GSS: GRAPH-BASED SUBSPACE LEARNING WITH SHOTS INITIALIZATION FOR FEW-SHOT RECOGNITION

Rui-Qi Wang[*†], *Xu-Yao Zhang*[*†], *and Cheng-Lin Liu*[*†‡]

[*]National Laboratory of Pattern Recognition (NLPR),
Institute of Automation of Chinese Academy of Sciences, Beijing 100190, China
[†]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
[‡]CAS Center for Excellence of Brain Science and Intelligence Technology, Beijing 100190, China
Email: {ruiqi.wang, xyz, liucl}@nlpr.ia.ac.cn

2021 IEEE International Conference on Multimedia and Expo (ICME) | 978-1-6654-3864-3/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICME51207.2021.9428122

ABSTRACT

Previous methods of few-shot Learning mostly solve different few-shot recognition tasks in an identical feature space. But identical features are hard to fit various tasks. Some works show that learning a unique subspace for each few-shot recognition task can improve the signal-noise ratio (SNR) of the features and boost the performance. However, there are still two problems remaining. First, in constructing the subspace for few-shot task, often some information (embeddings of queries or labels of shots) are discarded. Second, the eigendecomposition of covariance matrix is usually needed, which degrades the efficiency of the whole model. In this paper, we propose Graph-based Subspace learning with Shots initialization (GSS) for few-shot recognition to learn a better subspace efficiently. In GSS, the bases of the subspace are directly initialized with labels based on shots (given labeled samples) and iteratively updated for better discrimination based on a graph that connects bases and all samples. Extensive experiments on four few-shot benchmark datasets show that GSS reports better performance and higher efficient compared with previous subspace based methods and achieves state-of-the-art performance.

Index Terms— Few-shot learning, subspace learning

1. INTRODUCTION

The success of deep neural networks have promoted the research of many computer vision tasks tremendously [1, 2, 3]. However, it is notorious that the performance of DNN usually relies on a large amount of labeled data [4]. The generalization ability of learning from a few samples is still far behind real intelligence level of humans. This gap has triggered wide interests in research community as the problem of few-shot learning (FSL) [5, 6, 7].

To mimic the ability of efficient learning in human brain, FSL takes rich labeled data of *base* classes to learn the "prior knowledge", based on which the classification ability on *novel* classes can be obtained with only a few (e.g. 1 or 5 samples for each novel class) labeled samples. In FSL, the training set contains samples of *base* classes, and the testing set contains samples of *novel* classes that are unseen during training. To evaluate the trained model, the testing set is sampled into many few-shot tasks, each of which consists of k labeled samples for each of the n sampled novel classes (called n-way k-shot) and some queries to be classified. The performance is reported as the mean accuracy and the 95% confidence interval. Usually, the training set is also sampled into many training few-shot tasks to optimize the model [5, 8, 9].

Recently, subspace learning based methods for few-shot learning [6, 7] draws increasing attention. Because, with subspace learning, features with higher SNR (signal-noise ratio) can be extracted for each specific few-shot task [6]. However, there are still two problems remaining in previous subspace based methods that limit the performance and efficiency. First, they do not fully exploit available information of a few-shot task to construct subspaces. For example, TAFSSL [6] discards the labels of shots, and Adaptive Subspace [7] abandons the queries. Second, they involve the eigendecomposition of covariance matrix, which decreases the execution efficiency of the whole model.

Thus, we propose Graph-based Subspace learning with Shots initialization (GSS) to learn better subspaces fast for few-shot recognition tasks. In GSS, the bases of subspace are initialized with labels based on shots. Then, the representations are updated iteratively based on a graph that relates data points and bases. The relations (similarities) are computed in the low-dimensional subspace and used to update the representations in the high-dimensional space, because the subspace has higher SNR (signal-noise ratio) [6] and the original space has richer representation. Based on the updated representations, a new generation of subspace can be learned. After a few cycles of updates, the queries are finally classified in the last subspace. GSS fully exploits the information of few-shot tasks and avoids finding eigenvectors of matrices, thus boosts the performance with less computational overhead. In extensive experiments, GSS reports better performance and higher efficiency compared with previous methods based on subspace. Our contribution can be summarized as follows:

- We propose Graph-based Subspace learning with Shots initialization (GSS) to learn a unique subspace for each few-shot task using all available information to boost the performance.
- GSS iteratively learns the subspace with shots initialization and avoids the eigendecomposition of covariance matrix, thus boosts the efficiency of the whole model.
- Extensive experiments on four popular benchmark datasets demonstrate that GSS achieves high performance and efficiency.

2. RELATED WORK

Previous works mainly attack FSL from the view of meta-learning and metric learning. Recently, graph based methods also draw some attention. Meta-learning based methods [8, 10] focus on adapting the trained model to different few-shot tasks efficiently based on the support set. Metric learning based methods [11, 5, 12, 13] finds better similarity metrics for image embeddings. Graph based methods [14, 9] construct a complete graph that connects all samples of a few-shot task to perform information propagation and the queries are classified based on their relations to shots. Besides, there are also image retrieval based works [15] and reinforcement learning based ones [16].

However, most of them tackle different few-shot tasks in an identical feature space. The cross-used feature representation in different few-shot tasks introduces problems in both training and testing. Some features provide discriminant for a few-shot task but serve as noise in other tasks. Thus, the variances of these features are optimized to be larger for classification in some tasks but also smaller in other tasks to reduce in-class divergence, which brings unnecessary conflict. Further, in the testing scenario, the learned features are directly applied to novel classes with much noise.

Therefore, subspace learning draws more and more attention in the few-shot learning scenario [6, 7]. TAFSSL [6] shows that the signal-noise ratio (SNR) of the features can be increased when they are mapped into a subspace. Adaptive Subspace[7] construct a subspace for each class and the queries are classified based on the distances to their projections on each class subspace. However, they both fail to fully exploit the available information. TAFSSL [6] only uses samples, discarding labels of shots, and Adaptive Subspace [7] only uses shots with their labels, discarding the queries. Further, they both require finding eigenvectors of the covariance matrix to construct subspaces, which severely slows down the model. In contrast, we propose Graph-based Subspace learning with Shots initialization (GSS) that uses all samples and the labels of shots to learn the subspace in an iterative manner that avoids computing the covariance matrix.

3. METHOD

In this section, we first present some preliminaries for few-shot learning. Then, the proposed Graph-based Subspace learning with Shots initialization (GSS) is introduced.

3.1. Problem Definition

Given a few-shot task $\mathcal{T}$:

$$\begin{aligned} &\mathcal{T} = \left\{ \mathcal{S} = \{(x_s, y_s)\}_{s=1}^{N\times K} \right\} \cup \left\{ \mathcal{Q} = \{(x_q)\}_{q=1}^{N\times Q} \right\}, \\ &y_s \in \{1, \cdots, N\}, \mathcal{S} \cap \mathcal{Q} = \varnothing. \end{aligned} \tag{1}$$

where x is the raw instance and y is the given label. The support set $\mathcal{S}$ contains K (e.g. 1 or 5) labeled samples (shots) for each of N targeted classes. The query set $\mathcal{Q}$ contains many queries from the N classes to be classified. If the support set contains K labeled samples for each of N novel classes, the few-shot task is named N-way K-shot.

In FSL, the testing set $\mathcal{D}_{test}$ is sampled into many few-shot tasks $\mathcal{T}_{test}$ to evaluate the method. To ensure that the testing tasks are agnostic during training, the training set $\mathcal{D}_{train}$ and the testing set $\mathcal{D}_{test}$ contain different classes:

$$\begin{aligned} &\mathcal{D}_{train} = \{(x_i, y_i)\}, \qquad \mathcal{D}_{test} = \{(x_j, y_j)\}. \\ &y_i \in \{1, \cdots, C_{train}\}, \\ &y_j \in \{C_{train}+1, \cdots, C_{train}+C_{test}\}. \end{aligned} \tag{2}$$

The C_{train} classes in the training set are called *base* classes, and the C_{test} classes in the testing set are called *novel* classes. Usually, the training set $\mathcal{D}_{train}$ is also sampled into many few-shot tasks to train the model [5, 8, 9].

3.2. Graph-based Subspace Learning with Shots Initialization

The framework of the proposed GSS is shown in Fig. 1. Different from previous works, GSS exploits all available information of a few-shot task and the subspace is iteratively updated with bases initialized by shots, without computing covariance matrix.

Technically, all raw images are first embedded by an encoder f to obtain the representation in the high-dimensional space that provides general features. Then, the embeddings of shots are used to initialize bases with labels as shown in Fig. 2, with which all samples can be mapped into the subspaces according to their similarity to the bases. Relations are computed in the low-dimensional subspace with higher SNR (signal-noise ratio) [6] and used to update the representations in the high-dimensional space based on a graph shown

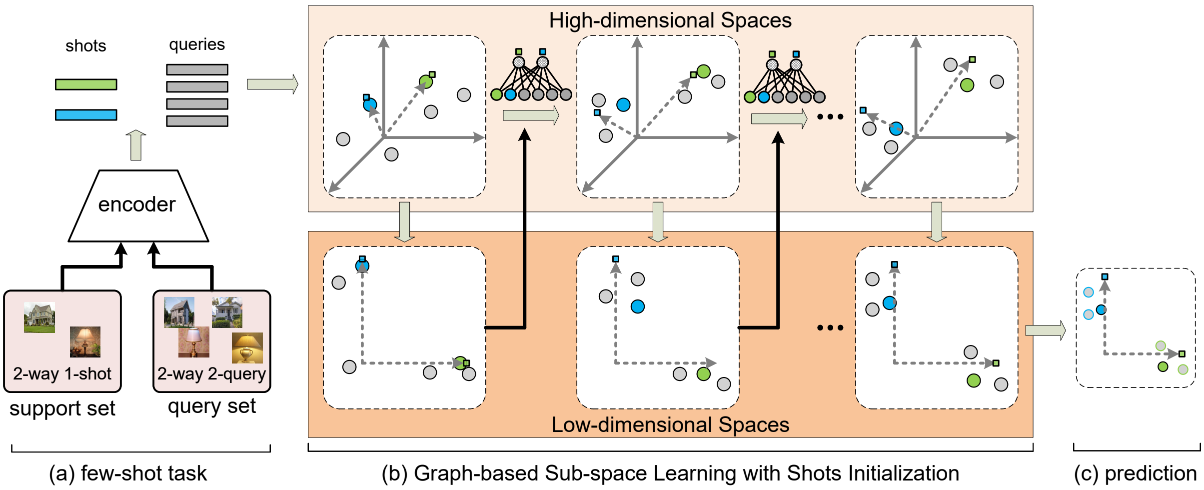

Fig. 1: The framework of the Graph-based Subspace learning with Shots initialization (GSS). The encoder f embeds raw images into the high-dimensional spaces. With bases initialized with labels by shots, all samples are mapped into the low-dimensional space. With the relations computed in the low-dimensional space, embeddings in the high-dimensional space are updated. After a few steps of iterative updates, the queries are classified. **Better viewed in color.**

in Fig. 3. With new representations in the high-dimensional space, the "mapping-update" process can be continued alternatively. Finally, the classes of queries are inferred in the low-dimensional spaces according to their relations with shots. In the following, we map the samples into the low-dimensional subspace and update the high-dimensional space.

Mapping to the Low-dimensional subspace. To map samples from high-dimensional space into a low-dimensional subspace, we need to find a group of bases. In traditional subspace learning methods such as PCA, the bases are found by eigendecomposition of the covariance matrix, which requires a large amount of computation. Differently, in the proposed GSS, the bases are directly initialized based on the embeddings of shots grouped by their classes:

$$\mathcal{B}^1 = \left\{(b_1^1, y_1), \cdots, (b_m^1, y_m)\right\} = g\left(\mathcal{S}; f\right). \tag{3}$$

where $\mathcal{B}^1$ notes the bases of the 1-generation of learned subspace. b_m^l is the m-th basis of the l-generation of subspace. f is the encoder and g is the function that generates bases. As shown in Fig. 2, we generate bases with labels based on shots. In this paper, we simply duplicate shots as the initialization of bases. With the bases $\mathcal{B}$, the all embeddings can be mapped into the low-dimensional subspace:

$$\begin{aligned} &z^l = \left[h(x^l, b_1^l), \cdots, h(x^l, b_m^l)\right] \in \mathbb{R}^m, \\ &b_m^l \in \mathcal{B}^l, \quad x^l \in \mathcal{S}^l \cup \mathcal{Q}^l \cup \mathcal{B}^l. \end{aligned} \tag{4}$$

where z^l notes the embedding in the l-th low-dimensional subspace and x^l notes the embedding in the l-th generation of high-dimensional space. b_m^l is the m-th basis for the l-th generation of subspace. h is the function to calculate the similarity between two embeddings, which is implemented as a three-layer MLP (Multilayer Perceptron) in this paper. m is the dimensionality of the low-dimensional space. $\mathcal{S}^l$, $\mathcal{Q}^l$, and $\mathcal{B}^l$ are the sets of shots embeddings, query embeddings, and bases in the l-th generation high-dimensional space.

Updating the High-dimensional Space with Relations Computed in Low-dimensional Space. The embeddings in the high-dimensional space are updated according to the relations computed in the low-dimensional subspace. The relations are computed as:

$$\begin{aligned} &e_{ij}^l = h'(z_i^l, z_j^l) \cdot \frac{e_{ij}^{l-1}}{\sum_k ||e_{ik}^l||}, \quad l > 1. \\ &z_i^l \in \mathcal{S}_z^l \cup \mathcal{Q}_z^l, \quad z_j^l \in \mathcal{B}_z^l. \end{aligned} \tag{5}$$

where e_{ij}^l is the edge information between two embeddings. h' is the function to measure the similarity between two embeddings in the low-dimensional space, which is implemented as a three-layer MLP in this paper. e_{ij}^0 is initialized with 1 if $y_i = y_j$ and $1/(N \cdot m)$ otherwise. $\mathcal{S}_z^l$, $\mathcal{Q}_z^l$, and $\mathcal{B}_z^l$ are the sets of shots embeddings, query embeddings, and bases in the l-th generation low-dimensional subspace. With the edges e_{ij} computed in the low-dimensional subspace, the embeddings in the high-dimensional space are updated as follows:

$$\begin{aligned} &x_i^{l+1} = u(\sum_{j=1}(\frac{e_{ij}^l}{\sum_{k=1} ||e_{ik}^l||} \cdot x_j^l), x_i^l). \\ &x_i^l \in \mathcal{S}^l \cup \mathcal{Q}^l \cup \mathcal{B}^l. \end{aligned} \tag{6}$$

where u is the function that updates the embedding based on gathered information from the graph, which is implemented as an MLP in this paper. The graph has edges only between bases and samples as shown in Fig. 3, which means that

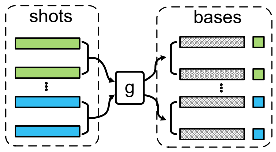

Fig. 2: Initialization of bases.

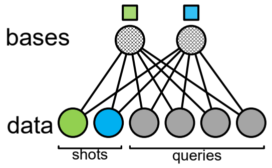

Fig. 3: The graph to update bases and embeddings.

the bases gather information from samples and the samples gather information from bases. With updated embeddings, a new generation of subspace can be learned based on Equation 4, and the mapping-update process can be performed alternatively.

Query Classification and Training Objective. For the classification of queries, we adopt the widely used similarity based method [5, 12, 13]. Technically, the posterior probability is derived according to the distances between the query and shots of each class:

$$p(\hat{y}_i|x_i) = softmax(\sum_{j=1}^{N\times K} d_{ij}^l \cdot one\_hot(y_j)), \quad x_i \in \mathcal{Q}_z^l, \qquad x_j \in \mathcal{S}_z^l. \tag{7}$$

where $p(\hat{y}_i|x_i)$ is the estimated probability distribution of query x_i over given classes. d_{ij}^l means the distance between z_i^l and z_j^l. one_hot is the function to convert label y into one-hot encoding.

The whole model is trained from scratch according to two losses. The fist is the query classification loss defined as follows:

$$L_c = CE(p(\hat{y}_i|x_i), y_i), \quad x_i \in \mathcal{Q}_z^l. \tag{8}$$

where CE notes the widely used cross-entropy loss function. The other loss measures the inferred relations between embeddings, which is defined as follows:

$$L_s = BCE(e_{ij}, \delta(y_i - y_j)). \quad \delta(t) = \begin{cases} 1, & t = 0, \\ 0, & others. \end{cases} \tag{9}$$

where e_{ij} is the edge between x_i and x_j (if the edge exists) and y_i and y_j are the corresponding labels. δ is the sampling function. And BCE notes the binary cross entropy loss.

4. EXPERIMENTS

In this section, the implementation details and datasets are first presented. Then, GSS is compared with previous methods on popular benchmark datasets. Further, the ablation study is discussed. All experiments are implemented with Pytorch [17] on TITAN RTX GPUs.

4.1. Implementation Details and Dataset

For fair comparisons with the existing methods, we use the convolutional part of ResNet-12 as the feature encoder, which is widely used in previous works [12, 13, 9]. The functions (h, h', u) used in GSS are all implemented as MLPs. We adopt 6 generations of updates to find the final subspace in the experiments if not specified. Our model is tested with 5-way 5-shot tasks and 5-way 1-shot tasks on four datasets. The model is trained with few-shot tasks with the same setting as testing tasks. The stochastic gradient descent (SGD) with an initial learning rate 0.1 is used to optimize the parameters. For more details, open source code [1] is available.

We conduct experiments of few-shot learning on four popular benchmark datasets: mini-ImageNet [11], tiered-ImageNet [18], CIFAR-FS [19], and FC100 [12].

Mini-ImageNet is the most popular benchmark dataset for FSL. It consists of a training set of 64 classes, a test set of another 20 classes, and a validation set of 16 classes. Each class contains 600 color images from ImageNet [20].

Tiered-ImageNet is similar to mini-ImageNet but with much more data and two-level class labels. It consists of a training set of 351 classes from 20 high-level classes, a testing set of 160 classes from 8 high-level classes, and a validation set of 97 classes from 6 high-level classes. The dataset contains more than 700K color images in total but not balanced between classes.

CIFAR-FS is a few-shot dataset based on CIFAR-100 [21]. It consists of 100 classes with 600 images for each and covers 3 spits: 64 classes for training, 20 classes for testing, and 16 classes for validation.

FC100 contains 60 classes of 12 high-level classes for training, 20 classes of 4 high-level classes for testing, and 20 classes of 4 high-level classes for validation. Each class contains 100 color images from CIFAR-100.

4.2. Main Results

We report the experimental results on mini-ImageNet and tiered-ImageNet in Table 1. Results on CIFAR-FS and FC100 are reported in Table 2. These results demonstrate that the proposed GSS achieves competitive performance compared with the previous state-of-the-art methods. Further, on the CIFAR-FS dataset, our method achieves the new state-of-the-art with a noticeable margin. In these results, GSS leads more in 5-shot tasks. We think this is a result of transductive learning, which also appears in the results of DPGN [9]. All other compared methods are inductive methods.

4.3. Ablation Studies

Comparison with other subspace learning based methods. In the ablation studies, we compare the proposed GSS with

[1] https://github.com/RuiqiWang95/GSS

Table 1: Results on ImageNet derivatives. Performances are reported with 95% confidence interval.

Method	Publication	Backbone	mini-ImageNet		tiered-ImageNet	
			5-shot	1-shot	5-shot	1-shot
PN [5]	NeurIPS 17'	ResNet-12	76.25±0.34	59.29±0.53	79.04±0.39	61.74±0.56
TADAM [22]	NeurIPS 18'	ResNet-12	76.70±0.30	58.50±0.30	-	-
MetaOptNet [12]	CVPR 19'	ResNet-12	78.63±0.46	62.46±0.61	81.56±0.53	65.99±0.72
Adaptive Subspace [7]	CVPR 20'	ResNet-12	81.65±0.69	67.09±0.68	83.32±0.66	68.44±0.77
DeepEMD [13]	CVPR 20'	ResNet-12	82.41±0.56	65.91±0.82	86.03±0.58	71.16±0.87
DPGN [9]	CVPR 20'	ResNet-12	**84.60±0.43**	67.77±0.32	**87.24±0.39**	72.45±0.51
Negative Margin [23]	ECCV 20'	ResNet-12	81.57±0.56	63.58±0.81	-	-
Distill [24]	ECCV 20'	ResNet-12	82.14±0.43	64.82±0.60	86.03±0.49	71.52±0.69
GSS	Ours	ResNet-12	83.99±0.43	**67.83±0.32**	86.59±0.38	**72.54±0.49**

Table 2: Results on CIFAR derivatives. Performances are reported with 95% confidence interval.

Method	Publication	Backbone	CIFAR-FS		FC100	
			5-shot	1-shot	5-shot	1-shot
PN [5]	NeurIPS 17'	ResNet-12	83.52±0.35	72.21±0.61	52.50±0.40	37.56±0.61
TADAM [22]	NeurIPS 18'	ResNet-12	-	-	56.1±0.4	40.1±0.4
MetaOptNet [12]	CVPR 19'	ResNet-12	84.2±0.5	72.0±0.7	55.5±0.6	41.1±0.6
Adaptive Subspace [7]	CVPR 20'	ResNet-12	87.3±0.6	**78.0±0.9**	-	-
DeepEMD [13]	CVPR 20'	ResNet-12	88.16±0.50	74.97±0.82	63.22±0.71	46.47±0.78
DPGN [9]	CVPR 20'	ResNet-12	89.2±0.4	77.9±0.5	-	-
Distill [24]	ECCV 20'	ResNet-12	86.9±0.5	73.9±0.8	60.9±0.6	44.6±0.7
GSS	Ours	ResNet-12	**91.55±0.33**	77.83±0.48	**64.13±0.56**	**47.32±0.63**

Table 3: Comparison with PCA and Adaptive Subspace [7]. Performances are reported with 95% confidence interval based on CIFAR-FS dataset 5-way 5-shot tasks.

Method	Performance	training time	testing time
PCA	86.73±0.40	3.9×	2.9×
Ada-Sub [7]	87.3±0.6	2.1×	1.3×
GSS (ours)	91.37±0.37	1×	1×

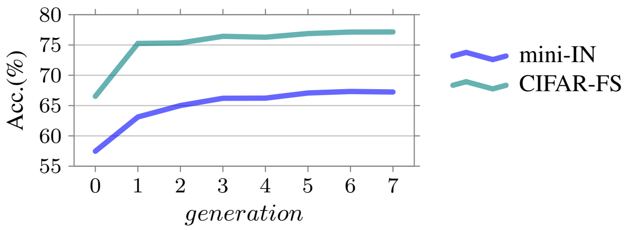

Fig. 4: Comparison of different generations. Performances are reported based on mini-ImageNet (mini-IN) and CIFAR-FS 5-way 1-shot tasks.

Adaptive subspace [7] and PCA as the traditional subspace learning methods on performance and time. The experiment settings remain the same for different methods. We adopt Pytorch [17] implementation of PCA and open-source code [2] of Adaptive subspace [7]. Results in Table 3 demonstrates that the proposed GSS achieves better performance with a margin of 4% compared with PCA and Adaptive subspace [7]. Further, GSS shows approximately 4 and 3 times higher efficiency during training and testing compared with PCA. It also shows 2 and 1.3 times higher efficiency compared with Adaptive subspace [7].

Update generations. Further, we analyze the performances with different generations to study the effect of the subspace update. 0 generation notes the performance of the original space. Results are visualized as a line chart in Fig. 4. The experimental results show that more generations does improve the performance and the first 3 generations provide the most significant performance gain, which validates the effect of the alternative update. The best performance is achieved at the 6-th generation.

5. CONCLUSION

In this paper, we propose a novel Graph-based Subspace learning with Shots initialization (GSS) for few-shot recognition to solve each few-shot task in a unique feature space. Different from past subspace based methods, GSS fully exploits information of a few-shot task and avoids computing eigenvector to find the subspace in an iterative manner, which boosts both the performance and the efficiency. Technically, the bases for the subspace is initialized based on the few la-

[2] https://github.com/chrysts/dsn_fewshot

beled samples. Then, based on the relations described in the low-dimensional subspace, the presentations in the high-dimensional space is updated. Thus, new subspace can be constructed based on updated representations, which makes the mapping-update process can be performed alternatively. Experimental results show that GSS achieves better performance with high efficiency compared with other methods. In the future, we will study more advanced subspace learning method for few-shot recognition.

6. ACKNOWLEDGEMENTS

This work has been supported by the Major Project for New Generation of AI under Grant No. 2018AAA0100400, the National Natural Science Foundation of China (NSFC) grants U20A20223, 61633021, 62076236, 61721004, the Key Research Program of Frontier Sciences of CAS under Grant ZDBS-LY-7004, and the Youth Innovation Promotion Association of CAS under Grant 2019141.

7. REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

[3] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[4] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *ICCV*, 2017, pp. 843–852.

[5] Jake Snell, Kevin Swersky, and Richard Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017, pp. 4077–4087.

[6] Moshe Lichtenstein, Prasanna Sattigeri, Rogerio Feris, Raja Giryes, and Leonid Karlinsky, "Tafssl: Task-adaptive feature sub-space learning for few-shot classification," in *ECCV*, 2020.

[7] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi, "Adaptive subspaces for few-shot learning," in *CVPR*, 2020, pp. 4136–4145.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine, "Model-agnostic metalearning for fast adaptation of deep networks," in *ICML*, 2017.

[9] Ling Yang, Liangliang Li, Zilun Zhang, Xinyu Zhou, Erjin Zhou, and Yu Liu, "Dpgn: Distribution propagation graph network for few-shot learning," in *CVPR*, 2020, pp. 13390–13399.

[10] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li, "Meta-sgd: Learning to learn quickly for few-shot learning," *arXiv preprint arXiv:1707.09835*, 2017.

[11] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra, "Matching networks for one shot learning," in *NIPS*, 2016, pp. 3630–3638.

[12] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto, "Meta-learning with differentiable convex optimization," in *CVPR*, 2019.

[13] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen, "Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers," in *CVPR*, 2020, pp. 12203–12213.

[14] Jongmin Kim, Taesup Kim, Sungwoong Kim, and Chang D Yoo, "Edge-labeling graph neural network for few-shot learning," in *CVPR*, 2019, pp. 11–20.

[15] Eleni Triantafillou, Richard Zemel, and Raquel Urtasun, "Few-shot learning through an information retrieval lens," in *NIPS*, 2017, pp. 2255–2265.

[16] Anton Puzanov and Kobi Cohen, "Deep reinforcement one-shot learning for artificially intelligent classification systems," *arXiv preprint arXiv:1808.01527*, 2018.

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

[18] 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," in *ICLR*, 2018.

[19] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi, "Meta-learning with differentiable closed-form solvers," in *ICLR*, 2019.

[20] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al., "Imagenet large scale visual recognition challenge," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[21] Alex Krizhevsky and Geoffrey Hinton, "Learning multiple layers of features from tiny images," Tech. Rep., Citeseer, 2009.

[22] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste, "Tadam: Task dependent adaptive metric for improved few-shot learning," in *NIPS*, 2018, pp. 721–731.

[23] Bin Liu, Yue Cao, Yutong Lin, Qi Li, Zheng Zhang, Mingsheng Long, and Han Hu, "Negative margin matters: Understanding margin in few-shot classification," in *ECCV*, 2020.

[24] 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 *ECCV*, 2020.