
| $\alpha$-TIM [39] | 69.80 | 84.80 | 76.00 | 87.80 |
| $\alpha$-TIM$_{\text{PLC}}$* [39] | 66.50 | 85.12 | 71.97 | 88.28 |
| $\alpha$-AM | **73.22** | 85.00 | **78.94** | 88.44 |
| $\alpha$-AM$_{\text{PLC}}$ | 71.98 | **85.66** | 78.75 | **88.69** |

Table 3. *Imbalanced transductive inference* on CUB. Results as reported by [39]. *: Results were reproduced using the official code provided by [39]. † Our reproduction of the imbalanced ProtoLP using the official code.

| METHOD | CUB | |
|---|---|---|
| | 1-shot | 5-shot |
| RESNET-18 | | |
| PT-MAP [12] | 65.10 | 71.30 |
| Entropy-min [7] | 67.50 | 82.90 |
| LaplacianShot [48] | 73.70 | 87.70 |
| BD-CSPN [22] | 74.50 | 87.10 |
| ProtoLP [47]† | 74.47 | 86.01 |
| TIM [2] | 74.80 | 86.90 |
| $\alpha$-TIM [39] | 75.70 | 89.80 |
| $\alpha$-TIM$_{\text{PLC}}$* [39] | 70.95 | 89.56 |
| $\alpha$-AM | **79.92** | 89.83 |
| $\alpha$-AM$_{\text{PLC}}$ | 78.62 | **89.86** |

effect on $\alpha$-TIM, providing only marginal improvement in the 5-shot while being detrimental in the 1-shot.

Table 4. *Balanced transductive inference* on *mini*ImageNet and *tiered*ImageNet. All results were reproduced using the official code provided by [39]. †: Our reproduction using the official code [47].

| METHOD | *mini*IMAGENET | | *tiered*IMAGENET | |
|---|---|---|---|---|
| | 1-shot | 5-shot | 1-shot | 5-shot |
| RESNET-18 | | | | |
| LaplacianShot [48] | 70.24 | 82.10 | 77.28 | 86.22 |
| BD-CSPN [22] | 69.36 | 82.06 | 76.36 | 86.18 |
| PT-MAP [12] | 76.88 | 85.18 | 82.89 | 88.64 |
| protoLP [47]† | 76.96 | 84.90 | 83.06 | 88.55 |
| TIM [2] | 73.81 | 84.91 | 80.13 | 88.61 |
| TIM$_{\text{PLC}}$ [2] | 69.33 | 84.53 | 76.36 | 88.33 |
| AM | 76.06 | 84.82 | 82.42 | 88.61 |
| AM$_{\text{PLC}}$ | **77.35** | **85.47** | **83.40** | **89.07** |
| WRN-28-10 | | | | |
| LaplacianShot [48] | 72.91 | 83.85 | 78.85 | 87.27 |
| BD-CSPN [22] | 72.16 | 83.78 | 77.88 | 87.23 |
| PT-MAP [12] | 80.35 | 87.37 | 84.84 | 89.86 |
| ProtoLP [47]† | 80.45 | 87.21 | 84.92 | 89.80 |
| TIM [2] | 77.78 | 87.43 | 82.28 | 89.84 |
| TIM$_{\text{PLC}}$ [2] | 73.52 | 86.95 | 78.23 | 89.56 |
| AM | 79.37 | 87.12 | 84.07 | 89.69 |
| AM$_{\text{PLC}}$ | **80.99** | **87.86** | **85.26** | **90.30** |

Table 5. *Balanced transductive inference* on CUB. All results were reproduced using the official code provided by [39]. †: Our reproduction using the official code [47].

| METHOD | CUB | |
|---|---|---|
| | 1-shot | 5-shot |
| LaplacianShot [48] | 79.55 | 88.96 |
| BD-CSPN [22] | 78.52 | 89.02 |
| PT-MAP [12] | 86.05 | 91.28 |
| ProtoLP [47]† | 86.39 | 91.07 |
| TIM [2] | 82.87 | 91.58 |
| TIM$_{\text{PLC}}$ [2] | 77.69 | 91.17 |
| AM | 85.59 | 91.24 |
| AM$_{\text{PLC}}$ | **86.64** | **91.78** |

**Balanced transductive few-shot learning** Tables 4 and 5 show that AM$_{\text{PLC}}$ outperforms all other methods, with its closest competitors being PT-MAP [12] and ProtoLP [47]. Notably, our superiority is not due to pre-processing since PT-MAP and ProtoLP also use PLC. AM$_{\text{PLC}}$ also significantly outperforms both versions of TIM. Interestingly, the performance of TIM always drops when PLC pre-processing is used, while AM always improves. Even without PLC, AM significantly outperforms TIM by $2-4\%$ in 1-shot, while being on par or slightly worse by $0.1-0.3\%$ in 5-shot.

We use the publicly available code and pre-trained WRN-28-10 provided by [17] to compare AM with other state of the art methods that were not included in the code pro-

Table 6. *Balanced transductive inference state of the art*. Results were reproduced using the official code provided by [17] and the official codes of these method [2, 46, 47]. *: Results as reported by [17].

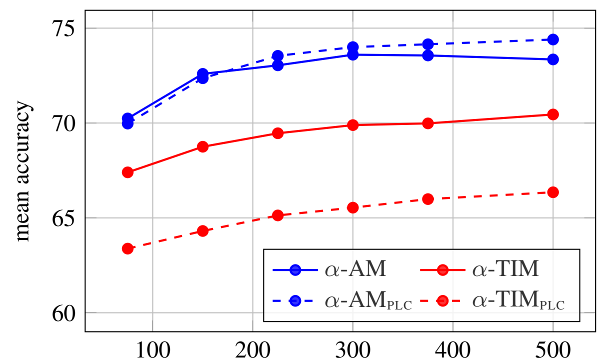
| METHOD | *mini*IMAGENET 1-shot | *mini*IMAGENET 5-shot | *tiered*IMAGENET 1-shot | *tiered*IMAGENET 5-shot | CIFAR-FS 1-shot | CIFAR-FS 5-shot | CUB 1-shot | CUB 5-shot |
|---|---|---|---|---|---|---|---|---|
| WRN-28-10 | | | | | | | | |
| PT+MAP [12]* | $82.88_{\pm 0.73}$ | $88.78_{\pm 0.40}$ | $88.15_{\pm 0.71}$ | $92.32_{\pm 0.40}$ | $86.91_{\pm 0.72}$ | $90.50_{\pm 0.49}$ | $91.37_{\pm 0.61}$ | $93.93_{\pm 0.32}$ |
| iLPC [17]* | $83.05_{\pm 0.79}$ | $88.82_{\pm 0.42}$ | $88.50_{\pm 0.75}$ | $92.46_{\pm 0.42}$ | $86.51_{\pm 0.75}$ | $90.60_{\pm 0.48}$ | $91.03_{\pm 0.63}$ | $94.11_{\pm 0.30}$ |
| EASE+SIAMESE [46] | $\mathbf{83.44}_{\pm 0.77}$ | $88.66_{\pm 0.43}$ | $\mathbf{88.69}_{\pm 0.73}$ | $92.47_{\pm 0.41}$ | $86.71_{\pm 0.77}$ | $90.28_{\pm 0.51}$ | $91.44_{\pm 0.63}$ | $93.85_{\pm 0.32}$ |
| EASE+SIAMESE$_{\text{PLC}}$ [46] | $82.13_{\pm 0.81}$ | $87.34_{\pm 0.46}$ | $88.42_{\pm 0.73}$ | $92.19_{\pm 0.41}$ | $86.74_{\pm 0.78}$ | $90.22_{\pm 0.51}$ | $\mathbf{91.49}_{\pm 0.63}$ | $93.32_{\pm 0.32}$ |
| ProtoLP [47] | $83.22_{\pm 0.80}$ | $88.64_{\pm 0.43}$ | $88.55_{\pm 0.75}$ | $92.42_{\pm 0.42}$ | $86.73_{\pm 0.79}$ | $90.21_{\pm 0.52}$ | $91.39_{\pm 0.65}$ | $93.93_{\pm 0.32}$ |
| TIM [2] | $77.65_{\pm 0.72}$ | $88.21_{\pm 0.40}$ | $83.88_{\pm 0.74}$ | $91.89_{\pm 0.41}$ | $82.63_{\pm 0.70}$ | $90.28_{\pm 0.46}$ | $87.50_{\pm 0.62}$ | $93.59_{\pm 0.30}$ |
| TIM$_{\text{PLC}}$ [2] | $75.77_{\pm 0.67}$ | $88.37_{\pm 0.40}$ | $83.22_{\pm 0.70}$ | $92.13_{\pm 0.40}$ | $80.52_{\pm 0.70}$ | $90.25_{\pm 0.46}$ | $85.58_{\pm 0.61}$ | $93.48_{\pm 0.31}$ |
| AM | $80.74_{\pm 0.81}$ | $87.75_{\pm 0.42}$ | $86.38_{\pm 0.78}$ | $91.85_{\pm 0.85}$ | $85.93_{\pm 0.74}$ | $90.13_{\pm 0.47}$ | $90.24_{\pm 0.65}$ | $93.43_{\pm 0.30}$ |
| AM$_{\text{PLC}}$ | $83.40_{\pm 0.74}$ | $\mathbf{89.08}_{\pm 0.40}$ | $88.31_{\pm 0.73}$ | $\mathbf{92.60}_{\pm 0.39}$ | $\mathbf{86.91}_{\pm 0.74}$ | $\mathbf{90.80}_{\pm 0.46}$ | $91.32_{\pm 0.60}$ | $\mathbf{94.14}_{\pm 0.29}$ |

vided by [39] such as EASE+SIAMESE [46] and iLPC [17]. Table 6 shows that AM$_{\text{PLC}}$ outperforms all methods in the majority of the experiments. Our superiority is not due to pre-processing since we provided results for TIM and EASE+SIAMESE using PLC while PT-MAP, iLPC and ProtoLP use PLC as part of their method.

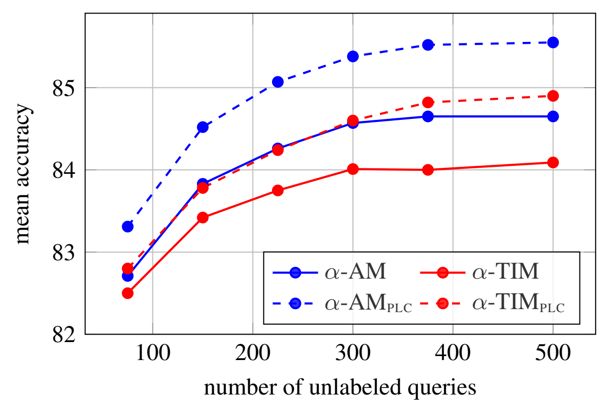
### 4.4. Effect of unlabeled data

We investigate the effect of the quantity of unlabeled queries $M$, comparing $\alpha$-AM against $\alpha$-TIM. It can be seen from Figure 3 that $\alpha$-AM outperforms $\alpha$-TIM in both 1-shot and 5-shot settings using both $\ell_2$-normalization and PLC. The performance gap is impressive in the 1-shot setting since $\alpha$-AM and $\alpha$-AM$_{\text{PLC}}$ outperform $\alpha$-TIM and $\alpha$-TIM$_{\text{plc}}$ by as much as $3.7\%$ and $8.7\%$ respectively when $M = 300$. Interestingly, the performance gap tends to increase as the number of unlabeled queries increases. This is because $\alpha$-AM exploits the data manifold through the $k$-nearest neighbour graph while $\alpha$-TIM works in Euclidean space.

Furthermore, the robustness of our method is evident from the performance gap between $\alpha$-AM and $\alpha$-AM$_{\text{PLC}}$, which is much smaller than the gap between $\alpha$-TIM and $\alpha$-TIM$_{\text{plc}}$ in the 1-shot setting (Figure 3a) while is about the same in the 5-shot setting (Figure 3b).




(a) 1-shot *mini*ImageNet




(b) 5-shot *mini*ImageNet

Figure 3. *Effect of number of unlabeled queries* $M$ on $\alpha$-AM and $\alpha$-TIM using ResNet-18.

## 5. Conclusion

In this work we propose a novel method named *Adaptive Manifold*, AM, that achieves new state of the art performance in the imbalanced transductive few-shot learning. Our method significantly outperforms other state of the art methods on multiple datasets using multiple backbones especially in the 1-shot setting. AM combines the complementary strengths of iterative class centroid adaptation and exploiting the underlying data manifold through label propagation. Specifically, our method leverages the manifold class similarities to measure class probabilities for the unlabeled query examples and iteratively adapts the manifold parameters by minimizing a specialized loss function. The robustness of our method is further validated by our findings that it can be combined effectively with PLC pre-processing and that it can outperform its competitors in other settings such as on the traditional balanced transductive few-shot setting as well as with more unlabeled queries.

## References


[1] Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018. 2

[2] Malik Boudiaf, Imtiaz Ziko, Jérôme Rony, Jose Dolz, Pablo Piantanida, and Ismail Ben Ayed. Information maximization for few-shot learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, 2020. 1, 2, 3, 4, 6, 7, 8

[3] Da Chen, Yuefeng Chen, Yuhong Li, Feng Mao, Yuan He, and Hui Xue. Self-supervised learning for few-shot image classification. *arXiv preprint arXiv:1911.06045*, 2019. 5

[4] Mengting Chen, Yuxin Fang, Xinggang Wang, Heng Luo, Yifeng Geng, Xinyu Zhang, Chang Huang, Wenyu Liu, and Bo Wang. Diversity transfer network for few-shot learning. In *AAAI*, 2020. 2

[5] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *ICLR*, 2019. 2, 5

[6] Zitian Chen, Yanwei Fu, Yu-Xiong Wang, Lin Ma, Wei Liu, and Martial Hebert. Image deformation meta-networks for one-shot learning. In *CVPR*, 2019. 2

[7] Guneet Singh Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *ICLR*, 2020. 7

[8] Li Fei-Fei. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006. 2

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 1, 2

[10] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 2

[11] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016. 2

[12] Yuqing Hu, Vincent Gripon, and Stéphane Pateux. Leveraging the feature distribution in transfer-based few-shot learning. *arXiv preprint arXiv:2006.03806*, 2020. 1, 2, 3, 5, 7, 8

[13] Yuqing Hu, Stéphane Pateux, and Vincent Gripon. Adaptive dimension reduction and variational inference for transductive few-shot classification. In *International Conference on Artificial Intelligence and Statistics*, pages 5899–5917. PMLR, 2023. 3, 5

[14] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *CVPR*, 2019. 1, 3

[15] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML workshop*, 2015. 1, 2

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, 2009. 5

[17] Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Iterative label cleaning for transductive and semi-supervised few-shot learning. In *ICCV*, 2021. 1, 2, 3, 5, 6, 7, 8

[18] Michalis Lazarou, Tania Stathaki, and Yannis Avrithis. Tensor feature hallucination for few-shot learning. In *WACV*, January 2022. 1, 2

[19] Kai Li, Yulun Zhang, Kunpeng Li, and Yun Fu. Adversarial feature hallucination networks for few-shot learning. In *CVPR*, 2020. 1, 2

[20] Yann Lifchitz, Yannis Avrithis, and Sylvaine Picard. Local propagation for few-shot learning. In *ICPR*. IEEE, 2021. 2

[21] Yann Lifchitz, Yannis Avrithis, Sylvaine Picard, and Andrei Bursuc. Dense classification and implanting for few-shot learning. In *CVPR*, 2019. 1

[22] Jinlu Liu, Liang Song, and Yongqiang Qin. Prototype rectification for few-shot learning. In *ECCV*, 2019. 1, 2, 7

[23] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. In *CVPR*, 2019. 2

[24] Qinxuan Luo, Lingfeng Wang, Jingguo Lv, Shiming Xiang, and Chunhong Pan. Few-shot learning via feature hallucination with variational inference. In *WACV*, 2021. 1, 2

[25] Puneet Mangla, Nupur Kumari, Abhishek Sinha, Mayank Singh, Balaji Krishnamurthy, and Vineeth N Balasubramanian. Charting the right manifold: Manifold mixup for few-shot learning. In *WACV*, 2020. 1, 2

[26] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017. 2

[27] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *ICML*, 2017. 2

[28] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 2

[29] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[30] Guodong Qi, Huimin Yu, Zhaohui Lu, and Shuzhao Li. Transductive few-shot classification on the oblique manifold. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8392–8402, 2021. 1, 2

[31] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. In *NeurIPS*, 2019. 2

[32] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016. 2

[33] Pau Rodríguez, Issam Laradji, Alexandre Drouin, and Alexandre Lacoste. Embedding propagation: Smoother manifold for few-shot classification. *ECCV*, 2020. 1, 2, 3

[34] 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. 2

[35] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *ICML*, 2016. 2

[36] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 1, 2, 3

[37] 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 *CVPR*, 2018. 2

[38] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *arXiv preprint arXiv:2003.11539*, 2020. 1, 2

[39] Olivier Veilleux, Malik Boudiaf, Pablo Piantanida, and Ismail Ben Ayed. Realistic evaluation of transductive few-shot learning. *Advances in Neural Information Processing Systems*, 34:9290–9302, 2021. 1, 2, 3, 4, 5, 6, 7, 8, 11

[40] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NIPS*, 2016. 1, 2, 5

[41] Yikai Wang, C. Xu, Chen Liu, Liyong Zhang, and Yanwei Fu. Instance credibility inference for few-shot learning. *CVPR*, 2020. 1, 2, 7

[42] Zhanyuan Yang, Jinghua Wang, and Yingying Zhu. Few-shot classification with contrastive learning. In *ECCV*. Springer, 2022. 2

[43] Hongguang Zhang, Jing Zhang, and Piotr Koniusz. Few-shot learning via saliency-guided hallucination of samples. In *CVPR*, 2019. 2

[44] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. MetaGAN: An adversarial approach to few-shot learning. *NeurIPS*, 2018. 2

[45] Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *NIPS*, 2003. 4

[46] Hao Zhu and Piotr Koniusz. Ease: Unsupervised discriminant subspace learning for transductive few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9078–9088, 2022. 1, 5, 8

[47] Hao Zhu and Piotr Koniusz. Transductive few-shot learning with prototype-based label propagation by iterative graph refinement. In *CVPR*, 2023. 2, 5, 7, 8

[48] Imtiaz Ziko, Jose Dolz, Eric Granger, and Ismail Ben Ayed. Laplacian regularized few-shot learning. In *International conference on machine learning*, pages 11660–11670. PMLR, 2020. 7