Neurocomputing 383 (2020) 224–234

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Meta-Learning based prototype-relation network for few-shot classification

Xiaoqian Liu, Fengyu Zhou*, Jin Liu, Lianjie Jiang

School of Control Science and Engineering, Shandong University, Jinan 250061 Shandong, PR China

ARTICLE INFO

Article history:
Received 5 August 2019
Revised 28 October 2019
Accepted 9 December 2019
Available online 12 December 2019

Communicated by Prof. WK Wong

Keywords:
Few-shot learning
Classification
Meta-learning
Prototype

ABSTRACT

Pattern recognition has made great progress under large amount of labeled data, while performs poorly on a very few examples obtained, named few-shot classification, where a classifier can identify new classes not encountered during training. In this paper, a simple framework named Prototype-Relation Network is presented for the few-shot classification. Moreover, a novel loss function compared with prototype networks is proposed which takes both inter-class and intra-class distance into account. During meta-learning, the model is optimized by end-to-end episodes, each of which is to imitate the test few-shot setting. The trained model is used to classify new classes by computing min distance between query images and the prototype of each class. Extensive experimental results demonstrate that our proposed meta-learning model is competitive and effective, which achieves the state-of-the-art performance on Omniglot and *mini*ImageNet datasets.

1. Introduction

In recent years, deep learning has made great progress in the field of pattern recognition, such as object detection [1,2], image classification [3–14], speech recognition [15,16] and machine translation [17]. Deep learning is a data-hungry technique, which needs large amounts of labeled data [18,19] to train model parameters, otherwise, over-fitting will occur [20], which results in poor robustness and generalization performance of the model. However, it is difficult to get labeled data in real life, and image annotation is time-consuming and laborious. In contrast, humans have the ability to recognize objects where only a few examples of each class are given with a high accuracy [21]. For example, children have no problem generalizing the concept of 'zebra' from a single picture in a book, or hearing its description as looking like a stripy horse [22], which is an innate meta-learning [23] or learning to learn [24] ability of humans. Meta-learning performs transfer learning from a pool of various classification problems generated from large quantities of available labeled data, to new classification problem from classes unseen at training time [25]. It is mainly to enable the model to learn a knowledge transfer ability, and to identify the sample classes that have not been encountered in the training process, so as to achieve few-shot classification.

Inspired by the deficiency of deep learning requiring large amounts of data and the meta-learning ability of human beings, few-shot learning [26–30] where the given classification problem is assumed to contain only a handful of labeled examples per class has attracted the interest of many researchers. Data augmentation [31–33] is one of the methods to reduce the over-fitting phenomenon in such a limited-data regime, but they do not solve it. Another effective method is to use only a small number of samples to learn a generalized representation of the data [34–37], which can be directly applied to the target data, i.e., transfer learning-based approaches. Currently, most exiting few-shot learning approaches learn a metric [20,22,38–40] in the space constructed by data sources, and then classification is completed by comparing the distance between images.

Two recent metric-based approaches have made significant progress in few-shot learning. The *Prototype Network* [20] considered that each class has a prototype in the embedding space. A non-linear mapping of the input was learned into an embedding space by using a neural network, and a class's prototype is the mean of its support set in the embedding space. Lastly they transformed the classification problem into the nearest neighbor problem. Instead of learning about a fixed metric, Sung et al. [22] designed a two-branch *Relation Network* to learn a transferrable deep metric: *Embedding* module and *Relation* module. *Embedding* module generated embeddings of the *query* and *sample* images, and these representations are compared by a *Relation* module through a MSE loss function. The two approaches above utilized sampled mini-batches called *episode* training strategy, as proposed in [39], which is an effective way to exploit the training set. Episode-based

* Corresponding author.
E-mail address: zhoufengyu@sdu.edu.cn (F. Zhou).

https://doi.org/10.1016/j.neucom.2019.12.034
0925-2312/

training is to mimic the few-shot learning setting during testing, thus making the training problem more faithful to the test environment and improving generalization.

Considering the problem of over-fitting [41,42], we adopt the idea of prototype. It is a metric-based method, which models the distance between samples. However, the loss function that [20] adopted only considers the proximity of samples of the same class, but not the similarity of samples of different classes. To solve the problems above, a novel loss function is proposed in this paper, which takes into account that samples of different classes are as far away as possible to better measure the similarity between *query* and *sample* images. Specifically, we design a two-branch Prototype-Relation Network (PRN) that performs few-shot learning by comparing the distance between the prototype of *query* images and few-shot labeled *sample* images. First a *Prototype* module generates the prototypes of the meta-task constructed based on episode. Then these prototypes are compared by a *Relation* module that determines if they are from matching categories or not.

The rest of this paper is organized as follows. In Section 2, we briefly overview some related work on meta-learning and metric learning. We present our proposed method in Section 3. Experiment results and discussions are shown in Section 4. The conclusion is in Section 5 with some discussion for the future works.

2. Related work

2.1. Meta-learning

Few-shot learning is the application of meta learning in the field of supervised learning. Meta Learning (learning to learn) decomposes datasets into different meta tasks in the stage of training to learn the generalization ability of the model in the case of class changes. In the stage of testing, classification can be completed without changing existing models in the face of brand new classes. Ren et al. [25] advanced few-shot classification paradigm towards a scenario where unlabeled examples were also available within each episode. It proposed novel extensions of prototype network that were augmented with the ability to use unlabeled example when producing prototype. Santoro et al. [43] trained a memory-augmented neural network to learn how to store and retrieve memories to use for each classification task. Andrychowicz et al. [44] utilized a LSTM to train a neural network. The difference is that they are interested in large-scale classification, whereas we are interested in the few-shot learning problem. Bertinetto et al. [45] trained a met-learner to map a training example to the weights of a neural network that was then used to classify future examples from this class, however, unlike our method the classifier network is directly produced rather than being fine-tuned after multiple training steps. Maclaurin et al. [46] tuned the hyper-parameters of gradient descent with momentum by back-propagating through the chain of gradient steps to optimize the validation performance.

2.2. Metric-learning

The literature on metric learning is vast. Neighborhood Components Analysis (NCA) [47] learned a Mahalanobis distance to maximize K-nearest-neighbor's (KNN) leave-one-out accuracy in the transformed space. Weinberger et al. [48] proposed large margin nearest neighbor (LMNN) classification to optimize KNN accuracy but does so using a hinge loss that encouraged the local neighborhood of a point to contain other points with the same label. Min et al. [49] proposed a margin-based method DNet-KNN by utilizing a neural network to perform the embedding instead of a simple linear transformation. Koch et al. [38] explored a method for learning *Siamese Neural Network* with shared weights, and used the features extracted from two branch networks to measure the similarity of two inputs for few-shot learning. Vinyals et al. [39] employed the *Matching Network* framework which mapped a small labelled support set and an unlabeled example to its label, obviating the need for fine-tuning to adapt to new class types. The most related methodologies to ours are the *Prototype Network* of [20] and the *Relation Network* of [22]. These approaches focus on learning embeddings that transform the data such that it can be recognized with a fixed nearest-neighbor or linear classifier [22]. The *Prototype Network* believed that there was a prototype for each class. Under the distribution of Bregman divergence, the prototype was the mean of each dimension of support set in the embedding space. And then transformed the classification problem into the nearest neighbor problem in the feature space. The idea of this prototype is simple but can effectively prevents over-fitting. The *Relation Network* provided a learnable metric by training a neural network rather than a fixed metric. And it thought that it is more like a regression problem, using *MSE* instead of cross-entropy loss. We adopt the idea of [20], and propose a novel loss function, which takes into account the distance between heterogeneous samples as far as possible, instead of only considering homogeneous samples. Compared with [22], we benefit from an episodic training strategy with end-to-end manner from scratch.

3. Methodology

3.1. Problem definition

In the few-shot classification task, there are generally three data sets: training set, support set and testing set. The training set has its own label space that is disjoint with support/testing set. The support set and testing set have the same label space.

An effective way to exploit the training set is to follow an episode paradigm, as proposed in [39], which is to simulate the types of few-shot problems that will be encountered at test time. Every episode of training is formed in this way: C class is randomly selected from the training set, K labeled samples of each of the C class are randomly selected to form the sample set $S = \{(x_i, y_i)\}_{i=1}^{m} (m = C \times K)$, then q labeled samples of the remaining samples of each of the C class are randomly selected to serve as query set $Q = \{(x_j, y_j)\}_{j=1}^{n} (n = C \times q)$. This is called C-way K-shot few-shot classification problem. This sample/query set is to mimic the support/test set that will be encountered at test time. The model obtained by this training strategy optimization has good generalization performance on the new class samples. Due to this analogy, training under this paradigm is often referred to as learning to learn or meta-learning. In our work, we adopt such an episode-based training strategy and consider one-shot and five-shot settings.

3.2. Network architecture

In this paper, we first put forward a Prototype-Relation Network similar to Relation Network (RN) [22] architecture, as shown in Fig. 1. It consists of two modules: *Prototype* module generates prototypes of each class samples, and *Relation* module calculates the relationship between each samples in query set and each prototype of each class.

3.2.1. Prototype module

Firstly, samples x_i in the sample set S and samples x_j in the query set Q constructed by episode-based are sent into *Embedding* network, and their feature vectors $f(x_i)$ and $f(x_j)$ will be obtained through *Embedding* network. Then we calculate the prototype of each class samples in the sample set.

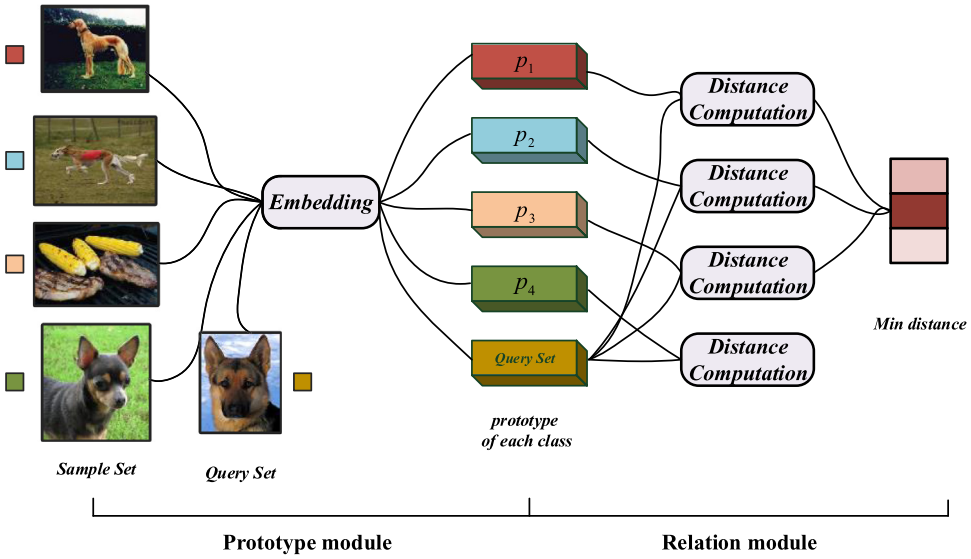


Fig. 1. Prototype Relation Networks Architecture.

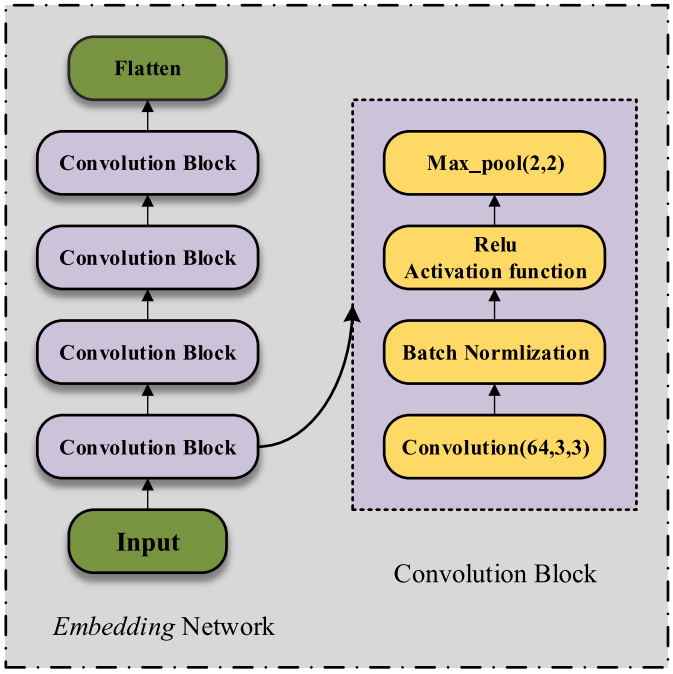


Fig. 2. *Embedding* network architecture of prototype relation.

3.2.2. Relation module

Calculate the Euclidean distance from each sample in the query set Q to each class prototype. During the training process, model parameters are optimized by minimizing the distance between same class samples and maximizing the distance between different class samples. In the test process, the classification problem is transformed into the nearest neighbor problem, and the prototype category which is the nearest to the test sample is output.

3.2.3. Embedding *network*

In order to make a fair comparison, the few-shot classification model in this paper also utilizes the form of four convolution blocks adopted in [20,22,39], as shown in Fig. 2. *Embedding* network consists of four convolution blocks and a *Flatten* layer. More concretely, each convolutional block contains a 64 filter 3×3 convolution, a batch normalization, a Relu activation function and a 2×2 max_pool layer. The output size of *Flatten* layer is $n = 64$.

3.3. Model

Firstly, we can get an n-dimensional representation $f(x_i)$ of each sample x_i in the sample set S. Each prototype is the mean vector of the embedded support points belonging to its class. In the C-way K-shot problem, the prototype p_k of class k follows the following rules:

$$p_k = \begin{cases} f(x_i) & , K = 1 \\ \frac{1}{N_s} \sum_{(x_i, y_i) \in S_k} f(x_i), & K > 1 \end{cases} \tag{1}$$

where N_s is the number of support examples per class.

Secondly, calculate the distance $d(f(x_j), p_k)$ from each sample x_j in the query set Q to each class prototype p_k. In this paper, the Euclidean distance is adopted. Snell et al. [20] proved that computing the class prototype as the mean of embedded support points is more naturally suited to Euclidean distances since cosine distance is not a Bregman divergence.

Thirdly, calculate the probability of each sample x_j in the query set Q to each class prototypes. The closer the sample is to the prototype, the more likely it is to belong to this class; otherwise, the less likely it is to belong to this class. The probability of the sample of the class k to the prototype of the class k is:

$$p(y = k | p_k) = \frac{\exp(-d(f(x_q), p_k))}{\sum_{k'=1}^{N_c} \exp(-d(f(x_q), p_{k'}))} \tag{2}$$

where N_c is the number of class per episode.

Then, define the loss of the sample of the class k to the prototype of the class k

$$\begin{aligned} Loss_k &= -\log p(y = k | p_k) - \lambda \sum_{k'=1, k' \neq k}^{N_c} \log [1 - p(y = k | p_{k'})] \\ &= -\log \frac{\exp(-d(f(x_q), p_k))}{\sum_{k'=1}^{N_c} \exp(-d(f(x_q), p_{k'}))} \\ &\quad -\lambda \sum_{k'=1, k' \neq k}^{N_c} \log \left[1 - \frac{\exp(-d(f(x_q), p_{k'}))}{\sum_{k'=1, k' \neq k}^{N_c} \exp(-d(f(x_q), p_{k'}))} \right] \end{aligned} \tag{3}$$

where λ is the regularization coefficient.

Algorithm 1
Episode formation and model training for Prototype-Relation Network. D_k represents all samples in the class k.

Input: Training set $D = (x_i, y_i)_{i=1}^{N} y_i \in \{1, \cdots, K\}$.
Output: The loss L for a randomly generated training episode.
Select N_c randomly from K classes to construct sample set S and query set Q.
for k in $\{1, \cdots, N_c\}$ **do**
  Select N_s randomly from D_k to constitute S_k
  Select N_q randomly from $(D_k - S_k)$ to constitute Q_k
  $S = S + S_k$
  $Q = Q + Q_k$
  $p_k = \frac{1}{N_s} \sum_{(x_i, y_i) \in S_k} f(x_i)$
end for
$L = 0$
for k in $\{1, \cdots, N_c\}$ **do**
  for (x_j, y_j) in Q_k **do**
    Optimize (SGD, Adam)
  end for
end for

Then, calculate the loss of all samples in the query set Q.

$$Loss = \frac{1}{N_c} \sum_{k=1}^{N_c} Loss_k \tag{4}$$

where N_q is the number of query images of each class per episode.

At last, SGD optimizer is used to minimize the loss and optimize the model parameters. The training process also utilizes episode-based strategy, which is the formation sample set and query set. Suppose there is a training set $D = (x_i, y_i)_{i=1}^{N} y_i \in \{1, \cdots, K\}$, where N is the number of examples in the training set, K is the number of classes in the training set. Training episodes are formed by randomly selecting $N_c \leq K$ classes from the training set, then choosing N_s samples within each class to serve as the support set and N_q samples of the remainder to act as query set. Pseudocode that forms an episode and training models is provided in Algorithm 1.

4. Experiments

For few-shot learning, we evaluate the performance of our model on two datasets: Omniglot [50] and the *mini*ImageNet version of ILSVRC-2012 [51]. All the experiments are implemented based on Tensorflow. All of our model were trained via SGD with Adam [52] using an initial learning rate of 10^{-3}. All our model are end-to-end trained from scratch with no additional dataset.

4.1. Omniglot few-shot classification

4.1.1. Dataset

Omniglot [50] is a dataset of 1623 handwritten characters (classes) from 50 different alphabets. There are 20 examples associated with each class, where each example is drawn by different people. We follow the few-shot classification setting proposed by Vinyals et al. [39], in which the grayscale images are resized to 28×28 pixels and rotations in multiples of 90° are applied to augmenting the character classes, yielding 6492 classes in total. These are split into 4112 training classes, 688 validation classes and 1692 testing classes.

4.1.2. The affection of regularization coefficient

Compared to [20], the loss function proposed in this paper considers both inter-class distance and intra-class distance, that is, the regularization term is introduced. In order to prove the influence of different λ values on the robustness of the model, the following comparative experiments under Omniglot dataset are conducted. In all experiments, no matter train and test, if the shot=1, the number of query samples per class is 19, i.e. query=19; if the shot=5, the number of query samples per class is 15, i.e. query=15. Tables 1 and 2 show the accuracy of different λ values under different C-way K-shot settings. $\lambda = 0$ is the baseline, i.e., the loss function in [20] is adopted. From Tables 1 and 2, we found that the accuracy of test set is improved to a certain extent after introducing intra-class distance loss. Except for individual experiments, when $\lambda = 0.05$, the model can achieve the best results. Therefore, 0.05 was selected as the best regularization coefficient in this paper on Omniglot dataset.

Furthermore, aiming at the few-shot classification of C-way K-shot, we define it as C_{train} and K_{train},C_{test} and K_{test}, such as "Train: 5-way 1-shot, Test: 5-way 1-shot" is defined as "C_{train} =5 K_{train} =1, C_{test} =5, K_{test}=1". Fig. 3(a)–(p) indicate the detailed boxplot diagram for the distribution of the detailed classification results over 10 runs for each episode setting under different λ values on Omniglot dataset. As indicated in Fig. 3(a)–(p), the proposed method under $\lambda = 0.05$ achieves superior performance and outperforms all the other λ values for almost episode setting significantly except setting (d) and setting (m).

In addition, we record the accuracy changes on the support set when $\lambda = 0$ and $\lambda = 0.05$ under different settings. In this paper, two groups of settings are randomly selected: setting1 is "Train: 5-way 1-shot, Test: 20-way 1-shot", setting2 is "Train: 5-way 5-shot, Test: 20-way 5-shot". From Fig. 4, it is observed that compared with $\lambda = 0$, the model at $\lambda = 0.05$ shows significant advantages, which can achieve faster convergence and higher accuracy on the test set.

4.1.3. Comparison with baseline models

In this section, we compare our approach with 10 baseline results on Omniglot dataset. We computed classification accuracy for our models averaged over 1000 randomly generated episode from the test set. Table 3 presents the results of our proposed method and baseline models. We achieve state-of-the-art performance under all experiments setting with higher averaged accuracies, except 5-way 1-shot where our model is 99.27% and 20-way 1-shot where our model is 95.97%, the best results are 99.6% and 97.6%, respectively.

4.2. mini*ImageNet few-shot classification*

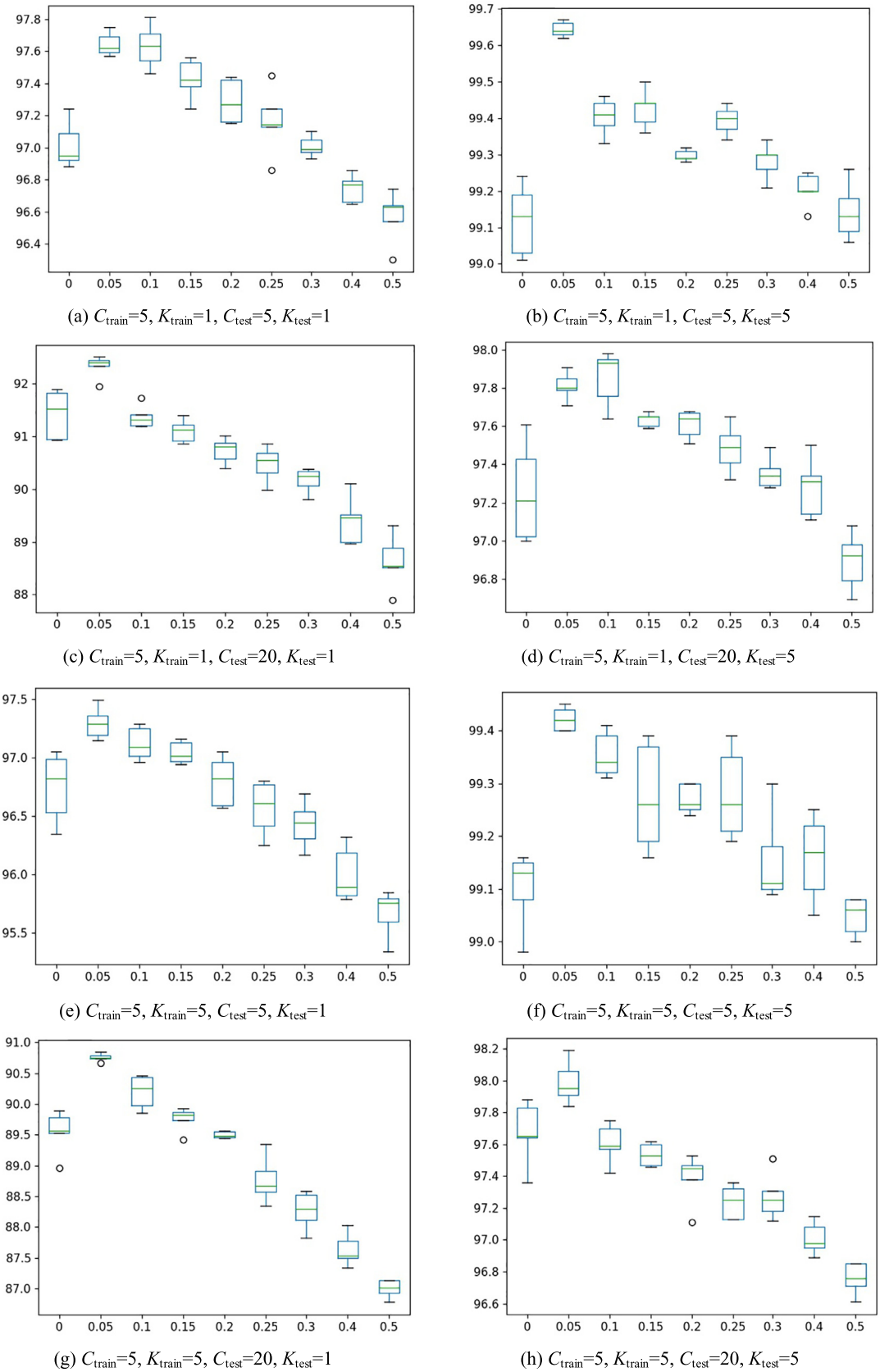
4.2.1. Dataset

The *mini*ImageNet dataset, originally proposed by Vinyals et al. [39] is a modified version of the ILSVRC-12 dataset [51], in which 600 images for each of 100 classes were randomly chosen to be part of the dataset. The splits used by Vinyals et al. [39] use 64 classes for training, 16 classes for validation and 20 classes for test. All images are of size 84×84 pixels.

4.2.2. The affection of regularization coefficient

In order to prove the influence of different λ values on the robustness of the model, the following comparative experiments under *mini*ImageNet dataset are conducted. In all experiments, the number of query samples per class is 15, i.e. query=15. Tables 4 and 5 show the accuracy of different λ values under different C-way K-shot settings.$\lambda = 0$ is the baseline, i.e., the loss function in [20] is adopted. From Tables 4 and 5, we found that the accuracy of test set is improved to a certain extent after introducing intra-class distance loss. Except for individual experiments, when $\lambda = 0.1$, the model can achieve the best results. Therefore, $\lambda = 0.1$ was selected as the best regularization coefficient in this paper on *mini*ImageNet dataset.

Furthermore, Fig. 5(a)–(h) indicate the detailed boxplot diagram for the distribution of the detailed classification results over 10 runs for each episode setting under different λ values on *mini*ImageNet dataset. As indicated in Fig. 5(a)–(h), the proposed



(a) C_{train}=5, K_{train}=1, C_{test}=5, K_{test}=1

(b) C_{train}=5, K_{train}=1, C_{test}=5, K_{test}=5

(c) C_{train}=5, K_{train}=1, C_{test}=20, K_{test}=1

(d) C_{train}=5, K_{train}=1, C_{test}=20, K_{test}=5

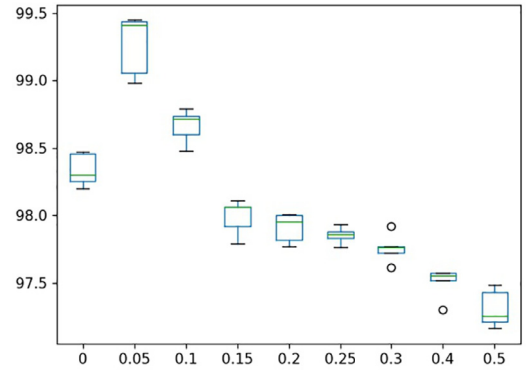
(e) C_{train}=5, K_{train}=5, C_{test}=5, K_{test}=1

(f) C_{train}=5, K_{train}=5, C_{test}=5, K_{test}=5

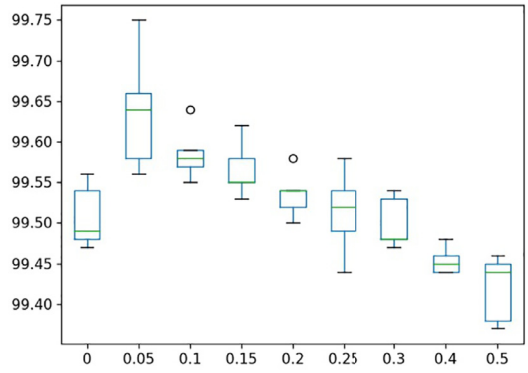
(g) C_{train}=5, K_{train}=5, C_{test}=20, K_{test}=1

(h) C_{train}=5, K_{train}=5, C_{test}=20, K_{test}=5

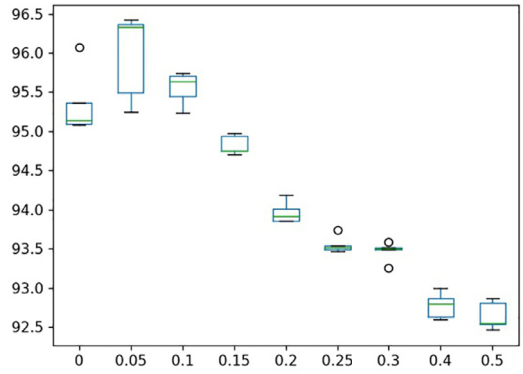
Fig. 3. Boxplot diagrams for the distribution of classification results for each episode setting under different λ values over 10 runs on Omniglot dataset.



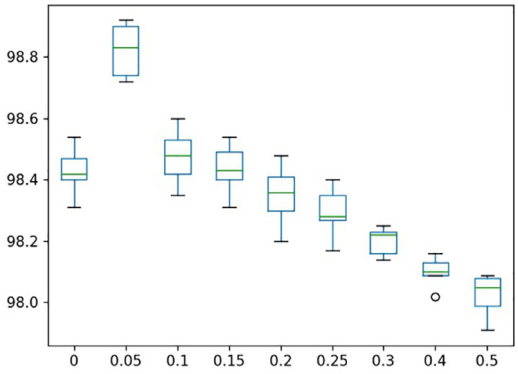
(i) C_{train}=20, K_{train}=1, C_{test}=5, K_{test}=1



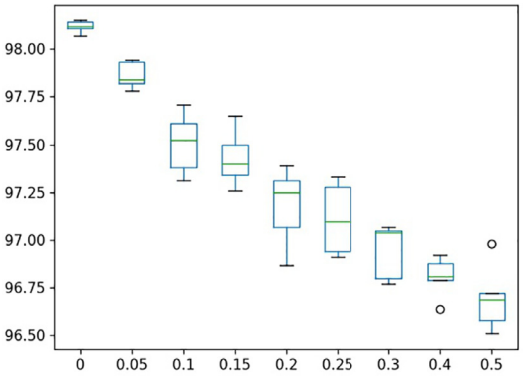
(j) C_{train}=20, K_{train}=1, C_{test}=5, K_{test}=5



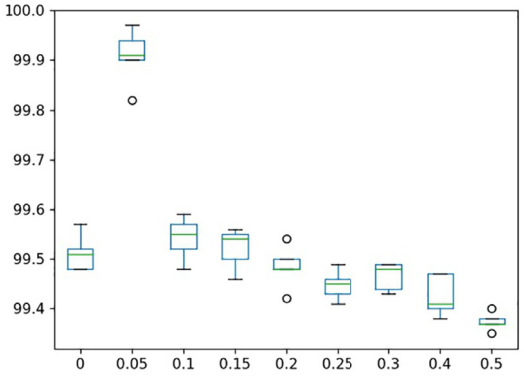
(k) C_{train}=20, K_{train}=1, C_{test}=20, K_{test}=1



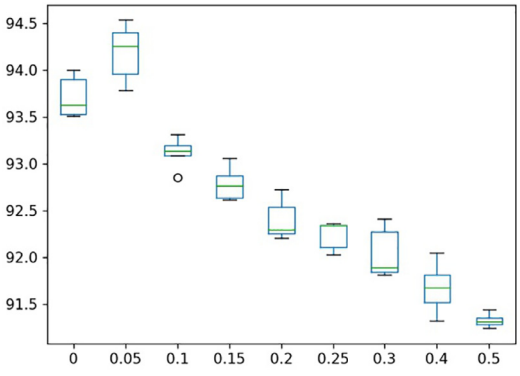
(l) C_{train}=20, K_{train}=1, C_{test}=20, K_{test}=5



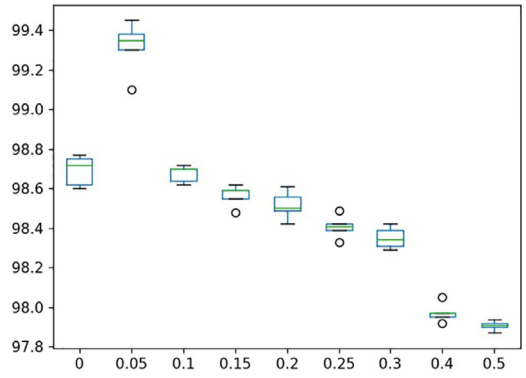
(m) C_{train}=20, K_{train}=5, C_{test}=5, K_{test}=1



(n) C_{train}=20, K_{train}=5, C_{test}=5, K_{test}=5



(o) C_{train}=20, K_{train}=5, C_{test}=20, K_{test}=1



(p) C_{train}=20, K_{train}=5, C_{test}=20, K_{test}=5

Fig. 3. Continued

Table 1
The experimental results of different λ values under Omniglot dataset.

λ	Train: 5-way 1-shot Acc. (%)				Train: 5-way 5-shot Acc. (%)			
	5-way		20-way		5-way		20-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
$\lambda = 0$	97.02	99.12	91.43	97.25	96.75	99.10	89.55	97.67
$\lambda = 0.05$	**97.68**	**99.64**	**92.33**	97.81	**97.30**	**99.42**	**90.76**	**97.99**
$\lambda = 0.1$	97.63	99.4	91.37	**97.85**	97.12	99.35	90.19	97.61
$\lambda = 0.15$	97.43	99.43	91.10	97.63	97.04	99.27	89.75	97.54
$\lambda = 0.2$	97.29	99.30	90.74	97.61	96.80	99.27	89.50	97.39
$\lambda = 0.25$	97.16	99.39	90.48	97.47	96.53	99.28	88.81	97.24
$\lambda = 0.3$	97.01	99.28	90.17	97.36	96.43	99.19	88.27	97.27
$\lambda = 0.4$	96.75	99.20	89.41	97.21	95.98	99.16	87.64	97.01
$\lambda = 0.5$	96.57	99.14	88.63	96.89	95.63	99.05	87.00	96.76

Table 2
The experimental results of different λ values under Omniglot dataset.

λ	Train: 20-way 1-shot Acc. (%)				Train: 20-way 5-shot Acc. (%)			
	5-way		20-way		5-way		20-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
$\lambda = 0$	98.34	99.51	95.35	98.43	**98.12**	99.51	93.72	98.69
$\lambda = 0.05$	**99.27**	**99.64**	**95.97**	**98.82**	97.86	**99.91**	**94.19**	**99.32**
$\lambda = 0.1$	98.67	99.59	95.55	98.48	97.51	99.54	93.12	98.68
$\lambda = 0.15$	97.99	99.57	94.82	98.43	97.43	99.52	92.79	98.57
$\lambda = 0.2$	97.91	99.54	93.97	98.35	97.18	99.48	92.41	98.52
$\lambda = 0.25$	97.85	99.51	93.55	98.29	97.11	99.45	92.24	98.41
$\lambda = 0.3$	97.76	99.50	93.47	98.20	96.95	99.47	92.05	98.35
$\lambda = 0.4$	97.50	99.45	92.77	98.10	96.81	99.43	91.68	97.97
$\lambda = 0.5$	97.31	99.42	92.65	98.02	96.70	99.37	91.32	97.91

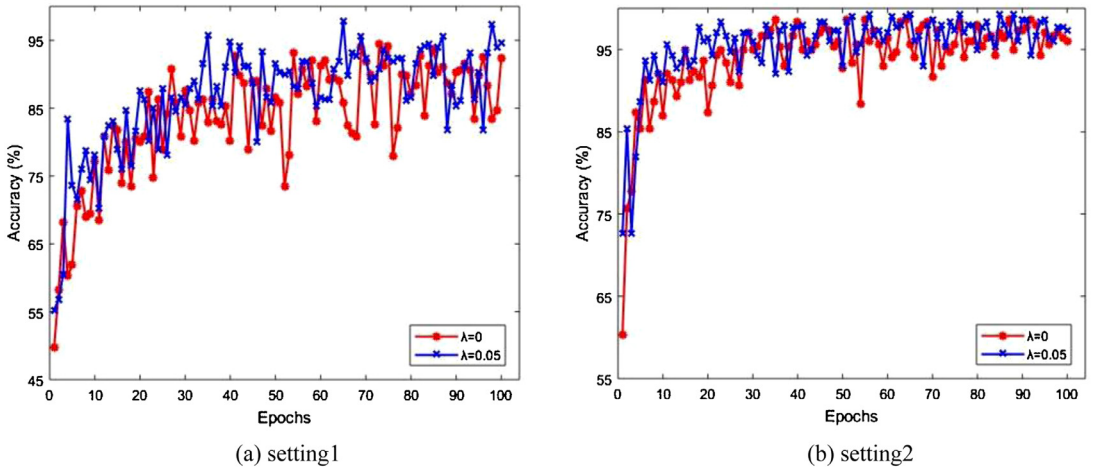


Fig. 4. The accuracy changes on the support set when $\lambda = 0$ and $\lambda = 0.05$ under different settings. Setting1 is "Train: 5-way 1-shot, Test: 20-way 1-shot". Setting2 is "Train: 5-way 5-shot, Test: 20-way 5-shot".

Table 3
Few-shot classification on Omniglot dataset. All accuracy results are averaged over 1000 test episodes and with 95% confidence intervals. The best performance method is highlighted, along with any others whose confidence intervals overlap. '-' Results is not reported.

Model	5-way Acc.		20-way Acc.	
	1-shot	5-shot	1-shot	5-shot
MANN [53]	82.8%	94.9%	–	–
CONVOLUTIONAL SIAMESE NETS [38]	96.7%	98.4%	88.0%	96.5%
MATCHING NETS [39]	98.1%	98.9%	93.8%	98.5%
SIAMESE NETS WITH MEMORY [54]	98.4%	99.6%	95.0%	98.6%
NEURAL STATISTICIAN [55]	98.1%	99.5%	93.2%	98.1%
META NETS [56]	98.95%	–	97.00%	–
GNN [59]	99.2%	99.7%	97.4%	99.0%
IMP [60]	98.4 ± 0.3%	99.5 ± 0.1%	95.0 ± 0.1%	98.6 ± 0.1%
RELATION NET [22]	**99.6 ± 0.2%**	99.8 ± 0.1%	**97.6 ± 0.2%**	99.1 ± 0.1%
PROTOTYPE NETS [20]	98.7%	99.6%	95.4%	98.8%
PROTOTYPE-RELATION NETS	99.27 ± 0.23%	**99.91 ± 0.06%**	95.97 ± 0.56%	**99.32 ± 0.13%**

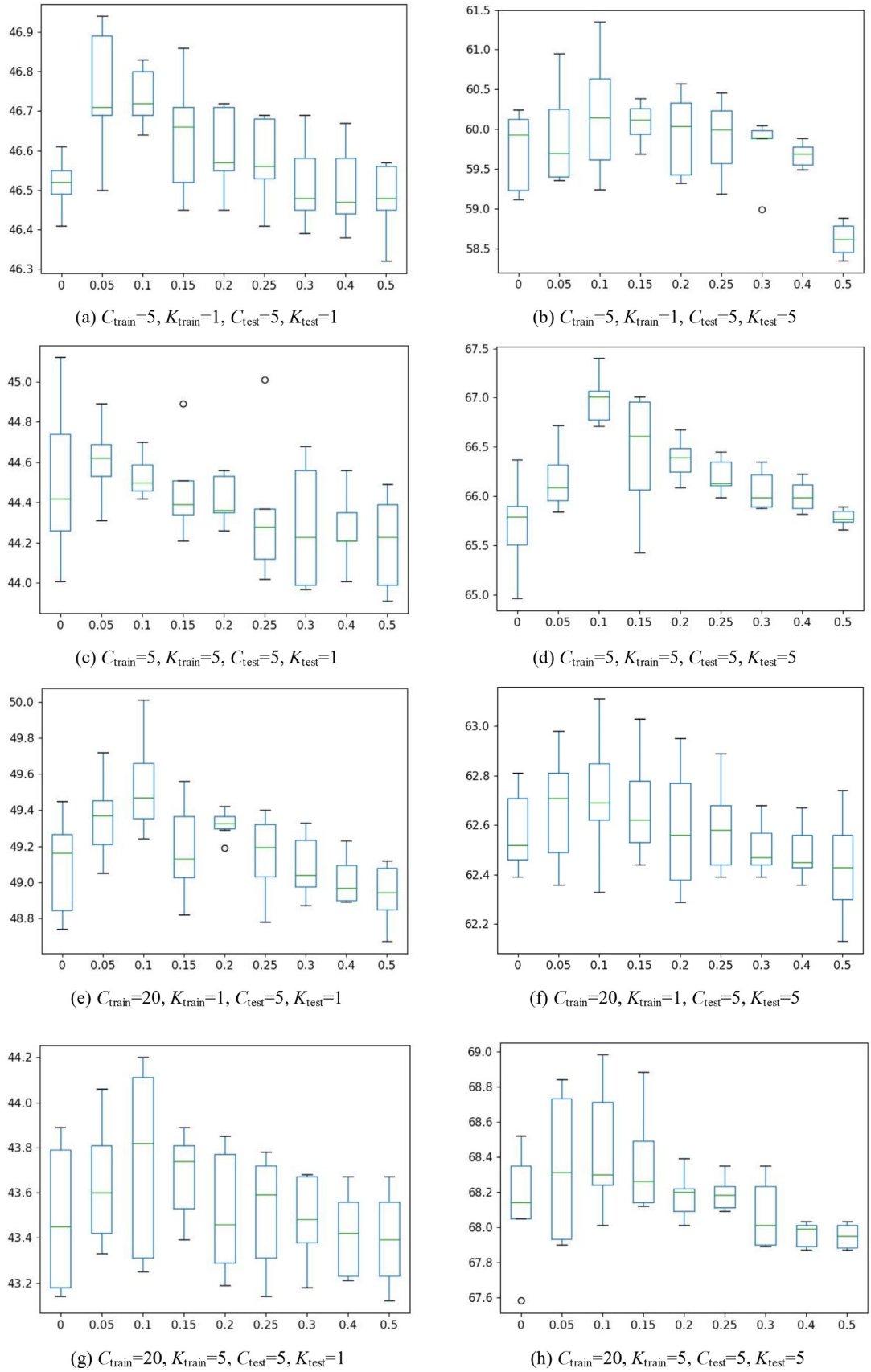


Fig. 5. Boxplot diagrams for the distribution of classification results for each episode setting under different λ values over 10 runs on *mini*ImageNet dataset.

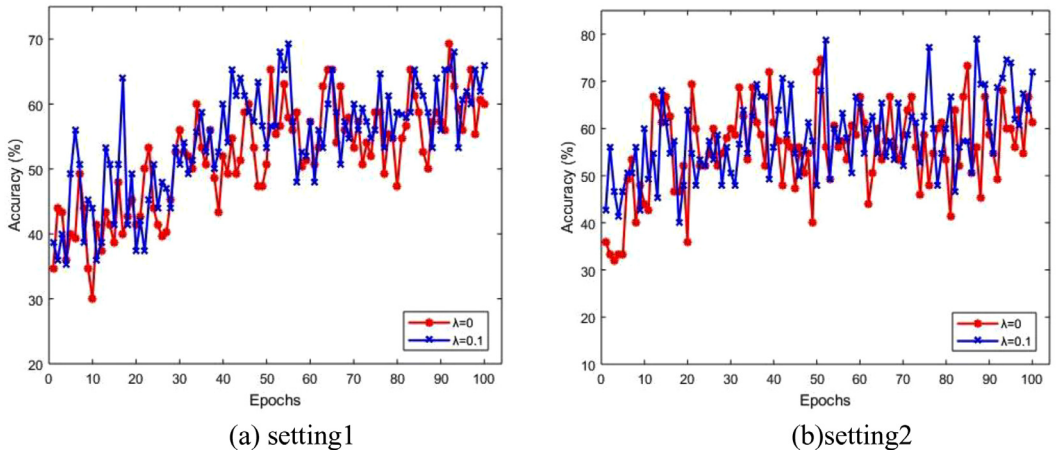


Fig. 6. The accuracy changes on the support set when $\lambda = 0$ and $\lambda = 0.1$ under different settings. Setting1 is "Train: 5-way 1-shot, Test: 20-way 1-shot". Setting2 is "Train: 5-way 5-shot, Test: 20-way 5-shot".

Table 4
The experimental results of different λ values under *mini*ImageNet dataset.

λ	Train: 5-way 1-shot Acc. (%)		Train: 5-way 5-shot Acc. (%)	
	5-way 1-shot	5way 5-shot	5-way 1-shot	5-way 5-shot
$\lambda = 0$	46.52	59.73	44.51	65.71
$\lambda = 0.05$	46.73	59.91	**44.61**	66.19
$\lambda = 0.1$	**46.74**	**60.19**	44.53	**66.59**
$\lambda = 0.15$	46.64	60.08	44.47	66.42
$\lambda = 0.2$	46.60	59.94	44.41	66.38
$\lambda = 0.25$	46.56	59.89	44.36	66.21
$\lambda = 0.3$	46.52	59.76	44.29	66.10
$\lambda = 0.4$	46.51	59.68	44.27	66.01
$\lambda = 0.5$	46.48	58.61	44.20	65.78

Table 5
The experimental results of different λ values under *mini*ImageNet dataset.

λ	Train: 20-way 1-shot Acc. (%)		Train: 20-way 5-shot Acc. (%)	
	5-way 1-shot	5way 5-shot	5-way 1-shot	5-way 5-shot
$\lambda = 0$	49.09	62.56	43.49	68.13
$\lambda = 0.05$	49.36	62.69	43.62	**68.34**
$\lambda = 0.1$	**49.54**	**62.72**	**43.68**	68.31
$\lambda = 0.15$	49.38	62.68	43.61	68.24
$\lambda = 0.2$	49.27	62.59	43.59	68.22
$\lambda = 0.25$	49.15	62.57	43.52	68.19
$\lambda = 0.3$	49.09	62.51	43.48	68.08
$\lambda = 0.4$	49.01	62.49	43.42	68.02
$\lambda = 0.5$	48.94	62.43	43.39	67.99

method under $\lambda = 0.1$ achieves superior performance and outperforms all the other λ values for almost episode setting significantly except setting (c) and setting (h).

In addition, we record the accuracy changes on the support set when $\lambda = 0$ and $\lambda = 0.1$ under different settings. In this paper, two groups of settings are randomly selected: setting1 is "Train: 5-way 1-shot, Test: 5-way 5-shot", setting2 is "Train: 20-way 1-shot, Test: 5-way 5-shot". From Fig. 6, we observed that compared with $\lambda = 0$, the model at $\lambda = 0.1$ shows faster convergence and higher accuracy on the test set.

4.2.3. Comparison with baseline models

In this section, we compare our approach with 7 baseline results on *mini*ImageNet dataset. We computed classification accuracy for our models averaged over 1000 randomly generated episode from the test set. Table 6 presents the results of our proposed method and baseline models. We achieved state-of-the-art performance under all experiments setting with higher averaged accuracies and lower standard deviations, except 5-way 1-shot where our model is 0.9% lower in accuracy than [22].

Table 6
Few-shot classification on *mini*ImageNet dataset. All accuracy results are averaged over 1000 test episodes and with 95% confidence intervals. The best performance method is highlighted, along with any others whose confidence intervals overlap.

Model	5-way Acc.	
	1-shot	5-shot
BASELINE NEAREST NEIGHBORS	28.86 ± 0.54%	49.79 ± 0.79%
MATCHING NETS [39]	43.40 ± 0.78%	51.09 ± 0.71%
MATCHING NETS FCE [39]	43.56 ± 0.84%	55.31 ± 0.73%
META-LEARNING LSTM [57]	43.44 ± 0.77%	60.60 ± 0.71%
META NETS [56]	49.21 ± 0.96%	–
MAML [58]	48.70 ± 1.84%	63.11 ± 0.92%
IMP [60]	49.6 ± 0.8%	68.1 ± 0.8%
GNN [59]	50.33 ± 0.36%	66.41 ± 0.63%
RELATION NET [22]	50.44 ± 0.82%	65.32 ± 0.70%
PROTOTYPICAL NETS [20]	49.42 ± 0.78%	68.20 ± 0.66%
PROTOTYPE-RELATION NETS	49.54 ± 0.09%	**68.34 ± 0.06%**

4.3. Relation of episode between train and test

We also compared Prototype-Relation Networks trained with a different number of classes per episode ("*C*-way") and a different number of points per class ("*K*-shot") on Omniglot and *mini*ImageNet datasets: 5-way vs. 20-way and 1-shot vs. 5-shot, i.e. the relation of C_{train} and C_{test}, K_{train} and K_{test}. We found that the construction of training episodes is an important consideration in order to achieve better results for few-shot classification.

From Table 7, we note that when K_{test} is larger than K_{train} under the same *C*-way, i.e. C_{test}=C_{train}, the model performance is increasing. For example, when C_{train}=C_{test}=5, K_{train} =1, the accuracy of K_{test}=5 (99.64%) is superior to the accuracy of K_{test}=1 (97.68%) on Omniglot dataset and the accuracy of K_{test}=5 (60.19%) is superior to the accuracy of K_{test}=1 (46.74%) on *mini*ImageNet dataset. Another example, when C_{train}=C_{test}=20, K_{train} =1, the accuracy of K_{test}=5 (98.82%) is superior to the accuracy of K_{test}=1 (95.97%) on Omniglot dataset. In addition, we have found that it can be greatly beneficial to train with a higher C_{train} than C_{test} under the same *K*-shot. For example, when K_{train}=K_{test}=1, C_{test}=5, the accuracy of C_{train}=20 (99.27%) is better than the accuracy of C_{train}=5 (97.68%) on Omniglot dataset and when K_{train}=K_{test}=5, C_{test}=5, the accuracy of C_{train}=20 (68.31%) is better than the accuracy of C_{train}=5 (66.59%) on *mini*ImageNet dataset, i.e. 20-way achieves higher accuracy than 5-way. It inferred that the model achieves

Table 7
Comparison of "way" of train and test, "shot" of train and test on Omniglot and *mini*ImageNet under 5-way vs. 20-way and 1-shot vs. 5-shot. Classification accuracy is averaged over 1000 randomly generated episodes from the test set and 95% confidence intervals are shown.

Dist.	Dataset	Train Episodes			C_{test}=5 Acc.		C_{test}=20 Acc.	
		C_{train}	K_{train}	*Query*	K_{test}=1	K_{test}=5	K_{test}=1	K_{test}=5
Euclid.	Omniglot	5	1	19	97.68%	99.64%	92.33%	97.81%
		5	5	15	97.30%	99.42%	90.76%	97.99%
		20	1	19	99.27%	99.64%	95.97%	98.82%
		20	5	15	97.86%	99.91%	94.10%	99.32%
	*mini*ImageNet	5	1	15	46.74%	60.19%	–	–
		5	5	15	44.53%	66.59%	–	–
		20	1	15	49.54%	62.72%	–	–
		20	5	15	43.68%	68.31%	–	–

better results on harder training tasks because more classes allow the model to have better generalization performance and it forces the model to make more fine-grained decisions in the embedding space.

5. Conclusion

In this paper, we proposed a simple network architecture named Prototype-Relation Network and a novel loss function which takes into account inter-class and intra-class distance for few-shot classification. The idea of meta-learning is adopted and the meta-task of each training is constructed based on episode paradigm. The approach is far simpler and more efficient than recent few-shot meta-learning approaches. We show that when the regularization coefficient $\lambda = 0.05$ for Omniglot dataset and $\lambda = 0.1$ for *mini*ImageNet, the model achieves the best performance under different *C*-way *K*-shot, which produces the state-of-the-art results. We further found that it is usual to train with a higher "way" and to test with a smaller "shot". And it is demonstrated that the construction of training episodes is an important consideration in order to achieve better results.

For future work, we would like to extend our method from supervised to semi-supervised learning to take advantage of the large amount of unlabeled data. In addition, we also would like to investigate the application of few-shot learning in facial expression recognition.

Declaration of Competing Interest

None.

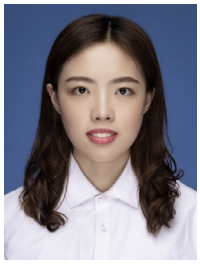
Acknowledgements

Project supported by The National Key R & D Program of China (Grant No. 2017YFB1302400), National Natural Science Foundation of China (Grant No. 61773242, No. 61803227), Major Agricultural Applied Technological Innovation Projects of Shandong Province (SD2019NJ014). Shandong Natural Science Foundation (ZR2019MF064), Intelligent Robot and System Innovation Center Foundation (2019IRS19).

References

[1] X. Xi, Y. Luo, P. Wang, et al., Salient object detection based on an efficient end-to-end saliency regression network, Neurocomputing 323 (2019) 265–276.
[2] Z. Liu, Q. Li, W. Li, Deep layer guided network for salient object detection, Neurocomputing 372 (2020) 55–63.
[3] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. (2012) 1097–1105.
[4] K. He, X. Zhang, S. Ren, et al., Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2016, pp. 770–778.
[5] K. He, X. Zhang, S. Ren, et al., Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2016, pp. 770–778.
[6] G. Huang, Z. Liu, L. Van Der Maaten, et al., Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2017, pp. 4700–4708.
[7] S. Qiao, C. Liu, W. Shen, et al., Few-shot image recognition by predicting parameters from activations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2018, pp. 7229–7238.
[8] C. Szegedy, W. Liu, Y. Jia, et al., Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2015, pp. 1–9.
[9] Y. Wang, L. Xie, C. Liu, et al., Sort: second-order response transform for visual recognition, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2017, pp. 1359–1368.
[10] Qiao S., Zhang Z., Shen W., et al., Gradually updated neural networks for large-scale image recognition. arXiv:1711.09280 (2017).
[11] X. Fang, S. Teng, Z. Lai, et al., Robust latent subspace learning for image classification, IEEE Trans. Neural Netw. Learn. Syst. 29 (6) (2017) 2502–2515.
[12] Y. Lu, Z. Lai, Y. Xu, et al., Low-rank preserving projections, IEEE Trans. Cybern. 46 (8) (2015) 1900–1913.
[13] Y. Lu, C. Yuan, W. Zhu, et al., Structurally incoherent low-rank nonnegative matrix factorization for image classification, IEEE Trans. Image Process. 27 (11) (2018) 5248–5260.
[14] Y. Lu, C. Yuan, Z. Lai, et al., Horizontal and vertical nuclear norm-based 2DLDA for image representation, IEEE Trans. Circ. Syst. Video Technol. 29 (4) (2018) 941–955.
[15] Oord A., Dieleman S., Zen H., et al., Wavenet: a generative model for raw audio. arXiv:1609.03499 (2016).
[16] G. Hinton, L. Deng, D. Yu, et al., Deep neural networks for acoustic modeling in speech recognition, IEEE Signal Process. Mag. 29 (2012) 82–97.
[17] Wu Y., Schuster M., Chen Z., et al., Google's neural machine translation system: bridging the gap between human and machine translation. arXiv:1609.08144 (2016).
[18] O. Russakovsky, J. Deng, H. Su, et al., Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[19] T.Y. Lin, M. Maire, S. Belongie, et al., Microsoft coco: common objects in context, in: Proceedings of the European Conference on Computer Vision, 2014, pp. 740–755.
[20] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Advances in Neural Information Processing Systems, 2017, pp. 4077–4087.
[21] B. Lake, R. Salakhutdinov, J. Gross, et al., One shot learning of simple visual concepts, in: Proceedings of the Annual Meeting of the Cognitive Science Society, 33, 2011 pp.
[22] F. Sung, Y. Yang, L. Zhang, et al., Learning to compare: relation network for few-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1199–1208.
[23] S. Thrun, Lifelong learning algorithms, in: Learning to Learn, 1998, pp. 181–209.
[24] S. Hochreiter, A.S. Younger, P.R. Conwell, Learning to learn using gradient descent, in: Proceedings of the International Conference on Artificial Neural Networks, 2001, pp. 87–94.
[25] Triantafillou E, Larochelle H, Snell J, et al. Meta-learning for semi-supervised few-shot classification, (2018).
[26] E.G. Miller, N.E. Matsakis, P.A. Viola, Learning from one example through shared densities on transforms, in: Proceedings IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2000, pp. 464–471.
[27] L. Fei-Fei, R. Fergus, P. Perona, One-shot learning of object categories, IEEE Trans. Pattern Anal. Mach. Intell. 28 (4) (2006) 594–611.
[28] W. Zheng, C. Gou, F.Y. Wang, A novel approach inspired by optic nerve characteristics for few-shot occluded face recognition, Neurocomputing (2019), doi:10.1016/j.neucom.2019.09.045.
[29] D. Wang, Y. Cheng, M. Yu, et al., A hybrid approach with optimization-based and metric-based meta-learner for few-shot learning, Neurocomputing 349 (2019) 202–211.
[30] J. Nie, N. Xu, M. Zhou, et al., 3D Model classification based on few-shot learning, Neurocomputing (2019), doi:10.1016/j.neucom.2019.03.105.
[31] Chen Z, Fu Y, Zhang Y, et al. Semantic feature augmentation in few-shot learning. arXiv:1804.05298, 2018, 86: 89.
[32] B. Hariharan, R. Girshick, Low-shot visual recognition by shrinking and halluci-

nating features, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 3018–3027.
[33] M. Dixit, R. Kwitt, M. Niethammer, et al., Aga: attribute-guided augmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 7455–7463.
[34] X. Fang, Y. Xu, X. Li, et al., Robust semi-supervised subspace clustering via non-negative low-rank representation, IEEE Trans. Cybern. 46 (8) (2015) 1828–1838.
[35] J. Wen, X. Fang, J. Cui, et al., Robust sparse linear discriminant analysis, IEEE Trans. Circ. Syst. Video Technol. 29 (2) (2018) 390–403.
[36] J. Wen, N. Han, X. Fang, et al., Low-rank preserving projection via graph regularized reconstruction, IEEE Trans. Cybern. 49 (4) (2018) 1279–1291.
[37] Y. Lu, Z. Lai, X. Li, et al., Low-rank 2-D neighborhood preserving projection for enhanced robust image representation, IEEE Trans. Cybern. 49 (5) (2018) 1859–1872.
[38] G. Koch, R. Zemel, R. Salakhutdinov, Siamese neural networks for one-shot image recognition, in: Proceedings of the ICML Deep Learning Workshop, 2, 2015.
[39] O. Vinyals, C. Blundell, T. Lillicrap, et al., Matching networks for one shot learning, in: Advances in Neural Information Processing Systems, 2016, pp. 3630–3638.
[40] X. Fang, N. Han, W.K. Wong, et al., Flexible affinity matrix learning for unsupervised and semisupervised classification, IEEE Trans. Neural Netw. Learn. Syst. 30 (4) (2018) 1133–1149.
[41] Y. Bengio, Learning deep architectures for AI, Found. Trends® Mach. Learn. 2 (1) (2009) 1–127.
[42] G.E. Hinton, S. Osindero, Y.W. Teh, A fast learning algorithm for deep belief nets, Neural Comput 18 (7) (2006) 1527–1554.
[43] Santoro A., Bartunov S., Botvinick M., et al., One-shot learning with memory-augmented neural networks, arXiv:1605.06065 (2016).
[44] M. Andrychowicz, M. Denil, S. Gomez, et al., Learning to learn by gradient descent by gradient descent, in: Advances in Neural Information Processing Systems, 2016, pp. 3981–3989.
[45] L. Bertinetto, J.F. Henriques, J. Valmadre, et al., Learning feed-forward one-shot learners, in: Advances in Neural Information Processing Systems, 2016, pp. 523–531.
[46] D. Maclaurin, D. Duvenaud, R. Adams, Gradient-based hyperparameter optimization through reversible learning, in: Proceedings of the International Conference on Machine Learning, 2015, pp. 2113–2122.
[47] J. Goldberger, G.E. Hinton, S.T. Roweis, et al., Neighbourhood components analysis, in: Advances in Neural Information Processing Systems, 2005, pp. 513–520.
[48] K.Q. Weinberger, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, J. Mach. Learn. Res. 10 (2) (2009) 207–244.
[49] R. Min, D.A. Stanley, Z. Yuan, et al., A deep non-linear feature mapping for large-margin knn classification, in: Proceedings of the Ninth IEEE International Conference on Data Mining, IEEE, 2009, pp. 357–366.
[50] B. Lake, R. Salakhutdinov, J. Gross, et al., One shot learning of simple visual concepts, in: Proceedings of the Annual Meeting of the Cognitive Science Society, 33, 2011.
[51] O. Russakovsky, J. Deng, H. Su, et al., Imagenet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (3) (2015) 211–252.
[52] Kingma D.P., Ba J., Adam: a method for stochastic optimization, arXiv:1412.6980 (2014).
[53] A. Santoro, S. Bartunov, M. Botvinick, et al., Meta-learning with memory-augmented neural networks, in: Proceedings of the International Conference on Machine Learning, 2016, pp. 1842–1850.
[54] Kaiser Ł., Nachum O., Roy A., et al., Learning to remember rare events, arXiv:1703.03129 (2017).
[55] Edwards H., Storkey A., Towards a neural statistician, arXiv:1606.02185 (2016).
[56] T. Munkhdalai, H. Yu, Meta networks, in: Proceedings of the 34th International Conference on Machine Learning, 2017, pp. 2554–2563.
[57] Ravi S., Larochelle H., Optimization as a model for few-shot learning, 2016.
[58] C. Finn, P. Abbeel, S. Levine, Model-agnostic meta-learning for fast adaptation of deep networks, in: Proceedings of the 34th International Conference on Machine Learning, 2017, pp. 1126–1135.
[59] Garcia V., Bruna J., Few-shot learning with graph neural networks, arXiv:1711.04043 (2017).
[60] Allen K.R., Shelhamer E., Shin H., et al., Infinite mixture prototypes for few-shot learning, arXiv:1902.04552 (2019).



Xiaoqian Liu received the B.S. degree in automation in 2017 from Shandong University of Science and Technology, Qingdao, China. She is currently pursuing the M.S. degree in control engineering at Shandong University, Jinan, China. Her research interests include data mining, pattern recognition, computer vision, image processing and deep learning techniques.

Fengyu Zhou received the Ph.D. degree in electrical engineering from Tianjin University, Tianjin, China, in 2008. He is currently a professor of the School of Control Science and Engineering at Shandong University, Jinan, China. His research interests include service robotics, automation, pattern recognition, image processing,computer vision and Cloud robotics.

Jin Liu received the B.S. degree in Control Technology and Instruments in 2017 from Qingdao University, Qingdao, China. He is currently pursuing the M.S. degree in control engineering at Shandong University, Jinan, China. His research interests include cloud platform, Particle Swarm Optimization (PSO), data mining, computer vision, and deep learning techniques.

Lianjie Jiang received the B.S. degree in automation in 2019 from Shandong University of Science and Technology, Qingdao, China. He is currently pursuing the M.S. degree in control engineering at Shandong University, Jinan, China. His research interests include time series forecasting, machine learning, computer vision, and AI techniques.