Supplementary Material of Tailoring Embedding Function to Heterogeneous Few-Shot Tasks by Global and Local Feature Adaptors

Su Lu, Han-Jia Ye, De-Chuan Zhan
National Key Laboratory for Novel Software Technology, Nanjing University
Nanjing, 210023, China
{lus, yehj, zhandc}@lamda.nju.edu.cn

Abstract

This is the supplementary material of paper "Tailoring Embedding Function to Heterogeneous Few-Shot Tasks by Global and Local Feature Adaptors". In this material, we provide more discussions and analyses about our proposed **G**lobal and **Lo**cal **F**eature **A**daptor (GLoFA), which generates bi-level task-adaptive feature masks for few-shot learning. There are three parts in this supplement: first, we review our proposed GLoFA; then, some discussions on algorithm details are given; last are experiment details.

Review of GLoFA

GLoFA belongs to metric-based meta-learning methods (Koch, Zemel, and Salakhutdinov 2015; Vinyals et al. 2016; Snell, Swersky, and Zemel 2017). These algorithms often meta-learn an embedding function from *meta-training* set, and then reuse the learned representation on *meta-testing* set. A main drawback of these algorithms is that all the tasks share a same embedding space, ignoring the diversity of tasks. Some researchers proposed task-specific embedding space in recent works (Oreshkin, López, and Lacoste 2018; Li et al. 2019; Ye et al. 2020). Different from these methods which apply a global transformation to all the instances, GLoFA incorporates extra local transformations, improving the representation ability of feature adaptors.

There are three components in GLoFA, namely embedding network $\phi(\cdot)$, feature adaptor $\mathcal{F}$ and mask combiner $g(\cdot)$. The embedding network extracts vector features from raw data. Global and local feature adaptors capture general and detailed characteristics respectively by generating feature masks. Instances from each class are projected into class-specific feature spaces by local masks. Global and local masks have different effects, and different masks should be emphasized for different tasks. To obtain an adaptive fusion of two masks, we balance them via a mask combiner.

In detail, for a task $\mathcal{T} = (\mathcal{S}, \mathcal{Q})$, task-level feature adaptor receives all the *support* instances as input and outputs a global mask, which is then applied to the whole task:

$$\mathbf{m}^{\text{task}} = f^{\text{task}}(\{\phi(\mathbf{x}_i) | \mathbf{x}_i \in \mathcal{S}\}) \tag{1}$$

Class-level feature adaptors generates local masks:

$$\mathbf{m}_n^{\text{cls}} = f^{\text{cls}}(\{\phi(\mathbf{x}_i) | \mathbf{x}_i \in \mathcal{S} \wedge y_i = n\}), n \in [N] \tag{2}$$

Mask combiner provides two smoothing parameters α^{task} and α^{cls} to balance the strengths of two masks:

$$[\alpha^{\text{task}}, \alpha^{\text{cls}}] = g(\{\phi(\mathbf{x}_i) | \mathbf{x}_i \in \mathcal{S}\}) \tag{3}$$

For each *support* instance $(\mathbf{x}_i, y_i) \in \mathcal{S}$, we compute its masked representation as Equation (4), which is then used to obtain the transformed class centres $\mathbf{e}_n$ in Equation (5).

$$\mathbf{z}_i = \phi(\mathbf{x}_i) \odot \left(\mathbf{1} + \frac{\mathbf{m}^{\text{task}}}{\alpha^{\text{task}}}\right) \odot \left(\mathbf{1} + \frac{\mathbf{m}_{y_i}^{\text{cls}}}{\alpha^{\text{cls}}}\right) \tag{4}$$

$$\mathbf{e}_n = \frac{1}{K} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{S} \wedge y_i = n} \mathbf{z}_i, n \in [N] \tag{5}$$

For each *query* instance $(\mathbf{x}_j, y_j) \in \mathcal{Q}$, we need to compute its distances to N masked class centres. Although we have no access to y_j, we can apply the global mask and the n-th local mask to it when comparing it to the n-th class center:

$$\mathbf{z}_j^n = \phi(\mathbf{x}_j) \odot \left(\mathbf{1} + \frac{\mathbf{m}^{\text{task}}}{\alpha^{\text{task}}}\right) \odot \left(\mathbf{1} + \frac{\mathbf{m}_n^{\text{cls}}}{\alpha^{\text{cls}}}\right) \tag{6}$$

The posterior probability of label is computed as Equation 7

$$p(\hat{y}_j = n | \mathbf{x}_j) = \frac{\exp\{-\text{dis}(\mathbf{z}_j^n, \mathbf{e}_n)\}}{\sum_{n'=1}^{N} \exp\{-\text{dis}(\mathbf{z}_j^{n'}, \mathbf{e}_{n'})\}} \tag{7}$$

Our main objective is to maximize log of label posterior probability on all *query* instances from sampled tasks.

$$\min_{\phi, \mathcal{F}, g} \sum_{\mathcal{T} \sim \mathcal{D}^{tr}} \sum_{(\mathbf{x}_j, y_j) \in \mathcal{Q}^{tr}} -\log p(\hat{y}_j = y_j | \mathbf{x}_j) \tag{8}$$

The whole flow of GLoFA is summarized in Algorithm 1.

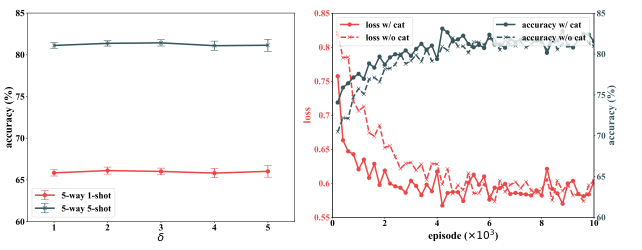
Discussion on Algorithm

In this section, we will focus on the following questions:

- How to set truncation hyper-parameter δ?
- Why to concatenate the output of $\mathbf{MLP}(\cdot)$ with $\phi(\mathbf{x})$ in $\mathcal{F}$ and $g(\cdot)$?
- What is the effect of pre-training embedding network?

Algorithm 1 Algorithm flow of GLoFA.

```
Require: Pre-trained embedding network $\phi(\cdot)$, randomly initialized feature adaptors $\mathcal{F} = \{f^{\text{task}}(\cdot), f^{\text{cls}}(\cdot)\}$, randomly initialized mask combiner $g(\cdot)$, meta-training set $\mathcal{D}^{tr}$, $N$, $K$, $M$.
  repeat
    Sample an $N$-way $K$-shot task $\mathcal{T} = (\mathcal{S}, \mathcal{Q})$ from $\mathcal{D}^{tr}$.
    Compute global mask $\mathbf{m}^{\text{task}}$ by Equation (1).
    Compute $N$ local masks $\mathbf{m}_n^{\text{cls}}, n \in [N]$ by Equation (2).
    Compute smoothing parameters $\alpha^{\text{task}}$ and $\alpha^{\text{cls}}$ by Equation (3).
    Get masked representations of support instances by Equation (4).
    Get masked representations of query instances by Equation (6).
    Compute query loss as Equation (8).
  until Convergence
```



(a) Influence of δ. (b) Effect of concatenation.

Figure 1: **(a)** Mean and standard deviation of accuracy for different δ values. The value of δ does not have a noticeable influence on model accuracy. Smaller δ tends to slightly reduce the variance of accuracy. **(b)** Convergence curves of GLoFA with (without) the residual operation. At the first few episodes, concatenating $\phi(\mathbf{x})$ with the output of the first $\mathbf{MLP}(\cdot)$ is beneficial to training.

Truncation Hyper-Parameter δ

In GLoFA, we assume that masks encode excess importance of each feature dimension and values in a mask are all positive. Thus, we project the masks into range $[0, \delta]$ by $h_\delta(\cdot) = \min(\delta, \max(0, \cdot))$. The reason why we introduce truncation hyper-parameter δ is computational. With assistance of this truncation hyper-parameter, all the mask values will be no larger than δ, preventing them from being infinity. Larger δ allows mask values to be in a larger range. But we find that absolute values of masks are not very influential to the model performance. In Figure 1a, we train our model on the *meta-training* set of *mini*ImageNet and report the accuracy on *meta-testing* set. Mean accuracy does not change a lot when δ varies from 1 to 5. With a smaller δ, the standard deviation of accuracy tends to be smaller. We set δ to 2 for all the experiments in this paper.

Residual Operation

In our concrete implementation of feature adaptors $\mathcal{F}$ and mask combiner $g(\cdot)$, the output of the first multi-layer linear network is concatenated with the raw representation $\phi(\mathbf{x})$. This can be seen as a kind of residual operation which is beneficial to training process (He et al. 2016; Xie et al. 2017). Feature adaptors $f^{\text{task}}(\cdot)$, $f^{\text{cls}}(\cdot)$ and mask combiner $g(\cdot)$ are all randomly initialized but the embedding network $\phi(\cdot)$ is pre-trained on the *meta-training* split. At the first few episodes, the outputs of $\mathcal{F}$ and $g(\cdot)$ are meaningless. Directly inputting $\phi(\mathbf{x})$ to the second multi-layer linear network provides more information and helps the model to converge faster. Figure 1b shows the convergence curves of GLoFA with (without) this residual operation. The model gets stable earlier with the assistance of this residual operation.

Pre-Training of $\phi(\cdot)$

As in many few-shot learning works (Li et al. 2019; Ye et al. 2020; Zhen et al. 2020), we pre-train our embedding network on the *meta-training* split using cross-entropy loss function. In Table 1, we show the accuracy of GLoFA on *mini*ImageNet and *tiered*ImageNet with (without) pre-training. We can see that pre-training significantly improves the model accuracy.

Experiment Details

We will provide detailed settings for all the experiments in this section. Our experiments contain three parts: heterogeneous tasks, benchmark evaluations, and further analyses.

Heterogeneous Tasks

Datasets. We construct a dataset mixed by 5 fine-grained classification sub-datasets, namely AirCraft (Maji et al. 2013), Car-196 (Krause et al. 2013), CUB-200-2011 (Wah et al. 2011), Stanford Dog (Khosla et al. 2011), and Indoor Scenes (Quattoni and Torralba 2009). All these fine-grained classification datasets are public. For each sub-dataset, we randomly select 20 classes from it, and then split the 20 classes into three parts: 10 classes for *meta-training*, 5 classes for *meta-validating*, and 5 classes for *meta-testing*. We randomly sample 100 images for each class. In total, there are 5000 images from 50 classes in *meta-training* set, 2500 images from 25 classes in *meta-validating* set, and 2500 images from 25 classes in *meta-testing* set. We will make this mixed dataset public once the paper is accepted.

Implementation details. For a 5-way K-shot task sampled from the mixed dataset, we define its granularity G as the number of sub-datasets involved in this task. When $G = 1$, all the classes in this task are all from a same sub-dataset, making the task extremely fine-grained. When $G = 5$, the task is very easy because there is a noticeable gap between each pair of classes. We take the commonly used ResNet-12 as the embedding network. The embedding network $\phi(\cdot)$ is pre-trained on the *meta-training* split of the mixed dataset. Other modules in GLoFA are randomly initialized. In *meta-training* phase, 40000 episodes are randomly sampled from the *meta-training* split. Due to the randomness in sampling, some of these tasks are coarse-grained and some are fine-grained. We optimize our model using SGD optimizer on these 40000 tasks. The momentum and

	*mini*ImageNet		*tiered*ImageNet	
	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
GLoFA w/ pre-training	**66.12 ± 0.42**	**81.37 ± 0.33**	**69.75 ± 0.33**	**83.58 ± 0.42**
GLoFA w/o pre-training	64.38 ± 0.39	80.54 ± 0.44	67.39 ± 0.28	82.70 ± 0.35

Table 1: Average test accuracies (%) with 95% confidence intervals on tasks sampled from *meta-testing* set of *mini*ImageNet and *tiered*ImageNet. Pre-training the embedding network on *meta-training* split significantly improves model accuracy.

weight decay of the optimizer are set to 0.9 and 0.0005 respectively. The initial learning rate for the pre-trained embedding network and other modules are set to 0.001 and 0.01 respectively. Two learning rates are decreased by 0.2 after 5000, 10000, 20000, and 30000 episodes. Truncation hyperparameter δ is set to 2. In *meta-testing* phase, we sample 600 episodes with G in $\{1, 2, 3, 4, 5\}$ from the *meta-testing* split. We compare our proposed method to three metric-based meta-learning algorithms: ProtoNet (Snell, Swersky, and Zemel 2017), TADAM (Oreshkin, López, and Lacoste 2018), and CTM (Li et al. 2019). We reimplement these methods using ResNet-12 backbone. The embedding networks in these compared methods are also pre-trained on the *meta-training* split.

Benchmark Evaluations

Datasets. In this part of experiment, we evaluate GLoFA on two widely used benchmark dataset, *i.e.*, *mini*ImageNet and *tiered*ImageNet. The *mini*ImageNet dataset was first proposed by (Vinyals et al. 2016) and it is a subset of ILSVRC-12 (Russakovsky et al. 2015). In this dataset, there are 100 classes and 600 images in each class. Each image in *mini*ImageNet is resized to 84×84. We follow (Ravi and Larochelle 2017) to split *mini*ImageNet, which means the total 100 classes are divided into *meta-training* set, *meta-validating* set, and *meta-testing* set, with 64, 16, and 20 classes respectively. The *tiered*ImageNet is a larger subset of ILSVRC-12. There are 608 classes and 779165 images in total. These classes are divided into 34 categories, with each category containing between 10 to 30 classes. The images in *tiered*ImageNet are also resized to 84×84. Following (Ren et al. 2018), we split *tiered*ImageNet into *meta-training*, *meta-validating* and *meta-testing* set, with 20, 6, and 8 categories respectively.

Implementation details. We use ResNet-12 as our embedding network. Number of *meta-training* episodes and *meta-testing* episodes are both 10000. We optimize our model using SGD optimizer on 10000 tasks. The momentum and weight decay of the optimizer are set to 0.9 and 0.0005 respectively. The initial learning rate for the pre-trained embedding network and other modules are set to 0.001 and 0.01 respectively. Two learning rates are decreased by 0.2 after every 2000 episodes. Truncation hyperparameter δ is set to 2. We also utilize data augmentation in *meta-training* phase. In detail, each image is randomly resized and cropped to 84×84, and then horizontally flipped with a probability 0.5. Finally, images are normalized with mean $[0.485, 0.456, 0.406]$ and standard deviation $[0.229, 0.224, 0.225]$. In *meta-testing* phase, we only center crop and normalize the images.

Further Analyses

In this part of experiment, we mainly focus on two questions. Firstly, does every single module in GLoFA improve the model performance? Secondly, can GLoFA successfully converge to a stable point? We conduct ablation study and convergence analysis on *mini*ImageNet. The detailed description about *mini*ImageNet dataset can be found in previous subsection. The implementation details are same as those in previous subsection.

References

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

Khosla, A.; Jayadevaprakash, N.; Yao, B.; and Li, F.-F. 2011. Novel Dataset for Fine-Grained Image Categorization: Stanford Dogs. In *24th IEEE Conference on Computer Vision and Pattern Recognition Workshop*, volume 2.

Koch, G.; Zemel, R.; and Salakhutdinov, R. 2015. Siamese Neural Networks for One-Shot Image Recognition. In *32nd International Conference on Machine Learning Workshop*, volume 2.

Krause, J.; Stark, M.; Deng, J.; and Fei-Fei, L. 2013. 3d Bbject Representations for Fine-Grained Categorization. In *14th International Conference on Computer Vision Workshop*, 554–561.

Li, H.; Eigen, D.; Dodge, S.; Zeiler, M.; and Wang, X. 2019. Finding Task-Relevant Features for Few-Shot Learning by Category Traversal. In *Proceedings of the 32nd IEEE Conference on Computer Vision and Pattern Recognition*, 1–10.

Maji, S.; Rahtu, E.; Kannala, J.; Blaschko, M.; and Vedaldi, A. 2013. Fine-Grained Visual Classification of Aircraft. *CoRR* abs/1306.5151.

Oreshkin, B.; López, P. R.; and Lacoste, A. 2018. Tadam: Task Dependent Adaptive Metric for Improved Few-Shot Learning. In *Advances in Neural Information Processing Systems 31*, 721–731.

Quattoni, A.; and Torralba, A. 2009. Recognizing Indoor Scenes. In *Proceedings of the 22nd IEEE Conference on Computer Vision and Pattern Recognition*, 413–420.

Ravi, S.; and Larochelle, H. 2017. Optimization as a Model for Few-Shot Learning. In *Proceedings of the 5th International Conference on Learning Representations*.

Ren, M.; Triantafillou, E.; Ravi, S.; Snell, J.; Swersky, K.; Tenenbaum, J. B.; Larochelle, H.; and Zemel, W. S. 2018. Meta-Learning for Semi-Supervised Few-Shot Classification. In *Proceedings of the 6th International Conference on Learning Representations*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* 115(3): 211–252.

Snell, J.; Swersky, K.; and Zemel, R. S. 2017. Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems 30*, 4077–4087.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Kavukcuoglu, K.; and Wierstra, D. 2016. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems 29*, 3630–3638.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The Caltech-UCSD Birds-200-2011 Dataset .

Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; and He, K. 2017. Aggregated Residual Transformations for Deep Neural Networks. In *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition*, 1492–1500.

Ye, H.-J.; Hu, H.; Zhan, D.-C.; and Sha, F. 2020. Few-Shot Learning via Embedding Adaptation With Set-to-Set Functions. In *Proceedings of the 33rd IEEE Conference on Computer Vision and Pattern Recognition*, 8808–8817.

Zhen, X.; Sun, H.; Du, Y.; Xu, J.; Yin, Y.; Shao, L.; and Snoek, C. 2020. Learning to Learn Kernels with Variational Random Features. In *Proceedings of the 37th International Conference on Machine Learning*, 11626–11636.