

# BSNet: Bi-Similarity Network for Few-shot Fine-grained Image Classification

Xiaoxu Li, Jijie Wu, Zhuo Sun, Zhanyu Ma, *Senior Member, IEEE*, Jie Cao, Jing-Hao Xue, *Member, IEEE*

**Abstract**—Few-shot learning for fine-grained image classification has gained recent attention in computer vision. Among the approaches for few-shot learning, due to the simplicity and effectiveness, metric-based methods are favorably state-of-the-art on many tasks. Most of the metric-based methods assume a single similarity measure and thus obtain a single feature space. However, if samples can simultaneously be well classified via two distinct similarity measures, the samples within a class can distribute more compactly in a smaller feature space, producing more discriminative feature maps. Motivated by this, we propose a so-called *Bi-Similarity Network (BSNet)* that consists of a single embedding module and a bi-similarity module of two similarity measures. After the support images and the query images pass through the convolution-based embedding module, the bi-similarity module learns feature maps according to two similarity measures of diverse characteristics. In this way, the model is enabled to learn more discriminative and less similarity-biased features from few shots of fine-grained images, such that the model generalization ability can be significantly improved. Through extensive experiments by slightly modifying established metric/similarity based networks, we show that the proposed approach produces a substantial improvement on several fine-grained image benchmark datasets.

Codes are available at: <https://github.com/spraise/BSNet>

**Index Terms**—Fine-grained image classification, Deep neural network, Few-shot learning, Metric learning

## I. INTRODUCTION

Deep learning models have achieved great success in visual recognition [1]–[3]. These enormous neural networks often require a large number of labeled instances to learn their parameters [4]–[6]. However, it is hard to obtain labeled image in many cases [7]. Furthermore, human can adapt fast based on transferable knowledge from previous experience and learn new concepts from only few observations [8]. Thus, it is of great importance to design deep learning algorithms to have such abilities [9]. Few-shot learning [10], [11] rises in response to these urgent demands, which aims to learn latent patterns from few labeled images [12].

Many few-shot classification algorithms can be grouped into two branches: meta-learning based methods [13]–[19] and metric-based methods [20]–[22]. Meta-learning based methods focus on how to learn good initializations [14], optimizers [23], or parameters. metric-based methods focus

on how to learn good feature embeddings, similarity measures [24], or both of them [11]. Due to their simplicity and effectiveness, metric-based methods have achieved the state-of-the-art performance on fine-grained images [22], [23].

Fine-grained image datasets are often used to evaluate the performance of few-shot classification algorithms [22], [23], [25]–[28], because they generally contain many sub-categories and each sub-category includes only a small amount of data [29]. Due to the similarity between these small sub-categories, a key consideration in few-shot fine-grained image classification is how to learn discriminative features from few labeled images [30]–[32].

Metric-based methods in few-shot classification are mostly built on a single similarity measure [22], [23], [25]–[28]. Intuitively, features obtained by adapting a single similarity metric are only discriminative in a single feature space. That is, using one single similarity measure may induce certain similarity bias that lowers the generalization ability of the model, in particular when the amount of training data is small. Thus, as illustrated in Figure 1, if the obtained features can simultaneously adapt two similarity measures of diverse characteristics, the samples within one class can be mapped more compactly into a smaller feature space. This will result in a model embedding two diverse similarity measures, generating more discriminative features than using a single measure.

Motivated by this, we develop a *Bi-Similarity Network (BSNet)*. Within each task generated from a meta-training dataset, the proposed *BSNet* employs two similarity measures, *e.g.*, the Euclidean distance and the cosine distance. To be more specific, the proposed *BSNet* contains two modules as illustrated in Figure 2. The first module is a convolution-based embedding module, generating representations of query and support images. The learned representations are then fed forward to the second module, *i.e.* our bi-similarity module, producing two similarity measurements between a query image and each class. During meta-training process, the total loss is the summation of the losses of both heads, which is used for back-propagating the network’s parameters jointly. We evaluate the proposed method on several fine-grained benchmark datasets. Extensive experiments demonstrate the state-of-the-art performance of the proposed *BSNet*. Our contributions are three-fold:

- 1) We propose a *BSNet* that leverages two similarity measures and significantly improves the performance of four state-of-the-art few-shot classification methods [11], [20]–[22] on four benchmark fine-grained image datasets.

X. Li, J. Wu and J. Cao are with the School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China.

Z. Ma is with the Pattern Recognition and Intelligent System Laboratory, School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing 100876, China.

Z. Sun and J.-H. Xue are with the Department of Statistical Science, University College London, London, WC1E 6BT, U.K.

X. Li, J. Wu and Z. Sun contribute equally.

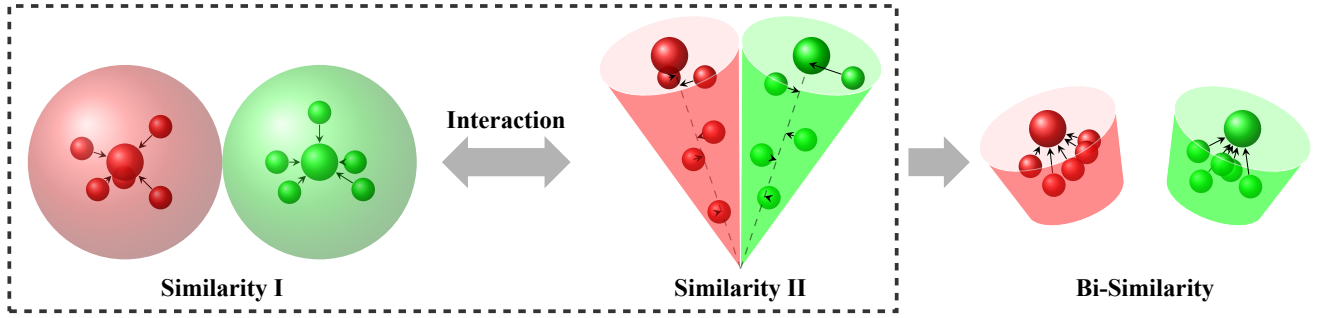


Figure 1. Motivation of the proposed *Bi-Similarity Network (BSNet)*. Here we use the Euclidean distance and the cosine distance as the similarity measures in feature spaces. The Euclidean distance and the cosine distance are for Similarity I and Similarity II, respectively. Different colors indicate different predicted labels; larger balls are class prototypes.

- 2) We demonstrate that the model complexity of *BSNet* is less than the mean value of model complexities of two single-similarity networks, even though *BSNet* contains more model parameters.
- 3) We demonstrate that the proposed *BSNet* learns discriminative areas of the input images via visualization.

## II. RELATED WORK

Recently, deep neural networks have been carefully redesigned to solve few-shot learning problems. In this work, we are particularly interested in metric-based few-shot learning models as they are most relevant to the proposed *BSNet*. Existing metric-based few-shot learning methods roughly fall into the following three groups according to their innovation:

a) *Learning feature embedding given a metric*: Koch et al. [33] used a Siamese convolutional neural network for one-shot learning, in which convolution layers are of VGG [34] styled structure and  $L_1$  distance is used to compute the distance between image features. *Matching Network* [20] introduced attention mechanisms along with the cosine metric on deep features of images. Most of existing methods, for instance [11], [21], [35], [36], have benefited from the episodic meta-training strategy proposed in [20], learning transferable knowledge from labeled support images to query images through this strategy. Wu et al. [37] added a deformed feature extractor and dual correlation attention module into the *Relation Network* [11], and proposed position-aware relation network (or *PARN*, for short). As a result, *PARN* outperformed the *Relation Network* on the Omniglot and Mini-ImageNet datasets. This group of methods mainly learn a feature representation that adapts to a fixed metric. Unlike them, the proposed *BSNet* learns metrics and a feature representation simultaneously.

b) *Learning a prototype*: Despite of similar meta-training procedures, these methods are built on different hypotheses. [21] proposed the *Prototype Network* that used the Euclidean distance as metric to classify query images to class-specific prototypes, producing linear decision boundaries in the learned embedding space as it implicitly assumed identical covariance matrices for all class-specific Gaussian distributions. Allen et al. [35] further extended the *Prototype Network* [21] by allowing multiple clusters for each class, and

hence proposed infinite mixture prototype networks. Unlike these methods, the proposed *BSNet* does not learn a prototype, but introduces the bi-similarity idea into the metric-based few-shot learning models.

c) *Learning a distance (or similarity) metric*: Instead of using an existing metric, Sung et al. [11] proposed the *Relation Network* to compute similarity scores to measure the similarity between each query image and support images from each class. Li et al. [22] constructed an image-to-class module after a CNN-based embedding module to compute the cosine similarity between deep local descriptors of a query image and its neighbors in each class, known as *Deep Nearest Neighbor Neural Network* (or *DN4* in short). Zhang et al. [38] learned Gaussian distributions for each class via incorporating evidence lower bound into the design of loss function and used the posterior densities of query images as the criterion, which produced quadratic decision boundaries as it allowed class-specific covariance matrices. It can be viewed as learning the Mahalanobis distance in deep feature space. The proposed *BSNet* is closely related to this group of methods. To the best of our knowledge, most current studies in few-shot learning only focus on a single similarity measure. However, the proposed work constructs a network leveraging one module with two similarity measures.

## III. THE PROPOSED BI-SIMILARITY NETWORK

### A. Problem Formulation

Few-shot classification problems are often formalized as  $C$ -way  $K$ -shot classification problems, in which models are given  $K$  labeled images from each of  $C$  classes, and required to correctly classify unlabeled images. Unlike traditional classification problems where the label set in training set is identical to those in the validation and test sets, few-shot classification requires to classify novel classes after training. This requires that images used for training, validation and testing should come from disjoint label sets. To be more specific, given a dataset  $\mathcal{D} = \{(x_i, y_i), y_i \in \mathcal{L}\}_{i=1}^N$ , we divide it into three parts, namely  $\mathcal{D}_{train} = \{(x_i, y_i), y_i \in \mathcal{L}_{train}\}_{i=1}^{N_{train}}$ ,  $\mathcal{D}_{val} = \{(\tilde{x}_i, \tilde{y}_i), \tilde{y}_i \in \mathcal{L}_{val}\}_{i=1}^{N_{val}}$ ,  $\mathcal{D}_{test} = \{(x_i^*, y_i^*), y_i^* \in \mathcal{L}_{test}\}_{i=1}^{N_{test}}$ , where  $(x_i, y_i)$  represents the raw feature vector and label information for the  $i$ th image. The label sets,  $\mathcal{L}_{train}$  and  $\mathcal{L}_{val}$  and  $\mathcal{L}_{test}$ , are disjoint and their union is  $\mathcal{L}$ .

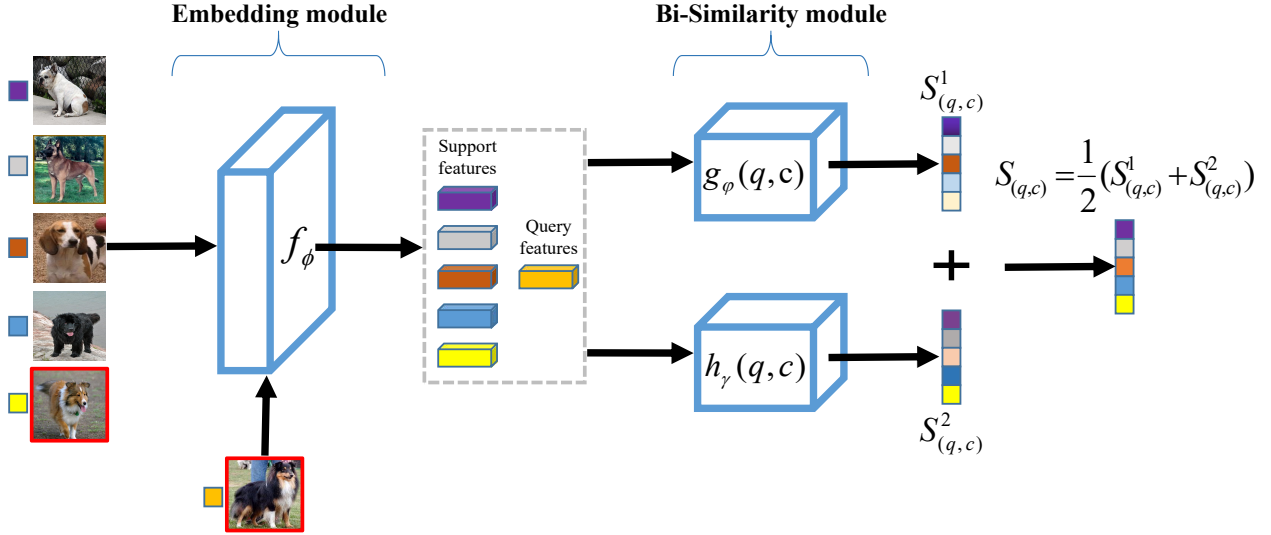


Figure 2. Illustration of the proposed *Bi-Similarity Network (BSNet)*. It consists of one embedding module  $f_\phi$ , followed by a bi-similarity module which outputs two similarity scores between a query image and  $C$  class prototypes in  $C$ -way  $K$ -shot problems.  $S_{(q,c)}^k$  denotes the  $k$ th similarity score between the  $q$ th query image  $x_q$  and the  $c$ th class in a task.

Following recent works [11], [20]–[23], we organized our experiments in an episodic meta-training setting. To form a task  $\mathcal{T}_i$  each time during episodic meta-training, we randomly select  $C$  classes from  $\mathcal{L}_{train}$ , and randomly select  $M$  images within each of these  $C$  classes from  $\mathcal{D}_{train}$ . Within each of these selected classes,  $M$  images are further separated into two sets with  $K$  and  $M-K$  images respectively, namely a support set  $\mathcal{S}_i$  and a query set  $\mathcal{Q}_i$ . Similarly, we can define tasks  $\tilde{\mathcal{T}}_j$  and  $\mathcal{T}_k^*$  on  $\mathcal{D}_{val}$  and  $\mathcal{D}_{test}$  for meta-validation and meta-testing scenarios. We aim to train our neural networks to learn transferable deep bi-similarity knowledge from those meta-learning tasks, tune hyper-parameters through meta-validation, and report the generalization accuracy by taking the average of model’s accuracy in meta-testing.

### B. Bi-Similarity Network

1) *An Overview:* Our *Bi-Similarity Network (BSNet)* has one embedding module  $f_\phi$ , followed by a bi-similarity module consisting of two similarity measurement branches,  $g_\phi$  and  $h_\gamma$ , as shown in Figure 2.

Although we further generalize the proposed *BSNet* by equipping it with other similarity measurements, i.e. [20]–[22], we only demonstrate the structure of the proposed neural network here to avoid redundant contents. The meta-training procedure of the proposed *BSNet* is summarized in Algorithm 1. Note that in Algorithm 1,  $\mathbb{1}(a == b) = 1$  if and only if all elements of  $a$  and  $b$  are identical since we are using one-hot vectors for true/predicted labeling information for every image. A similar procedure, with the exception of updating parameters, is also used for calculating the validation and test accuracy.

For each task  $\mathcal{T}_i$ , images  $\{x_k^{(i)}\}_{k=1}^M$  from support set  $\mathcal{S}_i$  and query set  $\mathcal{Q}_i$  are fed into  $f_\phi$ , generating  $f_\phi(x_k^{(i)})$  for all images in task  $\mathcal{T}_i$ . Then images’ representations are further fed forward to our bi-similarity module. Let  $x_{s,c}^{(i)}$  and  $x_q^{(i)}$  denote

the  $s$ th support image from class  $c$  and the  $q$ th query image in task  $\mathcal{T}_i$ , respectively.

2) *The Training Procedure:* During the meta-training process, we assign the  $q$ th query image in task  $\mathcal{T}_i$  according to each of the two similarity scores,  $S_{(q,c)}^1$  and  $S_{(q,c)}^2$ . Then we obtain two one-hot vectors,  $\hat{y}_q^{(i,1)}$  and  $\hat{y}_q^{(i,2)}$ , as two predictions. Here,  $S_{(q,c)}^1$  and  $S_{(q,c)}^2$  refer to the two similarity scores between  $x_q^{(i)}$  and the  $c$ th class, generated by the proposed *BSNet*. The  $\hat{y}_q^{(i,1)}$  and  $\hat{y}_q^{(i,2)}$  only have element 1 at  $\arg \max_c S_{(q,c)}^d$ ,  $d \in \{1, 2\}$ , for  $c \in \{1, \dots, C\}$ . That is,

$$\begin{aligned} \hat{y}_q^{(i,1)} &= [0, \dots, \underbrace{1}_{\arg \max_c S_{(q,c)}^1, c \in \{1, \dots, C\}}, \dots, 0], \\ \hat{y}_q^{(i,2)} &= [0, \dots, \underbrace{1}_{\arg \max_c S_{(q,c)}^2, c \in \{1, \dots, C\}}, \dots, 0]. \end{aligned} \quad (1)$$

Then we compute two loss values of the query image  $x_q^{(i)}$  by comparing the two assignments  $\hat{y}_q^{(i,1)}$  and  $\hat{y}_q^{(i,2)}$  with the one-hot vector  $y_q^{(i)}$  representing its ground-truth label, and take the average of the two values for back-propagation. The two loss values are given by

$$\begin{aligned} l_q^{(i,1)} &= \sum_j^C (\hat{y}_{q,j}^{(i,1)} - y_{q,j}^{(i)})^2, q = 1, \dots, |\mathcal{Q}_i|, \\ l_q^{(i,2)} &= \sum_j^C (\hat{y}_{q,j}^{(i,2)} - y_{q,j}^{(i)})^2, q = 1, \dots, |\mathcal{Q}_i|, \\ l_q^{(i)} &= (l_q^{(i,1)} + l_q^{(i,2)}), \end{aligned} \quad (2)$$

where  $|\mathcal{Q}_i|$  denotes the total number of query images in  $\mathcal{Q}_i$ .

3) *The Validation/Testing Procedure:* During the validation/test process, we assign the query image to the class

with the maximum average similarity score and generate a corresponding one-hot vector  $\hat{y}_q^{(i)}$ . That is,

$$\hat{y}_q^{(i)} = [0, \dots, \underbrace{1}_{\arg \max_c \frac{1}{2} (S_{(q,c)}^1 + S_{(q,c)}^2)}, \dots, 0]. \quad (3)$$

**Algorithm 1** Meta-Training procedure for BSNet under  $C$ -way  $K$ -shot scenarios

**Require:** Input Model  $\mathcal{M}$ , Optimizer, Number of episodes  $B$ ,  $\mathcal{D}_{train}$ ,  $C$ ,  $K$ .

- 1: **for**  $i$  from 1 to  $B$  **do**
- 2:   Generate a task  $\mathcal{T}_i$  from  $\mathcal{D}_{train}$                     $\triangleright$  Details in Sec. III-A
- 3:   Further split  $\mathcal{T}_i$  into  $\mathcal{S}_i$  and  $\mathcal{Q}_i$     $\triangleright C \times K$  images for  $\mathcal{S}_i$ ,  $C \times (M - K)$  images for  $\mathcal{Q}_i$
- 4:    $L^{(i)} \leftarrow 0$                     $\triangleright$  Initializing loss for the  $i$ th task
- 5:    $A^{(i)} \leftarrow 0$                 $\triangleright$  Initializing accuracy for the  $i$ th task
- 6:   **for**  $x_q^{(i)}$  in  $\mathcal{Q}_i$  **do**  $\triangleright x_q^{(i)}$  is the  $q$ th query sample in  $\mathcal{Q}_i$
- 7:     Compute the  $C \times 2$  similarity scores for  $x_q^{(i)}$ , i.e.  $S_{(q,c)}^d$ , for  $d = 1, 2$  and  $c = 1, \dots, C$ .
- 8:     Compute two class assignments for  $x_q^{(i)}$ :  $\arg \max_c S_{(q,c)}^1$  and  $\arg \max_c S_{(q,c)}^2$
- 9:     Generate two corresponding one-hot vectors  $\hat{y}_q^{(i,1)}$  and  $\hat{y}_q^{(i,2)}$  according to Equation (1).
- 10:    Compute the loss of  $\mathcal{M}$  for  $\mathcal{S}_i$  and  $x_q^{(i)}$  according to Equation (2), denoted by  $l_q^{(i)}$ .
- 11:     $L^{(i)} \leftarrow L^{(i)} + l_q^{(i)}$
- 12:     $A^{(i)} \leftarrow A^{(i)} + \mathbb{1}(\hat{y}_q^{(i)} == y_q^{(i)})$     $\triangleright \hat{y}_q^{(i)}$  is  $\mathcal{M}$ 's prediction for  $x_q^{(i)}$
- 13:     $L^{(i)} \leftarrow \frac{L^{(i)}}{C \times |\mathcal{Q}_i|}$
- 14:     $A^{(i)} \leftarrow \frac{A^{(i)}}{|\mathcal{Q}_i|}$
- 15:    Use  $L^{(i)}$  to update  $\mathcal{M}$ 's parameters according to the optimizer                    $\triangleright$  e.g Adam optimizer
- 16: Stop training  $\mathcal{M}$ . Output mean accuracy  $\frac{1}{B} \sum_{i=1}^B A^{(i)}$ .

### C. The Design of BSNet

In the proposed *BSNet*, the feature embedding module and the bi-similarity module are key parts that affect overall performance. In terms of the feature embedding module  $f_\phi$ , we adopt convolution-based architecture, and will introduce the detailed design in Sec. IV-B.

For the design of bi-similarity module, both of the two chosen similarity measurements should have good fitting ability. We take the *relation module* as an example. The *relation module* was proposed in [11], in which a query image's representation  $f_\phi(x_q^{(i)})$  is concatenated to every support image's representation  $f_\phi(x_{s,c}^{(i)})$ . In terms of  $C$ -way 1-shot scenario ( $|\mathcal{S}_i| = 1$ ), for each query image  $x_q^{(i)}$ ,  $C \times 1$  concatenated features are directly fed forward into two further convolution blocks and two fully connected layers to get  $C$  similarity scores:

$$S_{(q,c)}^1 = g_\varphi \left( [f_\phi(x_{1,c}^{(i)}) || f_\phi(x_q^{(i)})] \right), \quad c = 1, \dots, C, \quad (4)$$

where  $||$  denotes the concatenation operator. While for the  $K$ -shot scenario, we take the mean of support images' feature maps from each class to act as  $C$  prototypes and concatenate them ahead of the query image's feature map, and still obtain  $C$  similarity scores for this query image:

$$S_{(q,c)}^1 = g_\varphi \left( \left[ \frac{1}{K} \sum_{s=1}^K f_\phi(x_{s,c}^{(i)}) || f_\phi(x_q^{(i)}) \right] \right), \quad c = 1, \dots, C. \quad (5)$$

In addition, we also introduce a cosine similarity module designed by our own (details in Sec. IV-B). Actually, many previous studies in few-shot classification have adopted the cosine distance [22], [23]. In this work, we combine the self-designed cosine similarity module with the similarity modules of *Matching Network* [20], *Prototype Network* [21], *DN4* [22] in order to justify effectiveness of the proposed *BSNet*. The details of these similarity modules are present in Sec. IV-B.

Our cosine similarity module contains two convolution blocks ( $h_\gamma^{em}$ ) and a cosine similarity layer ( $h_\gamma^{cos}$ ). The cosine similarity between a query image  $x_q^{(i)}$  and the  $c$ th class, conditioned on the support images from class  $c$ , is given as

$$S_{(q,c)}^2 = h_\gamma^{cos} \left( h_\gamma^{em} \left( \frac{1}{K} \sum_{s=1}^K f_\phi(x_{s,c}^{(i)}) \right), h_\gamma^{em} (f_\phi(x_q^{(i)})) \right). \quad (6)$$

Note that a convolution block in our experiments refers to a convolution layer, a batch norm layer, a *ReLU* layer and with or without a pooling layer. After the model computes two similarity scores between the query image and each class, we have two strategies for training and validation/testing processes, respectively, which have already been demonstrated in Sec. III-B.

### D. The Empirical Rademacher Complexity of BSNet

We start by introducing some notation. We define a single similarity network as a network consisting of a feature embedding module and a similarity module. Let  $I(x; \mathbf{W}_{em1}, \mathbf{W}_{s1}) \in \mathcal{I}$ , and  $J(x; \mathbf{W}_{em2}, \mathbf{W}_{s2}) \in \mathcal{J}$  denote two different similarity networks which contain the feature embedding module with the same structure, where  $\mathbf{W}_{em1}$ ,  $\mathbf{W}_{em2}$  are the parameters of the feature embeddings of  $I$  and  $J$ , respectively, and  $\mathbf{W}_{s1}$  and  $\mathbf{W}_{s2}$  are the parameters of the similarity modules of  $I$  and  $J$ , respectively. We define the *BSNet* as learning a common feature embedding for  $I$  and  $J$  and adopting their similarity modules. Therefore, the Bi-similarity can be denoted as  $Z(x; \mathbf{W}_{s1}, \mathbf{W}_{s2}, \mathbf{W}_{em}) \in \mathcal{Z}$ , where  $\mathbf{W}_{em}$  is the parameter of the common feature embedding. The Rademacher complexity measures the richness of a family of functions which is defined as follows:

**Definition 1.** [39] Let  $\mathcal{I}$  denote a family of real valued function and  $S = \{x_1, x_2, \dots, x_M\}$  be a fixed sample of size  $M$ . Then, the empirical Rademacher complexity of  $\mathcal{I}$  with respect to  $S$  is defined as  $\hat{R}_S(\mathcal{I}) = \mathbf{E}_\sigma [\sup_{I \in \mathcal{I}} \frac{1}{M} \sum_{i=1}^M \sigma_i I(x_i)]$ , where  $\sigma = \{\sigma_1, \sigma_2, \dots, \sigma_M\}$  with  $\sigma_i$  independently distributed according to  $P(\sigma_i = 1) = P(\sigma_i = -1) = \frac{1}{2}$ .

**Theorem 1.** Let  $\hat{R}_S(\mathcal{I})$ ,  $\hat{R}_S(\mathcal{J})$ , and  $\hat{R}_S(\mathcal{Z})$  denote the Rademacher complexities of the single similarity network

family  $\mathcal{I}$ , the single similarity network family  $\mathcal{J}$ , and the Bi-similarity network family  $\mathcal{Z}$ . We have  $\hat{R}_S(\mathcal{Z}) \leq \frac{1}{2}(\hat{R}_S(\mathcal{I}) + \hat{R}_S(\mathcal{J}))$ .

*Proof.* Here, we introduce a network family  $\mathcal{P}$ ,  $P(x) = \frac{1}{2}(I(x) + J(x)) \in \mathcal{P}$ , then  $P(x)$  can be denoted by  $P(x; \mathbf{W}_{s1}, \mathbf{W}_{s2}, \mathbf{W}_{em1}, \mathbf{W}_{em2})$ . Following the notation above, for  $\forall Z \in \mathcal{Z}$ ,  $\exists P \in \mathcal{P}$  subject to  $Z(x) = P(x)$ , so we have  $\mathcal{Z} \subset \mathcal{P}$ . Thus, we have

$$\begin{aligned} \hat{R}_S(\mathcal{Z}) &= \mathbf{E}_\sigma \left[ \sup_{Z \in \mathcal{Z}} \frac{1}{M} \sum_{i=1}^M \sigma_i Z(x_i) \right] \\ &\leq \mathbf{E}_\sigma \left[ \sup_{P \in \mathcal{P}} \frac{1}{M} \sum_{i=1}^M \sigma_i P(x_i) \right] \\ &= \mathbf{E}_\sigma \left[ \sup_{I \in \mathcal{I}, J \in \mathcal{J}} \frac{1}{M} \sum_{i=1}^M \sigma_i \left( \frac{1}{2}I(x_i) + \frac{1}{2}J(x_i) \right) \right] \\ &= \frac{1}{2} \mathbf{E}_\sigma \left[ \sup_{I \in \mathcal{I}} \frac{1}{M} \sum_{i=1}^M \sigma_i I(x_i) \right] + \frac{1}{2} \mathbf{E}_\sigma \left[ \sup_{J \in \mathcal{J}} \frac{1}{M} \sum_{i=1}^M \sigma_i J(x_i) \right] \\ &= \frac{1}{2} (\hat{R}_S(\mathcal{I}) + \hat{R}_S(\mathcal{J})). \end{aligned} \tag{7}$$

□

This theorem states that even though *BSNet* has more model parameters, the Rademacher complexity of the proposed *BSNet* is no more than the average of the Rademacher complexities of two individual networks.

#### IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

We evaluate the proposed approach on few-shot classification tasks. This evaluation serves four purposes:

- To compare the proposed *BSNet* with the state-of-the-art methods on few-shot fine-grained classification (Sec. IV-C);
- To investigate the generalization ability of *BSNet* by changing the backbone network (Sec. IV-D);
- To study the effectiveness of each similarity module in the proposed *BSNet* (Sec. IV-E);
- To demonstrate that the proposed *BSNet* can learn a reduced number of class-discriminative features (Sec. IV-F);
- To investigate the effect of changing similarity modules on *BSNet* (Sec. IV-G).

##### A. Datasets

We conducted all the experiments on four benchmark fine-grained datasets, *FGVC-Aircraft*, *Stanford-Cars*, *Stanford-Dogs* and *CUB-200-2011*. For each dataset, we divided them into meta-training set  $\mathcal{D}_{train}$ , meta-validation set  $\mathcal{D}_{val}$  and meta-test set  $\mathcal{D}_{test}$  in a ratio of 2 : 1 : 1. All images in the four datasets are resized to  $84 \times 84$ . We did not use boundary boxes for all the images.

*FGVC-Aircraft* [40] is a classic dataset in fine-grained image classification and recognition research. It contains four-level hierarchical notations: *model*, *variant*, *family* and *manufacturer*. According to the *variant* level, it can be divided

into 100 categories (classification annotation commonly used in fine-grained image classification), which are the annotation level we used. *Stanford-Cars* [41] is also a benchmark dataset for fine-grained classification, which contains 16,185 images of 196 classes of cars, and categories of this dataset are mainly divided based on the brand, model, and year of the car. *Stanford-Dogs* [42] is a challenging and large-scale dataset that aims at fine-grained image categorization, including 20,580 annotated images of 120 breeds of dogs from around the world. *CUB-200-2011* [43] contains 11,788 images from 200 bird species. This dataset is also a classic benchmark for fine-grained image classification.

##### B. Implementation Details

In order to prove the effectiveness of the proposed method, we choose several metric/similarity based few-shot classification networks, *Matching Network* [20], *Prototype Network* [21], *Relation Network* [11], *Deep Nearest Neighbor Neural Network* (or *DN4*, for short) [22], to be the baseline of the proposed method. At the same time, we combine these diverse similarity measurement modules with a self-designed cosine module to construct the proposed *BSNet*. We present details of our experiments in this section, which include information about experiment parameters, the matching module, the prototype module, the relation module, the image-to-class module and the bi-similarity module. Together with Table I, it provides a detailed summary for all implementations in our experiments.

All methods are trained from scratch. Some important parameters during training are shown in Table I. In particular, for the optimizer of *DN4* and bi-similarity experiments involved *DN4*, the learning rate is reduced by half every 100,000 episodes, while the other methods do not adopt this strategy. In all experiments, we use the meta-validation set to select the optimal setting. At the meta-testing stage, we report the mean accuracy of 600 randomly generated testing episodes as well as 95% confidence intervals. i.e.,  $\text{Mean} \pm 1.96 \times \frac{\text{Std}}{\sqrt{600}}$ . Particularly, in terms of *DN4*, the above test procedure is repeated five times, and the mean accuracy with 95% confidence intervals are reported. We will introduce the embedding module and several similarity measurement modules used in our experiments in detail below, as well as the self-defined Cosine module and Bi-similarity module proposed by us.

**Embedding module:** We construct four convolution blocks to be our embedding module  $f_\phi$  to learn a shared representation in the proposed *BSNet*. Following [23], we used the embedding module *Conv4* for *Matching Network*, *Prototype Network*, *Relation Network* and *Cosine Network*. Specifically, each convolution block of *Conv4* has  $3 \times 3$  convolution of 64 filters, followed by batch normalization and a *ReLU* activation function. Following [22], we used the embedding module *Conv64F* for *DN4*. In *Conv64F*, each convolution block employs 64  $3 \times 3$  convolution filters without padding, followed by batch normalization and a *Leaky ReLU* activation function. The  $2 \times 2$  max-pooling layer is used in the first two blocks. After passing through the *Conv64F* module, a  $3 \times 84 \times 84$  input image is now represented by a  $64 \times 21 \times 21$  feature map.

Table I  
EXPERIMENT CONFIGURATION (SHOWN IN COLUMNS).

Experiment Configuration					
	Matching Network [20]	Prototype Network [21]	Relation Network [11]	Cosine Network	DN4 [22]
Input size	3 × 84 × 84				
Embedding module $f_\phi$	Conv4				Conv64F
Feature map size	64 × 19 × 19				64 × 21 × 21
Similarity module	Matching module	Prototype module	Relation module	Cosine module	Image-to-Class module
Optimizer	Adam (initial learning rate = $10^{-3}$ , weight decay = 0)				
Loss function	NLL Loss	Cross Entropy Loss	MSE Loss		Cross Entropy Loss
Data augment	Random Sized Crop, Image Jitter, Random Horizontal Flip				
Training episodes	1-shot: 60, 000, 5-shot: 40, 000				300, 000
Training query sample size	16				1-shot: 15, 5-shot: 10

Table II  
FIVE-WAY FEW-SHOT CLASSIFICATION PERFORMANCE ON THE *Stanford-Cars* (CARS), *Stanford-Dogs* (DOGS), *FGVC-Aircraft* (AIRCRAFT) AND *CUB-200-2011* (CUB) DATASETS. THE METHODS INCLUDE: *Matching Network* (MATCHING), *Prototype Network* (PROTOTYPE), *Relation Network* (RELATION) AND *Deep Nearest Neighbor Neural Network* (DN4) AND THE PROPOSED *BSNet*. WE REPORT THE MEAN ACCURACY FOR EACH METHOD, ALONG WITH ITS 95% CONFIDENCE INTERVAL.

Model	5-Way 5-shot Accuracy (%)				5-Way 1-shot Accuracy (%)			
	Cars	Dogs	Aircraft	CUB	Cars	Dogs	Aircraft	CUB
Matching [20]	<b>64.74 ± 0.72</b>	59.79 ± 0.72	73.75 ± 0.69	74.57 ± 0.73	44.73 ± 0.77	<b>46.10 ± 0.86</b>	<b>56.74 ± 0.87</b>	60.06 ± 0.88
BSNet (M&C)	63.58 ± 0.75	<b>61.61 ± 0.69</b>	<b>75.92 ± 0.76</b>	<b>74.68 ± 0.71</b>	<b>44.93 ± 0.80</b>	45.91 ± 0.81	56.53 ± 0.81	<b>60.73 ± 0.94</b>
Prototype [21]	62.14 ± 0.76	61.58 ± 0.71	71.27 ± 0.67	75.06 ± 0.67	36.54 ± 0.74	40.81 ± 0.83	46.68 ± 0.81	50.67 ± 0.88
BSNet (P&C)	<b>63.72 ± 0.78</b>	<b>62.61 ± 0.73</b>	<b>77.35 ± 0.68</b>	<b>76.34 ± 0.65</b>	<b>44.56 ± 0.83</b>	<b>43.13 ± 0.85</b>	<b>52.48 ± 0.88</b>	<b>55.81 ± 0.97</b>
Relation [11]	68.52 ± 0.78	66.20 ± 0.74	75.18 ± 0.74	77.87 ± 0.64	46.04 ± 0.91	47.35 ± 0.88	62.04 ± 0.92	63.94 ± 0.92
BSNet (R&C)	<b>73.47 ± 0.75</b>	<b>68.60 ± 0.73</b>	<b>80.25 ± 0.67</b>	<b>80.99 ± 0.63</b>	<b>54.12 ± 0.96</b>	<b>51.06 ± 0.94</b>	<b>64.83 ± 1.00</b>	<b>65.89 ± 1.00</b>
DN4 [22]	<b>87.47 ± 0.47</b>	69.81 ± 0.69	<b>84.07 ± 0.65</b>	84.41 ± 0.58	34.12 ± 0.68	39.08 ± 0.76	60.65 ± 0.95	57.45 ± 0.89
BSNet (D&C)	86.88 ± 0.50	<b>71.90 ± 0.68</b>	83.12 ± 0.68	<b>85.39 ± 0.56</b>	<b>40.89 ± 0.77</b>	<b>43.42 ± 0.86</b>	<b>62.86 ± 0.96</b>	<b>62.84 ± 0.95</b>

**Matching module:** The matching module is composed of three parts: a memory network, a metric module and an attention module. The memory network uses a simple bidirectional *LSTM*, which further processes the support features from the embedding module  $f_\phi$ , so that these features can contain the context information. The metric module uses a cosine distance measure, and a softmax layer is used in the attention module. In the process of model training, a single support set and a query sample will be used as input such that feature extraction will be carried out at the same time. After that, these features will be further processed through the memory network, and then the final prediction value of a query image depends on distance measurements and the attention module.

**Prototype module:** The prototype module is a module which includes an Euclidean distance algorithm to calculate the distance of query features and support features from the embedding module  $f_\phi$ . The specific calculation formula is  $-\|f_\phi(x_q) - f_\phi(x_s)\|^2$ , where  $f_\phi(x_q)$  represents the flattened vector of a query feature and  $f_\phi(x_s)$  represents the flattened vector of a support prototype feature. For the 1-shot classification, the feature of a support sample within a class is the prototype of the class. For 5-shot, the mean of the features of all 5 support samples in the same class is the corresponding class prototype. The Euclidean distance between a query feature and a class prototype represents the dissimilarity between the query sample and the class.

**Relation module:** The relation module consists of two

convolution blocks and two full connection layers (or *FC*, for short). Each convolution block is composed of  $3 \times 3$  convolution of 64 filters, followed by batch normalization, a *ReLU* non-linearity function and a  $2 \times 2$  max-pooling. The padding parameter of both blocks is set to be 0. The first *FC* layer is followed by a *ReLU* function, and the second *FC* layer is followed by a sigmoid function. The input sizes of the first and the second *FC* layers are 576 and 8, respectively, and the final output size is 1. During the training process, we concatenated the feature of a query feature to every class prototype (i.e. mean of support image’s features within each class), resulting in 128-dimensional relation pairs. By passing these relation pairs into the relation module, it computes similarity scores between the query sample and each class.

**Image-to-Class module:** Different from other methods, we use *Conv64F* to be the embedding module of image-to-class module, of which the output size is  $64 \times 21 \times 21$ . We divide the features into 441 ( $21 \times 21$ ) 64-dimensional local features. The image-to-class module consists of a cosine measurement module and a *K-Nearest-Neighbours* classifier (*KNN* for short). The cosine measurement module calculates the cosine distance between each local features of the query sample and each local features of the support samples. In terms of 5-way 1-shot classification problems, we calculate the cosine distance of the local features between the query sample and each of the 5 support samples, get the cosine distance of size  $5 \times 441 \times 441$ . Then the *KNN* classifier takes

Table III  
FIVE-WAY FEW-SHOT CLASSIFICATION PERFORMANCE OF THE PROPOSED *BSNet* BY ADJUSTING THE WEIGHTS IN THE LOSS FUNCTION,  $l_q$ , ON THE CUB-200-2011 DATASET. HERE  $l_q = \lambda \times l_q^{(1)} + \beta \times l_q^{(2)}$ . WE REPORT THE MEAN ACCURACY OF 600 RANDOMLY GENERATED TESTING EPISODES FOR *BSNet* (R&C), WITH ITS 95% CONFIDENCE INTERVAL.

$\lambda$	$\beta$	5-Way Accuracy (%)	
		CUB-200-2011	
		1-shot	5-shot
1	0.1	65.65 $\pm$ 0.97	78.80 $\pm$ 0.70
1	0.3	63.86 $\pm$ 0.91	78.83 $\pm$ 0.62
1	0.5	64.75 $\pm$ 0.97	80.01 $\pm$ 0.65
1	0.7	63.55 $\pm$ 0.97	79.08 $\pm$ 0.64
1	0.9	64.48 $\pm$ 1.00	79.08 $\pm$ 0.63
0.1	1	64.36 $\pm$ 0.94	79.22 $\pm$ 0.67
0.3	1	64.38 $\pm$ 1.00	79.55 $\pm$ 0.68
0.5	1	64.96 $\pm$ 0.98	79.41 $\pm$ 0.68
0.7	1	63.97 $\pm$ 1.02	78.59 $\pm$ 0.67
0.9	1	64.92 $\pm$ 0.96	78.53 $\pm$ 0.69
<b>1</b>	<b>1</b>	<b>65.89 <math>\pm</math> 1.00</b>	<b>80.99 <math>\pm</math> 0.63</b>

the category corresponding to the maximum of sum of top-3 cosine distances in each class as the prediction category of the query sample (i.e. using  $K = 3$ ). For 5-shot experiments, we provide 5 support features of each class with 2205 ( $21 \times 21 \times 5$ ) 64-dimensional local features.

**Cosine module:** The cosine module (i.e. the cosine similarity branch in Section III-C) consists of two convolution blocks and a cosine similarity measurement layer. Each convolution block is composed of  $3 \times 3$  convolution of 64 filters, followed by batch normalization, a *ReLU* non-linearity activation function and a pooling layer. The padding of each block is 1. The first convolution block is followed by a  $2 \times 2$  max-pooling, and the second convolution block is followed by a  $2 \times 2$  avg-pool. Our cosine module does not require to concatenate representations of a query image and class prototypes, and feature maps of the query image and class prototypes are fed independently to the convolution blocks. Then these new feature maps are flattened and pass through a cosine similarity layer to compute the cosine distance between the query image and each class. Cosine similarity scores represent the similarity between query samples and each class’s prototype given support samples. For 5-shot learning, a class prototype is the mean of the feature maps of 5 support images from an identical class, while it is directly the feature map of a support image in 1-shot scenarios.

**Bi-similarity module:** We combine the self-proposed cosine module with other modules to build our *bi-similarity module*. That is,  $h_\gamma$  is our cosine module, and  $g_\varphi$  can be any other modules mentioned above. Note that the parameter setting of *BSNet*’s experiments is the same as that of  $g_\varphi$  similarity experiments. In this way, features from the embedded module  $f_\phi$  enter into  $h_\gamma$  and  $g_\varphi$  respectively, producing bi-similarity prediction values. At the meta-training stage, we update our network with the average of the prediction losses of the two similarity branches (Equation (2)). At the meta-testing stage, we use the average of two similarity scores generated by the two branches to produce the final prediction (Equation (3)).

### C. Comparison with State-of-the-art Methods on Few Shot Classification

To evaluate the performance of the proposed *BSNet*, we compare it with several metric/similarity based few-shot classification networks, i.e., *Matching Network* [20], *Prototype Network* [21], *Relation Network* [11], *DN4* [22], on the four images datasets mentioned in Sec. IV-A. Few shot classification results are shown in Table II. *BSNet* is implemented by keeping the  $h_\gamma$  branch of Bi-similarity module to be our cosine module, replacing the  $g_\varphi$  branch by an arbitrary module among Matching module, Prototype module, Relation module, and Image-to-Class module in Sec. IV-B, resulting in *BSNet* (M&C), *BSNet* (P&C), *BSNet* (R&C), and *BSNet* (D&C) in Table II, respectively.

Our *BSNet* outperforms *Matching Network* on *Stanford-Dogs*, *FGVC-Aircraft* and *CUB-200-2011*, and achieves comparable generalization performance on *Stanford-Cars* (63.58% versus 64.74%). To compare with *DN4*, we combine its similarity measurement (i.e. the Image-to-Class module in Sec. IV-B) with our cosine similarity measurement (i.e. the Cosine module in Sec. IV-B), which achieves notable 85.39% on the *CUB-200-2011* dataset. It also demonstrates that the proposed *BSNet* consistently exceeds the *Prototype Network* and *Relation Network* on all of the four fine-grained datasets.

A tougher problem is fine-grained 1-shot classification. We also conducted 5-way 1-shot experiments on the four fine-grained datasets. Table II demonstrates that the proposed *BSNet* achieves the state-of-the-art classification performance on fine-grained datasets in 1-shot scenarios. That is, the proposed *BSNet* achieves the best classification accuracy on each dataset, 54.12%, 51.06%, 64.83% and 65.89%, respectively.

It is found that the proposed *BSNet* has great improvements by combining the deep relation measurement (i.e. the Relation module in Sec. IV-B) and our Cosine module, which consistently obtains state-of-the-art performance on all of the four fine-grained datasets in 1-shot scenarios. It is also found that the combination of our Cosine module with other few-shot networks boosts the performance (around 2% to 8% in mean accuracy) with the exception of *Matching Network*.

For the two addends in the loss function of *BSNet* (Equation 2), we also tried to tune their weights according to  $l_q = \lambda \times l_q^{(1)} + \beta \times l_q^{(2)}$ . Table III lists the classification performance of *BSNet* (R&C) on the *CUB-200-2011* dataset when  $\lambda$  and  $\beta$  are set to different values. Firstly, the highest mean value of the classification accuracies is achieved when the weights of the two addends both equal 1. Meanwhile, we can only observe tiny difference of the standard deviation between different settings of the weights. Thus, we simply select equal weights for the two addends in the loss function.

### D. Few-shot Classification with Different Backbone Networks

In the previous experiments, we used feature embedding modules *Conv4* for the *Matching Network*, *Prototype Network* and *Relation Network*, and *Conv64F* for *DN4*. To further investigate the effectiveness of the proposed *BSNet*, We changed the feature embedding module to *Conv4*, *Conv6*, and *Conv8*.

Table IV

FEW-SHOT CLASSIFICATION OF CHANGING THE BACKBONE NETWORK ON THE STANFORD-CARS AND CUB-200-2011 DATASETS. WE REPORT THE MEAN ACCURACY FOR EACH METHOD, ALONG WITH ITS 95% CONFIDENCE INTERVAL.

Backbone	Model	5-Way Accuracy (%)			
		5-shot		1-shot	
		<i>Stanford-Cars</i>	<i>CUB-200-2011</i>	<i>Stanford-Cars</i>	<i>CUB-200-2011</i>
Conv4	Matching Network [20]	<b>64.74 ± 0.72</b>	74.57 ± 0.73	44.73 ± 0.77	60.06 ± 0.88
	BSNet (M&C)	63.58 ± 0.75	<b>74.68 ± 0.71</b>	<b>44.93 ± 0.80</b>	<b>60.73 ± 0.94</b>
	Prototype Network [21]	62.14 ± 0.76	75.06 ± 0.67	36.54 ± 0.74	50.67 ± 0.88
	BSNet (P&C)	<b>63.72 ± 0.78</b>	<b>76.34 ± 0.65</b>	<b>44.56 ± 0.83</b>	<b>55.81 ± 0.97</b>
	Relation Network [11]	68.52 ± 0.78	77.87 ± 0.64	46.04 ± 0.91	63.94 ± 0.92
	BSNet (R&C)	<b>73.47 ± 0.75</b>	<b>80.99 ± 0.63</b>	<b>54.12 ± 0.96</b>	<b>65.89 ± 1.00</b>
	DN4 [22]	86.59 ± 0.54	82.97 ± 0.66	59.57 ± 0.87	64.02 ± 0.92
Conv6	BSNet (D&C)	<b>86.68 ± 0.54</b>	<b>84.18 ± 0.64</b>	<b>61.41 ± 0.92</b>	<b>65.20 ± 0.92</b>
	Matching Network [20]	71.65 ± 0.72	76.36 ± 0.60	57.00 ± 0.94	61.05 ± 0.93
	BSNet (M&C)	<b>74.50 ± 0.75</b>	<b>78.99 ± 0.65</b>	<b>58.32 ± 0.95</b>	<b>67.60 ± 0.91</b>
	Prototype Network [21]	74.55 ± 0.71	<b>80.59 ± 0.61</b>	53.92 ± 0.93	63.63 ± 0.93
	BSNet (P&C)	<b>74.61 ± 0.71</b>	80.51 ± 0.62	<b>54.25 ± 0.96</b>	<b>64.93 ± 0.99</b>
	Relation Network [11]	72.86 ± 0.73	79.16 ± 0.68	47.36 ± 0.93	62.93 ± 0.97
	BSNet (R&C)	<b>76.90 ± 0.70</b>	<b>80.32 ± 0.62</b>	<b>55.27 ± 1.00</b>	<b>66.19 ± 0.98</b>
Conv8	DN4 [22]	<b>87.04 ± 0.54</b>	<b>83.81 ± 0.65</b>	42.77 ± 0.81	66.13 ± 0.94
	BSNet (D&C)	86.17 ± 0.57	83.43 ± 0.64	<b>57.16 ± 0.97</b>	<b>67.58 ± 0.95</b>
	Matching Network [20]	71.84 ± 0.73	74.84 ± 0.62	57.41 ± 0.96	63.44 ± 0.95
	BSNet (M&C)	<b>72.26 ± 0.72</b>	<b>76.96 ± 0.63</b>	<b>57.86 ± 0.91</b>	<b>66.06 ± 0.98</b>
	Prototype Network [21]	<b>76.69 ± 0.68</b>	81.44 ± 0.61	<b>57.66 ± 0.99</b>	<b>65.54 ± 0.97</b>
	BSNet (P&C)	76.30 ± 0.70	<b>81.77 ± 0.61</b>	53.19 ± 0.96	65.50 ± 0.99
	Relation Network [11]	73.09 ± 0.75	78.63 ± 0.66	47.45 ± 0.94	62.47 ± 0.99
ResNet-10	BSNet (R&C)	<b>75.29 ± 0.72</b>	<b>80.60 ± 0.63</b>	<b>53.77 ± 0.94</b>	<b>63.23 ± 1.01</b>
	DN4 [22]	<b>88.14 ± 0.50</b>	<b>84.41 ± 0.63</b>	<b>61.73 ± 0.97</b>	64.26 ± 1.01
	BSNet (D&C)	84.66 ± 0.59	81.06 ± 0.72	61.20 ± 0.97	<b>64.28 ± 1.01</b>
	Relation Network [11]	83.85 ± 0.64	81.67 ± 0.58	63.55 ± 1.04	<b>69.95 ± 0.97</b>
	BSNet (R&C)	<b>84.09 ± 0.66</b>	<b>82.85 ± 0.61</b>	<b>66.97 ± 0.99</b>	69.73 ± 0.97
ResNet-18	Relation Network [11]	83.61 ± 0.68	83.04 ± 0.60	<b>61.02 ± 1.04</b>	69.58 ± 0.97
	BSNet (R&C)	<b>85.28 ± 0.64</b>	<b>83.24 ± 0.60</b>	60.36 ± 0.98	<b>69.61 ± 0.92</b>
ResNet-34	Relation Network [11]	<b>89.07 ± 0.56</b>	<b>84.21 ± 0.55</b>	<b>76.35 ± 1.00</b>	<b>71.89 ± 1.04</b>
	BSNet (R&C)	85.30 ± 0.68	82.12 ± 0.59	72.72 ± 1.02	68.98 ± 1.04

We run all the compared methods and *BSNet* on the *Stanford-Cars* and *CUB-200-2011* datasets, respectively. Regarding the structure of *Conv4* (introduced in Section IV-B), *Conv6*, and *Conv8*, we used the same setting as [23]. The classification results are listed in Table IV, from which we can make the following observations:

Firstly, when all the methods use *Conv4* as the feature embedding module, for the 5-way 5-shot classification on the *Stanford-Cars* dataset, the proposed *BSNet* is inferior to the *Matching Network*. But in other cases, the proposed method performs better than the *Matching Network*, *Prototype Network*, *Relation Network*, and *DN4* on the *Stanford-Cars* and *CUB-200-2011* datasets in both the 5-way 5-shot classification and the 5-way 1-shot classification scenarios. When all the methods use *Conv6* as the feature embedding module, the proposed *BSNet* performs slightly worse than the *Prototype network* and *DN4* on *CUB-200-2011* dataset, and performs slightly worse than *DN4* on *Stanford-Cars* in the 5-way 5-

shot classification. In other cases, *BSNet* performs better than all the compared methods in the 5-way 5-shot classification. Moreover, *BSNet* performs the best in the 5-way 1-shot tasks. Secondly, when all the methods use *Conv8* as the feature embedding module, the proposed *BSNet* performs worse than *Prototype Network* and *DN4* in 5-way 1-shot and 5-way 5-shot tasks, respectively. However, it performs better than *Matching Network* and *Relation Network* on *Stanford-Cars* and *CUB-200-2011* for both the 5-way 5-shot and 5-way 1-shot tasks.

In addition to traditional convolutional backbone networks, we test feature embedding by *ResNet-10*, *ResNet-18* or *ResNet-34*, for *Relation Network* and *BSNet* on the *Stanford-Cars* and *CUB-200-2011* datasets (see Table IV). We used the same setting as [23], except that the epoch number of *ResNet-34* is set to 1,800 since this feature embedding network is difficult to converge on the training data, especially for *BSNet*. From Table IV, we find that, when *Relation Network* and *BSNet* use *ResNet-10* or *ResNet-18* as the feature embedding module,



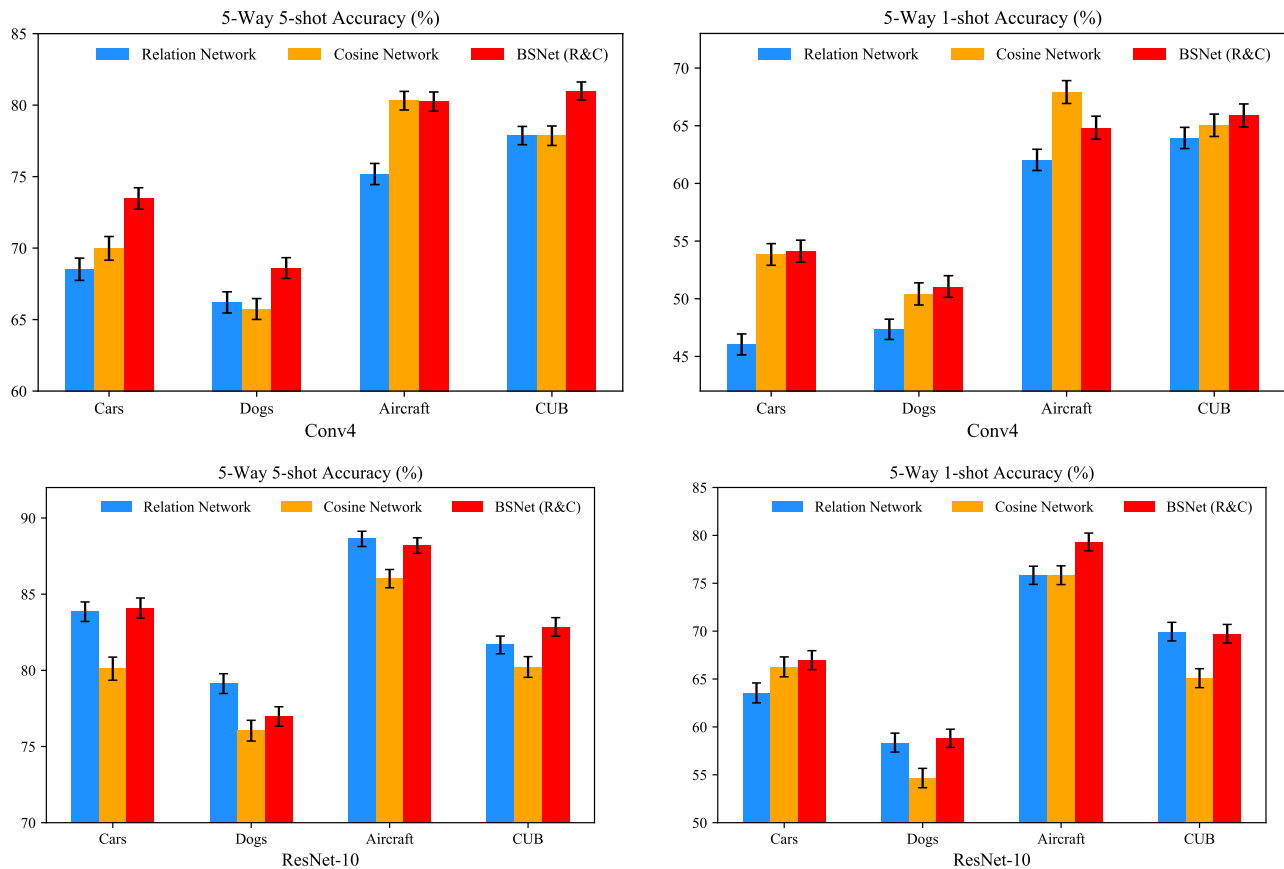


Figure 3. Ablation study: effectiveness of the two-branch similarity. Five-way few-shot classification performance of *Relation Network*, *Cosine Network* and the proposed *BSNet (R&C)*, on the *Stanford-Cars* (Cars), *Stanford-Dogs* (Dogs), *FGVC-Aircraft* (Aircraft) and *CUB-200-2011* (CUB) datasets. The mean accuracy of 600 randomly generated testing episodes for each method is reported.

*BSNet* generally outperforms *Relation Network*. When *ResNet-34* is chosen, however, *Relation Network* is better.

In brief, we can conclude that, firstly, when we change the feature embedding to *Conv4*, *Conv6*, *Conv8*, *ResNet-10*, and *ResNet-18*, *BSNet* outperforms the compared methods in most of the cases. Secondly, the performance of all compared methods will increase when the embedding layer becomes deeper, which is consistent with the findings in [23].

#### E. Ablation Study on Effectiveness of Bi-Similarity Module

To further explore the effect of Bi-similarity module, in this section, we prune either of two similarity branches in Bi-similarity module. If only keeping Relation similarity branch and pruning the Cosine similarity branch, we recover the *Relation Network* [11]. Similarly, if only keeping our cosine branch, we obtain a single Cosine module based network, which is denoted by *Cosine Network* in this work.

We compare the performance of *Relation Network*, *Cosine Network* and the proposed *BSNet* with different embedding modules here, i.e., *Conv4* and *Resnet-10*, on the four fine-grained datasets. Experimental results of 5-way 5-shot and 5-way 1-shot tasks are presented in Figure 3. From Figure 3, it can be found that firstly, in some cases *Relation Network* performs better than *Cosine Network*, and in other cases *Cosine Network* performs better than *Relation Network*. Secondly,

Table V  
FIVE-WAY FEW-SHOT CLASSIFICATION PERFORMANCE ON THE *Stanford-Cars* (CARS) AND *CUB-200-2011* (CUB) DATASETS. THE METHODS INCLUDE: *Matching Network*, *Prototype Network*, *Relation Network*, AND THE PROPOSED *BSNet* BUT WITH TWO OTHER SIMILARITY METRICS. WE REPORT THE MEAN ACCURACY FOR EACH METHOD, ALONG WITH ITS 95% CONFIDENCE INTERVAL.

Shot	Model	5-Way Accuracy (%)	
		Cars	CUB
5-shot	Matching [20]	64.74 ± 0.72	74.57 ± 0.73
	Prototype [21]	62.14 ± 0.76	75.06 ± 0.67
	Relation [11]	68.52 ± 0.78	77.87 ± 0.64
	<b>BSNet (M&amp;P)</b>	<b>68.69 ± 0.70</b>	<b>78.20 ± 0.64</b>
	BSNet (M&R)	67.32 ± 0.71	77.95 ± 0.69
	BSNet (P&R)	63.73 ± 0.74	76.83 ± 0.65
1-shot	Matching [20]	44.73 ± 0.77	60.06 ± 0.88
	Prototype [21]	36.54 ± 0.74	50.67 ± 0.88
	<b>Relation [11]</b>	<b>46.04 ± 0.91</b>	<b>63.94 ± 0.92</b>
	BSNet (M&P)	38.44 ± 0.75	52.10 ± 0.90
	BSNet (M&R)	45.92 ± 0.79	62.70 ± 0.92
	BSNet (P&R)	41.20 ± 0.83	52.95 ± 0.98

in most cases, the proposed *BSNet* outperforms both of the

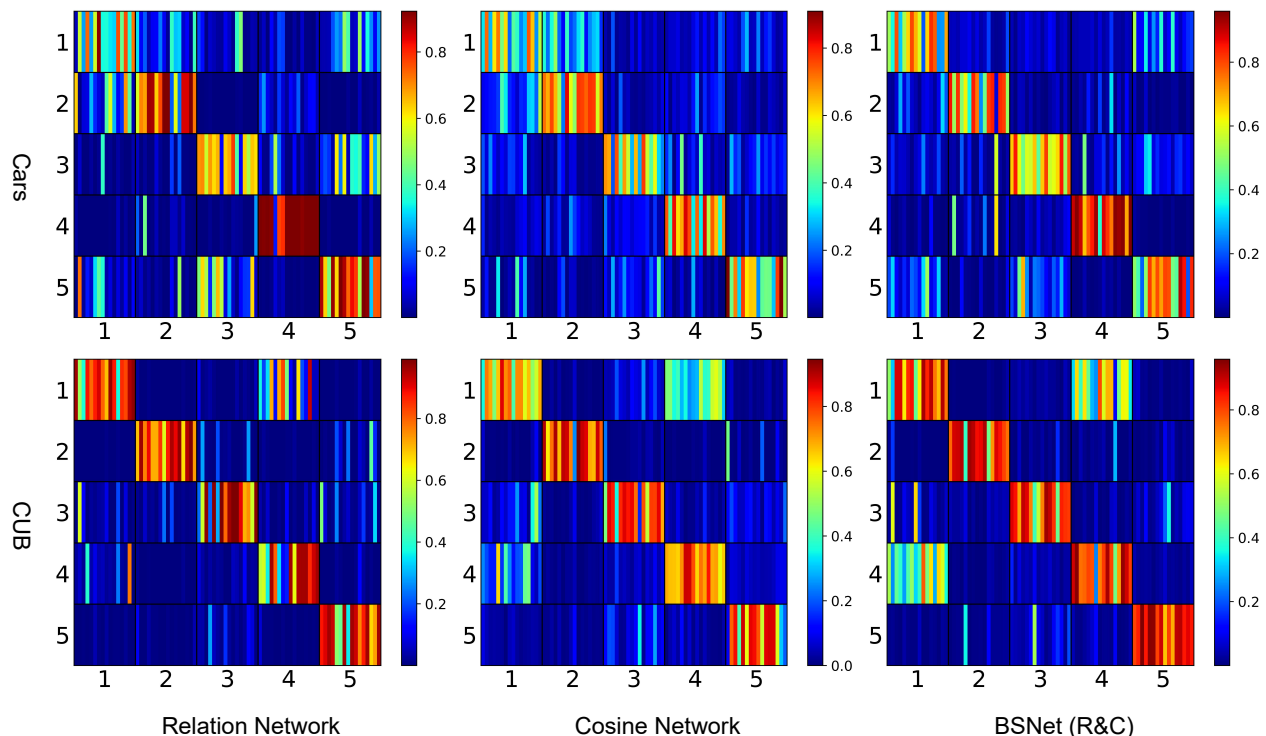


Figure 4. Visualization of similarity scores predicted by *Relation Network* [11], *Cosine Network* and the proposed *BSNet* on the *Stanford-Cars* (Cars) and *CUB-200-2011* (CUB) datasets. In each confusion matrix, the vertical axis shows 5 classes in a task, the horizontal axis shows query samples in the 5 classes, each class contains 16 query samples. Warmer color means larger similarity score.

Table VI

FIVE-WAY FEW-SHOT CLASSIFICATION PERFORMANCE ON THE *Stanford-Cars* AND *CUB-200-2011* DATASETS. THE METHODS INCLUDE: *Matching Network*, *Prototype Network*, *Relation Network*, *Cosine Network*, AND THE PROPOSED *BSNet* WITH THREE OR FOUR SIMILARITY METRICS. WE REPORT THE MEAN ACCURACY FOR EACH METHOD, ALONG WITH ITS 95% CONFIDENCE INTERVAL.

Model	5-Way 5-shot Accuracy (%)		5-Way 1-shot Accuracy (%)	
	<i>Stanford-Cars</i>	<i>CUB-200-2011</i>	<i>Stanford-Cars</i>	<i>CUB-200-2011</i>
Matching Network [20]	64.74 ± 0.72	74.57 ± 0.73	44.73 ± 0.77	60.06 ± 0.88
Prototype Network [21]	62.14 ± 0.76	75.06 ± 0.67	36.54 ± 0.74	50.67 ± 0.88
Relation Network [11]	68.52 ± 0.78	77.87 ± 0.64	46.04 ± 0.91	63.94 ± 0.92
Cosine Network	69.98 ± 0.83	77.86 ± 0.68	53.84 ± 0.94	65.04 ± 0.97
BSNet (M&P&R)	71.53 ± 0.73	79.83 ± 0.63	45.56 ± 0.83	60.28 ± 0.94
BSNet (M&P&C)	71.50 ± 0.75	79.30 ± 0.61	44.33 ± 0.83	59.18 ± 0.93
BSNet (M&R&C)	72.70 ± 0.71	80.84 ± 0.67	<b>54.82 ± 0.89</b>	<b>66.13 ± 0.90</b>
BSNet (P&R&C)	69.20 ± 0.76	79.26 ± 0.62	47.09 ± 0.85	58.98 ± 0.96
BSNet (M&P&R&C)	<b>72.78 ± 0.73</b>	<b>80.94 ± 0.63</b>	48.00 ± 0.87	59.50 ± 0.96

*Relation Network* and the *Cosine Network*.

We visualize similarity scores obtained by *Relation Network*, *Cosine Network*, and the proposed *BSNet* in the 5-way 5-shot experiments. In particular, we fixed the testing tasks from the meta-testing set of the *Stanford-Cars* and *CUB-200-2011* datasets, respectively, so that the sequences of testing tasks for different methods on each dataset are the same. On the *Stanford-Cars* dataset, the prediction accuracy of *Relation Network*, *Cosine Network* and the proposed *BSNet* are  $68.52 \pm 0.78\%$ ,  $69.98 \pm 0.83\%$  and  $73.47 \pm 0.75\%$ , respectively. On the *CUB-200-2011* dataset, the prediction accuracy of *Relation Network*, *Cosine Network* and the proposed *BSNet* are

$77.87 \pm 0.64\%$ ,  $77.86 \pm 0.68\%$  and  $80.99 \pm 0.63\%$ , respectively. For each dataset, we randomly select a testing task and show the similarity scores of its query images via confusion matrices. Please refer to Figure 4 for details.

From Figure 4, it can be observed that on the *Stanford-Cars* dataset, for the query images from the 4th class, the predicting similarity scores of *Relation Network* are higher than those of *Cosine Network*; on the *CUB-200-2011* dataset, for the query images from the 3rd class, the predicting similarity scores of *Cosine Network* are higher than those of *Relation Network*. However, in both cases, *BSNet* can predict query images correctly by synthesizing the Relation module and the

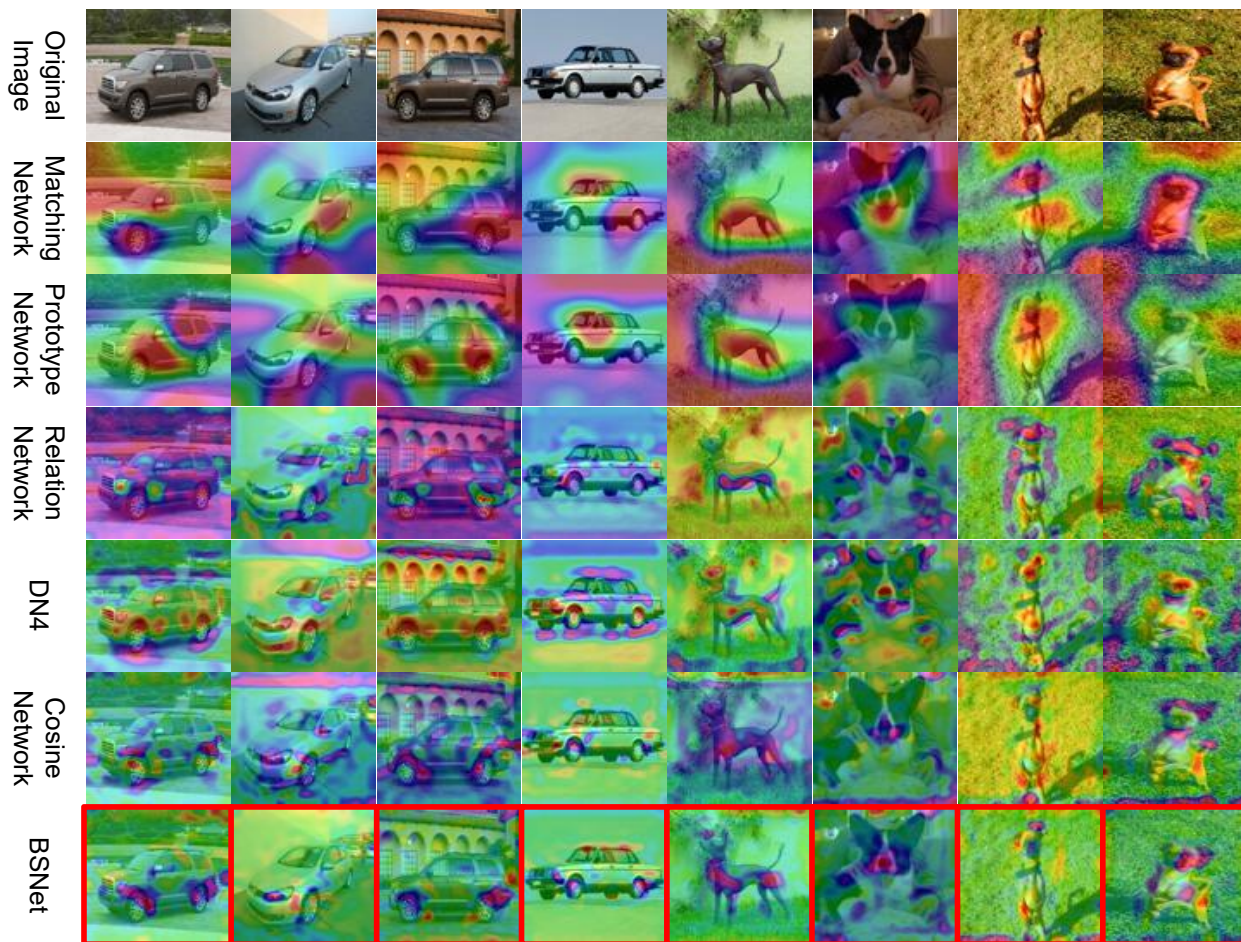


Figure 5. Feature visualization under *Matching Network* [20], *Prototype Network* [21], *Relation Network* [11], *DN4* [22], *Cosine Network* and the proposed *BSNet* on the *Stanford-Cars* and *Stanford-Dogs* datasets. The redder the region, the more class-discriminative it is.

Cosine module. Similar patterns can also be found in other tasks.

These results further show that our bi-similarity idea is reasonable and the proposed *BSNet* is less biased on similarity.

#### F. Feature Visualization

To further demonstrate that the features learned by the proposed *BSNet* are distributed in a smaller feature space and are more discriminative, we use a gradient-based technique, *Grad-CAM* [44], to visualize the important regions in the original images, which is illustrated in Figure 1.

In Figure 5, we randomly select 8 images (4 from *Stanford-Cars*, 4 from *Stanford-Dogs*) and resize the original images to the same size as the output of the embedding layer  $f_\phi$ . The resized raw images are compared to the outputs of *Grad-CAM* under the setting of *Matching Network*, *Prototype Network*, *Relation Network*, *DN4 Network*, our *Cosine Network* and the proposed *BSNet*. Figure 5 shows that in comparison with other compared methods, the proposed *BSNet* consistently has a reduced number of class-discriminative regions concentrated in the regions of “cars” or “dogs”, thus the features learned by *BSNet* are more robust and efficient.

#### G. Effect of Changing Similarity Modules on *BSNet*

To further show the applicability of *BSNet*, we implemented *BSNet* based on two similarity metrics among *Prototype* module, *Relation* module and *Matching* module (see Table V). In addition, we also extended *BSNet* to a network with three or four similarity modules (see Table VI).

From Table V, we can observe that, for 5-way 5-shot classification, either *BSNet* is better than single similarity metric networks, e.g., *BSNet(M&P)* outperforms *Matching Network* and *Prototype Network* on the *Stanford-Cars* dataset, or *BSNet* performs in between the two single similarity metric networks, e.g., *BSNet(M&R)* underperforms *Matching Network* but outperforms *Relation Network* on the *Stanford-Cars* dataset. For 5-way 1-shot classification, the accuracy of *BSNet* is in between those of the two single similarity metric networks on the two datasets.

From Table VI, we can see that, when we extend the proposed *BSNet* to a network with three similarity metrics, it still works well on the *Stanford-Cars* and *CUB-200-2011* datasets. *BSNet* either outperforms all single similarity networks, or performs better than the worst one of three single similarity networks but worse than the best one of three single similarity networks. When we equip *BSNet* with four similarity metrics, similar patterns can be found.

In short, the above results indicate that: firstly, when we replace the similarity branch with different similarity metric modules, the improvement over a single similarity network is consistent with that of the *BSNet* with Relation module and Cosine module. Secondly, when we extend *BSNet* to multiple similarity metrics, the improvement is also consistent with the bi-similarity *BSNet*.

#### H. Discussion

The experimental results have demonstrated the effectiveness of *BSNet*. As listed in Table II and Table IV, in most cases, the proposed *BSNet* can improve the state-of-the-art metric-based few-shot learning methods. This can be attributed to the following properties of *BSNet*. Firstly, compared with the single-similarity network, even though the proposed *BSNet* contains more model parameters, it does not necessarily increase the empirical Rademacher complexity according to the Theorem 1. When a single-similarity based few-shot method has larger model complexity, bi-similarity can reduce the model complexity, and thus resulting in better generalization performance. Therefore, when a single-similarity network is excessively flexible, a *BSNet* with better generalization can be constructed by simply adding an additional similarity module with lower model complexity onto the single-similarity network. Secondly, *BSNet* can learn more discriminative features than other single-similarity networks, as the feature embedding in *BSNet* needs to meet two distinct similarity measures.

The experimental results demonstrate the applicability of *BSNet*. As listed in Table V and VI, the proposed *BSNet* either performs better than all individual similarity networks, or performs better than the worst one of three individual similarity networks. We can interpret this pattern from the perspective of the Rademacher complexity: the Rademacher complexity of the proposed *BSNet* is no more than the average of the Rademacher complexities of two individual networks. Hence, according to the relationship between a model generalization error bound and an empirical Rademacher complexity (Theorem 3.1 in [45]), when the Rademacher complexity of the proposed *BSNet* is lower than both two individual networks, the proposed *BSNet* may perform better than the two individual networks; otherwise, *BSNet* may perform in between the two individual networks. Thus, in terms of the similarity metric selection for *BSNet*, it is desirable to have each similarity metric module with lower Rademacher complexity.

#### V. CONCLUSION

In this paper, we proposed a novel neural network, namely *Bi-Similarity Network*, for few-shot fine-grained image classification. The proposed network contains a shared embedding module and a bi-similarity module, the structure of which contains a parameter sharing mechanism. The *Bi-Similarity Network* can learn fewer but more discriminative regions compared with other single metric/similarity based few-shot learning neural networks. Extensive experiments demonstrate that the proposed *BSNet* outperforms or matches previous state-of-the-art performance on fine-grained image datasets.

#### REFERENCES

- [1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.
- [2] N. Dvornik, C. Schmid, and J. Mairal, "Diversity with cooperation: Ensemble methods for few-shot classification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3723–3731.
- [3] W. Lin, X. He, X. Han, D. Liu, J. See, J. Zou, H. Xiong, and F. Wu, "Partition-aware adaptive switching neural networks for post-processing in hevc," *IEEE Transactions on Multimedia*, 2019.
- [4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
- [5] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, and G. Wang, "Recent advances in convolutional neural networks," *arXiv preprint arXiv:1512.07108*, 2015.
- [6] W. Lin, Y. Zhou, H. Xu, J. Yan, M. Xu, J. Wu, and Z. Liu, "A tube-and-droplet-based approach for representing and analyzing motion trajectories," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 8, pp. 1489–1503, 2016.
- [7] X. Li, D. Chang, Z. Ma, Z. Tan, J. Xue, J. Cao, J. Yu, and J. Guo, "Oslnet: Deep small-sample classification with an orthogonal softmax layer," *IEEE Transactions on Image Processing*, vol. 29, pp. 6482–6495, 2020.
- [8] X. Li, Z. Sun, J.-H. Xue, and Z. Ma, "A concise review of recent few-shot meta-learning methods," *Neurocomputing*, 2020.
- [9] X. Li, L. Yu, X. Yang, Z. Ma, J.-H. Xue, J. Cao, and J. Guo, "Remarnet: Conjoint relation and margin learning for small-sample image classification," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.
- [10] Y. Lifchitz, Y. Avrithis, S. Picard, and A. Bursuc, "Dense classification and implanting for few-shot learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9258–9267.
- [11] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to compare: Relation network for few-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1199–1208.
- [12] S. Gidaris, A. Bursuc, N. Komodakis, P. Pérez, and M. Cord, "Boosting few-shot visual learning with self-supervision," in *IEEE International Conference on Computer Vision*, 2019, pp. 8059–8068.
- [13] L. Bertinetto, J. F. Henriques, J. Valmadre, P. Torr, and A. Vedaldi, "Learning feed-forward one-shot learners," in *Advances in Neural Information Processing Systems*, 2016, pp. 523–531.
- [14] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*, 2017, pp. 1126–1135.
- [15] C. Finn, K. Xu, and S. Levine, "Probabilistic model-agnostic meta-learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 9516–9527.
- [16] H. Qi, M. Brown, and D. G. Lowe, "Low-shot learning with imprinted weights," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5822–5830.
- [17] H. Li, W. Dong, X. Mei, C. Ma, F. Huang, and B.-G. Hu, "LGM-Net: Learning to generate matching networks for few-shot learning," *arXiv preprint arXiv:1905.06331*, 2019.
- [18] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap, "Meta-learning with memory-augmented neural networks," in *International conference on machine learning*, 2016, pp. 1842–1850.
- [19] T. Munkhdalai and H. Yu, "Meta networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 2554–2563.
- [20] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 3630–3638.
- [21] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 4077–4087.
- [22] W. Li, L. Wang, J. Xu, J. Huo, Y. Gao, and J. Luo, "Revisiting local descriptor based image-to-class measure for few-shot learning," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7260–7268.
- [23] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," in *International Conference on Learning Representations*, 2019.
- [24] W. Lin, Y. Shen, J. Yan, M. Xu, J. Wu, J. Wang, and K. Lu, "Learning correspondence structures for person re-identification," *IEEE Transactions on Image Processing*, vol. 26, no. 5, pp. 2438–2453, 2017.

- [25] L. Qiao, Y. Shi, J. Li, Y. Wang, T. Huang, and Y. Tian, "Transductive episodic-wise adaptive metric for few-shot learning," in *IEEE International Conference on Computer Vision*, 2019, pp. 3603–3612.
- [26] P. Tokmakov, Y.-X. Wang, and M. Hebert, "Learning compositional representations for few-shot recognition," in *IEEE International Conference on Computer Vision*, 2019, pp. 6372–6381.
- [27] F. Hao, F. He, J. Cheng, L. Wang, J. Cao, and D. Tao, "Collect and select: Semantic alignment metric learning for few-shot learning," in *IEEE International Conference on Computer Vision*, October 2019.
- [28] X. Li, L. Yu, D. Chang, Z. Ma, and J. Cao, "Dual cross-entropy loss for small-sample fine-grained vehicle classification," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4204–4212, 2019.
- [29] X. Zhang, H. Xiong, W. Zhou, W. Lin, and Q. Tian, "Picking deep filter responses for fine-grained image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1134–1142.
- [30] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z. Si, and J. Guo, "Fine-grained vehicle classification with channel max pooling modified cnns," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3224–3233, 2019.
- [31] X. Zhang, H. Xiong, W. Zhou, W. Lin, and Q. Tian, "Picking neural activations for fine-grained recognition," *IEEE Transactions on Multimedia*, vol. 19, no. 12, pp. 2736–2750, 2017.
- [32] D. Chang, Y. Ding, J. Xie, A. K. Bhunia, X. Li, Z. Ma, M. Wu, J. Guo, and Y.-Z. Song, "The devil is in the channels: Mutual-channel loss for fine-grained image classification," *IEEE Transactions on Image Processing*, vol. 29, pp. 4683–4695, 2020.
- [33] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2, 2015.
- [34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.
- [35] K. R. Allen, E. Shelhamer, H. Shin, and J. B. Tenenbaum, "Infinite mixture prototypes for few-shot learning," *arXiv preprint arXiv:1902.04552*, 2019.
- [36] J. Kim, T. Kim, S. Kim, and C. D. Yoo, "Edge-labeling graph neural network for few-shot learning," pp. 11–20, 2019.
- [37] Z. Wu, Y. Li, L. Guo, and K. Jia, "PARN: Position-aware relation networks for few-shot learning," in *IEEE International Conference on Computer Vision*, 2019, pp. 6659–6667.
- [38] J. Zhang, C. Zhao, B. Ni, M. Xu, and X. Yang, "Variational few-shot learning," in *IEEE International Conference on Computer Vision*, 2019, pp. 1685–1694.
- [39] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.
- [40] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013.
- [41] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *IEEE International Conference on Computer Vision Workshops*, 2013, pp. 554–561.
- [42] A. Khosla, N. Jayadevaprakash, B. Yao, and F.-F. Li, "Novel dataset for fine-grained image categorization: Stanford dogs," in *CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, vol. 2, no. 1, 2011.
- [43] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 dataset," 2011.
- [44] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *IEEE International Conference on Computer Vision*, 2017, pp. 618–626.
- [45] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2012.