Self-Attention Message Passing for Contrastive Few-Shot Learning

Shirekar, Ojas Kishorkumar; Singh, Anuj; Jamali-Rad, Hadi

DOI
10.1109/WACV56688.2023.00539

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)

2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) | 978-1-6654-9346-8/23/$31.00 ©2023 IEEE | DOI: 10.1109/WACV56688.2023.00539

Self-Attention Message Passing for Contrastive Few-Shot Learning

Ojas Kishorkumar Shirekar[1,2], Anuj Singh[1,2], Hadi Jamali-Rad[1,2]
[1]Delft University of Technology (TU Delft), The Netherlands
[2]Shell Global Solutions International B.V., Amsterdam, The Netherlands
{o.k.shirekar, a.r.singh}@student.tudelft.nl, h.jamalirad@tudelft.nl

Abstract

Humans have a unique ability to learn new representations from just a handful of examples with little to no supervision. Deep learning models, however, require an abundance of data and supervision to perform at a satisfactory level. Unsupervised few-shot learning (U-FSL) is the pursuit of bridging this gap between machines and humans. Inspired by the capacity of graph neural networks (GNNs) in discovering complex inter-sample relationships, we propose a novel self-attention based message passing contrastive learning approach (coined as `SAMP-CLR`*) for U-FSL pre-training. We also propose an optimal transport (OT) based fine-tuning strategy (we call* `OpT-Tune`*) to efficiently induce task awareness into our novel end-to-end unsupervised few-shot classification framework (*`SAMPTransfer`*). Our extensive experimental results corroborate the efficacy of* `SAMPTransfer` *in a variety of downstream few-shot classification scenarios, setting a new state-of-the-art for U-FSL on both miniImageNet and tieredImageNet benchmarks, offering up to* $7\%+$ *and* $5\%+$ *improvements, respectively. Our further investigations also confirm that* `SAMPTransfer` *remains on-par with some supervised baselines on miniImageNet and outperforms all existing U-FSL baselines in a challenging cross-domain scenario. Our code can be found in our GitHub repository:* `https://github.com/ojss/SAMPTransfer/`.

1. Introduction

Deep learning models have become increasingly large and data hungry to be able to guarantee acceptable downstream performance. Humans need neither a ton of data samples nor extensive forms of supervision to understand their surroundings and the semantics therein. Few-shot learning has garnered an upsurge of interest recently as it underscores this fundamental gap between humans' adaptive learning capacity compared to data-demanding deep learning methods. In this realm, few-shot classification is cast as the task of predicting class labels for a set of unlabeled data points (*query set*) given only a small set of labeled ones (*support set*). Typically, query and support data samples are drawn from the same distribution.

Few-shot classification methods usually consist of two sequential phases: (i) *pre-training* on a large dataset of *base* classes, regardless of this pre-training being supervised or unsupervised. This is followed by (ii) *fine-tuning* on an unseen dataset consisting of *novel* classes. Normally, the classes used in the pre-training and fine-tuning are mutually exclusive. In this paper, we focus on the self-supervised setting (also interchangeably called "unsupervised" in literature) where we have no access to the actual class labels of the "base" dataset. Our motivation to tackle unsupervised few-shot learning (U-FSL) is that it poses a more realistic challenge, closer to humans' learning process.

The body of work around U-FSL can be broadly classified into two different approaches. The first approach relies on the use of *meta-learning* and episodic pre-training that involves the creation of synthetic "tasks" to mimic the subsequent episodic fine-tuning phase [1, 16, 23–25, 29, 56]. The second approach follows a *transfer learning* strategy, where the network is trained non-episodically to learn optimal representations in the pre-training phase from the abundance of unlabeled data and is then followed by an episodic fine-tuning phase [14, 32, 39]. To be more specific, a feature extractor is first pre-trained to capture the structure of unlabeled data (present in base classes) using some form of representation learning [5, 6, 32, 39]. Next, a prediction layer (a linear layer, by convention) is fine-tuned in conjunction with the pre-trained feature extractor for a swift adaptation to the novel classes. The better the feature extractor models the distribution of the unlabeled data, the less the predictor requires training samples, and the faster it adapts itself to the unseen classes in the fine-tuning and eventual testing phases. Some recent studies [11, 32, 42] argue that transfer learning approaches outperform meta-learning counterparts in standard in-domain and cross-domain settings, where base and novel classes come from totally different distributions.

On the other side of the aisle, supervised FSL approaches that follow the episodic training paradigm may include a certain degree of *task awareness*. Such approaches exploit

2642-9381/23/$31.00 ©2023 IEEE
DOI 10.1109/WACV56688.2023.00539

the information available in the query set during the training and testing phases [9, 54, 57] to alleviate the model's sample bias. As a result, the model learns to generate task-specific embeddings by better aligning the features of the support and query samples for optimal metric based label assignment. Some other supervised approaches do not rely purely on convolutional feature extractors. Instead, they use graph neural networks (GNN) to model instance-level and class-level relationships [26, 37, 55, 58]. This is owing to the fact that GNN's are capable of exploiting the manifold structure of the novel classes [52]. However, looking at the recent literature, one can barely see any GNN based architectures being used in the unsupervised setting.

Recent unsupervised methods use a successful form of *contrastive learning* [6] in their self-supervised pre-training phase. Contrastive learning methods typically treat each image in a batch as its own class. The only other images that share the class are the augmentations of the image in question. Such methods enforce similarity of representations between pairs of an image and its augmentations (positive pairs), while enforcing dissimilarity between all other pairs of images (negative pairs) through a *contrastive loss*. Although these methods work well, they overlook the possibility that within a randomly sampled batch of images there could be several images (apart from their augmentations) that in reality belong to the same class. By applying the contrastive loss, the network may inadvertently learn different representations for such images and classes.

To address this problem, recent methods such as SimCLR [6] introduce larger batch sizes in the pre-training phase to maximize the number of negative samples. However, this approach faces two shortcomings: (i) increasingly larger batch sizes, mandate more costly training infrastructure, and (ii) it still does not ingrain intra-class dependencies into the network. Point (ii) still applies to even more recent approaches, such as ProtoCLR [32]. A simple yet effective remedy of this problem proposed in C^3LR [39] where an intermediate clustering and re-ranking step is introduced, and the contrastive loss is accordingly adjusted to ingest a semblance of class-cognizance. However, the problem could be approached from a different perspective, where the network explores the structure of data samples per batch.

We propose a novel U-FSL approach (coined as `SAMPTransfer`) that marries the potential of GNNs in learning the global structure of data in the pre-training stage, and the efficiency of optimal transport (OT) for inducing task-awareness in the following fine-tuning phase. More concretely, with `SAMPTransfer` we introduce a novel self-attention message passing contrastive learning (`SAMP-CLR`) scheme that uses a form of *graph attention* allowing the network to learn refined representations by looking beyond single-image instances per batch. Furthermore, the proposed OT based fine-tuning strategy (we call `OpT-Tune`) aligns the distributions of the support and query samples to improve downstream adaptability of the pre-trained encoder, without requiring any additional parameters. Our contributions can be summarized as:

1. We propose `SAMPTransfer`, a novel U-FSL approach that introduces a self-attention message passing contrastive learning (`SAMP-CLR`) paradigm for unsupervised few-shot pre-training.

2. We propose applying an optimal transport (OT) based fine-tuning (`OpT-Tune`) strategy to efficiently induce task-awareness in both fine-tuning and inference stages.

3. We present a theoretical foundation for `SAMPTransfer`, as well as extensive experimental results corroborating the efficacy of `SAMPTransfer`, and setting a new state-of-the-art (to our best knowledge) in both *mini*ImageNet and *tiered*ImageNet benchmarks, we also report competitive performance on the challenging CDFSL benchmark [20].

2. Related Work

Self-Supervised learning. Self-supervised learning (SSL) is a term used for a collection of unsupervised methods that obtain supervisory signals from within the data itself, typically by leveraging the underlying structure in the data. The general technique of self-supervised learning is to predict any unobserved (or property) of the input from any observed part. Several recent advances in the SSL space have made waves by eclipsing their fully supervised counterparts [18]. Some examples of seminal works include SimCLR [6], BYOL [19], SWaV [5], MoCo [21], and SimSiam [7]. Our pre-training method `SAMP-CLR` is inspired by SimCLR [6], ProtoTransfer [32] and C^3LR [39].

Metric learning. Metric learning aims to learn a representation function that maps the data to an embedding space. The distance between objects in the embedding space must preserve their similarity (or dissimilarity) - similar objects are closer, while dissimilar objects are farther. For example, unsupervised methods based on some form of contrastive loss, such as SimCLR [6] or NNCLR [15], guide objects belonging to the same potential class to be mapped to the same point and those from different classes to be mapped to different points. Note that in an unsupervised setting, each image in a batch is its own class. This process generally involves taking two crops of the same image and encouraging the network to emit an identical representation for the two, while ensuring that the representations remain different from all other images in a given batch. Metric learning methods have been shown to work quite well for few-shot learning. AAL-ProtoNets [1], ProtoTransfer [32], UMTRA [25], and certain GNN methods [37] are excellent examples that use metric learning for few-shot learning.

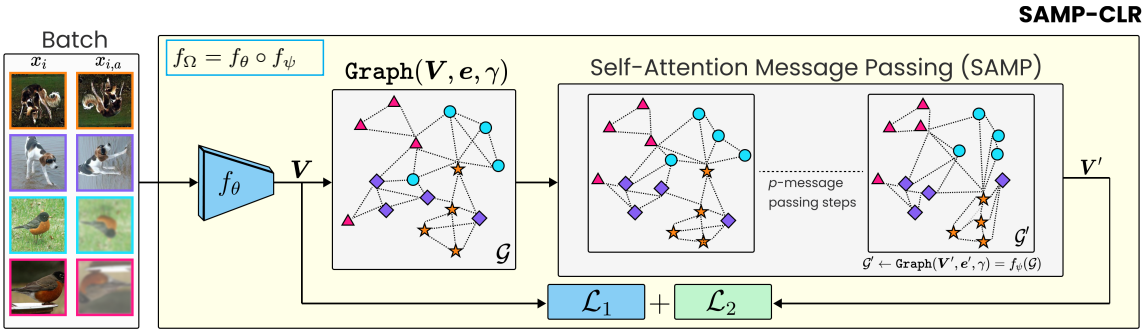

Figure 1: SAMP-CLR schematic view and pre-training procedure. In the figure, $x_{i,a}$ is an image sampled from the augmented set $\mathcal{A}$. The p-message passing steps refine the features extracted using a CNN encoder.

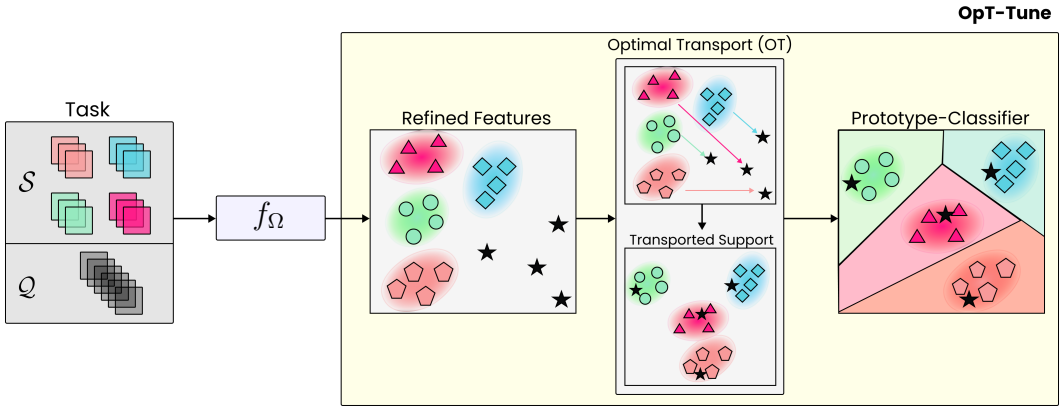

Figure 2: Features extracted from the pre-trained CNN are used to build a graph. The features are first refined using the pre-trained SAMP layer(s). Then OpT-Tune aligns support features with query features.

Graph Neural Networks for FSL. Since the first use of graphs for FSL in [37], there have been several advancements and continued interest in using graphs for supervised FSL. In [37], each node corresponds to one instance (labeled or unlabeled) and is represented as the concatenation of a feature embedding and a label embedding. The final layer of their model is a linear classifier layer that directly outputs the prediction scores for each unlabeled node. There has also been an increase in methods that use transduction. TPN [31] is one of those methods that uses graphs to propagate labels [52] from labeled samples to unlabeled samples. Although methods such as EGNN [26] make use of edge and node features, earlier methods focused only on using node features. Graphs are attractive, as they can model intra-batch relations and can be extended for transduction, as used in [26, 31]. In addition to transduction and relation modeling, graphs are highly potent as task adaptation modules. HGNN [58] is an example in which a graph is used to refine and adapt feature embeddings. It must be noted that most graph-based methods have been applied in the supervised FSL setting. To the best of our knowledge, we are the first to use it in any form for U-FSL. More specifically, we use a message passing network as a part of our network architecture and pre-training scheme.

3. Proposed Method (SAMPTransfer)

In this section, we first describe our problem formulation. We then discuss the two subsequent phases of the proposed approach: (i) self-supervised pre-training (SAMP-CLR), and (ii) the optimal transport based episodic supervised fine-tuning (OpT-Tune). Together, these two phases constitute our overall approach, which we have coined as SAMPTransfer. The mechanics of the proposed pre-training and fine-tuning procedures are illustrated in Figs. 1 and 2, respectively.

3.1. Preliminaries

Let us denote the training data of size D as $\mathcal{D}_{\text{tr}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{D}$ with $(\boldsymbol{x}_i, y_i)$ representing an image and its class label, respectively. In the pre-training phase, we sample L random images from $\mathcal{D}_{\text{tr}}$ and augment each sample A times by randomly sampling augmentation functions $\zeta_a(.), \forall a \in [A]$ from the set $\mathcal{A}$. This results in a mini-batch of size $B = (A+1)L$ total samples. Note that in the unsupervised setting, we have no access to the data labels in the pre-training phase. Next, we fine-tune our model episodically [47] on a set of randomly sampled tasks $\mathcal{T}_i$ drawn from the test dataset $\mathcal{D}_{\text{tst}} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{D'}$ of size D'. A task, $\mathcal{T}_i$,

is comprised of two parts: (i) the support set $\mathcal{S}$ from which the model learns, (ii) the query set $\mathcal{Q}$ on which the model is evaluated. The support set $\mathcal{S} = \{\boldsymbol{x}_i^s, y_i^s\}_{i=1}^{NK}$ is constructed by drawing K labeled random samples from N different classes, resulting in the so-called (N-way, K-shot) settings. The query set $\mathcal{Q} = \{\boldsymbol{x}_j^q\}_{j=1}^{NQ}$ then contains NQ unlabeled samples. By convention, we denote $\mathcal{T}_i = \mathcal{S}_i \cup \mathcal{Q}_i$ by (N, K).

3.2. Self-Attention Message Passing (SAMP)

Our network architecture consists of a convolutional (CNN) feature extractor f_θ and a message passing network based on self-attention, f_ψ. The CNN feature extractor f_θ, parameterized by θ, is used to extract features $\boldsymbol{V} = f_\theta(\boldsymbol{X})$, where $\boldsymbol{V} \in \mathbb{R}^{B\times d}$ is the set of B features each of size d and $\boldsymbol{X} \in \mathbb{R}^{B\times C\times H\times W}$ is a batch of B images of size $C \times H \times W$. To help refine the features and use batch-level relationships, we create a graph $\mathcal{G} = \texttt{Graph}(\boldsymbol{V}, \boldsymbol{e}, \gamma)$ where $\boldsymbol{V}$ is treated as a set of initial node features, $\boldsymbol{e}$ is the pairwise distance between all nodes based on a given distance metric and γ is a threshold on the values in $\boldsymbol{e}$ that determines whether two nodes will be connected or not. Note that $|\mathcal{G}| = B$, as we build the graph over the B samples in our batch. We use a **s**elf-**a**ttention **m**essage **p**assing neural network (we call SAMP) f_ψ, parameterized by ψ, to refine the initial feature vectors by exchanging and amalgamating information between all pairs of connected nodes. From now on, as can be seen in Figs. 1 and 2, we refer to the combination of the feature extractor f_θ and the SAMP module f_ψ as $f_\Omega = f_\psi \circ f_\theta$ where $\Omega = \{\theta, \psi\}$ is the collection of all parameters. The SAMP layers, f_ψ operate on the graph $\mathcal{G}$.

To allow an effective exchange of information to refine initial node features $\boldsymbol{V}$, we make use of graph attention in a slightly different manner than the standard graph attention defined in [46]. The graph attention in [46] uses a single weight matrix $\boldsymbol{W}$ that acts as a shared linear transformation for all nodes. Instead, we choose to use scaled dot-product self-attention as defined in [38, 45]. The major benefit of this design choice is that it enhances the network with more expressivity, as shown in [4, 27]. Notably, the use of three separate representations (query, key, and value) instead of just a single weight matrix to linearly transform the data is key to modeling relationships between data points.

We apply p successive message passing steps similar to [38, 46]. In each step, we pass messages between the connected nodes of $\mathcal{G}$ and obtain updated features in $\boldsymbol{V}^{p+1}$, at step $p+1$. Here, the i-th row of $\boldsymbol{V}^{p+1}$ is given by $\boldsymbol{V}_i^{p+1} = \sum_{j\in\mathcal{N}_i} \lambda_{ij}^p \boldsymbol{W}^p \boldsymbol{V}_j^p$, where λ_{ij} is the attention score between the nodes i and j, $\boldsymbol{W}^p \in \mathbb{R}^{d\times d}$ is the message passing weight matrix at step p, and $\mathcal{N}_i$ denotes the set of neighboring nodes of node i. In this way, $\lambda_{i,j}$ allows our update mechanism to flexibly weight every sample w.r.t every other sample in the batch. We employ scaled dot-product self-attention to compute attention scores, leading to: $\lambda_{ij}^p = \mathrm{softmax}(\boldsymbol{W}_q^p \boldsymbol{V}_i^p (\boldsymbol{W}_k^p \boldsymbol{V}_j^p)^T / \sqrt{d})$ where $\boldsymbol{W}_k^p$ and $\boldsymbol{W}_q^p$, both $\in \mathbb{R}^{d\times d}$, are the weight matrices corresponding to the sending and receiving nodes, respectively. To allow the message-passing neural network to learn a diverse set of attention scores, we apply H scaled dot-product self-attention heads in every message-passing step and concatenate their results. To this end, instead of using single weight matrices $\boldsymbol{W}_q^p, \boldsymbol{W}_k^p$ and $\boldsymbol{W}^p$, we use $\boldsymbol{W}_q^{p,h}, \boldsymbol{W}_k^{p,h}$ and $\boldsymbol{W}^{p,h}$ all $\in \mathbb{R}^{d/H\times d}$ for each attention head, resulting in:

$$\boldsymbol{V}_i^{p+1} = \left[\sum_{j\in\mathcal{N}_i} \lambda_{ij}^{p,1} \boldsymbol{W}^{p,1} \boldsymbol{V}_j^p, \ldots, \sum_{j\in\mathcal{N}_i} \lambda_{ij}^{p,H} \boldsymbol{W}^{p,H} \boldsymbol{V}_j^p\right],$$

note that $\boldsymbol{V}_i^{p+1}$ still has the same dimension $\mathbb{R}^d$.

3.3. Self-Supervised Pre-Training (SAMP-CLR)

The fact that we do not have access to the true class labels of the training data underscores the need to use a self-supervised pre-training scheme. As briefly discussed in Section 1, we build on the idea of employing contrastive prototypical transfer learning with some inspiration from [6, 32, 39]. Standard contrastive learning enforces embeddings of augmented images to be close to the embeddings of their source images in the representation space. The *key idea* of SAMP-CLR is not only to perform contrastive learning (the "CLR" component) on the source and augmented image embeddings, but also to ensure that images in the mini-batch belonging to potentially the same class have similar embeddings. This is where the "SAMP" module comes to rescue, enabling the model to look beyond single instances and their augmentations. SAMP allows the model to extract richer semantic information across multiple images present in a mini-batch. Concretely speaking, we apply a contrastive loss on the SAMP refined features (generated by f_ψ), and on the standard convolutional features (generated by f_θ). Let us walk you through the process in more detail.

Algorithm 1 begins with batch generation: each mini-batch consists of L random samples $\{\boldsymbol{x}_i\}_{i=1}^L$ from $\mathcal{D}_{\text{tr}}$, where $\boldsymbol{x}_i$ is treated as a 1-shot support sample for which we create A randomly augmented versions $\tilde{\boldsymbol{x}}_{i,a}$ as query samples (lines 2 to 3), leading to a batch size of $B = (A+1)L$. Then the embeddings $\boldsymbol{Z} \in \mathbb{R}^{L\times d}$ and $\bar{\boldsymbol{Z}} \in \mathbb{R}^{LA\times d}$ are generated (line 4) by passing the source images and augmented images through a feature extraction network f_θ, respectively. We then construct $\mathcal{G} = \texttt{Graph}(\boldsymbol{V}, \boldsymbol{e}, \gamma)$ with $\boldsymbol{V} = [\boldsymbol{Z}^\top, \bar{\boldsymbol{Z}}^\top]^\top$ of size $B \times d$ concatenating source and augmented image embeddings $\boldsymbol{Z}$ and $\bar{\boldsymbol{Z}}$ (line 5-6), $\boldsymbol{e}$ is the vector of centered shift/scale-invariant cosine similarities $d'[.]$ (line 5) [44], and γ is defined earlier. The graph $\mathcal{G}$ is then passed through the SAMP layer(s) f_ψ resulting in a updated graph $\mathcal{G}'$ with refined node features $\boldsymbol{V}'$ (line 7). $\boldsymbol{V}'$ is spliced into the updated source and augmented image

Algorithm 1: SAMP-CLR

Require: $\mathcal{A}$, f_θ, f_ψ, Ω, α, β, η, $d[.]$, $d'[.]$

1 **while** *not done* **do**

2 Sample minibatch $\{\boldsymbol{x}_i\}_{i=1}^{L} \in \mathcal{D}_{\text{tr}}$

3 Augment samples: $\bar{\boldsymbol{x}}_{i,a} = \zeta_a(\boldsymbol{x}_i)$; $\zeta_a \sim \mathcal{A}$.

4 $\boldsymbol{Z}, \bar{\boldsymbol{Z}} \leftarrow f_\theta\left(\{\boldsymbol{x}_i\}_{i=1}^{L}\right), f_\theta\left(\{\bar{\boldsymbol{x}}_{i,\,a}\}_{i=1,a=1}^{L,A}\right)$

5 $\boldsymbol{V} = [\boldsymbol{Z}^\top, \bar{\boldsymbol{Z}}^\top]^\top, \boldsymbol{e} = \{d'[\boldsymbol{V}_i, \boldsymbol{V}_j], \forall i,j \in [B]\}$

6 $\mathcal{G} \leftarrow \texttt{Graph}(\boldsymbol{V}, \boldsymbol{e}, \gamma)$

7 $\mathcal{G}' \leftarrow \texttt{Graph}(\boldsymbol{V}', \boldsymbol{e}', \gamma) = f_\psi(\mathcal{G})$

8 $\boldsymbol{Z}', \bar{\boldsymbol{Z}}' \leftarrow \boldsymbol{V}'_{1:L}, \boldsymbol{V}'_{L+1:B}$

9 $\ell(i,a) = -\log \frac{\exp\left(-d\left[\bar{\boldsymbol{Z}}_{(a-1)L+i}, \boldsymbol{Z}_i\right]\right)}{\sum_{k=1}^{L}\exp\left(-d\left[\bar{\boldsymbol{Z}}_{(a-1)L+i}, \boldsymbol{Z}_k\right]\right)}$

10 $r(i,a) = -\log \frac{\exp\left(-d\left[\bar{\boldsymbol{Z}}'_{(a-1)L+i}, \boldsymbol{Z}'_i\right]\right)}{\sum_{k=1}^{L}\exp\left(-d\left[\bar{\boldsymbol{Z}}'_{(a-1)L+i}, \boldsymbol{Z}'_k\right]\right)}$

11 $\mathcal{L}_1 = {}^1\!/_{LA}\sum_{i=1}^{L}\sum_{a=1}^{A}\ell(i,a)$

12 $\mathcal{L}_2 = {}^1\!/_{LA}\sum_{i=1}^{L}\sum_{a=1}^{A} r(i,a)$

13 $\mathcal{L} = \beta\mathcal{L}_1 + \mathcal{L}_2$

14 $\Omega \leftarrow \Omega - \eta\nabla_\Omega\mathcal{L}$

15 **end**

embeddings ($\boldsymbol{Z}'$ and $\bar{\boldsymbol{Z}}'$), respectively (lines 8). In lines 9 to 12, we then apply contrastive losses $\mathcal{L}_1$ (between $\boldsymbol{Z}$ and $\bar{\boldsymbol{Z}}$) and $\mathcal{L}_2$ (between $\boldsymbol{Z}'$ and $\bar{\boldsymbol{Z}}'$). Here, $\mathcal{L}_1$ encourages the feature extractor to cluster the embeddings of augmented query samples $\bar{\boldsymbol{Z}}$ around their prototypes (namely, source embeddings) $\boldsymbol{Z}$, which in turn provides a good initial set of embeddings for the SAMP projector module to refine. $\mathcal{L}_2$ enforces the same constraints as $\mathcal{L}_1$ but for embeddings generated by the SAMP layer. Both loss terms use a Euclidean distance metric in the embedding space, denoted by $d[.]$. Finally, the overall loss is given by $\mathcal{L} = \beta\mathcal{L}_1 + \mathcal{L}_2$, which is optimized with mini-batch stochastic gradient descent w.r.t all the parameters in $\Omega = \{\theta, \psi\}$ where β is a scaling factor, and η the learning rate.

3.4. Supervised Fine-tuning (OpT-Tune)

We propose a two-stage supervised fine-tuning consisting of (i) a transportation stage followed by (ii) a prototypical fine-tuning and classification stage. The transportation stage involves using optimal transport (OT) [10, 34]. As sketched in Fig. 2, OT helps projecting embeddings of the support set, $\boldsymbol{Z}^s = f_\Omega(\{\boldsymbol{x}_i^s\}_{i=1}^{NK}) \in \mathbb{R}^{NK\times d}$, so that they overlap better with the query set embeddings, $\boldsymbol{Z}^q = f_\Omega(\{\boldsymbol{x}_j^q\}_{j=1}^{NQ}) \in \mathbb{R}^{NQ\times d}$ upon transportation. This increases the spread of $\boldsymbol{Z}^s$ in the query set's domain, which in turn creates more representative prototypes for each of the N classes in $\mathcal{S}$. We show in Section 6 that this results in a significant boost in the downstream classification performance.

Algorithm 2: OpT-Tune

Require: $d[\cdot]$, $\boldsymbol{Z}^s$, $\boldsymbol{Z}^q$

$\boldsymbol{M}_{i,j} = d[\boldsymbol{Z}_i^s, \boldsymbol{Z}_j^q], \forall i \in [NK], j \in [NQ]$

$\boldsymbol{\pi}^\star \leftarrow$ Solving Eq. (1) using Sinkhorn-Knopp [10]

$\hat{\boldsymbol{\pi}}^\star_{i,j} \leftarrow \boldsymbol{\pi}^\star_{i,j} \,/\, \sum_j \boldsymbol{\pi}^\star_{i,j}, \forall i \in [NK], j \in [NQ]$

Solve Eq. (2)

Return: $\hat{\boldsymbol{Z}}^s$

OT based feature alignment. We provide a basic intuition for OT in the context of SAMPTransfer. Let $\boldsymbol{r} \in \mathbb{R}^{NK}$ and $\boldsymbol{c} \in \mathbb{R}^{NQ}$ be two probability simplexes defined over $\boldsymbol{Z}_i^s, \forall i \in [NK]$ and $\boldsymbol{Z}_j^q, \forall j \in [NQ]$, respectively. $\boldsymbol{r}$ denotes the distribution of the support embeddings, whereas $\boldsymbol{c}$ denotes the distribution of the query embeddings. Consider $\boldsymbol{\Pi}(\boldsymbol{r}, \boldsymbol{c})$ to be a set of $NK \times NQ$ doubly stochastic matrices where all rows sum up to $\boldsymbol{r}$ and all columns sum up to $\boldsymbol{c}$ as:

$$\boldsymbol{\Pi}(\boldsymbol{r}, \boldsymbol{c}) = \left\{ \boldsymbol{\pi} \in \mathbb{R}_+^{NK\times NQ} \mid \boldsymbol{\pi}\mathbf{1}_{NQ} = \boldsymbol{r}, \boldsymbol{\pi}^\top\mathbf{1}_{NK} = \boldsymbol{c} \right\}.$$

Intuitively, $\boldsymbol{\Pi}(\boldsymbol{r}, \boldsymbol{c})$ is a collection of all transport "plans", where a transport plan is defined as a potential strategy specifying how much of each support embedding is allocated to every query embedding and vice-versa. Our goal here is to find the most optimal transport plan, out of all possible transport plans $\boldsymbol{\Pi}(\boldsymbol{r}, \boldsymbol{c})$, that allocates NK support embeddings to NQ query embeddings with maximum overlap between their distributions.

Given a cost matrix $\boldsymbol{M}$, the cost of mapping $\boldsymbol{Z}^s$ to $\boldsymbol{Z}^q$ using a transport plan $\boldsymbol{\pi}$ can be quantified as $\langle \boldsymbol{\pi}, \boldsymbol{M} \rangle_F$ and the OT problem can then be stated as,

$$\boldsymbol{\pi}^\star = \underset{\boldsymbol{\pi}\in\boldsymbol{\Pi}(\boldsymbol{r},\boldsymbol{c})}{\operatorname{argmin}} \langle \boldsymbol{\pi}, \boldsymbol{M} \rangle_F - \varepsilon\mathbb{H}(\boldsymbol{\pi}), \tag{1}$$

where $\boldsymbol{\pi}^\star$ denotes the most optimal transportation plan, $\langle \cdot, \cdot \rangle_F$ is the Frobenius dot product, and ε is the weight on the entropic regularizer $\mathbb{H}$. The cost matrix $\boldsymbol{M}$ quantifies the overlap between the two distributions by measuring the distance between each support and query embedding pair: $\boldsymbol{M}_{i,j} = d[\boldsymbol{Z}_i^s, \boldsymbol{Z}_j^q]$. The entropic regularization promotes "smoother" transportation plans [10]. Equation (1) is then solved using the time-efficient Sinkhorn-Knopp algorithm [10, 40]. Notice that $\boldsymbol{\pi}^\star$ is also referred to as *Wasserstein metric* [10, 34]. To adapt $\boldsymbol{Z}^s$ to $\boldsymbol{Z}^q$ with cost matrix $\boldsymbol{M}$, we compute $\hat{\boldsymbol{Z}}^s$ as the *projected mapping* of $\boldsymbol{Z}^s$, given by:

$$\begin{aligned} \hat{\boldsymbol{Z}}^s &= \hat{\boldsymbol{\pi}}^\star \boldsymbol{Z}^q, \\ \hat{\boldsymbol{\pi}}^\star_{i,j} &= \frac{\boldsymbol{\pi}^\star_{i,j}}{\sum_j \boldsymbol{\pi}^\star_{i,j}}, \forall i \in [NK], j \in [NQ], \end{aligned} \tag{2}$$

where $\widehat{\boldsymbol{\pi}}^\star$ is the normalized transport. The *projected support* embeddings $\hat{\boldsymbol{Z}}^s$ are an estimation of $\boldsymbol{Z}^s$ in the region occupied by the query embeddings $\boldsymbol{Z}^q$. Specifically, it is a

barycentric mapping of the support features $\boldsymbol{Z}^s$. Algorithm 2 shows this process in a succinct manner.

Prototypical classification. The projected support embeddings, $\hat{\boldsymbol{Z}}^s$, are used for prototype creation and classification of the query points. To this end, following [32, 43] we concatenate f_Ω with a single layer nearest mean classifier f_ϕ (resulting in an architecture similar to ProtoNet [41]) and only fine-tune this last layer. In this stage, for each class $k \in \mathcal{C}$ in the support set, we compute the class prototype $\boldsymbol{c}_k$ for class k using the projected support embeddings $\hat{\boldsymbol{Z}}^{s,k}$ belonging to class k:

$$\boldsymbol{c}_k = \frac{1}{\left|\hat{\boldsymbol{Z}}^{s,k}\right|} \sum_{\hat{\boldsymbol{z}} \in \hat{\boldsymbol{Z}}^{s,k}} \hat{\boldsymbol{z}}, \text{ for } k \in \mathcal{C}.$$

Following [32, 43], we initialize the classification layer f_ϕ with weights set to $\boldsymbol{W}_k = 2\boldsymbol{c}_k$ and biases set to $b_k = -\|\boldsymbol{c}_k\|^2$. To finetune this layer, we sample a subset of supports from $\mathcal{S}$ and train f_ϕ with a standard cross-entropy loss; more details are given in Section 4.

4. Experimental Setup

Datasets. To benchmark the performance of our method `SAMPTransfer`, we conduct "in-domain" experimentation on two most commonly adopted few-shot learning datasets: *mini*ImageNet [47] and *tiered*ImageNet [36]. *Mini*ImageNet contains 100 classes with 600 samples in each class. This equals a total of $60,000$ images that we resize to 84×84 pixels. Out of the 100 classes, we use 64 classes for training, 16 for validation, and 20 for testing. *Tiered*ImageNet is a larger subset of ILSVRC-12 [13] with 608 classes with a total of $779,165$ images of size 84×84. We use 351 for training, 97 for validation, and 8 for testing, out of the 608 classes. The augmentation strategy follows the one proposed in [2]. We also compare our method on a recent more challenging "cross-domain" few-shot learning (CDFSL) benchmark [20], which consists of several datasets. This benchmark has four datasets with increasing similarities to *mini*ImageNet. In that order, we have grayscale chest X-ray images from ChestX [50], dermatological skin lesion images from ISIC2018 [8], aerial satellite images from EuroSAT [22], and crop disease images from CropDiseases [33]. We also used the Caltech-UCSD Birds (CUB) dataset [48] for further analysis of cross-domain performance. The CUB dataset is made up of $11,788$ images from 200 unique species of birds. We use 100 classes for training, 50 for both validation and testing.

Training strategy. In Fig. 1, as feature extractor, we use the standard `Conv4` model following [25, 32, 47]. It is followed by a single `SAMP` layer with 4 attention heads. Note that we also use a slightly modified version of the `Conv4` network which we call `Conv4b`, where we increase the number of filters from $(64, 64, 64, 64)$ to $(96, 128, 256, 512)$ [17] and average pool the final feature map returning a smaller embedding dimension $d = 512$ instead of $d = 1600$. The networks are pre-trained using `SAMP-CLR` on the respective training splits of the datasets, with an initial learning rate of $\eta = 0.0005$, annealed by a cosine scheduler via the Adam optimizer [28] and $L = 128$. Experiments involving CDFSL benchmark follow [20, 32, 39], where we pre-train a ResNet-10 encoder using `SAMP-CLR` on *mini*ImageNet images of size 224×224 for 400 epochs with the Adam optimizer and a constant learning rate of $\eta = 0.0001$. Similar to the `Conv4` encoder, the ResNet-10 uses the same `SAMP` configuration. During validation and testing, as defined in Section 3.4, we initialize and fine-tune f_ϕ for 15 iterations where we sample a subset of examples from $\mathcal{S}$ in each iteration. For validation, we create 15 (N-way, K-shot) tasks using the validation split of the respective dataset.

Evaluation scenarios and baseline. Our testing scheme uses 600 test episodes, each with 15 query shots per class, on which the pre-trained encoder (`SAMP-CLR`) is fine-tuned using `OpT-Tune` and tested. All our results indicate 95% confidence intervals over 3 runs, each with 600 test episodes. Therefore, the standard deviation values are calculated according to the 3 runs to provide more concrete measures for comparison. For our in-domain benchmarks, we test on (5-way, 1-shot) and (5-way, 5-shot) classification tasks, while our cross-domain testing is performed using (5-way, 5-shot) and (5-way, 20-shot) classification tasks following [20]. We compare our performance with a suite of recent unsupervised few-shot baselines such as U-MlSo [60], C^{3}LR [39], Meta-GMVAE [29], and Revisiting UML [56] to name a few. Furthermore, we also compare with a set of supervised approaches (such as MetaQDA [61] and TransductiveCNAPS [3]), the best of which are expected to outperform ours and other unsupervised methods.

5. Performance Evaluation

In-domain experiments. Table 1 summarizes our performance evaluation results on the *mini*ImageNet dataset for (N-way, K-shot) scenarios with $N = 5$ and $K = 1, 5$. The top section compares the performance of the proposed approach (`SAMPTransfer`) with the most recent unsupervised competitors. We outperform our closest competitors by approximately 7%+ and 2%+ in the (5-way, 1-shot) and (5-way, 5-shot) settings, respectively. More interestingly, our method matches or outperforms some of the supervised baselines (bottom section of the table), especially SimpleCNAPS which uses a much more powerful ResNet-18 backbone. Obviously, the state-of-the-art supervised few-shot learning approaches have the advantage of having access to the true labels. When it comes to *tiered*ImageNet, our approach shows considerable gains over recent competitors such as C^{3}LR [39] with a 3%+ improvement in the (5-way, 1-shot) setting and a 5%+ improvement in the (5-way,5 shot) setting. As such, `SAMPTransfer` sets a new state-of-the-art for both

Table 1: Accuracy (%± std.) for (N-way, K-shot) classification tasks. Style: **best** and second best.

Method(N, K)	**Backbone**	***mini*ImageNet** (5,1)	***mini*ImageNet** (5,5)
CACTUs-MAML [23]	Conv4	39.90 ± 0.74	53.97 ± 0.70
CACTUs-Proto [23]	Conv4	39.18 ± 0.71	53.36 ± 0.70
UMTRA [25]	Conv4	39.93	50.73
AAL-ProtoNet [1]	Conv4	37.67 ± 0.39	40.29 ± 0.68
AAL-MAML++ [1]	Conv4	34.57 ± 0.74	49.18± 0.47
UFLST [24]	Conv4	33.77 ± 0.70	45.03 ± 0.73
ULDA-ProtoNet [35]	Conv4	40.63 ± 0.61	55.41 ± 0.57
ULDA-MetaNet [35]	Conv4	40.71 ± 0.62	54.49 ± 0.58
U-SoSN+ArL [59]	Conv4	41.13 ± 0.84	55.39 ± 0.79
U-MlSo [60]	Conv4	41.09	55.38
ProtoTransfer [32]	Conv4	45.67 ± 0.79	62.99 ± 0.75
CUMCA [53]	Conv4	41.12	54.55
Meta-GMVAE [29]	Conv4	42.82	55.73
Revisiting UML [56]	Conv4	48.12 ± 0.19	65.33 ± 0.17
CSSL-FSL_Mini64 [30]	Conv4	48.53 ± 1.26	63.13 ± 0.87
C^3LR [39]	Conv4	47.92 ± 1.2	64.81 ± 1.15
`SAMPTransfer` (**ours**)	Conv4	55.75 ± 0.77	68.33 ± 0.66
`SAMPTransfer`* (**ours**)	Conv4b	**61.02** ± 1.0	**72.52** ± 0.68
Supervised Methods			
MAML [16]	Conv4	46.81 ± 0.77	62.13± 0.72
ProtoNet [41]	Conv4	46.44± 0.78	66.33± 0.68
MMC [36]	Conv4	50.41 ± 0.31	64.39 ± 0.24
FEAT [57]	Conv4	55.15	71.61
SimpleShot [51]	Conv4	49.69 ± 0.19	66.92 ± 0.17
Simple CNAPS [3]	ResNet-18	53.2 ± 0.9	70.8 ± 0.7
Transductive CNAPS [3]	ResNet-18	55.6 ± 0.9	73.1 ± 0.7
MetaQDA [61]	Conv4	56.41 ± 0.80	72.64 ± 0.62
Pre+Linear [32]	Conv4	43.87 ± 0.69	63.01 ± 0.71

Table 2: Accuracy (%± std.) for (N-way, K-shot) classification tasks. Style: **best** and second best.

Method(N, K)	**Backbone**	***tiered*ImageNet** (5,1)	***tiered*ImageNet** (5,5)
C^3LR [39]	Conv4	42.37 ± 0.77	61.77 ± 0.25
ULDA-ProtoNet [35]	Conv4	41.60 ±0.64	56.28 ± 0.62
ULDA-MetaOptNet [35]	Conv4	41.77 ± 0.65	56.78 ± 0.63
U-SoSN+ArL [59]	Conv4	43.68 ± 0.91	58.56 ± 0.74
U-MlSo [60]	Conv4	43.01 ± 0.91	57.53 ± 0.74
`SAMPTransfer` (**ours**)	Conv4	45.25 ± 0.89	59.75 ± 0.66
`SAMPTransfer`* (**ours**)	Conv4b	**49.10** ± 0.94	**65.19** ± 0.82

*tiered*ImageNet and *mini*ImageNet datasets.

Cross-domain experiments. We focus on the recent CDFSL benchmark [20] to investigate the performance of `SAMPTransfer` in cross-domain scenarios. This outcome is summarize in Table 3. Here, we pre-train on *mini*ImageNet and fine-tune on ChestX [50], ISIC2018 [8], EuroSAT [22], and CropDiseases [33]. We compare the performance against C^3LR[39], ProtoTransfer [32] along with its two variants using UMTRA [25] (also proposed in [32]), as well as ConFeSS [11] and ATA [49] - two of the latest methods *dedicated* to solving the cross-domain few-shot learning problem. Note that we also compare with a couple of related supervised approaches from [20], as a reference. Our method consistently keeps up with ConFeSS [11], but scores higher in 5 and 20 shot CropDiseases tasks by 2%+ and about 1%, respectively. Except for EuroSAT, our method is consistently competitive ($\sim$ 1% difference) to the performance of ConFeSS in ChestX and ISIC. In ISIC, which is the second least similar dataset to *mini*ImageNet, our method is better by 1%+ in the (5-way, 20-shot) setting. Note that `SAMPTransfer` outperforms another recent dedicated method ATA [49] in all but one CDFSL benchmark settings, with the exception being the EuroSAT (5-way, 5-shot) setting.

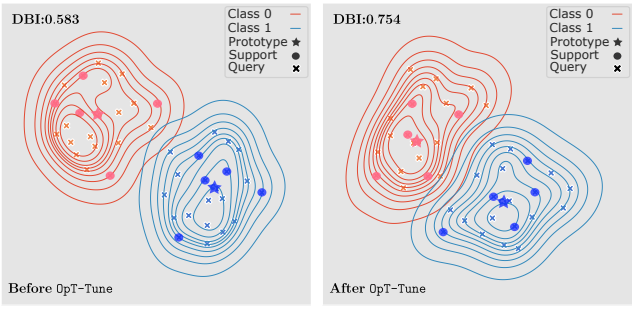

Figure 3: Before (left) and after applying OT (right). Prototypes (★), supports (●) and queries (✖). OT helps better align the distribution of support and query samples.

6. Ablation Study and Robustness Analysis

Table 4 investigates the performance of the proposed method against various choices of important hyperparameters. We use the (5-way, 5-shot) *mini*ImageNet benchmark to analyze the robustness of our method and demonstrate the importance of our design choices.

`OpT-Tune` is crucial. To illustrate the effect of using `OpT-Tune` on the classification performance, we perform experiments with `OpT-Tune` disabled. For a fair comparison, we use the same pre-trained models in the test runs with `OpT-Tune` enabled or disabled. The best performing model (a `Conv4b`) uses 1 `SAMP` layer with 4 attention heads and a batch size of 128, resulting in accuracy of 72.52% with `OpT-Tune` enabled. The same model, with `OpT-Tune` disabled, loses 9% accuracy. Even with `OpT-Tune` disabled, our method remains competitive with some of the latest methods in Table 1. This observation suggests that the process described in Section 3.4 is an efficient technique to incorporate task awareness and improve the quality of prototypes. This is further corroborated in Fig. 3 where a task with $N = 2$ is used to showcase the effect of `OpT-Tune`. We observe that the support embeddings are more evenly spread out over the distribution of the query embeddings. This is also backed by the DBI score [12] which increases

Table 3: Accuracy (%± std.) of (N-way, K-shot) classification on the CDFSL benchmark. Style: **best** and second best.

Method(N, K)	(5,5)	(5,20)	(5,5)	(5,20)	(5,5)	(5,20)	(5,5)	(5,20)
	ChestX		**ISIC**		**EuroSAT**		**CropDiseases**	
UMTRA-ProtoNet [32]	24.94 ± 0.43	28.04 ± 0.44	39.21 ± 0.53	44.62 ± 0.49	74.91 ± 0.72	80.42 ± 0.66	79.81 ± 0.65	86.84 ± 0.50
UMTRA-ProtoTune [32]	25.00 ± 0.43	30.41 ± 0.44	38.47 ± 0.55	51.60 ± 0.54	68.11 ± 0.70	81.56 ± 0.54	82.67 ± 0.60	92.04 ± 0.43
ProtoTransfer [32]	26.71 ± 0.46	33.82 ± 0.48	45.19 ± 0.56	59.07 ± 0.55	75.62 ± 0.67	86.80 ± 0.42	86.53 ± 0.56	95.06 ± 0.32
C^3LR [39]	26.00 ± 0.41	33.39 ± 0.47	45.93 ± 0.54	59.95 ± 0.53	80.32 ± 0.65	88.09 ± 0.45	87.90 ± 0.55	95.38 ± 0.31
`SAMPTransfer` (**ours**)	26.27 ± 0.44	**34.15** ± 0.50	47.60 ± 0.59	**61.28** ± 0.56	**85.55** ± 0.60	88.52 ± 0.50	**91.74** ± 0.55	**96.36** ± 0.28
ConFeSS [11] (dedicated)	**27.09**	33.57	**48.85**	60.10	84.65	**90.40**	88.88	95.34
ATA [49] (dedicated)	24.43 ± 0.2	-	45.83 ± 0.3	-	83.75 ± 0.4	-	90.59 ± 0.3	-
ProtoNet [20] (sup.)	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
Pre+Mean-Cent. [20] (sup.)	26.31 ± 0.42	30.41 ± 0.46	47.16 ± 0.54	56.40 ± 0.53	82.21 ± 0.49	87.62 ± 0.34	87.61 ± 0.47	93.87 ± 0.68
Pre+Linear [20] (sup.)	25.97 ± 0.41	31.32 ± 0.45	48.11 ± 0.64	59.31 ± 0.48	79.08 ± 0.61	87.64 ± 0.47	89.25 ± 0.51	95.51 ± 0.31

Table 4: Ablation study of various parameters on accuracy.

Backbone	p	H	L	β	**OT**	**Accuracy**
Conv4b	1	4	64	1.0	✓	71.42 ± 0.73
Conv4b	1	4	64	0.7	✓	71.41 ± 0.71
Conv4b	1	8	64	1.0	✓	71.27 ± 0.75
Conv4b	1	8	64	0.7	✓	69.87 ± 0.72
Conv4b	2	1	64	0.7	✓	68.99 ± 0.71
Conv4b	2	4	64	0.7	✓	67.01 ± 0.69
Conv4	1	4	64	0.7	✓	69.61 ± 0.71
Conv4	1	4	64	1.0	✓	67.60 ± 0.62
Conv4	1	8	64	1.0	✓	63.59 ± 0.68
Conv4b	1	4	128	0.7	✓	72.52 ± 0.72
Conv4	1	4	128	0.7	✓	68.33 ± 0.71
Conv4	1	4	128	0.0	✓	52.81 ± 0.66
Conv4b	1	4	128	0.0	✓	72.44 ± 0.69
Conv4b	1	4	64	0.7	✗	64.29 ± 0.63
Conv4b	1	4	128	0.7	✗	63.47 ± 0.64
Conv4	1	4	64	0.7	✗	66.73 ± 0.65

Table 5: Accuracy (%± std.) for (N-way, K-shot) classification on *mini*ImageNet with pre-training on CUB.

Training	**Testing**	(5,1)	(5,5)
ProtoTransfer [32]	ProtoTune	35.37 ± 0.63	52.38 ± 0.66
C^3LR [39]	ProtoTune	39.61 ± 1.11	55.53 ± 1.42
`SAMPTransfer` (**ours**)	OpT-Tune	**49.32** ± 0.75	**56.10** ± 0.60

from 0.583 to 0.754 after `OpT-Tune` is applied.

SAMP layers and attention heads. In Table 4, we also investigate the robustness of our method when the number of `SAMP` layers (p) and attention heads (H) vary. The best performance is achieved with a single `SAMP` layer with four attention heads. Increasing p leads to a significant decrease in performance; however, increasing H leads to a small performance degradation. Notably, the observations here are consistent with those reported in [38, 46].

Embedding dimension. We measure the performance of the model in relation to two commonly used (by a majority of the existing baselines) embedding dimensions: 512 and 1600. As can be seen in Table 4, the network performs best with an embedding dimension of 512 (`Conv4b`). Performance is notably lower with an embedding dimension of 1600 (`Conv4`). We hypothesize that this behavior can be attributed to the lower number of channels in the final feature map of a `Conv4` network, which is limited to 64.

Effect of loss scaling factor β on $\mathcal{L}_1$. We observe that when $\beta = 0$ the `Conv4` based model suffers the most as it loses 15% accuracy compared to $\beta = 0.7$, suggesting that training the CNN with a contrastive loss is crucial. However, the `Conv4b` model is not affected as strongly by the presence of this loss function. Regardless, we set $\beta = 0.7$ for both models (`Conv4` and `Conv4b`).

Cross-domain robustness. For the sake of completeness, and to further analyze the cross-domain performance of `SAMPTransfer`, in addition to Table 3, we trained a `Conv4` model on CUB and tested it on tasks derived from *mini*ImageNet. CUB consists of 200 classes of only birds, while *mini*ImageNet consists of 64 classes, of which only 3 training classes are birds. Thus, CUB has a diminished class diversity compared to *mini*ImageNet. Table 5 demonstrates that when training classes are diversity constrained, our method offers a better cross-domain transfer accuracy compared to the only two other competing baselines that report experimental results on this setting.

7. Concluding Remarks

We introduced `SAMP-CLR`, a novel contrastive pre-training method for unsupervised few-shot classification. `SAMP-CLR` learns its representations by looking beyond single-image instances owing to a built-in self-attention message passing (`SAMP`) module. We also propose an optimal transport (OT) based fine-tuning strategy (`OpT-Tune`) which enables the creation of more representative prototypes by inducing task-awareness. Together, they construct our overall unsupervised FSL approach (coined as `SAMPTransfer`). We demonstrate that `SAMPTransfer` sets a new state-of-the-art for unsupervised FSL in both *mini*ImageNet and *tiered*ImageNet datasets, as well as offering competitive performance on the challenging CDFSL benchmark [20]. As future work, we are investigating the idea of incorporating memory modules in `SAMP-CLR` pre-training to help better approximate the data distribution.

References

[1] Antreas Antoniou and Amos Storkey. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *arXiv preprint arXiv:1902.09884*, 2019.

[2] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*, 32, 2019.

[3] Peyman Bateni, Jarred Barber, Jan-Willem van de Meent, and Frank Wood. Enhancing few-shot image classification with unlabelled examples. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2796–2805, 2022.

[4] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*, 2021.

[5] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[8] 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.

[9] Wentao Cui and Yuhong Guo. Parameterless transductive feature re-representation for few-shot learning. In *International Conference on Machine Learning*, pages 2212–2221. PMLR, 2021.

[10] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[11] Debasmit Das, Sungrack Yun, and Fatih Porikli. ConfeSS: A framework for single source cross-domain few-shot learning. In *International Conference on Learning Representations*, 2022.

[12] David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[13] 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*, pages 248–255. Ieee, 2009.

[14] Guneet S Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. *arXiv preprint arXiv:1909.02729*, 2019.

[15] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9588–9597, 2021.

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

[17] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8059–8068, 2019.

[18] Priya Goyal, Mathilde Caron, Benjamin Lefaudeux, Min Xu, Pengchao Wang, Vivek Pai, Mannat Singh, Vitaliy Liptchinsky, Ishan Misra, Armand Joulin, and Piotr Bojanowski. Self-supervised pretraining of visual features in the wild. 3 2021.

[19] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[20] Yunhui Guo, Noel CF Codella, Leonid Karlinsky, John R Smith, Tajana Rosing, and Rogerio Feris. A New Benchmark for Evaluation of Cross-Domain Few-Shot Learning. *arXiv preprint arXiv:1912.07200*, 2019.

[21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[22] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification, 2017.

[23] Kyle Hsu, Sergey Levine, and Chelsea Finn. Unsupervised learning via meta-learning. *arXiv preprint arXiv:1810.02334*, 2018.

[24] Zilong Ji, Xiaolong Zou, Tiejun Huang, and Si Wu. Unsupervised few-shot learning via self-supervised training. *arXiv preprint arXiv:1912.12178*, 2019.

[25] Siavash Khodadadeh, Ladislau Boloni, and Mubarak Shah. Unsupervised meta-learning for few-shot image classification. *Advances in neural information processing systems*, 32, 2019.

[26] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo. Edge-labeling graph neural network for few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11–20, 2019.

[27] Jinwoo Kim, Tien Dat Nguyen, Seonwoo Min, Sungjun Cho, Moontae Lee, Honglak Lee, and Seunghoon Hong. Pure transformers are powerful graph learners. *arXiv preprint arXiv:2207.02505*, 2022.

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] Dong Bok Lee, Dongchan Min, Seanie Lee, and Sung Ju Hwang. Meta-gmvae: Mixture of gaussian vae for unsupervised meta-learning. In *ICLR*, 2021.

[30] Jianyi Li and Guizhong Liu. Few-shot image classification via contrastive self-supervised learning. *arXiv preprint*

arXiv:2008.09942, 2020.

[31] Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, Eunho Yang, Sung Ju Hwang, and Yi Yang. Learning to propagate labels: Transductive propagation network for few-shot learning. *arXiv preprint arXiv:1805.10002*, 2018.

[32] Carlos Medina, Arnout Devos, and Matthias Grossglauser. Self-supervised prototypical transfer learning for few-shot classification. *arXiv preprint arXiv:2006.11325*, 2020.

[33] 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.

[34] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[35] Tiexin Qin, Wenbin Li, Yinghuan Shi, and Yang Gao. Diversity helps: Unsupervised few-shot learning via distribution shift-based data augmentation. *arXiv preprint arXiv:2004.05805*, 2020.

[36] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

[37] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018.

[38] Jenny Denise Seidenschwarz, Ismail Elezi, and Laura Leal-Taixé. Learning intra-batch connections for deep metric learning. In *International Conference on Machine Learning*, pages 9410–9421. PMLR, 2021.

[39] Ojas Kishore Shirekar and Hadi Jamali-Rad. Self-supervised class-cognizant few-shot classification. *arXiv preprint arXiv:2202.08149*, 2022.

[40] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

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

[42] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer, 2020.

[43] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-Dataset: A Dataset of Datasets for Learning to Learn from Few Examples. *arXiv preprint arXiv:1903.03096*, 2020.

[44] Stijn Van Dongen and Anton J Enright. Metric distances derived from cosine similarity and pearson and spearman correlations. *arXiv preprint arXiv:1208.3145*, 2012.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

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

[48] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[49] Haoqing Wang and Zhi-Hong Deng. Cross-domain few-shot classification via adversarial task augmentation. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 1075–1081. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Main Track.

[50] 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*, pages 2097–2106, 2017.

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

[52] Zhu Xiaojin and Ghahramani Zoubin. Learning from labeled and unlabeled data with label propagation. *Tech. Rep., Technical Report CMU-CALD-02–107, Carnegie Mellon University*, 2002.

[53] Hui Xu, Jiaxing Wang, Hao Li, Deqiang Ouyang, and Jie Shao. Unsupervised meta-learning for few-shot learning. *Pattern Recognition*, 116:107951, 2021.

[54] Weijian Xu, Yifan Xu, Huaijin Wang, and Zhuowen Tu. Attentional constellation nets for few-shot learning. In *International Conference on Learning Representations*, 2021.

[55] Ling Yang, Liangliang Li, Zilun Zhang, Xinyu Zhou, Erjin Zhou, and Yu Liu. Dpgn: Distribution propagation graph network for few-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13390–13399, 2020.

[56] Han-Jia Ye, Lu Han, and De-Chuan Zhan. Revisiting Unsupervised Meta-Learning via the Characteristics of Few-Shot Tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2022.

[57] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8808–8817, 2020.

[58] Tianyuan Yu, Sen He, Yi-Zhe Song, and Tao Xiang. Hybrid graph neural networks for few-shot learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3179–3187, 2022.

[59] Hongguang Zhang, Piotr Koniusz, Songlei Jian, Hongdong Li, and Philip HS Torr. Rethinking class relations: Absolute-relative supervised and unsupervised few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9432–9441, 2021.

[60] Hongguang Zhang, Hongdong Li, and Piotr Koniusz. Multi-

level second-order few-shot learning. *IEEE Transactions on Multimedia*, pages 1–1, 2022.

[61] Xueting Zhang, Debin Meng, Henry Gouk, and Timothy M. Hospedales. Shallow bayesian meta learning for real-world few-shot recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 651–660, October 2021.