# Appendix for Tree Structure-Aware Few-Shot Image Classification via Hierarchical Aggregation

Min Zhang[124] Siteng Huang[24] Wenbin Li[3] Donglin Wang[24*]

[1]Zhejiang University [2]Westlake University
[3]State Key Laboratory for Novel Software Technology, Nanjing University
[4]Institute of Advanced Technology, Westlake Institute for Advanced Study
liwenbin@nju.edu.cn, {zhangmin,huangsiteng,wangdonglin}@westlake.edu.cn

## Appendix

The supplementary material is organized as:

1. The full algorithm for few-shot image classification with HTS is shown in Appendix A.1.
2. The formulations of meta-learning based FSL methods are shown in Appendix A.2.
3. Additional experimental setups are shown in Appendix A.3.
4. Additional experimental results for RQ2 (main paper) are shown in Appendix A.4.
5. Additional experimental results for RQ4 (main paper) are shown in Appendix A.5.

## A.1 Full HTS Algorithm

For easy reproduction, we present the full algorithm for few-shot image classification with HTS in Algorithm 1. Once trained, with the learned model parameters, we can perform the meta-testing phase over the test episodes with Eq. (9).

## A.2 Meta-Learning based FSL Models

In the main paper, we have introduced the ProtoNet [7] as the FSL model of the HTS in Sec. 3.2. Due to the flexibility of the HTS (*i.e.*, it can be combined with any meta-learning based FSL model), we also show the experimental results obtained by using other popular meta-learning based models in Sec. 5.3. Next, we detail these methods how work with the proposed HTS method.

**Graph Neural Network (GNN).** GNN [5,11,12,13] uses the posterior inference over a graphical model determined by the inputs and labels to formally propagate information. It constructs a fully-connected graph $G_{\mathcal{T}_e} = (V, E)$,

* Corresponding author

**Algorithm 1** Hierarchical Tree Structure-Aware Method

```
Require: The training set $\mathcal{D}_b$, the pretext-task operators $\mathcal{G}$
         The loss weight hyperparameter $\{\beta_j | j = 1, \cdots, J\}$
 1: Randomly initialize all learnable parameters $\{\phi, \theta^r | r = 0, \cdots, J\}$
 2: for iteration=1,$\cdots$, MaxIteration do
 3:     Randomly sample episode $\mathcal{T}_e$ from $\mathcal{D}_b$
 4:     Generate the set of augmented episodes $\mathcal{T}_{agg}$ from $\mathcal{T}_e$ using $\mathcal{G}$
 5:     Extract image features of $\mathcal{T}_e$ and $\mathcal{T}_{agg}$ using shared encoder $E_\phi$
 6:     // Hierarchical tree constructing component
 7:     The form of the tree $\{E_\phi(x_i) \xrightarrow{g_1} E_\phi(x_i^1) \xrightarrow{g_2} \cdots \xrightarrow{g_j} E_\phi(x_i^j) \cdots \xrightarrow{g_J} E_\phi(x_i^J)\}$ is used
        to construct tree structure
 8:     // Gated selection aggregating component
 9:     The learning process of the tree $\{h_i^0 \xleftarrow{agg} h_i^1 \xleftarrow{agg} \cdots \xleftarrow{agg} h_i^r \cdots \xleftarrow{agg} E_\phi(x_i^J)\}$ is
        used to hierarchically aggregate node information
10:     // Meta-training phase
11:     if Data augmentation setting then
12:         Update the FSL model using Eq. (7);
13:     else if Self-supervised learning setting then
14:         Update the FSL model using Eq. (8);
15:     end if
16: end for
```

where nodes $v_n \in V$ correspond to the images present in each episode $\mathcal{T}_e = \{\mathcal{S}_e, \mathcal{Q}_e\}$. $e_{n,n'} \in E$ represents the similarity of image $x_n$ and $x_{n'}$ and uses a parametric model (*i.e.* MLP) to learn. GNN contains a feature encoder $E_\phi$ with learnable parameters $\phi$ (*e.g.*, CNN) to extract the image features. Then these features are used to calculate the similarities between images using an MLP network. Over the weighted graph $G$, a GNN is used to propagate and update the node information. Finally, updated feature representations of query images are input a classifier to predict the labels. The training process of GNN is shown as:

$$\begin{aligned} L_{FSL}(\mathcal{S}_e, \mathcal{Q}_e) &= \frac{1}{|\mathcal{Q}_e|} \sum\nolimits_{(x_i, y_i \in \mathcal{Q}_e)} -log\ p_{y_i}, \\ p_{y_i} &= GNN(MLP(E_\phi(\mathcal{S}_e), E_\phi(x_i))). \end{aligned} \tag{A.1}$$

GNN adopts the message propagating process, which iteratively aggregates the neighborhood information (*i.e.*, support labeled information). Formally, the aggregation and propagation process of the $k$-th layer in GNN is two steps:

$$\begin{aligned} m_{k,n} &= AGGREGATE(\{a_{k-1,u} : u \in \mathcal{N}_{(n)}\}), \\ a_{k,n} &= UPDATE(a_{k-1}, m_{k,n}, a_{0,n}), \end{aligned} \tag{A.2}$$

where $\mathcal{N}_n$ is the set of neighbors of node $v_n$, $a_{0,n}$ is initial features and AGGREGATE is a permutation invariant function. After K message-passing layers, the final node embeddings $A_K$ are used to infer the query image label. GNN combined with the proposed HTS method only need to replace $L_{FSL}$ (Eq. (7) - Eq. (9) in the main paper) with Eq. (A.1).

**Matching Network (MatchingNet).** MatchingNet [9] contains a feature encoder $E_\phi$ and a attention or memory network. In each episode $\mathcal{T}_e = \{\mathcal{S}_e, \mathcal{Q}_e\}$, MatchingNet labels each query image as a consine distance-weighted linear combination of the support labels and uses a attention mechanism to calculate. The training process of MatchingNet is shown as:

$$\begin{aligned} L_{FSL}(\mathcal{S}_e, \mathcal{Q}_e) &= \frac{1}{|\mathcal{Q}_e|} \sum\nolimits_{(x_i, y_i \in \mathcal{Q}_e)} -log\ p_{y_i}, \\ p_{y_i} &= \sum\nolimits_{\hat{x}_i, \hat{y}_i \in \mathcal{S}_e} a(x_i, \hat{x}_i) \cdot \mathbb{I}(\hat{y}_i = y_i), \end{aligned} \tag{A.3}$$

where $a(\cdot,\cdot)$ represents an attention mechanism and it fully specifies the classifier. $\mathbb{I}$ represents the indicator function with its output being 1 if the input is true or 0 otherwise. Our proposed HTS method with MatchingNet only need to replace $L_{FSL}$ (Eq. (7) - Eq. (9) in the main paper) with Eq. (A.3).

**Relation Network (RelationNet).** RelationNet [8] is comprised of a feature encoder $E_\phi$ as usual, and a relation module $R_\omega$ parameterized by $\omega$. They first embed each support and query using $E_\phi$ and create a prototype $\tilde{p}_c$ for each class $c$ by averaging its support embeddings, as shown in Eq.(1) in the main paper. Each prototype $\tilde{p}_c$ is concatenated with each embedded query and fed through the relation module which outputs a number in $[0, 1]$ representing the predicted probability that that query belongs to class $c$. The query loss is then defined as the mean square error of that prediction compared to the (binary) ground truth. Both the encoder and the relation module are trained to minimize this loss.

$$\begin{aligned} L_{FSL}(\mathcal{S}_e, \mathcal{Q}_e) &= \frac{1}{|\mathcal{Q}_e|} \sum\nolimits_{(x_i, y_i \in \mathcal{Q}_e)} -log\ p_{y_i}, \\ p_{y_i} &= \sum\nolimits_{c \in \mathcal{C}_e} (r_{c,i} - \mathbb{I}(y_i = y_c))^2, \\ r_{c,i} &= R_\omega(Ca(\tilde{p}_c, E_\phi(x_i))), \end{aligned} \tag{A.4}$$

where $Ca(\cdot,\cdot)$ is a concatenated operator. The proposed HTS with MatchingNet only need to replace $L_{FSL}$ (Eq. (7) - Eq. (9) in the main paper) with Eq. (A.4).

## A.3 Additional Experimental Setup

**Benchmark Datasets.** Four benchmark few-shot learning datasets are used to evaluate our HTS: (1) ***mini*ImageNet** contains 600 images per class over 100 classes. These classes are divided into 64/16/20 for train/val/test [9]. (2) ***tiered*ImageNet** is much larger compared with *mini*ImageNet with 608 classes and each class about 1,300 samples. These classes were grouped into 34 higher-level concepts and then partitioned into 20/6/8 disjoint sets for train/val/test to achieve a larger domain difference between training and testing phases [4]. (3) **CUB-200-2011** is initially designed for fine-grained classification. It contains 200 classes and each class has around 60 samples. We divided these classes into 100/50/50 for train/val/test following the previous works [10] . (4) **CIFAR-FS** is a subset of CIFAR-100 and has 600 images per class over 100 classes. The train/val/test classes are same to *mini*ImageNet [1]. We also list the images number, classes number, images resolution and train/val/test splits following the criteria of previous works in Tab. A.1.

| Datasets | Images | Classes | Train/Val/Test | Resolution |
|---|---|---|---|---|
| *mini*ImageNet | 60000 | 100 | 64/16/20 | 84×84 |
| *tiered*ImageNet | 779165 | 608 | 351/97/160 | 84×84 |
| CUB-200-2011 | 11788 | 200 | 100/50/50 | 84×84 |
| CIFAR-FS | 60000 | 100 | 64/16/20 | 32×32 |

**Table A.1.** The details for four benchmark datasets.

**Implementation Details.** Following [2], we apply Conv4 and ResNet12 as the convolution backbone networks to fairly compare the results. The Conv4 consists of a stack of four Conv-BN-ReLU convolutional blocks and each convolutional block with 64 filters. ResNet12 mainly has four blocks, which include one residual block. The last features of all backbone networks are processed by a global average pooling, then followed by a fully-connected layer with batch normalization [3] to obtain a 64-dimensions instance embedding. We use Adam optimizer with an initial learning rate of $10^{-3}$, and reduce the learning rate by 15K episodes of all datasets. The weight decay is set to $5e^{-4}$.

## A.4 Additional Experiments for RQ2

In this section, we report additional experiments to further prove the effectiveness of our proposed HTS method.

**Visualization of CAM.** To evaluate whether a model exploits an important or actual object when making a prediction, Fig. A.1 visualizes the gradient-weighted class activation mapping (Grad-CAM) [6] from ProtoNet and HTS under a Conv4-64 feature encoder. It is observed that using HTS makes the model pay more attention to the local object features than the ProtoNet. It also further shows that the hierarchical aggregated representations can avoid semantic bias caused by pretext tasks. Therefore, the proposed HTS helps the meta-learning based methods to use the correct visual features for prediction.

**Performance on cross domain.** In Fig. A.2, we give the cross-domain performance of four datasets as the training and testing domains respectively. To clearly show the performance, we use the same color bar with the same number of shots based on baseline and our method. From Fig. A.2, our method outperforms the ProtoNet under all domain settings including singe domain (diagonal line) and cross domain (off-diagonal line). This indicates that our HTS can improve the generalization ability of the FSL model with only a few labeled images.

## A.5 Additional Experimental for RQ4

More experiment results are shown under different pretext tasks with the same child nodes to supplement RQ4.

**The effect of different pretext tasks.** In the main paper, we have reported the results of *mini*ImageNet and CUB-200-2011 with rotation and color permutation tasks under different pretext tasks. Using the same setting with Fig. 6 (a) and (b), we show the results of *tiered*ImageNet and CIFAR-FS datasets in Fig. A.3. From (a)-(d), we find that pretext tasks can bring semantic structure information and improve the generalization ability on all classification datasets.

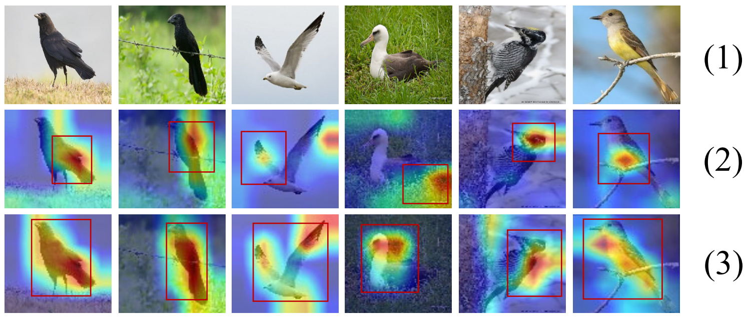

**Fig. A.1.** Grad-CAM visualizations of CUB-200-2011 dataset. Each column shows the result of same test image, and each row shows: (1) The results of test images. (2) The results of ProtoNet. (3) The results of our HTS. Best viewed in color and our proposed HTS method achieves better results for all test images.

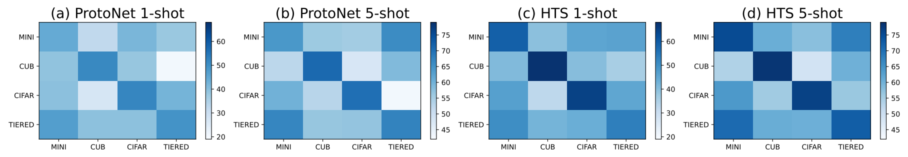

**Fig. A.2.** Cross-domain evaluation with **rotation3** under 5-way 1-shot and 5-shot settings. The horizontal axis represents training domains and the vertical axis is testing domains on miniImagenet (MINI), CUB-200-2011 (CUB), CIFAR-FS (CIFAR) and tieredImagenet (TIERED). To clearly show the performance, we use the same color bar with the same number of shots under baseline and our method.

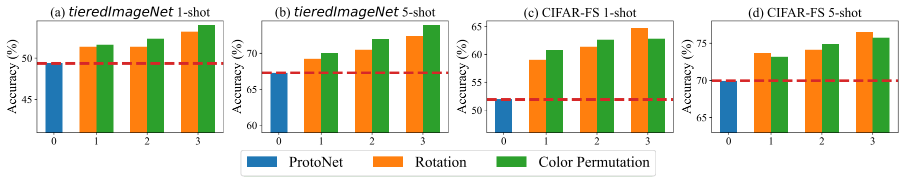

**Fig. A.3.** (a)-(d) indiccate different pretext tasks with the same number of child nodes, where (a) and (b) represent *tiered*ImageNet under 5-way 1-shot and 5-shot settings. (c) and (d) represent CIFAR-FS under 5-way 1-shot and 5-shot settings. Red dotted lines represent the performance of baseline (ProtoNet).

## References

1. Bertinetto, L., Henriques, J.F., Torr, P.H.S., Vedaldi, A.: Meta-learning with differentiable closed-form solvers. In: International Conference on Learning Representations, ICLR (2019)
2. Huang, S., Zhang, M., Kang, Y., Wang, D.: Attributes-guided and pure-visual attention alignment for few-shot recognition. In: Association for the Advancement of Artificial Intelligence, AAAI (2021)
3. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: ICML. pp. 448–456. PMLR (2015)
4. Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J.B., Larochelle, H., Zemel, R.S.: Meta-learning for semi-supervised few-shot classification. In: International Conference on Learning Representations, ICLR (2018)
5. Satorras, V.G., Estrach, J.B.: Few-shot learning with graph neural networks. In: International Conference on Learning Representations, ICLR (2018)
6. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: International Conference on Computer Vision , ICCV (2017)
7. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems, NeurIPS (2017)
8. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H.S., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2018)
9. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: Advances in Neural Information Processing Systems, NeurIPS (2016)
10. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. California Institute of Technology (CNS-TR-2011-001) (2011)
11. Zhao, F., Huang, T., Wang, D.: Graph few-shot learning via restructuring task graph. IEEE Transactions on Neural Networks and Learning Systems (2022)
12. Zhao, F., Wang, D.: Multimodal graph meta contrastive learning. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 3657–3661 (2021)
13. Zhao, F., Wang, D., Xiang, X.: Multi-initialization graph meta-learning for node classification. In: Proceedings of the 2021 International Conference on Multimedia Retrieval. pp. 402–410 (2021)