doi:10.3772/j.issn.1006-6748.2022.01.006

Smoother manifold for graph meta-learning①

ZHAO Wencang (赵文仓), WANG Chunxin②, XU Changkai
(College of Automation and Electronic Engineering, Qingdao University of Science and Technology, Qingdao 266061, P. R. China)

Abstract

Meta-learning provides a framework for the possibility of mimicking artificial intelligence. However, data distribution of the training set fails to be consistent with the one of the testing set as the limited domain differences among them. These factors often result in poor generalization in existing meta-learning models. In this work, a novel smoother manifold for graph meta-learning (SGML) is proposed, which derives the similarity parameters of node features from the relationship between nodes and edges in the graph structure, and then utilizes the similarity parameters to yield smoother manifold through embedded propagation module. Smoother manifold can naturally filter out noise from the most important components when generalizing the local mapping relationship to the global. Besides suiting for generalizing on unseen low data issues, the framework is capable to easily perform transductive inference. Experimental results on MiniImageNet and TieredImageNet consistently show that applying SGML to supervised and semi-supervised classification can improve the performance in reducing the noise of domain shift representation.

Key words: meta-learning, smoother manifold, similarity parameter, graph structure

0 Introduction

The difference between human intelligence and artificial intelligence is that humans can learn quickly and accurately from a small number of examples, because humans have the ability to learn from prior knowledge[1]. In contrast, traditional deep learning has made tremendous progress in various fields, such as image classification[2-3], object detection[4-5] and image semantic segmentation[6-7]. However, the strong deep learning algorithm heavily relies on a lot of labeled data. The human label cost and the scarcity of some classes tremendously limit the further development of deep learning. Just as humans can use the previous background and knowledge to effectively learn new knowledge, artificial intelligence algorithms also need to learn quickly and efficiently to contain new and invisible information. So meta-learning has aroused strong interest in various fields, especially few-shot learning[8-10], learn to learn[11-12] and continual learning[13-14]. This work focuses on few-shot learning, which aims to identify novel categories with very few labeled examples.

A key challenge of meta-learning is inference ability, inferring the new data representation by the efficient and fast neural network based on training. Because the operation of few-shot learning needs to make full use of the information in the dataset, especially it is sensitive to the slight change of data distribution, and graph neural network (GNN) can perform neighborhood feature aggregation iteratively to express the complex interaction between data instances[15]. Therefore, the research on solving few-shot learning with GNN has attracted more and more attention in related fields. For example, Ref. [16] obtained the prior knowledge of classifier by training similar few-shot learning tasks. Ref. [17] proposed a model based on node labeling framework to obtain node classification tasks. In addition, there are a large number of literatures containing regularization techniques, such as dropout[18], batch normalization[19], and manifold mixup[20-21], which analyzed the generalization capability.

This paper tries to use simple smoother manifold combined with GNN to address the few-shot classification problem. A method called smoother manifold for graph meta-learning (SGML) is proposed. Specifically, it is believed that the similarity of node features can be obtained by iteratively updating node features and

① Supported by the National Natural Science Foundation of China (No. 61171131), the Key R&D Program of Shandong Province (No. YD01033) and the China Scholarship Council Project (No. 021608370049).
② To whom correspondence should be addressed. E-mail: wcxchunxin@163.com.
Received on Oct. 20, 2020

edge features in the graph network, then map them to the embedded propagation network to obtain a set of interpolation features, and then the image is labeled by the classifier. The embedded propagation network can increase the smoothness of the embedded manifold.

1 Related works

Because GNN is extremely powerful in modeling the dependencies between nodes, some revolutionary breakthroughs have been made in the study of graph structure. Refs[22,23] combined graph node characteristics with graph topology to classify graph nodes. On this basis, Ref. [24] proposed an edge-labeling graph neural network (EGNN) to make full use of edge information, which solves the problem that graph convolutional network (GCN) can only handle one-dimensional edge features and that noise with original adjacency matrix in graph attention network (GAT) model may not be optimal. However, combining graph structure and few-shot learning are still unnoticed, and there are few unreliable solutions to classify for unknown categories with only a small amount of data. Ref. [25] proposed a label-based propagation transmit network, built an undirected graph containing both unlabeled data and labeled data, and obtained labels of all unlabeled data by means of label propagation. By learning to predict edge labels, the network can explore the direct connection state by using intra-cluster dissimilarity and inter-cluster similarity. By iteratively updating edge labels, the clustering representation can be advanced and the classification performance can be improved by combining with transductive propagation network (TPN). This approach is used to construct the network by using the relationship between edges and nodes.

When constructing machine learning, if a mess of complex neural network area is used to fit the data, it is easy to cause over-fitting, which will cause the generalization ability of the model to decline. At this time, regularization is needed. The absence of the free lunch theorem has clearly show that there is no optimal regularization method, and a more appropriate regularization can only be chosen from different tasks. Ref. [26] proposed the thought of batch normalization expression, which is more universal and can solve problems such as slow training, gradient disappearance and gradient explosion caused by inconsistent distribution. Ref. [27] showed that self-supervised technology can fully exert the advantages of manifold, especially when used with manifold mixup, it can significantly improve the few-shot learning performance. Ref. [21] proposed the embedding propagation network (EPNet) utilizing embedded interpolation to capture high-order feature interactions, so as to use manifold smoothing to solve the problem of poor generalization ability caused by different data set distribution in few-shot learning. This work tries to combine this manifold smoothing with GNN and achieves unexpected results.

Semi-supervised learning and transductive learning are similar to the core ideas of meta-learning, so they are widely used in this field. The former enables learners to use unlabeled samples to improve their learning performance without relying on external interaction, while the latter can train specific training samples to predict the samples to be determined. Semi-supervised learning is applied in few-shot learning in order to improve classification performance through adding unlabeled data. Three ProtoNets semi-supervised variants were proposed based on k-means method, which adjusted clustering information by using a large amount of unlabeled data[28]. Based on this, Ref. [29] proposed a meta-learned cherry-picking method which based on the self-training and meta gradient descent method without redesigning the semi-supervised network. Experiments have proved that transductive learning achieves better performance than inductive learning in the few-shot classification task[30]. They proposed a transduction network for a meta-learning framework that uses feature embedding, graph construction, label propagation and the four steps of loss calculation realizing end-to-end learning. Furthermore, the meta-learned confidence transduction (MCT) was proposed in Ref. [31] in order to update the confidence weighted average of all supporting samples and query samples of each prototype class and perform meta-learning on the parameters of distance measurement to improve the performance of transduction inference. The transduction learning method is applied to the framework to improve performance and improve semi-supervised learning results.

As shown in Fig. 1, the L iterations update of node

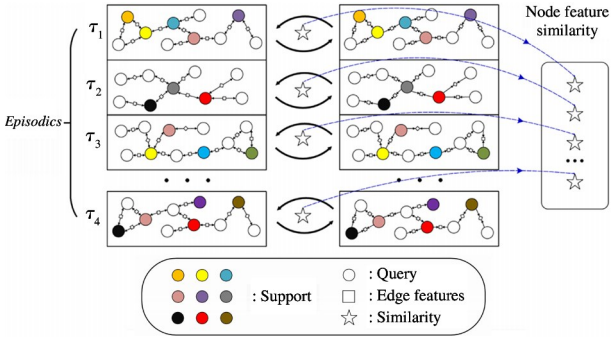


Fig. 1 Extract similarity from mutually updated node and edge features

features and edge features is completed in each episodes, and then the similarity features between the two nodes are extracted as the input of the next stage.

2 Method

In this section, the few-shot classification task is defined and the proposed smoother manifold for graph meta-learning is described. Firstly, the node features are initialized by using feature extractor for inputting images and edge features. Then, node feature transformation network and edge feature transformation network are adopted to update each other with new node features and edge features, and get the similarity between nodes. Finally, the similarity is input into the embedding propagation module to map the features to a set of interpolated features and into the label propagation module to get logit value for each query examples of the class.

2.1 Problem definition

Episodic training is utilized to complete classification task T, where T includes a support set S with labeled samples (k samples per class), and a query set Q with unlabeled samples (q samples per class). And query set does not contain the samples of support set i.e. $S \cap N = \varnothing$.

More specifically, the training and testing tasks for n-way k-shot problems are as follows. $T = S \cup V \cup Q$, $S = \{(x_i, y_i)\}_{i=1}^{n\times k}$, where x_i and $y_i \in \{C_1, \cdots, C_N\} = C_T \in C$ are the ith input image and its label. $Q = \{(x_j, y_j)\}_{j=1}^{n\times k+t}$ are the jth input image and its label, t represents the number of samples from the query set. Although S and Q are uniformly sampled from the dataset, they do not included in each other, i.e. $S \cap Q = \varnothing$. The samples in the verification dataset V intersect with neither S nor Q, but are only used for hyperparameter search. In each episode, the training model is optimized in the support set Q with the labels, and then the loss of the model is predicted in the query set Q. The training program iterates until the convergence according to the given episodes.

2.2 Model

This section describes the SGML framework for the few shot classification. Firstly, the convolutional neural network is used to extract the feature representation of the input image to construct the full connection graph $G = (P, W; M)$, nodes and edges represent samples and types of relationships between samples, respectively, where M represents amount of samples task. $P := \{P_i\}_{i=1,\cdots,|M|}$ and $W := \{E_{ij}\}_{i,j=1,\cdots,|M|}$ respectively represent nodes set and edges set, where p_i is the node feature of P_i and e_{ij} is the edge feature of E_{ij}. Then apply the obtained similarity between the nodes in the embedded propagation matrix and the label propagation matrix and finally use the classifier to obtain the loss.

2.2.1 Preparing

Firstly, the node features are initialized by the output of feature extractor network $v_i^0 = f_\theta\{x_i\}$. Then initialize the edge labels as follows.

$$e_{ij}^0 = \begin{cases} [1 \,\|\, 0] & y_{ij} = 1 \text{ and } i, j \leqslant N \times K \\ [0 \,\|\, 1] & y_{ij} = 0 \text{ and } i, j \leqslant N \times K \\ [0.5 \,\|\, 0.5] & \text{otherwise} \end{cases} \tag{1}$$

The similarity parameters of the node features are obtained by iterations between node features and edge features. Specifically, set the number of iterations be L. Defining node feature and edge feature of the $L-1$ layer as v_i^{L-1} and e_{ij}^{L-1}, then update the node features v_i^L during the neighborhood aggregation process as follows.

$$v_i^L = f_v^L\{[\sum_j \tilde{e}_{ij1}^{L-1} v_j^{L-1} \,\|\, \sum_j \tilde{e}_{ij2}^{L-1} v_j^{L-1}]; \theta_v^L\} \tag{2}$$

where $\tilde{e}_{ijd}^{L-1} = \dfrac{e_{ijd}}{\sum_k e_{ikd}}$, and f_v^L is the node feature transformation network, as shown in Fig. 2(a).

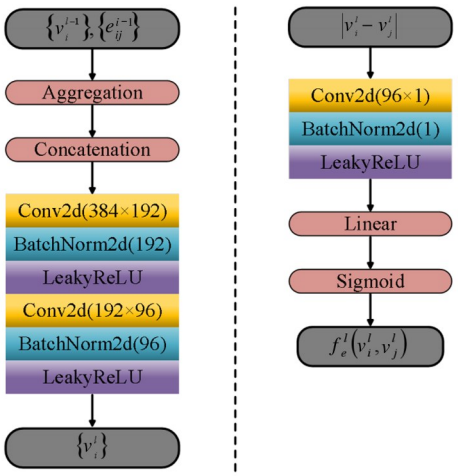
Then, the updated node feature and edge feature are used to obtain newly edge feature and the similarity between the nodes.

$$\bar{e}_{ij}^L = \frac{f_e^L(v_i^L, v_i^L; \theta_e^L) e_{ij}^{L-1}}{\sum_k f_e^L(v_i^L, v_k^L; \theta_e^L) e_{ik1}^{L-1} / (\sum_k \theta_e^L)} \tag{3}$$

$$e_{ij}^L = \bar{e}_{ij}^L / \| \bar{e}_{ij}^L \|_1 \tag{4}$$

$$z^L = \frac{1}{1 + e^{-f_e^L(v_i^L, v_i^L; \theta_e^L)}} \tag{5}$$

where f_e^L is the edge feature transformation network, as shown in Fig. 2(b).



(a) Node feature transformation (b) Edge feature transformation

Fig. 2 Detailed network structure of feature transformation

The node feature and edge feature are iterated with each other L times to obtain the final similarity feature z^L between the node pairs as the input of the embedded propagation module. The purpose of embedding is increasing the smoothness of manifolds, especially at the classification boundary of low density. Firstly, the module calculates the prototype distance and the adjacency of each similarity feature (i, j) respectively as $d^2_{i,j} = \| z_i - z_j \|^2_2$ and $A_{ij} = \exp(-\frac{d^2_{ij}}{Var(d^2_{ij})})$. Then the Laplace of the adjacency matrix is calculated by the prototype distance.

$$C = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}, \quad D_{ii} = \sum_j A_{ij} \tag{6}$$

Then, the propagation matrix is obtained by using the label propagation formula proposed by Ref. [32].

$$P = (\boldsymbol{I} - \alpha C)^{-1} \tag{7}$$

where $\alpha \in [0,1]$ is the scaling factor, the value is 0.5. $\boldsymbol{I}$ is the identity matrix. To remove the irrelevant noise in the input information and obtain the embedded information, the propagation matrix $\boldsymbol{P}$ is weighted and operated as follows.

$$\tilde{z}_i = \sum_j \boldsymbol{P}_{ij} z_j \tag{8}$$

The one-hot encoding is used to process data. Let $\boldsymbol{P}_{\tilde{Z}}$ be matrix of embedding propagation by employing Eq. (6) and Eq. (8), where $\tilde{\boldsymbol{Z}} \in \mathbb{R}^{n \times k+t}$ is sampled at S and Q. Moreover, let $Y_S \in \mathbb{R}^{n \times k}$ be the labels from the S of one-shot encoding. Finally, the label for query set Y_Q with the largest logit value is obtained by applying label propagation as follows.

$$\begin{bmatrix} \hat{Y}_S \\ \hat{Y}_Q \end{bmatrix} = \log\left(\boldsymbol{P}_{\tilde{Z}} \begin{bmatrix} Y_S \\ 0 \end{bmatrix}\right) \tag{9}$$

2.2.2 Training

SGML will conduct two stages of training. Firstly, the model will be pre-trained in the pre-training stage to learn the general feature representation, and then the model will be fine-tuned in the episodic learning stage to inference to unseen classes.

Pre-training phase First, let a classifier parameterized by W_l and W_r with a linear layer of softmax activation function to predict the sample label category in support set S and optimize it with the minimum cross entropy loss.

$$L_c(x_i, y_i) = -\ln p(y_i \mid \tilde{z}_i, W_l) \tag{10}$$

In addition, self-supervision is added to improve the effectiveness of the model, so the same classifier as above is used to predict the loss of rotating images.

$$L_r(x_i, r_j) = -\ln p(y_i \mid \tilde{z}_i, W_r) \tag{11}$$

In conclusion, stochastic gradient descent (SGD) is used to optimize the losses of the above two classifiers as follows.

$$\underset{\theta, W_l, W_r}{\operatorname{argmin}} \sum_{i=1}^{b} \sum_{j=1}^{4} L_c(x_i, y_i) + L_r(x_i, r_j) \tag{12}$$

where b is the batches of size, 4 rotations per image and $r_j \in \{0°, 90°, 180°, 360°\}$.

Episodic learning phase At this stage, the model obtained in the pre-training stage is deduced to novel categories, and two classifiers are still used to optimize SGML. The first classifier uses softmax to calculate the class probability based on label propagation as follows.

$$L_p(x_i, y_i) = -\ln p(y_i \mid \tilde{z}_i, \tilde{Z}, Y_S) \tag{13}$$

The second classifier uses the same classifier as in the pre-training stage to retain the distinguish feature representation. The loss optimization is as follows.

$$L_1 = \frac{1}{|S|} \sum_{(x_i, y_i) \in Q} L_p(x_i, y_i) \tag{14}$$

$$L_2 = \frac{1}{|S \cup Q|} \sum_{(x_i, y_i) \in S \cup Q} \frac{1}{2} L_c(x_i, y_i) \tag{15}$$

$$\underset{\theta, W_l}{\operatorname{argmin}} (L_1 + L_2) \tag{16}$$

Framework of SGML is shown as Algorithm 1.

Algorithm 1 Framework of SGML

Input: $T = S \cup V \cup Q$, where $S = \{(x_i, y_i)\}_{i=1}^{n \times k}$, $Q = \{(x_j, y_j)\}_{j=1}^{n \times k+t}$.
Initialize: $v_i^0 = f_\theta\{x_i\}$, e_{ij}^0 by Eq. (1).
For $l = 1, \cdots, L$ do:
  v_i^L = NodeUpdate $\{v_i^{L-1}, e_{ij}^{L-1}\}$ by Eq. (3);
  e_{ij}^L, z^L = EdgeUpdate $\{v_i^{L-1}, e_{ij}^{L-1}\}$ by Eq. (4) and Eq. (5).
End
$\tilde{z}_i$ = propagation_module $\{z^L, \theta_e\}$ by Eq. (5), Eq. (6), Eq. (7) and Eq. (8), where θ_e is the is corresponding parameter sets.
Pre-training phase:
Predict the class labels of examples in support set S by Eq. (10), Eq. (11) and Eq. (12).
Output: argmin $\sum \sum (L_c + L_r)$.
Episodic learning phase:
Learn to inference unseen classes by Eq. (13), Eq. (14), Eq. (15) and Eq. (16).
Output: argmin $\sum \sum (L_P + \frac{1}{2} L_C)$.

3 Experiments

This section mainly describes the experimental results on two commonly used datasets, MiniImageNet and TieredImageNet.

3.1 Datasets

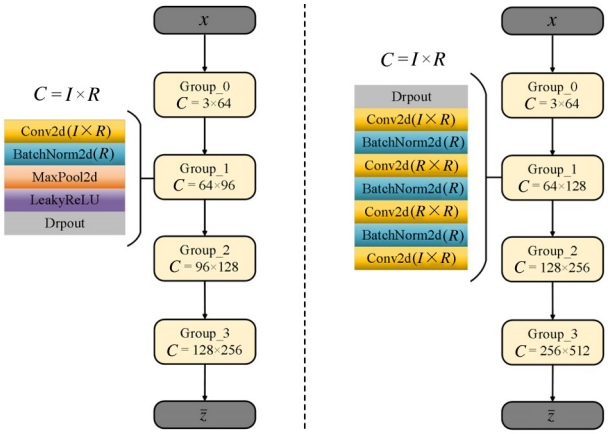
MiniImageNet is one of the most commonly used few-shot classification datasets proposed by Ref. [33].

Total of 100 categories, each category has 600 images, all in RGB color, adjusted to 84 × 84 pixels. Divide it into three (64, 16 and 20) disjoint parts for training, verification and testing.

TieredImageNet is a subset of the alternative imagenet dataset that has a larger number of images than MiniImageNet[34]. There are a total of 608 categories (all sampled from the 34 high-level categories in imagenet), all RGB colors, adjusted to 84 × 84 pixels. It is divided into 351 (20 high-level categories), 97 (6 high-level categories), and 160 (8 high-level categories) groups for training, verification, and testing.

3.2 Experimental setting

Baseline methods For feature extractors module, two common convolutional neural networks are used. One is a basic embedding network Conv-4 (Fig. 3(a)), the other one is a much more complicated network ResNet-12 (Fig. 3(b)).



(a) Conv-4 (b) ResNet-12

Fig. 3 Detailed network structure of feature extractors

$v_i^0 = f_\theta(x_i)$

Implementation details To verify the accuracy of the implementation on these two datasets, 5-way 1-shot and 5-way 5-shot experiment will be conducted. To assess the fairness of the experiment, 15 query images will be selected from each class for 5-way 1-shot and 5-way 5-shot in each episode. The proposed SGML model is optimized using SGD with initial learning rate of 5×10^{-4} and weight decay of 10^{-1}. For MiniImageNet, the model iterated 100 epochs in total, and the batch size of the meta-training task is set to 60. For TieredImageNet, because it is a larger dataset, more iterations are needed to better converge. The model iterated 150 epochs and the batch size of the task is set to 40. All of the experiments were carried out in Pytorch and run with two NVIDIA GTX 1080Ti GPUs.

Few-shot classification The proposed SGML with the latest meta-learning model are compared in Table 1. In which, all models are tested by using transductive and non-transductive settings; T means transductive setting, where it classifies all query samples at once at a single episode; F means non-transductive setting, where it performs the query inference one by one. Notably, almost all the transductive settings improve the performance of SGML. Furthermore, in the transductive setting, SGML performed better than EGNN[15] with 50 parameters, which only classifies the results by predicting edge labels. Likewise, since TPN[25] has fixed node features during the label propagation process, its label propagation equation requires additional parameters. On the contrary, the node features and edge features are adopted to the given task in the process of mutual update. The accuracy of model-agnostic meta-learning (MAML)[35] has been greatly improved after transduction, but it is still insufficient compared with other models. It can be guessed that because MAML is prone to overfitting when processing high-dimensional data, SGML is better for this problem. MetaOpt[8] increases the amount of calculation to improve the generalization ability under high-dimensional embedding, which is inconsistent with the ideas

Table 1 Classification results in 5-way setting

Model	Transduction	1-shot Conv4	5-shot Conv4	1-shot ResNet-12	5-shot ResNet-12
		MiniImageNet			
MAML	F	50.76	54.37	52.96	59.88
MAML	T	53.84	60.86	57.43	65.16
TPN	F	51.28	60.18	53.76	67.77
TPN	T	55.43	68.29	60.75	74.96
MetaOpt	T	-	-	63.27	77.40
EGNN	F	53.18	62.68	-	-
EGNN	T	58.18	76.37	-	-
EPNet	T	59.32	74.26	68.74	79.52
SGML	F	54.61	68.46	62.37	71.85
SGML	T	59.75	74.88	70.65	80.71
		TieredImageNet			
MAML	F	52.21	61.26	51.86	58.94
MAML	T	57.21	69.24	59.62	70.53
TPN	F	53.43	68.76	62.94	69.13
TPN	T	58.96	74.93	68.12	76.73
MetaOpt	T	-	-	68.72	79.38
EGNN	F	57.86	65.86	-	-
EGNN	T	62.40	78.25	-	-
EPNet	T	61.36	76.51	74.63	79.36
SGML	F	58.49	69.73	62.13	72.05
SGML	T	62.58	77.45	75.65	81.12

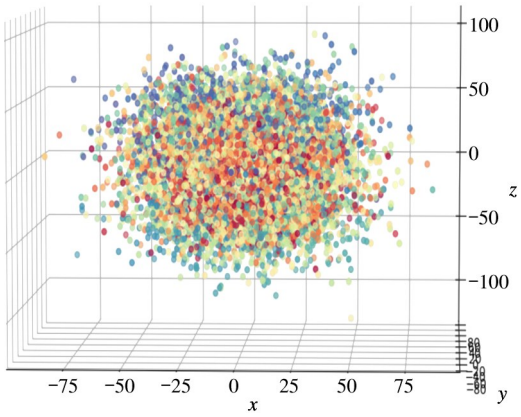
of this work. Different from EPNet[21], this work uses graph network to generate smoother embedding manifold, SGML network architecture is bigger and the effect is better. It is worth mentioning that the worst experimental results of SGML will not lag behind the reported performance by more than 1.5% (74.88% vs. 76.37%).

Semi-supervised classification This work conducts a fair comparison with the same settings in the 5-way 5-shot experiment, that is, all classes in the support set samples are labeled with the same number. In order to fully demonstrate the effectiveness of the proposed method, this work conducts experiments on MiniImageNet and TieredImageNet datasets with Conv-4. The experimental results are shown in Table 2. SGML can increase the maximum accuracy by 2.81% on MiniImageNet (66.84% vs. 64.03%, when 60% labeled) and up to 4.89% on TieredImageNet (72.41% vs. 67.52%, when 60% labeled). It is worth noting that SGML performs the worst at 20% labeled, and only has a 0.18% improvement at 40% labeled (64.50% vs. 64.32%). It can be assumed that better accuracy may be achieved with Resnet-12, which will be studied in future work.

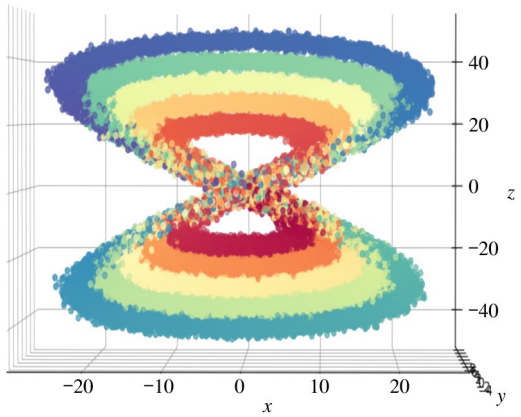
Table 2 Semi-supervised results for Conv-4

Method	Transduction	Labeled ratio (5-way 5-shot) 20%	40%	60%	100%
		MiniImageNet			
TPN	T	57.22	59.89	64.03	98.29
EGNN	T	63.62	64.32	66.36	76.37
EPNet	T	58.34	62.36	65.87	74.26
SGML	T	57.21	64.50	66.84	74.88
		TieredImageNet			
TPN	T	60.37	61.48	67.52	74.93
EGNN	T	63.74	64.56	71.01	78.25
EPNet	T	59.63	65.58	70.80	76.51
SGML	T	61.52	63.77	72.41	77.45

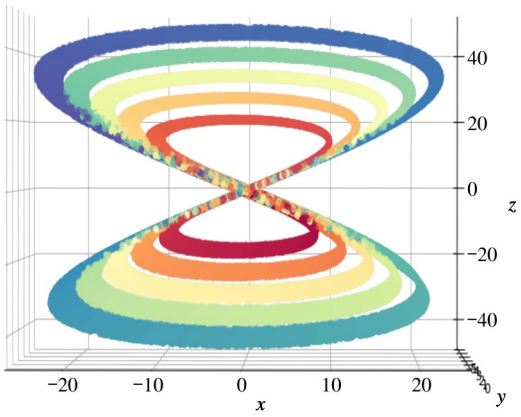
Visualization Fig. 4 shows the effect of embedding manifolds. In Fig. 4(a), all the samples are scattered in the characteristic space, with different colors representing different classes. In the pre-training stage, all samples were classified. As shown in Fig. 4, different categories are distributed in different circles, and the classification boundary is relatively obvious. The density of nodes also shows that the relationship between nodes in the feature space is complex and interactive. In the next stage, the sample interaction is spread more intensively, the relationship between nodes is more complex, the density and smoothness of the decision boundary are also increased, and the noise representation of the previous stage is reduced (the noise may cause missing manifolds, which may affect the embedding result in severe cases). GNN is a deep architecture composed of several nodes and edges. To validate the idea that interaction between task samples should be easier as the number of iterations increases.



(a) Initial phase



(b) Pretraining phase

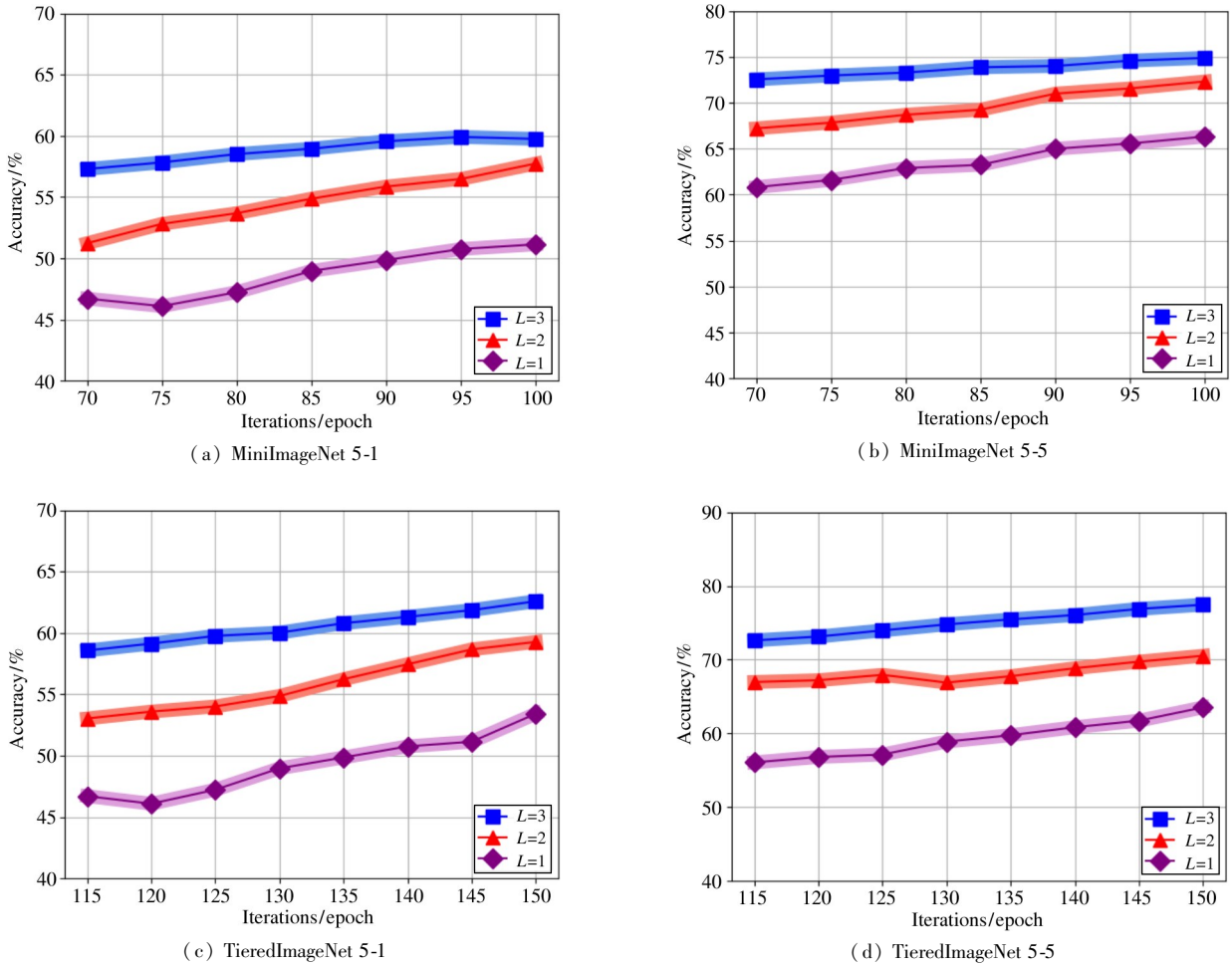


(c) Episodic learning phase

Fig. 4 Visualization of feature space at different phase

The SGML is compared at different layers, and the results are shown in Fig. 5. Since TieredImageNet

is a larger dataset, more iterations are required to converge (100 epochs vs. 150 epochs). With the increase of the number of iteration layers L between node features and edge features, the classification effect of the two datasets is also better. When $L = 3$, the classification accuracy is the best, and the curve is relatively flat; when $L = 2$, the classification accuracy is reduced, and the curve is relatively tortuous; when $L = 1$, the classification accuracy is the worst. This indicates that SGML tends to show good clustering in more iterations, but it does not indicate that more iterations will make the effect better.



(a) MiniImageNet 5-1 (b) MiniImageNet 5-5

(c) TieredImageNet 5-1 (d) TieredImageNet 5-5

Fig. 5 Classification accuracy on different layers

4 Conclusion

A novel smoother manifold for graph meta-learning is proposed, which aims to addressed the problem of few-shot classification task. In the process of SGML, the similarity between nodes is obtained by updating nodes and edges features with different parameter layers, and then the similarity is sent to the embedded propagation module to complete the classification task. The experiment results show that the proposed method not only can improve the classification performance, but also can increase the smooth classification boundaries. Future work aims to explore other way of adding graph networks to manifold in meta-learning such as graph embedding, that will address more challenging problems to larger number of shots.

Reference

[1] LAKE B, ULLMAN T, TENENBAUM J, et al. Building machines that learn and think like people[J]. *Behavioral and Brain Sciences*, 2016,40(10):1-10

[2] TAN M, LE Q. EfficientNet: rethinking model scaling for convolutional neural networks[C]// Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019:6105-6114

[3] XIE S, GIRSHICK R, DOLLR P, et al. Aggregated residual transformations for deep neural networks[C]// Conference on Computer Vision and Pattern Recognition, Honolulu, USA, 2017:5987-5995

[4] PEREZ-CABO D, JIMENEZ-CABELLO D, COSTA-PAZO A, et al. Deep anomaly detection for generalized face anti-spoofing[C]// Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019:1591-1600

[5] SHAKERI M, ZHANG H. Moving object detection under

discontinuous change in illumination using tensor low-rank and invariant sparse decomposition[C]// Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019:7221-7230
[6] LONG J, SHELHAMER E, DARRELL T. Fully convolutional networks for semantic segmentation[J]. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015, 39(4):640-651
[7] GARCIA-GARCIA A, ORTS-ESCOLANO S, OPREA S, et al. A review on deep learning techniques applied to semantic segmentation[EB/OL]. https://arxiv.org/pdf/1704.06857.pdf:arXiv, (2017-04-22),[2020-10-20]
[8] LEE K, MAJI S, RAVICHANDRAN A, et al. Meta-learning with differentiable convex optimization[C]// Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019: 10657-10665
[9] SUN Q, LIU Y, CHUA T, et al. Meta-transfer learning for few-shot learning[C]// Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019: 403-412
[10] LIU B, YU X, YU A, et al. Deep few-shot learning for hyperspectral image classification[J]. *IEEE Transactions on Geoscience and Remote Sensing*, 2018, 57(4):2290-2304
[11] BELLO I, ZOPH B, VASUDEVAN V, et al. Neural optimizer search with reinforcement learning[EB/OL]. https://arxiv.org/pdf/1611.01578.pdf: arXiv, (2017-02-15),[2020-10-20]
[12] WICHROWSKA O, MAHESWARANATHAN N, HOFFMAN M W, et al. Learned optimizers that scale and generalize[C]// International Conference on Machine Learning, Sydney, Australia, 2017:3751-3760
[13] VUORIO R, CHO D Y, KIM D, et al. Meta Continual Learning[EB/OL]. https://arxiv.org/pdf/1806.06928.pdf: arXiv, (2018-06-11),[2020-10-20]
[14] XU J, Zhu Z X. Reinforced continual learning[C]// Neural Information Processing Systems, Montreal, Canada, 2018:899-908
[15] KIM J, KIM T, KIM S, et al. Edge-labeling graph neural network for few-shot learning[C]// Conference on Computer Vision and Pattern Recognition, Long Beach, USA, 2019:11-20
[16] FAN Z, CHENGTAI C, KUNPENG Z, et al. Meta-GNN on few-shot node classification in graph meta-learning[C]// International Conference on Information and Knowledge Management, Beijing, China, 2019:2357-2360
[17] GARCIA V, BRUNA J. Few-shot learning with graph neural networks[C]// International Conference on Learning Representations, Toulon, French, 2017: 1-13
[18] SRIVASTAVA N, HINTON G, KRIZHEVSKY A, et al. Dropout: a simple way to prevent neural networks from overfitting[J]. *Journal of Machine Learning Reserarch*, 2014, 15(1):1929-1958
[19] HUANG L, YANG D, LANG B, et al. Decorrelated batch normalization[C]// Computer Vision and Pattern Recognition, Salt Lake City, USA, 2018:791-800
[20] VERMA V, LAMB A, BECKHAM C, et al. Manifold mixup: better representations by interpolating hidden states[C]// International Conference on Machine Learning, Long Beach, USA, 2019:6438-6447
[21] RODRGUEZ P, LARADJI I, DROUIN A, et al. Embedding propagation: smoother manifold for few-shot classification[C]// European Conference on Computer Vision, Glasgow, UK, 2020:121-138
[22] DEFFERRARD M, BRESSON X, VANDERGHEYNST P. Convolutional neural networks on graphs with fast localized spectral filtering[C]// Advances in Neural Information Processing Systems, Barcelona, Spain, 2016: 3844-3852
[23] KIPF T, WELLING M. Semi-supervised classification with graph convolutional networks[EB/OL]. https://arxiv.org/pdf/1609.02907.pdf: arXiv, (2017-02-22),[2020-10-20]
[24] GONG L Y, CHENG Q. Exploiting edge features for graph neural networks[C]// Computer Vision and Pattern Recognition, Long Beach, USA, 2019:9211-9219
[25] LIU Y B, LEE J H, PARK M, et al. Transductive propagation network for few-shot learning[C]// International Conference on Learning Representations, New Orleans, USA, 2019:1-14
[26] IOFFFFE S, SZEGEDY C. Batch normalization: accelerating deep network training by reducing internal covariate shift[C]// International Conference on Machine Learning, Miani, USA, 2015:448-456
[27] MANGLA P, SINGH M, SINHA A, et al. Charting the right manifold: manifold mixup for few-shot learning[C]// Winter Conference on Applications of Computer Vision, Snowmass, USA, 2020:2218-2227
[28] SNELL J, SWERSKY K, ZEMEL R. Prototypical networks for few-shot learning[C]// Conference and Workshop on Neural Information Processing Systems, Montreal, Canada, 2017:4080-4090
[29] LI X, SUN Q, LIU Y, et al. Learning to self-train for semi-supervised few-shot classification[C]// In Neural Information Processing Systems, Vancouver, Canada, 2019:10276-10286
[30] AFONSO B K D A, BERTON L. Identifying noisy labels with a transductive semi-supervised leave-one-out filter[J]. *Pattern Recognition Letters*, 2020, 140:127-134
[31] KYE S M, LEE H B, KIM H, et al. Meta-learned confidence for few-shot learning[EB/OL]. https://arxiv.org/pdf/2002.12017v2.pdf: arXiv, (2020-06-24),[2020-10-20]
[32] ZHOU D, BOUSQUET O, LAL T, et al. Learning with local and global consistency[C]// Neural Information Processing Systems, Vancouver, USA, 2004:321-328
[33] ORIOL V, CHARLES B, TIM L, et al. Matching networks for one shot learning[C]// Neural Information Processing Systems, Montreal, Canada, 2016:3630-3638
[34] OLGA R, DENG J, SU H, et al. Imagenet large scale visual recognition challenge[J]. *International Journal of Computer Vision*, 2015, 115(3):211-252
[35] CHELSEA F, PIETER A, SERGEY L. Model agnostic meta-learning for fast adaptation of deep networks[C]// International Conference on Machine Learning, Sydney, Australia, 2017: 1126-1135

ZHAO Wencang, born in 1973. He received his Ph. D degree from Information Science Department of Ocean University of China in 2005. He also received his B. E. and M. E. degrees from Qingdao University of Science and Technology and Shandong University in 1995 and 2002 respectively. He is mainly engaged in image processing and pattern recognition, cognitive informatics, neural computing, machine learning and other research work.