MASP: MODEL-AGNOSTIC SAMPLE PROPAGATION FOR FEW-SHOT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Few-shot learning aims to train a classifier given only a few samples per class that are highly insufficient to describe the whole data distribution. These few-shot samples not only introduce high variance to the training but also may include outliers near the class boundaries. Directly feeding these samples to training algorithms can lead to unstable optimization and even incorrect gradient descent direction. In this paper, we improve the robustness to "outliers" by learning to propagate and refine the representations of few-shot samples to form a more compact data distribution before using them to train a classifier. We develop a mutual calibration among few-shot samples' representations by graph propagation, for which we learn an attention mechanism to build the graph and determine the propagation weights. On both clean datasets and datasets containing noisy labels, we show that our sample propagation generally improves different types of existing few-shot learning methods in multiple few-shot learning settings.

1 INTRODUCTION

Over the past decade, machine learning models have demonstrated their generalization power to unseen data if trained on sufficient data drawn from the same distribution for the same task. However, human intelligence is more powerful in adapting the wisdom learned from those accomplished tasks to new and unseen tasks in the future. Recent studies of meta-learning (or "learning to learn") (Finn et al., 2017; Snell et al., 2017; Ravi & Larochelle, 2016) aim to extract this type of task-agnostic knowledge, collect it from multiple tasks, and leverage it to make the learning of unseen tasks with highly deficient data possible.

This raises a question: is this set of samples a good deputy of this task and can these samples well-represent the data distribution of every class? As shown in Figure 1, if we directly represent each class by a prototype, e.g., the average feature representation of a few samples from the class (Snell et al., 2017), the prototype may suffer from high variance at the presence of outliers and lose the capability to represent the whole class's distribution. This is a joint result of data deficiency and the direct usage of a pre-trained model in producing the feature representations, which are trained on sufficient data and spans an ambient space for the data distribution. Hence, when we only know a few samples for a class in the space, it is almost impossible to reconstruct the true distribution of the class and any sample far away from some others could be an outlier. This could lead to inaccurate distance-based classifier and accumulated errors if the prototype needs to get further refined, for example, by prototype propagation (Liu et al., 2019a). Similarly, the gradients generated by a small minibatch of samples may lead to a suboptimal direction and make the optimization unstable. This is especially evident in the optimization for "learning to learn" where higher order derivatives are computed or approximated: the model gets to a suboptimal saddle point, from which the loss is computed and a second order derivatives are backpropagated (Finn et al., 2017).

We propose *model-agnostic sample propagation* (MASP) calibrating the representations produced for few-shot samples to a more compact space before using them to train a task-specific model. MASP learns to correct the high variance existing in the representations of few-shot samples produced by a pre-trained model. In MASP, we train an iterative propagation scheme that sends messages across all training samples for the task, which deploys an attention module to build connections between samples and compute the weight of propagating a sample's representation to another. The sample propagation can be applied over all samples in one task so that both intra-class distribution

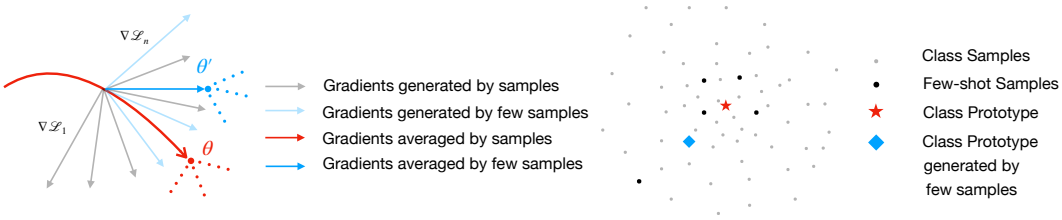

Figure 1: **LEFT:** Gradients generated by few samples can lead to suboptimal saddle point (blue). **RIGHT:** Averaging the feature representations of few samples in one class can generate inaccurate class prototypes affected by outliers.

and inter-class distribution/relationships can be taken into the considerations. It is also easy to be extended to the transductive setting, where the query samples can also be included in the propagation.

In experiments, we show that MASP can serve as a plug-and-play module between a pre-trained model and any few-shot/meta-learning model, and can consistently improve their original few-shot learning performance. For example, we successfully plug MASP into the most popular models, i.e., prototypical network (Snell et al., 2017) and MAML (Finn et al., 2017) for traditional few-shot setting. MASP can also bring improvements to the Gated Propagation Network (Liu et al., 2019a) for the graph meta-learning setting, where outliers are more distracting since there are follow-up prototype refinement steps applied to the prototypes. We not only evaluate MASP on the standard benchmarks for few-shot learning, we also test it on a more practical dataset, i.e., WebVision (Li et al., 2017), which are web-labeled data with real-world noises as true outliers. We test if MASP can remove the noises from the dataset by the learned propagation. To understand how sample propagation modifies the representations and their distribution, we visualize them before and after propagation and observe that MASP moves the outliers closer to the cluster of other samples and thus provide more compact representations easier for the task-specific model to learn.

2 Related Works

Meta-learning has shown advantages in few-shot learning for tasks in regression, classification and reinforcement learning. Early works propose to use LSTM (Ravi & Larochelle, 2016) as a meta-learner or a mechanism of learning to match (Vinyals et al., 2016) to mitigate the few-shot challenges in the new tasks. MAML (Finn et al., 2017) and Prototypical Networks (Snell et al., 2017) are widely-used methods that are representatives of gradient-based methods and non-parametric methods, respectively. We propose a sample propagation module and try to plug it onto both methods.

Graph in Meta-learning Meta-learning has been applied to few-shot learning using graphs for tasks such as link prediction (Bose et al., 2019) and image classification (Liu et al., 2019a; Garcia & Bruna, 2018; Yao et al., 2019). In (Bose et al., 2019), they learn to initialize a graph neural network using the meta-learning framework along with a learned graph signature function for few-shot link prediction. In the task for few-shot image classification, previous works apply graph structures to propagate messages between the connected nodes, where the nodes can be images (Garcia & Bruna, 2018), classes (Liu et al., 2019a) or tasks (Yao et al., 2019). (Yao et al., 2019) propose to cluster tasks in a hierarchical structure so that the initialization of the model for gradient-based meta-learning methods can be shared within a cluster instead of within all tasks (Finn et al., 2017). (Yao et al., 2020) extends their work by developing an automated way to build the hierarchy, which is hand-crafted in their previous version. (Garcia & Bruna, 2018) propose to build a densely-connected graph with an image as a node. In this way, all images in a one-shot learning task are connected and a GNN is applied to make the prediction based on the graph. The weights of the edges are assigned in (Kim et al., 2019) so that various relationships between images can be considered. For our work, we propose a sample propagation module which is algorithm-agnostic and can be applied on the feature representations of a CNN followed by any classification algorithm.

Task Adaptive Representation in Meta-Learning Another line of work tries to adapt the representations conditioned on the current task. The same sample can have different representations by

making it conditioned on a representation of the current task (Oreshkin et al., 2018; Wang et al., 2019). The feature space can be projected to another space (Yoon et al., 2019). As for sample propagation, the model learns to linearly compose the feature representations of the samples in one task without mapping them to another hidden space. (Li et al., 2019) also uses the structure of attention for a task-specific representation. However, they try to apply attention to mask the feature representations based on inter-class commonality and inter-class uniqueness while the attention in our work is for generating the weights and the connections for propagation. Kye et al. (2020) meta-learns an input-adaptive distance metric over a task distribution under various model and data perturbations for the transductive setting. Comparatively, we try to improve the sample representation and is agnostic to different few-shot learning settings. Allen et al. (2019) try to increase the number of prototypes to alleviate the high variance in few-shot learning, while we try to improve the sample representation.

3 MASP: MODEL-AGNOSTIC SAMPLE PROPAGATION

In this section, we first introduce the background knowledge in Section 3.1. Then, we introduce our sample propagation layer and model in Section 3.2. Next, we introduce a natural extension of our MASP from the inductive few-shot setting to the transductive few-shot setting in Section 3.3. Lastly, we will briefly introduce how to utilize the propagated features in Section 3.4.

3.1 PRELIMINARY

In this section, we introduce a typical few-shot classification setting. Given a dataset with data-label pairs $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}$ where $\boldsymbol{x}_i \in \mathbb{R}^d$ is the feature vector of a sample and $y_i \in C$, where C denotes the set of classes. This set of classes is divided into *base classes* C_b and *novel classes* C_n, where $C_b \cap C_n = \emptyset$ and $C_b \cup C_n = C$. The most popular way to build a task is called an N-way-K-shot task (Vinyals et al. (2016)), where N classes are sampled from the novel set and only K (e.g., 1 or 5) labeled samples are provided for each class. The few available labeled data are called *support set* $\mathcal{S} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N\times K}$ and the model is evaluated on another query set $\mathcal{Q} = \{(\boldsymbol{x}_i, y_i)\}_{i=N\times K+1}^{N\times K+N\times q}$, where every class in the task has q test cases. The goal for a meta-learning algorithm is to train a model on multiple N-way-K-shot tasks and extract the common knowledge from them. The generalization ability of the model is evaluated on multiple tasks sampled from the novel classes C_n, where each task is a N-way-K-shot task. The performance of a model is evaluated as the averaged accuracy on (the query set of) multiple tasks sampled from the novel classes.

3.2 SAMPLE PROPAGATION LAYER

Sample propagation layer learns to propagate the representation of samples within a task to each other. Here we denote $\boldsymbol{x}_i[t]$ as the feature representation of the i-th sample after the t-th propagation layers. Formally, the procedure of one sample propagation layer can be formulated as:

$$\boldsymbol{x}_i[t+1] \triangleq \sum\nolimits_{(\boldsymbol{x}_j, y_j)\in\mathcal{D}^T} w_{i,j}[t] \times \boldsymbol{x}_j[t], \tag{1}$$

where y_j is the label of j-th sample and $\boldsymbol{x}_i[0]$ is output of the neural network backbone with the input of the i-th sample. $\mathcal{D}^T$ denotes the data-lablel pairs used for task T. The weight for propagating the representation from the j-th sample to the i-th sample, i.e., $w_{i,j}[t]$, is calculated by an attention mechanism based on their original representation:

$$w_{i,j}[t] = \frac{\exp(u_{i,j} \times a(\boldsymbol{x}_i[t], \boldsymbol{x}_j[t]))}{\sum_k \exp(u_{i,k} \times a(\boldsymbol{x}_i[t], \boldsymbol{x}_k[t]))}, \tag{2}$$

The representation of the samples from the same class can potentially share more than that from other classes. In order to distinguish the samples from the same class, the attention score of within-class samples are up-weighted by a hyperparameter α:

$$u_{i,j} = \begin{cases} \alpha, & \text{if } y_i = y_j. \\ 1, & \text{if } y_i \neq y_j. \end{cases} \tag{3}$$

Algorithm 1 Training of Model-Agnostic Sample Propagation model MASP

Input: sample propagation model f with S layers, a few-shot learning model, episodes τ_{total};
1: **Initialization:**
MASP parameters Θ, few-shot learning model parameters $\Theta^{\dagger}$, current epoch $\tau \leftarrow 0$;
2: **for** $\tau \in \{1, \cdots, \tau_{total}\}$ **do**
3: Sample a few-shot task T with its support set $\mathcal{S}$ and query set $\mathcal{Q}$ from a task distribution $\mathcal{T}$;
4: Initialize the representation $\boldsymbol{x}$ for each sample in $\mathcal{S}$ and $\mathcal{Q}$ by a pre-trained model;
5: **Apply sample propagation with MASP on the support set** $\mathcal{S}$ by Equation 1;
6: Compute the meta-objective of the few-shot learning model on the query set $\mathcal{Q}$;
7: Back-propagation updates Θ and $\Theta^{\dagger}$ by a training strategy for the few-shot learning model;
8: **end for**

The raw attention score is calculated by a dot product attention function $a(\cdot, \cdot)$ as follows:

$$a(p, q) = \frac{\langle h_1(p), h_2(q) \rangle}{\|h_1(p)\| \times \|h_2(q)\|}, \tag{4}$$

where h_1 and h_2 are linear transformation functions.

The MASP model is stacked by multiple sample propagation layers. Suppose there are S layers for our MASP model, it takes the original representation of all samples in a task T as input, and outputs $\boldsymbol{x}[S]$ for each sample as its propagated feature representation. We will introduce how to utilize the propagated feature representation below.

3.3 Extending the Scope to Transductive Setting

Sample propagation can be easily applied to unlabeled samples, so that the query set can be leverged in the propagation procedure as well. This is another popular setting in few-shot learning, i.e., transductive few-shot learning (Liu et al., 2019b). In this case, the sample propagation will include samples from both the query set D^T_{valid} and the support set D^T. As a result, Equation 1 can be reformulated as:

$$\boldsymbol{x}_i[t+1] \triangleq \sum_{(\boldsymbol{x}_j, y_j) \in \mathcal{D}^T \cup \mathcal{D}^T_{valid}} w_{i,j}[t] \times \boldsymbol{x}_j[t]. \tag{5}$$

The attention mechanism uses the same structure as Equation 4. Note that the unlabeled data is not upweighted since there is no accurate label for every sample and giving more weights on them may make the optimization unstable especially in the beginning stage:

$$u^T_{i,j} = \begin{cases} \alpha, & \text{if } y_i = y_j \text{ and } (\boldsymbol{x}_i, y_i) \in \mathcal{D}^T. \\ 1, & \text{otherwise.} \end{cases} \tag{6}$$

Thus, the sample propagation in transductive setting generally follows the same procedure as the inductive setting and sets the weights of the query set as default.

3.4 How to use features after sample propagation?

Our sample propagation module can be plugged to the beginning of most existing meta-learning models as preprocessing to the representations generated by a pre-trained model. In particular, given the few-shot training data for a specified task, we first initialize the representations of each sample by a pre-trained model. Then, we apply MASP to the representations to gain a more compact representation space with possible outliers removed. Afterwards, we can apply any few-shot learning model that takes the calibrated representations as inputs and produce a task-specific model (test phase) or train the meta-learner model by back-propagation (training phase).

MASP does not depend on any assumption to the data type since its feature calibration only depends on the propagation between samples rather than any structured transformation such as convolution. It bridges the pre-trained model and the meta-learning model and can be applied to any combination of the two. Hence, we argue that MASP is model-agnostic and a general technique that potentially improves most existing few-shot learning models. Moreover, it is also compatible with most training strategies. In our experiments, we apply it to a diverse set of few-shot learning models and achieve improvements on all of them.

4 TRAINING STRATEGY

The full training algorithm of a general few-shot learning framework with MASP is shown in Algorithm 1. It elaborates how to plug MASP to an existing framework of few-shot learning, regardless of which exact pre-trained model and few-shot learning model are used. Afterwards, they can be trained in an end-to-end manner with an initialization learned by MASP. Specifically, few-shot tasks are sampled based on the task generation mechanisms introduced in Experiments. Then MASP learns to preprocess the representations of the few-shot samples of the support set, which are usually produced by a pre-trained model, using sample propagation at line 4 of Alg. 1. The preprocessed representations are then fed to a few-shot learning model as input features for the few-shot training samples. We then compute the validation loss and apply back-propagation to update both the MASP module and the meta-learner. The model can be trained by back-propagating the cross-entropy loss to the backbone network, meta-learner and MASP module. Since the sample propagation is modeled by an attention network, these modules can be trained in an end-to-end way efficiently.

5 EXPERIMENTS

In this section, we mainly want to address if MASP can be general and consistently improve the performance in various settings. Specifically, we answer the following questions:

- Can MASP generally be applicable to different kinds of meta learning approaches?
- Can MASP generalize to different algorithms in different meta-learning settings/problems?
- Can MASP be robust to different datasets and improve the performance?
- Can MASP consistently improve the performance under both inductive and transductive settings?

5.1 DATASETS AND EXPERIMENTAL SETUPS

Datasets. To investigate the ability of avoiding noise for MASP, we first try it on miniWebVision – **a dataset with noisy labels**. The miniWebVision dataset is created for meta-learning from the WebVision 1.0 (Li et al., 2017) dataset. Specifically, we randomly sampling 64 classes and 16 classes from the original training classes as the meta train classes and the meta validation classes, respectively. We also randomly sample 20 classes from the original validation classes of WebVision 1.0 to serve as the meta test classes. All images come from the resized Google image database. Therefore, there are 56K training images, 16K validation images, and 1K test images. To test the performance of MASP for conventional meta-learning, we then use the Omniglot (Lake et al., 2015) dataset following the most popular setting for conventional meta-learning. Omniglot is a hand-written character dataset consisting of 1623 characters from 50 alphabets (super classes). We follow the split proposed in Vinyals et al. (2016) and also the way of resizing the image to 28*28 and augmenting the number of classes by rotating images of multiple 90°. 1200 characters (and their rotations) are used for training, and the remaining classes are for testing.

To test the performance of MASP for graph meta-learning, we use the datasets: *tiered*ImageNet-Close and *tiered*ImageNet-Far as proposed in Liu et al. (2019a). *tiered*ImageNet-Close and *tiered*ImageNet-Far are datasets of images with information for class hierarchy. The images come from ImageNet and the hierarchy is extracted from WordNet. The training sets for these two datasets are the same but the test sets differ in that the test classes are 1 to 4 hops away from the training classes in *tiered*ImageNet-Close based on WordNet while *tiered*ImageNet-Far has a test set that is further away at 5 to 10 hops from the training set. These two datasets provide comprehensive test on how the techniques of graph meta-learning will improve the performance of few-shot learning under various situations of relatedness/closeness between classes.

Experimental Setup In order to make an apple-to-apple comparison, we follow the same setup as in (Snell et al., 2017) and (Liu et al., 2019a). Specifically, for conventional meta-learning, the backbone is a CNN stacked with four convolutional blocks. Each block comprises a 64-filter 3*3 convolution, batch normalization layer, a ReLU non-linearity and a 2*2 max-pooling layer. The model is training with an Adam (Kingma & Ba, 2015) optimizer with initial learning rate as 0.001 and decrease by 0.5 every 25K episodes. We train the model by 200K episodes in total.

Table 1: Baseline vs. baseline with MASP – comparison w.r.t. evaluation accuracy (mean±CI$\%95$) on 600 test tasks on the miniWebVision dataset.

Model	5way1shot (%)	5way5shot (%)	10way1shot (%)	10way5shot (%)
Prototypical Net (Snell et al., 2017)	41.13±1.68	63.29±0.90	31.37±1.11	50.20±0.62
Prototypical Net w/ MASP	44.27±1.65 **Δ+3.1**	65.36±0.91 **Δ+2.1**	33.10±1.12 **Δ+1.7**	51.11±0.65 **Δ+0.9**
MAML (Finn et al., 2017)	37.53±1.55	52.65±0.91	26.60±1.05	24.43±0.52
MAML w/ MASP	38.46±1.66 **Δ+0.9**	52.83±0.99 **Δ+0.2**	26.45±1.12 **Δ-0.2**	26.28±0.55 **Δ+1.9**

For graph meta-learning, we train our model for τ_{total} =350k episodes using Adam with an initial learning rate of 10^{-3} and weight decay 10^{-5}. The learning rate is reduced by a factor of $0.9\times$ every $10k$ episodes starting from the $20k$-th episode. The prototypes are learned by sample propagation for $S = 2$ steps and the prototypes with learned initialization are then followed by prototype propagation for $S^{'} = 2$ steps. More steps may result in higher performance with a resulting outcome of more computations. The hyperparameter α for upweighting samples is 5. The hyper-parameter λ is 0.5. We use linear transformation for all transformations in attention, i.e., $h_1(\cdot)$, $h_2(\cdot)$, $h_1^{'}(\cdot)$ and $h_2^{'}(\cdot)$. To ensure a fair comparison, we follow the same setting in Liu et al. (2019a): using the same backbone ResNet-08; using the same setup of the training tasks, i.e., N-way-K-shot, for all methods including baselines in our experiments; using the same random sampling strategy for task generation. The test stage is conducted over 600 tasks with the size of the query set the same as that of its support set.

5.2 Empirical Analysis for Few-shot Learning for Clean and Noisy Datasets

Table 2: Validation accuracy on the Omniglot dataset with 50 ways.

Model	50way5shot	50way10shot
Prototypical Net	96.5%	97.1%
Prototypical Net w/ MASP	**96.8**%	**97.3**%

Our MASP is compatible with any kinds of few-shot learning methods. We choose two representative methods, Prototypical Net and MAML, as our baselines. We evaluate "baseline" vs "baseline with MASP" on the 5-way-1-shot, 5-way-5-shot, 10-way-1-shot, and 10-way-5-shot settings on miniWebVision in Table 1. In most scenarios, our MASP significantly improves the accuracy of the baseline models, e.g., 3.1% accuracy gain for Prototypical Net in 5-way-1-shot and 1.8% accuracy gain for MAML in 10-way-5-shot. The only failure case is for MAML in 10-way-1-shot: MAML + MASP drops the accuracy 0.15 within the range of variance.

We also investigate the effect of MASP on the commonly used Omniglot (Lake et al., 2015) when the number of classes within each task is high – 50. Here we only use the Prototypical Net as the baseline, as MAML requires much memory for this high-way setting. We show the results in Table 2. MASP improves the relative accuracy by 8.5% on 50-way-5-shot and 6.8% on 50-way-10-shot. *In sum, for the conventional few-shot learning problem, MASP consistently improve the performance of different few-shot methods.*

5.3 Empirical Analysis for Graph Meta-learning

MASP achieves the SoTA accuracy on *tiered*ImageNet-**Close** and *tiered*ImageNet-**Far**. We investigate the effect of MASP over different baseline models and compare it with other methods. We show the results on *tiered*ImageNet-**Close** and *tiered*ImageNet-**Far** in Table 3 and Table 4, respectively. In each table, we compare methods with (+) / without hierarchy during the test stage.

Table 3: Evaluation accuracy (mean±CI%95) on 600 test tasks on *tiered*ImageNet-**Close**.

Model	Reasoning setting	5way1shot	5way5shot	10way1shot	10way5shot
Prototypical Net	Inductive	42.87±1.67%	62.68±0.99%	30.65±1.15%	48.64±0.70%
GNN	Inductive	42.33±0.80%	59.17±0.69%	30.50±0.57%	44.33±0.72%
Closer Look	Inductive	35.07±1.53%	47.48±0.87%	21.58±0.96%	28.01±0.40%
PPN	Inductive	41.60±1.59%	63.04±0.97%	28.48±1.09%	48.66±0.70%
GPN	Inductive	48.37±1.80%	64.14±1.00%	33.23±1.05%	50.50±0.70%
GPN w/ MASP	Inductive	**49.90±1.79%**	**65.90±1.03%**	**34.90±1.19%**	**51.00±0.73%**
GPN+	Inductive	50.54±1.67%	65.74±0.98%	34.74±1.05%	**51.50±0.70%**
GPN+ w/ MASP	Inductive	**52.50±1.71%**	**66.00±0.98%**	**37.00±1.18%**	51.20±0.71%
TPN	Transductive	43.50±0.01%	61.80±0.01%	32.80±0.01%	46.70±0.01%
GPN w/ MASP	Transductive	53.20±1.95%	67.00±0.97%	35.50±1.19%	**51.00±0.72%**
GPN+ w/ MASP	Transductive	**54.10±1.72%**	**67.50±0.95%**	**40.30±1.15%**	50.70±0.70%

Table 4: Evaluation accuracy (mean±CI%95) on 600 test tasks onon *tiered*ImageNet-**Far**.

Model	Reasoning setting	5way1shot	5way5shot	10way1shot	10way5shot
Prototypical Net	Inductive	44.30±1.63%	61.01±1.03%	30.63±1.07%	47.19±0.68%
GNN	Inductive	43.67±0.69%	59.33±1.04%	30.17±0.47%	43.00±0.66%
Closer Look	Inductive	42.27±1.70%	58.78±0.94%	22.00±0.99%	32.73±0.41%
PPN	Inductive	43.63±1.59%	60.20±1.02%	29.55±1.09%	46.72±0.66%
GPN	Inductive	47.54±1.68%	64.20±1.01%	31.84±1.10%	48.20±0.69%
GPN w/ MASP	Inductive	**49.80±1.83%**	**64.70±0.97%**	**35.20±1.10%**	**48.90±0.65%**
GPN+	Inductive	47.49±1.67%	64.14±1.02%	31.95±1.15%	48.65±0.66%
GPN+ w/ MASP	Inductive	**51.70±1.73%**	**65.80±1.03%**	**34.60±1.08%**	**49.80±0.68%**
TPN	Transductive	48.10±0.01%	62.00±0.02%	33.00±0.01%	45.30±0.01%
GPN w/ MASP	Transductive	52.20±1.78%	65.20±1.05%	34.90±1.12%	**47.70±0.63%**
GPN+ w/ MASP	Transductive	**53.50±1.78%**	**67.10±0.99%**	**37.10±1.19%**	**47.70±0.65%**

We observe: (1) MASP achieves the highest accuracy on these two datasets and the improvements over other methods is non-trival. (2) MASP can consistently improve the performance of few-shot learning under all settings by 1∼4% compared to the baseline. (3) The original GPN is infeasible to handle the transductive graph few-shot setting. With the equipment of MASP, we can make it be feasible for the transductive setting and further improves the accuracy.

5.4 DISCUSSION

Visualization. We visualize the prototypes of the before and after in Figure 2 to visualize what modifications the propagation makes in the representation space. Here the transparent dots denotes the samples from the support set and each class is represented by a color. The mean of the representations, denoted by triangles, are affected by the outliers and are dragged away. Some triangles result in overlapping to each other and can have inaccurate classification boundaries. The sample propagation can get rid of the noise from the outliers and learn a prototype which lies in the majority of the support set. After the sample propagation, the boundaries are more clear.

Advantages of MASP. Our MASP is a light-weight module that can be plug-in into any few-shot learning models to improve their accuracy. There are only four hyper-parameters of our MASP – the number of propagation layers, the number of hidden dimensions, the keep ratio, and α. As we observed in experiments, the same hyperparameters work well across different models, datasets, and settings. It further demonstrates the effectiveness and flexibility of our MASP To be reproducible, we describe our PyTorch implementation below. It is quite simple and easy to implement – just 22 lines of code, then you can obtain up to 4% of the accuracy gain.

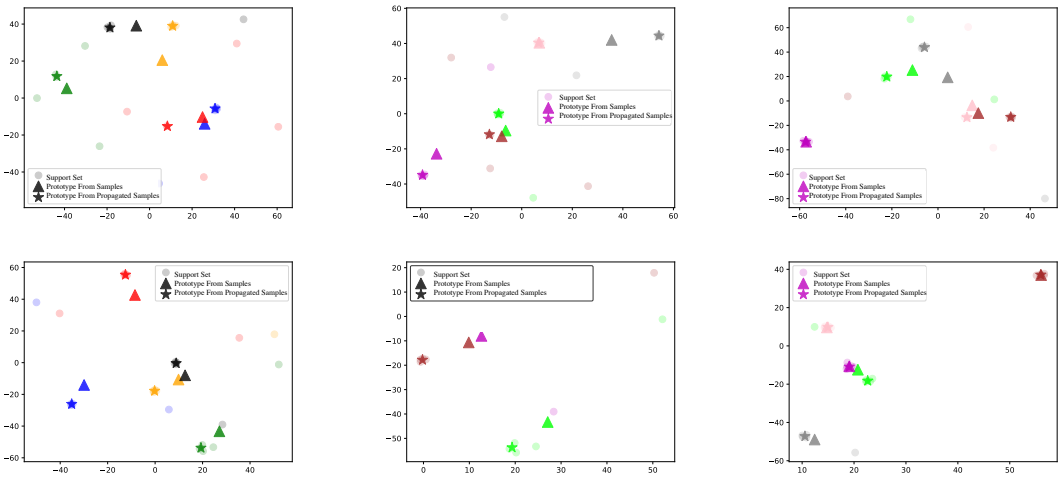

Figure 2: The t-distributed stochastic neighbor embedding (t-SNE) analysis for MASP. We use t-SNE to visualize the original representation, and its sample propagated representation. For each class, we also visualize its prototype – the mean feature representation of all samples.

PyTorch Implementation – just 22 lines of code to get up to 4% accuracy gain on *tiered*ImageNet-**Close** and *tiered*ImageNet-**Far**.

```
class SamplePropagationLayer(nn.Module):

  def __init__(self, in_dim, hidden=128, keep_ratio=0.5, alpha=5):
    super(SamplePropagationLayer, self).__init__()
    self.q = nn.Linear(in_dim, hidden, bias=False)
    self.k = nn.Linear(in_dim, hidden, bias=False)
    self.keep = keep_ratio
    self.alpha = alpha

  def forward(self, vectors, classes):
    vectors_q, vectors_k = self.q(vectors), self.k(vectors)
    num_samples, num_feats = vectors.shape
    raw_attention = F.cosine_similarity(
        vectors_q.view(num_samples, 1, -1),
        vectors_k.view(1, num_samples, -1), dim=-1)
    downweight = torch.ones_like(raw_attention)
    is_same_classes = classes.view(num_samples, 1) == classes.view(1,
    num_samples)
    downweight[is_same_classes] = self.alpha
    attention = F.softmax(raw_attention * downweight, dim=-1)
    prop_vectors = torch.matmul(attention, vectors)
    final_vectors = (1 - self.keep) * prop_vectors + self.keep * vectors
    return final_vectors
```

6 Conclusion

In this paper, we propose a mechanism called "model-agnostic sample propagation (MASP)"that leans to preprocess the representations of few-shot samples produced by a pre-trained model for the purpose of outlier removal and more compact representations. It improves the training of task-specific model and meta-learner in the scenario of few-shot data when the data distribution is hard to be captured in the pre-trained representation space. It can be generally applied to any existing few-shot learning models and settings without specifying the meta-learner and task-learner models' architecture or their training strategies. We show that sample propagation is a general technique and can consistently improve performance under various meta-learning setting, e.g., inductive and transductive, conventional meta-learning and graph meta-learning.

REFERENCES

Kelsey Allen, Evan Shelhamer, Hanul Shin, and Joshua Tenenbaum. Infinite mixture prototypes for few-shot learning. In *International Conference on Machine Learning*, pp. 232–241, 2019.

Avishek Joey Bose, Ankit Jain, Piero Molino, and William L. Hamilton. Meta-graph: Few shot link prediction via meta learning, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *The International Conference on Machine Learning (ICML)*, 2017.

Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo. Edge-labeling graph neural network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Seong Min Kye, Hae Beom Lee, Hoirin Kim, and Sung Ju Hwang. Meta-learned confidence for few-shot learning, 2020.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Hongyang Li, David Eigen, Samuel Dodge, Matthew Zeiler, and Xiaogang Wang. Finding task-relevant features for few-shot learning by category traversal. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

Lu Liu, Tianyi Zhou, Guodong Long, Jing Jiang, and Chengqi Zhang. Learning to propagate for graph meta-learning. In *The Conference on Neural Information Processing Systems (NeurIPS)*, 2019a.

Yanbin Liu, Juho Lee, Minseop Park, Saehoon Kim, and Yi Yang. Transductive propagation network for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2019b.

Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *The Conference on Neural Information Processing Systems (NeurIPS)*, 2018.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2016.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *The Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *The Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

Xin Wang, Fisher Yu, Ruth Wang, Trevor Darrell, and Joseph E Gonzalez. Tafe-net: Task-aware feature embeddings for low shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

Huaxiu Yao, Ying Wei, Junzhou Huang, and Zhenhui Li. Hierarchically structured meta-learning. In *The International Conference on Machine Learning (ICML)*, 2019.

Huaxiu Yao, Xian Wu, Zhiqiang Tao, Yaliang Li, Bolin Ding, Ruirui Li, and Zhenhui Li. Automated relational meta-learning. In *International Conference on Learning Representations (ICLR)*, 2020.

Sung Whan Yoon, Jun Seo, and Jaekyun Moon. Tapnet: Neural network augmented with task-adaptive projection for few-shot learning. In *The International Conference on Machine Learning (ICML)*, 2019.