Hybrid Pooling Networks for Few-shot Learning

Shaoqing Tan, Ruoyu Yang
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210023, China
tansq2018@gmail.com, yangry@nju.edu.cn

***Abstract*—We propose Hybrid Pooling Networks (HPN) for few-shot learning, where a classifier must learn to recognize new classes not seen in the training set, given only few examples from each new class. HPN learns deep similarity metrics between labeled and unlabeled instances, consisting of two independent networks named Bilinear Pooling Network (BPN) and Element-wise Interaction Network (EIN). By using bilinear pooling, BPN produces a combination of the inputs containing comprehensive information about pair-wise similarities. In EIN, we apply "alignment" on features, making feature-wise interactions on input images. This operation contributes to capturing finer similarities. Both of BPN and EIN bring significant improvements on miniImageNet and Omniglot few-shot datasets. We incorporate the two networks to form HPN by simply averaging their outputs, showing much better performance. Besides, all of our models are easily extended to zero-shot learning. Experiments on CUB and AWA2 demonstrate that our approaches can achieve state-of-the-art performance on zero-shot tasks.**

I. Introduction

Deep learning models have achieved impressive results in areas such as image classification [1], [2], natural language processing [3], and speech recognition [4]. However, these models require extensive, incremental training on large scale datasets. By contrast, large amounts of labeled data are difficult to obtain in those fields which need much professional knowledge (*e.g*. labeling medical images). Although data augmentation and regularization techniques can alleviate overfitting, they do not solve this problem. Besides, even children can acquire a new concept with few examples in a short time while deep learning models seem too "stupid" to learn well with limited data. Motivated by the learning ability of human and the drawbacks of deep learning on sparse data problems, few-shot [5]–[10] and zero-shot [11]–[16] learning have been actively pursued recently.

In few-shot learning, the model is expected to categorize unseen data with very few labeled samples provided for each class. Similarly, zero-shot learning aims to recognize those unseen objects, given only a high-level description each class. Training a conventional classifier on very small amounts of labeled data is unrealistic because of severe overfitting. Therefore, instead of attacking overfitting directly, researchers have explored ways to leverage a distribution of similar tasks, inspired by human learning [17]. This defines a learning setup named "meta-learning" where the input-output pairs of a model are no longer given by instances and their associated labels, but by instance pairs and their associated similarity. In most contemporary approaches, transferrable knowledge is first learned in the training set whose label space is disjoint with the test set, including appropriate representations [8], [10], good initial configurations [18], and nonlinear similarity metrics [7], [9]. Then in the test set, models learn to fine-tune on the specific problem [18] or calculate the similarity between pairs directly [7]–[10].

However, most works on few-shot learning either use complicated models which imitate the process of human cognition [6], [19] or implement the inference mechanisms [20], [21], or need to fine-tune on each specific target problem [18]. Recently, research related to metric and embedding learning has made significant progress [5], [7]–[10], [17]. These works learn similarity metrics essentially, which first gain the representations of input images via a CNN, and then learn a way to combine the representations of labeled and unlabeled images to propagate the label information toward the target images. Among them, some works use fixed linear distance metrics (*e.g*. Euclidean, cosine or Mahalanobis) [5], [8], [10], [17] while the others [7], [9] apply nonlinear metrics to calculating the similarity. Although these nonlinear metrics have some advantages over the former, their improvements are not obvious.

In this paper, we propose two networks, *Bilinear Pooling Network* (BPN) and *Element-wise Interaction Network* (EIN) which use nonlinear metrics, to learn similarity between image pairs on few-shot tasks. Experimental results in our works show better performance than the previous works using nonlinear metrics. The two networks differ in the way of incorporating the representations of images and processing the combination. In BPN, we use bilinear pooling [22] to get the joint representations of image pairs. This pooling method is capable of making comprehensive interactions between all the elements of two inputs, which contributes to finding global similarities. The key operation of EIN is aligning two feature vectors such that the same kind of features can be packaged together to make further processing. In this way, EIN concentrates on the comparison of the same kind of features only. This feature-by-feature way can capture subtler similarities and differences than BPN. By fusing the two complementary networks further, our eventual two-stream model, named Hybrid Pooling Networks (HPN, see Figure 1), achieves fantastic performance on *mini*ImageNet and Omniglot few-shot benchmarks. Besides, with minor modifications on our models, they can be transferred on zero-shot tasks as well, which beat other works on CUB and AWA2.

Corresponding author: Ruoyu Yang, yangry@nju.edu.cn

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

Overall, our contribution is two-fold. First, we propose EIN and BPN respectively, which are applicable to both few-shot and zero-shot learning. Second, we observe that the two networks have complementary properties to some extent, urging the formation of HPN which improves the performance further. Evaluation on four benchmarks proves that our models have great advantages over other approaches.

II. Related work

The concept of one-shot or few-shot classification was first proposed by [20]. Early works on few-shot learning usually utilized generative models to make complex iterative inference on targets [20], [21]. With the success of image classification on large-scale datasets by using deep models, generalizing deep learning approaches to few-shot classification has aroused interest. Inspired by the fact that children are able to rapidly learn the concept of "zebra" because they have already obtained other knowledges, *e.g*. recognizing a horse, many of recent approaches extract some transferrable knowledge from a set of auxiliary tasks, contributing to few-shot learning task without directly learning from few examples which may easily lead to overfitting. This strategy is called *meta-learning* [23], [24], referring to a scenario where an agent learns at two levels, each associated with different time scales. One level is rapid learning occuring *within* a task, *e.g*. when learning to accurately classify within a particular dataset. Another level accrue knowledge gradually *across* tasks, which can guide rapid learning in a specific task and capture the way that task structure varies across target domains [25].

Imitating Human Memory An important property of human-level learning is using memory to interact with environment. Some approaches try to imitate human memory by leveraging external or internal memories (*e.g*. recurrent neural networks) [6], [19]. Given a new example, their models compare it to historic information stored in the memory such that a memory is retrieved by calculating a weighted sum over all the elements, which can be used for prediction. After getting the ground truth of this example, memory will be updated according to it. However, as the number of new examples increases, which means long-term learning, it is difficult to store all the useful information and erase redundant ones reliably. By contrast, instead of imitating the learning mechanism of human by using complex RNN models and external memories, our models only consists of CNNs and fully-connected layers.

Fine-Tuning Between Tasks The knowledge about "horse" in a human brain can be transferred to what about "zebra" with few modifications. Another category of approaches leverage fine-tuning on few-shot problem. An approach named MAML [18] meta-learns initial values of neural network weights which is conducive to fine-tune on few-shot tasks. Specifically, they sample individual tasks from a given multiple task training set, then a base neural network model learns to fine-tune to solve each target problem. The success at each task will update parameters in the base model respectively, which essentially drives the production of an initial condition easily to fine-tune. Although their method is model-agnostic, it still need to fine-tune on each specific target task. In our approach, there is no need to do any fine-tuning operations toward target problem.

Learning Similarity Learning the similarity between image pairs seems more directly when labeled data is limited. This type of method is relevant to distance metric learning [26]. In early studies, Koch *et al*. [5] used a deep siamese network for few-shot learning, which was first proposed by Bromley *et al*. [27] to perform signature verification. Deep metric learning has made progress by using different forms of loss since then. In particular, training is performed with triplet loss to enforce a distance ranking in [28]. Kihyuk [29] used multi-class N-pair loss to train the whole model, which improves upon the triplet loss by pushing away multiple negative examples jointly at each update and has been utilized in most metric-based few-shot learning works. Vinyals *et al*. [10] presented an end-to-end trainable *k*NN model by using cosine distance, with an attention mechanism over a learned embedding of the labeled data to predict the categories of the unlabeled points. Snell *et al*. [8] extended this work by using euclidean distance as a similarity metric and they builded a prototype representation for each class, showing significant improvements. More recently, applying nonlinear metrics to few-shot tasks has been of interest. Sung *et al*. [9] proposed a relation module which concatenates the feature maps of two images and send this concatenation to CNNs to learn the similarity. Garcia and Bruna [7] introduced an approach named graph neural networks, where the input samples form the nodes of the graph and edge weights are computed as a nonlinear function of the absolute difference between node features. Despite some novelty, these approaches do not show obvious improvements.

Our work is most related to the above approaches based on nonlinear metrics. We argue that neural networks could be more powerful in finding pair-wise similarities by using appropriate pooling methods. Bilinear pooling, which is used in our BPN, has been applied on fine-grained visual recognition [30] and visual question answering [31] recently, which shows great performance in their fields. Our experiments demonstrate that this method also has advantages on both few-shot and zero-shot tasks. In addition, we propose a novel technique named *element-wise convolution* utilized in EIN, applying to few-shot tasks. Both BPN and EIN are competitive compared to other methods. In particular, EIN performs much better than BPN on fine grained few-shot tasks (see Section IV-C), which proves that EIN can capture finer similarities and differences.

III. Method

A. Problem Set-up

The aim of few-shot recognition is to classify images whose categories can not be seen before, only with few labeled images as examples in each class. If there are K labeled examples for each of C unseen classes, the target few-shot task is called C-way K-shot.

Few-shot classification is different from conventional classification that the $K \times C$ images are not suitable for training

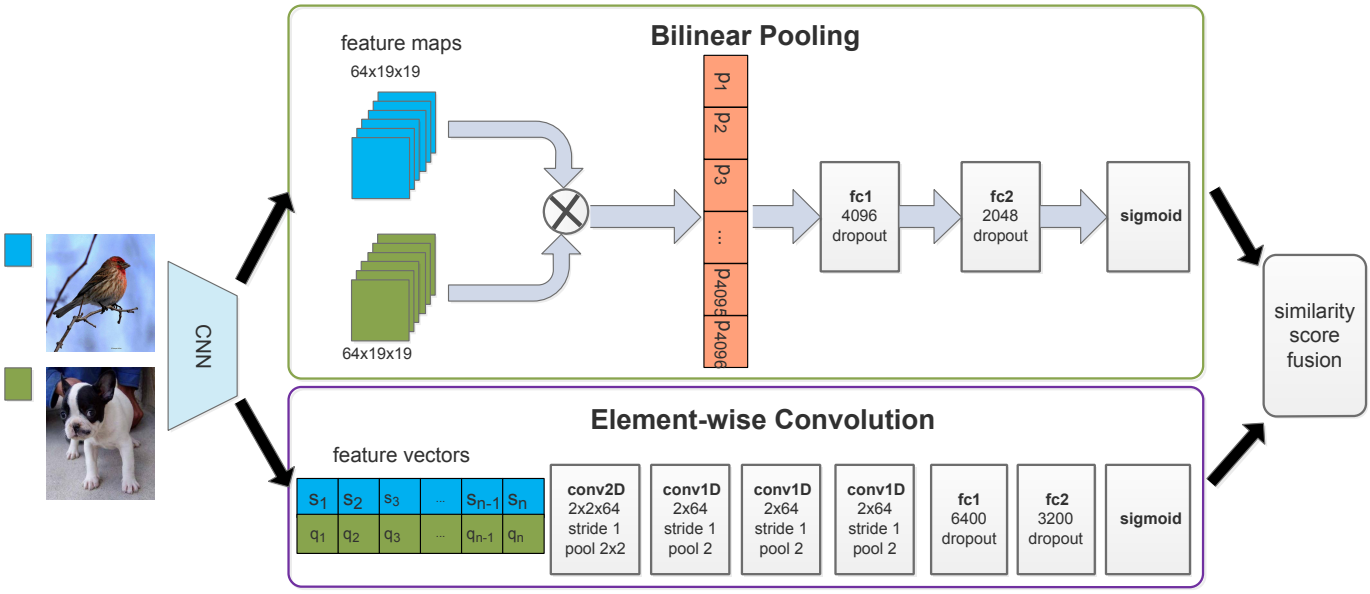

Fig. 1. Hybrid Pooling Networks architecture for few-shot learning.

directly, which easily leads to overfitting. A more effective way is to form an "episode" during training which mimics the test procedure [10]. Specifically, training set has its own label space which is disjoint with the test set. During one episode in the training procedure, $K \times C$ labeled images as above mentioned are sampled from training set, as well as several unlabeled images for each of C classes as query data. At the time of test, the same procedure will be performed except the data set is replaced by the test set. Formally, the $K \times C$ labeled images are called *support set* while the unlabeled ones are named as *query set*.

B. Bilinear Pooling Network

Bilinear models were first introduced by Tenenbaum and Freeman [22] to separate style and content. The key operation, *bilinear pooling*, takes the outer product of two vectors $x_1 \in \mathbb{R}^{n_1}$ and $x_2 \in \mathbb{R}^{n_2}$ (*e.g.* $x_1 {x_2}^T$), allowing a multiplicative interaction between all the elements of both vectors, which produces a very comprehensive combination. It has been considered for semantic segmentation and fine grained recognition using both hand-tuned and learned features, showing its great advantages in this field. Inspired by Fukui *et al.* [31] utilizing bilinear pooling to incorporate representations from two different modalities (*e.g.* textual and visual information) in VQA, we exploit bilinear pooling as one of our incorporating method in both few-shot and zero-shot tasks.

The main difference between few-shot and zero-shot learning is that the inputs of the former are the same modalities (image pairs) while the latter has different ones (images and attribute vectors). Therefore, there are some differences in solving few-shot and zero-shot problems.

Few-shot For few-shot tasks, a CNN consisting of a hierarchy of convolutional and pooling layers is used as feature extractors (see Figure 2). We compute the outer product between two sets of feature maps instead of feature vectors. In this way, each element in the pooling result is produced by two feature maps, thus a multiplicative interaction emerges between the feature maps of both sets. Besides, it also avoids the high dimension by directly applying outer product for two flattened feature vectors. Let $x_i, x_j \in \mathbb{R}^{c \times h \times w}$ denote the output of the CNN for two images, where c is the number of the feature maps and h, w are the height and the width of it. We flatten every feature map in x_i, x_j to form $x_i', x_j' \in \mathbb{R}^{c \times l}$, where $l = h \times w$. Then the bilinear pooling matrix can be calculated as: $\psi_{i,j} = x_i {x_j}^T \in \mathbb{R}^{c \times c}$. After flattening $\psi_{i,j}$ to a $c \times c$ length vector, we use it as features sent into fully-connected layers (see Figure 1).

Zero-shot In zero-shot learning, the inputs are images and attribute vectors. We apply L_2-normalization on the original attribute vectors, and then get the outer product between the normalized vectors and the image representations to form the pooling features. Except that, the processing of the other parts is the same as what in few-shot learning.

C. Element-wise Interaction Network

In this section, we demonstrate Element-wise Interaction Network (EIN) for few-shot and zero-shot learning. The basic idea of EIN is to align two vectors to learn their similarities. Before this operation, we need the input feature vectors share the same length. For few-shot, embeddings are inherently equal in length because images are processed by the same feature extractors. For zero-shot, we can project images and attribute vectors into the same embedding space. Therefore, the specific methods of EIN are partly different for few-shot and zero-shot learning.

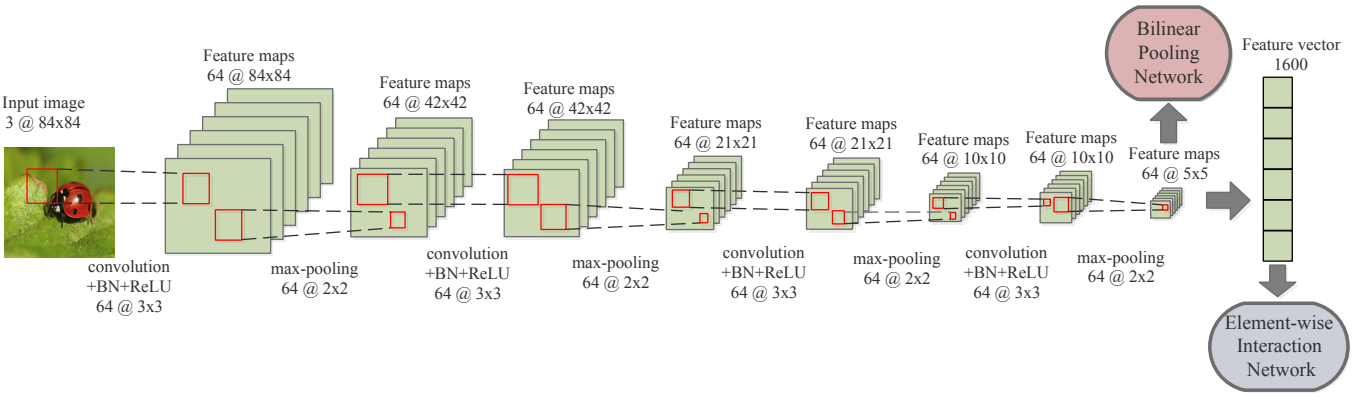

Fig. 2. Convolutional blocks for feature extraction. "BPN" only denotes batch normalization in this figure.

Few-shot We propose an incorporating method specifically for single modality, which named *element-wise convolution* and is used in EIN for few-shot learning. Note that the feature extractors of EIN share the same architecture with BPN. First, we get two feature vectors $v_1, v_2 \in \mathbb{R}^n$ of the input image pairs via the CNN. Then, the two vectors are concatenated to form a matrix of size $2 \times n$. We treat it as a single channel image, using a 2D CNN to extract local features of this matrix, after which a hierarchy of 1D convolutional and pooling layers are connected to extract local sequential feature further (see Figure 1).

The two vectors can be processed in this way because image pairs are fed into the same feature extractors, and each feature map at the top of the feature extractors can be explained as a kind of feature of the image, thus two feature maps from the same position (*i.e.* extracted by the same filter) in the two sets of feature maps represent the same kind of feature. As depicted in Figure 3, a pair of feature maps extracted by the same filter is visually more similar if their original images come from the same class. By flattening feature maps to vectors and combining two vectors element by element, the same features are aligned. Unlike using fixed metrics that feature vectors are solely compared in an element-wise way [8], [10], we apply a 2D convolutional layer to learning similarity between same kinds of features in the element-wise combined matrix, followed by three 1D convolutional layers. Since CNN is good at extracting local features, we argue that a 2D CNN is able to capture local similarities between two sets of features, and by adding 1D CNNs, the local similarities can be extracted further.

Zero-shot We can in principle project attribute vectors into visual embedding space and use *element-wise convolution* as in few-shot learning. However, because they still come from handcrafted features which are far from features learned by CNNs, the performance of such a model is not satisfactory. In spite of that, the projection for attribute vectors can still bring some properties which may make the attribute embeddings weakly aligned to image embeddings. We thus apply a common method in VQA [32] to replacing *element-wise convolution*, namely *element-wise product* which does not call for same modalities. By projecting attribute vectors into visual embedding space, it simply makes inner product of an embedded attribute vector and an image representation as the incorporating feature. We use this method in EIN for zero-shot learning, which show its great performance.

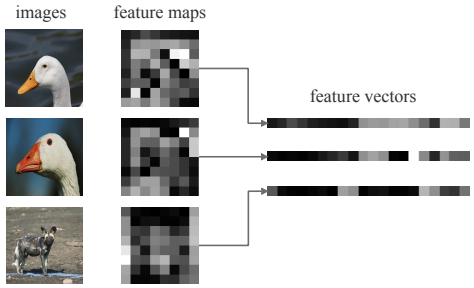

Fig. 3. Visualization of feature maps and vectors. We sample three images in *mini*ImageNet dataset, where the first two belong to the same class. We resize these images to 128×128 and get the feature maps through a randomly selected filter in the last convolution of our feature extractors (see Figure 2). We depict the first 20 elements in the feature vectors after flattening them.

D. Hybrid Pooling Networks

BPN allows comprehensive interactions among feature elements, while EIN focuses on the comparison between the same kinds of features. We devise our eventual architecture accordingly, fusing the two networks because of their complementary properties, namely *Hybrid Pooling Networks* (HPN), as shown in Figure 1. In order to show the two streams are greatly complementary, we just simply average the outputs of them to see how much improvement the fused model can make.

Few-shot Unlike conventional multi-class classification using cross entropy as loss function after a softmax layer, we add a sigmoid activation to the output of the model, which maps onto the interval $[0, 1]$. Thus the final output is $s_{i,j}$, a scalar denoting the similarity between image x_i and x_j, and naturally a mean square error loss function is used.

For C-way 1-shot, we sample Q query images each class. Since there are $N = C^2 \times Q$ image pairs sent into our model and we have C classes, the loss is called *C-class N-pair loss* [29]:

$$J(\phi) = \frac{1}{N} \sum_{i=1}^{C} \sum_{j=1}^{C \times Q} (\mathbb{I}(y_i == y_j) - s_{i,j})^2 \quad (1)$$

Where ϕ denotes the parameters in the networks, and y_i is the label of x_i. Let $\mathcal{B}(e_i, e_j)$ denote *bilinear pooling* and $\mathcal{E}(e_i, e_j)$ denote *element-wise convolution* for two image embeddings. The similarity score can be computed as equation 2, where $\mathcal{F}$ is the fusion method.

$$s_{i,j} = \mathcal{F}(\mathcal{B}(e_i, e_j), \mathcal{E}(e_i, e_j)) \quad (2)$$

Note that we just use averaging as the fusion method in our model, because we do not rely on the capability of ensemble methods to improve the performance, although various ensemble approaches are applicable.

For C-way K-shot, where $K > 1$, $e_{m,n}$ is the embedding (feature maps or feature vector) of the nth image in the mth class. We construct a *prototype* [8] for each class:

$$c_m = \frac{1}{K} \sum_{n=1}^{K} e_{m,n} \quad (3)$$

Then the classification is performed for an embedded query image by simply finding the most similar class prototype, which can be considered as a C-way 1-shot task.

Zero-shot Instead of providing a support set of training images, zero-shot learning gives a manually annotated attribute vector or a textual description [33] for each class. Therefore, they are totally different from images (*i.e.* different modalities). As discussed in Section III-B and Section III-C, we use the zero-shot version of BPN and EIN to perform zero-shot experiments.

Let v_i denote the attribute vector of the ith class, the similarity score for each embedded image e_j will be:

$$s_{i,j} = \mathcal{F}(\mathcal{B}(v_i, e_j), \mathcal{E}_p(f(v_i), e_j)) \quad (4)$$

Where f is the embedding network to project attribute vectors into visual space and $\mathcal{E}_p$ denotes the element-wise product module.

E. Network Architecture

The overview of our network for few-shot learning is shown in Figure 1. As most previous works on few-shot learning use four convolutional blocks as feature extractors without fine tuned [5], [8]–[10], we follow their architectures so as to fair comparison. More specifically, each convolutional block contains a 64-filter 3×3 convolution, a 2×2 max-pooling layer, a batch normalization layer and a rectified linear unit. We depict an example above (Figure 2), where the input image is sampled from *mini*ImageNet resized to 84×84. In BPN, we use the feature maps of the last convolutional block. In EIN, these feature maps are flattened as the input. Note that BPN and EIN do not share the parameters of the feature extractors.

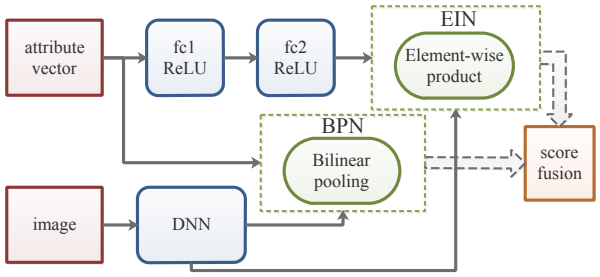

Fig. 4. Architecture for zero-shot learning. Where DNN is an pre-trained network on ImageNet (*e.g.* Densenet or Resnet).

In BPN, we use the bilinear pooling vector (see Section III-B) as the input of two fully-connected layers whose dimensions of the output are 8 and 1, with a ReLU non-linearity layer after the first fully-connected layer. For EIN with *element-wise convolution*, we use one 2D and three 1D convolutional blocks, where the setting of 2D one is the same as what in the Figure 2. Each 1D convolutional block consists of a 64-filter 3 convolution, a 2 max-pooling layer, a batch normalization layer and a rectified linear unit. The output of the last convolutional block is flattened and sent to two fully-connected layers whose hidden length and output dimension are the same as what in BPN. Both of the above two networks apply a sigmoid activation to their output layer, producing similarity scores in range of 0 to 1.

Our zero-shot learning architecture is shown in Figure 4. A deep neural network (DNN) pre-trained on ImageNet is utilized to extract image features. For a given attribute vector $v_i \in \mathbb{R}^{l_1}$ and an embedded image $e_i \in \mathbb{R}^{l_2}$, we send them to BPN to get the outer product of them, generating a $l_1 \times l_2$ incorporating feature. For another stream, we use two fully-connected layers to project attribute vectors into visual space, after that we compute the inner product between embedded attribute vectors and image features, producing another different incorporating feature. Finally, a fusion method is applied to fusing the above two outputs, which is the same as the one in the few-shot learning.

IV. Experiments

For few-shot learning, we evaluate our approaches on Omniglot [21] and *mini*ImageNet [10], [34]. For zero-shot learning, we perform experiments on Caltech-UCSD Birds-200-2011(CUB) [35] and Animals with Attributes 2 (AwA2) [36]. In addition, we conduct one-shot experiments on CUB to evaluate our models on fine-grained tasks. Note that all the models are trained from scratch on each dataset without additional data, except a pre-trained DNN in zero-shot learning.

A. Few-shot Classification

1) **mini*ImageNet*:** The *mini*ImageNet dataset is derived from ILSVRC-12 dataset [37], originally proposed by Vinyals *et al.* [10]. This dataset contains 60,000 color images divided into 100 classes, each having 600 images. In our experiments, we use the splits proposed by Ravi and Larochelle [34]

Model	Fine Tune	5-way Acc.	
		1-shot	5-shot
MATCHING NETS [24]	N	43.56 ± 0.84%	55.31 ± 0.73%
META NETWORK [19]	N	49.21 ± 0.96%	-
MEAT-LEARN LSTM [34]	N	43.44 ± 0.77%	60.60 ± 0.71%
MAML [18]	Y	48.70 ± 1.84%	63.11 ± 0.92%
PROTOTYPICAL NETS [8]	N	49.42 ± 0.84%	68.20 ± 0.66%
GRAPH NEURAL NETS [7]	N	50.33 ± 0.36%	66.41 ± 0.63%
RELATION NET [9]	N	50.44 ± 0.82%	65.32 ± 0.70%
BPN	N	53.29 ± 0.62%	68.34 ± 0.61%
EIN	N	52.78 ± 0.72%	68.29 ± 0.64%
HPN	N	**55.17 ± 0.61%**	**71.26 ± 0.69%**

TABLE I
FEW-SHOT CLASSIFICATION ACCURACIES ON *mini*IMAGENET. ALL ACCURACY RESULTS ARE AVERAGED OVER 600 TEST EPISODES AND ARE REPORTED WITH 95% CONFIDENCE INTERVALS. FOR EACH TASK, THE BEST-PERFORMING METHOD IS HIGHLIGHTED. '-': NOT REPORTED.

like most works. 100 classes are divided into 64 training, 16 validation, and 20 test classes in their splits, where the 16 validation classes are used for monitoring generalization performance only.

Training We train BPN and EIN respectively and then combine the two networks to form HPN. We just average the score produced by the two networks as the final prediction for the similarity of a certain image pair, although other choices are possible.

Following most few-shot learning works, we conduct 5-way 1-shot and 5-way 5-shot classification on this dataset. In addition to K support images in K-shot task, we sample 15 query images each class for 1-shot learning and 10 query images each class for 5-shot learning. All the images are resized to 84×84.

Results At the time of test, we follow [9] that we batch 15 query images per class in each episode for evaluation in both 1-shot and 5-shot task. The classification accuracies are computed by averaging over 600 randomly generated episodes from the test set.

The results are shown in Table I. HPN achieves state-of-the-art performance on both 1-shot and 5-shot accuracy. Note that BPN and EIN beat all the previous methods on 5-way 1-shot task respectively, and their results are only slightly lower than prototypical networks [8] on 5-shot task. However, prototypical networks are trained by applying a method that they use far more than 5 classes (*i.e*. numbers of way $\gg$ 5) to train and only 5 classes to test, which makes training much harder than test. If not using this trick and making the numbers of way the same in training and test like us, they only got $46.14 \pm 0.77\%$ for 1-shot evaluation and $65.77 \pm 0.70\%$ for 5-shot as they reported. We can conclude that BPN and EIN are superior to other models, and after fusing them, the performance can be much better because of their complementary properties.

2) Omniglot: Omniglot consists of 1623 kinds of handwritten characters (*i.e*. 1623 classes) collected from 50 alphabets. Each class consists of 20 different samples. we follow the procedure of [8]–[10], resizing all the images to 28×28 and augmenting data by rotating original images $90°$, $180°$ and $270°$. We use 1200 characters plus rotations (*i.e*. 4800 classes) for training and 423 characters plus rotations (*i.e*. 1692 classes) for test.

Training The training procedure of Omniglot is similar to *mini*ImageNet. As there are 20 samples in each class, we provide 19 query images for 1-shot tasks and 15 query images for 5-shot tasks.

Results We conduct 5-way 1-shot, 5-way 5-shot, 20-way 1-shot and 20-way 5-shot experiments on this dataset. Following previous work [8], [9], classification accuracies on Omniglot are computed by averaging over 1000 randomly generated episodes from the test set. From Table II, most of recent models can achieve nearly 100% in above mentioned tasks, for this dataset is extremely simple which consists of single-channel character images, similar to images in MNIST. Since the intra-class variations of this dataset are too small and the content of the images are simple, the improvements of our approaches are limited. Despite that, EIN beats all the other models and BPN shows its superiority over previous works as well. More simple images result in more simple features and lower intra-class variations make these features more similar such that a CNN is easy to learn local similarity on a combination of two feature vectors in EIN. Although BPN performs worse than EIN, they can still complement each other. By fusing the two networks, HPN achieves state-of-the-art performance especially on 20-way 1-shot, which is the hardest task among the above four.

B. Zero-shot Classification

1) Datasets: Our experiments are performed on two zero-shot datasets: **CUB** [35] and **AWA2** [36]. **CUB** (Caltech-UCSD Birds-200-2011) dataset contains 11,788 images of 200 bird species, divided into 150 seen classes and 50 unseen classes respectively. For each class, a continuous 312-dimension attribute vector is provided. **AWA2** (Animals with Attributes 2) is a coarse-grained dataset consisting of 37,322 images of 50 classes of animals, which is divided into 40 seen classes and 10 unseen classes respectively. AWA2 uses 85 binary and continuous class attributes.

There are two kinds of splits for these datasets. One is called *standard splits* (SS) in [36], which has been used in most works. Under this splits, however, some of the test classes also exist in the ImageNet, which is used to train the image feature extractors (*e.g*. Resnet) for zero-shot learning, violating the assumption that training and test classes are disjoint. To solve this problem, Xian *et al*. [36] propose *proposed splits* (PS) which ensures that the test classes do not appear in the ImageNet. Therefore, we follow **PS** for more reasonable comparison.

2) Experiments Details: As discussed in Section III-E, we still use two networks in zero-shot learning by simply replace *element-wise convolution* with *element-wise product* in EIN. All the image features in both CUB and AWA2 are extracted by ResNet101 [46]. We compute the outer product of a class attribute vector and an image feature vector to get the bilinear pooling feature. A Multi-Layer Perceptron (MLP) is utilized

Model	Fine Tune	5-way Acc.		20-way Acc.	
		1-shot	5-shot	1-shot	5-shot
SIAMESE NETWORK [5]	Y	97.3%	98.4%	88.2%	97.0%
MATCHING NETWORK [10]	Y	97.9%	98.7%	93.5%	98.7%
MATCHING NETWORK [10]	N	98.1%	98.9%	93.8%	98.5%
CONVNET WITH MEMORY [38]	N	98.4%	99.6%	95.0%	98.6%
PROTOTYPICAL NETWORK [8]	N	98.8%	99.7%	96.0%	98.9%
META NETWORK [19]	N	98.9%	-	97.0%	-
MAML [18]	Y	98.7 ± 0.4%	99.9 ± 0.1%	95.8 ± 0.3%	98.9 ± 0.2%
GRAPH NEURAL NETWORKS [7]	N	99.2%	99.7%	97.4%	99.0%
RELATION NETWORK [9]	N	99.6 ± 0.2%	99.8 ± 0.1%	97.6 ± 0.2%	99.1 ± 0.1%
BPN	N	99.6± 0.2%	99.6 ± 0.1%	98.1 ± 0.2%	99.3 ± 0.1%
EIN	N	99.7 ± 0.1%	**99.9 ± 0.1%**	98.3 ± 0.1%	99.5 ± 0.1%
HPN	N	**99.8 ± 0.1%**	**99.9 ± 0.1%**	**98.6 ± 0.2%**	**99.6 ± 0.1%**

TABLE II
FEW-SHOT CLASSIFICATION ACCURACIES ON OMNIGLOT. ALL ACCURACY RESULTS ARE AVERAGED OVER 1000 TEST EPISODES AND ARE REPORTED WITH 95% CONFIDENCE INTERVALS. FOR EACH TASK, THE BEST-PERFORMING METHOD IS HIGHLIGHTED. '-': NOT REPORTED.

Model	CUB	AWA2
DAP [12]	40.0%	46.1%
IAP [12]	24.0%	35.9%
CONSE [39]	34.3%	44.5%
SSE [40]	43.9%	61.0%
DEVICE [41]	52.0%	59.7%
SJE [42]	53.9%	61.9%
ESZSL [43]	53.9%	58.6%
SYNC [44]	55.6%	46.6%
SAE [45]	33.3%	54.1%
RELATION NET [9]	55.6%	64.2%
DEM [16]	51.7%	67.1%
BPN	58.3%	70.6%
EIN	58.6%	72.2%
HPN	**60.4%**	**72.6%**

TABLE III
ZERO-SHOT CLASSIFICATION ACCURACIES ON CUB AND AWA2.

for projecting attribute vector to visual space such that the inner product between image representations and the attribute vectors can be calculated. The hidden length of this MLP is 1,024 and the output dimension is 2,048 which is equal to the length of the top-layer pooling units of ResNet101. At last, we fuse the outputs of BPN and EIN with the same method used in our few-shot model, *i.e*. averaging.

Results We compare our model against various zero-shot models in Table III. EIN and BPN show their better performance than others respectively. By fusing the two streams, HPN further improves the results, outperforming existing models by a big margin on both CUB and AWA2 datasets. Compared to the EIN using *element-wise product* , HPN has increased by nearly 2% on CUB but only increase 0.4% on AWA2 after combining the two networks. This is because AWA2 is simpler than CUB, which is a coarse-grained dataset while CUB is a fine-grained one. Besides, AWA2 only has 10 unseen classes during test while CUB contains 50 ones. Therefore, the improvement of fusion is limited on AWA2.

C. CUB One-shot Trial

On few-shot learning tasks, our model show great performance on *mini*ImageNet and Omniglot. However, these datasets are coarse-grained, of which the difference between classes is obvious. To see the performance on a fine-grained dataset, we conduct 1-shot experiments on CUB, which may clarify the superiority of our networks with nonlinear metrics more clearly.

We still use PS in zero-shot learning to split the dataset into 100 training, 50 validation, and 50 test classes, where the validation set is only used for monitoring generalization performance. Since Prototypical Nets [8] are the best-performed linear-metric approaches and Relation Net [9] is the state-of-the-art method with nonlinear metric, We do the same experiments with them as baselines. Note that we set the same number of the way to train and test for all the models.

The results are reported at Table IV. Models based on nonlinear metrics beat Prototypical Nets using linear metric by a large margin, which is more obvious than what in *mini*ImageNet. Although the result of Relation Net on 5-way task is very close to BPN, it performs much worse when the number of way increases, which demonstrates that BPN is superior to Relation Net on harder tasks. In contrast to the results of *mini*ImageNet, EIN performs the best among all the approaches except HPN in CUB, whose results are much higher than BPN in all the tasks, demonstrating its superiority to capture subtle similarities and differences. In addition, further improvements of the averaging fusion of BPN and EIN appear in HPN as expected.

V. CONCLUSION

In this paper, we have proposed Hybrid Pooling Networks, a competitive model for few-shot and zero-shot learning, composed of Bilinear Pooling Network and Element-wise Interaction Network. In the two streams of the model, Bilinear Pooling Network can generate a comprehensive combination between two inputs which share the same modality or not; Element-wise Interaction Network relies on the alignment of the similar features. For inputs with the same modality,

Model	Metric	1-shot Acc.			
		5-way	10-way	15-way	20-way
PROTOTYPICAL NETS [8]	linear	53.72 ± 0.93%	34.74 ± 0.63%	26.64 ± 0.47%	22.94 ± 0.39%
RELATION NET [9]	nonlinear	61.68 ± 0.88%	42.64 ± 0.54%	33.23 ± 0.46%	29.27 ± 0.45%
BPN	nonlinear	62.34 ± 0.97%	47.01 ± 0.62%	37.34 ± 0.45%	32.43 ± 0.46%
EIN	nonlinear	67.89 ± 0.92%	51.51± 0.79%	42.60 ± 0.67%	37.14 ± 0.65%
HPN	nonlinear	**68.94 ± 0.92%**	**53.21± 0.83%**	**45.06 ± 0.79%**	**39.79 ± 0.85%**

TABLE IV
ONE-SHOT CLASSIFICATION ACCURACIES ON CUB. ALL ACCURACY RESULTS ARE AVERAGED OVER 600 TEST EPISODES AND ARE REPORTED WITH 95% CONFIDENCE INTERVALS. FOR EACH TASK, THE BEST-PERFORMING METHOD IS HIGHLIGHTED.

EIN captures local similarities via CNNs. For multimodal case, inner product is used for element-wise interaction. The two networks have been capable of outperforming other approaches on various few-shot and zero-shot classification tasks. After fusing BPN and EIN, HPN brings the state of the art to a new level. Overall, HPN is an effective, simple and flexible approach for few-shot and zero-shot learning.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China under Grant Nos. 61672273.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.
[3] T. Mikolov, M. Karafiát, L. Burget, J. Černockỳ, and S. Khudanpur, "Recurrent neural network based language model," in *ISCA*, 2010.
[4] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, 2012.
[5] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML*, 2015.
[6] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. P. Lillicrap, "Meta-learning with memory-augmented neural networks," in *ICML*, 2016.
[7] V. G. Satorras and J. B. Estrach, "Few-shot learning with graph neural networks," in *ICLR*, 2018.
[8] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017.
[9] 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 *CVPR*, 2018.
[10] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, *et al.*, "Matching networks for one shot learning," in *NIPS*, 2016.
[11] Z. Ding, M. Shao, and Y. Fu, "Low-rank embedded ensemble semantic dictionary for zero-shot learning," in *CVPR*, 2017.
[12] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *TPAMI*, 2014.
[13] H. Larochelle, D. Erhan, and Y. Bengio, "Zero-data learning of new tasks.," in *AAAI*, 2008.
[14] J. Lei Ba, K. Swersky, S. Fidler, *et al.*, "Predicting deep zero-shot convolutional neural networks using textual descriptions," in *ICCV*, 2015.
[15] M. Rohrbach, M. Stark, and B. Schiele, "Evaluating knowledge transfer and zero-shot learning in a large-scale setting," in *CVPR*, 2011.
[16] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," in *CVPR*, 2017.
[17] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, 2015.
[18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.
[19] T. Munkhdalai and H. Yu, "Meta networks," in *ICML*, 2017.
[20] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *TPAMI*, 2006.
[21] B. Lake, R. Salakhutdinov, J. Gross, and J. Tenenbaum, "One shot learning of simple visual concepts," in *CogSci*, 2011.
[22] J. B. Tenenbaum and W. T. Freeman, "Separating style and content with bilinear models," *Neural computation*, 2000.
[23] S. Thrun, "Lifelong learning algorithms," in *Learning to learn*, 1998.
[24] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artif. Intell. Rev.*, 2002.
[25] C. Giraud-Carrier, R. Vilalta, and P. Brazdil, "Introduction to the special issue on meta-learning," *Machine learning*, 2004.
[26] L. Yang and R. Jin, "Distance metric learning: A comprehensive survey," *Michigan State Universiy*, 2006.
[27] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a siamese time delay neural network," in *NIPS*, 1993.
[28] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*, 2015.
[29] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *NIPS*, 2016.
[30] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *ICCV*, 2015.
[31] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, "Multimodal compact bilinear pooling for visual question answering and visual grounding," in *EMNLP*, 2016.
[32] Z. Yang, X. He, J. Gao, L. Deng, and A. Smola, "Stacked attention networks for image question answering," in *CVPR*, 2016.
[33] M. Elhoseiny, B. Saleh, and A. Elgammal, "Write a classifier: Zero-shot learning using purely textual descriptions," in *ICCV*, 2013.
[34] S. Ravi and H. Larochelle, "Optimization as a model for few-shot learning," in *ICLR*, 2017.
[35] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-ucsd birds 200," 2010.
[36] Y. Xian, H. C. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly," *TPAMI*, 2018.
[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, 2015.
[38] Ł. Kaiser, O. Nachum, A. Roy, and S. Bengio, "Learning to remember rare events," *ICLR*, 2017.
[39] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean, "Zero-shot learning by convex combination of semantic embeddings," in *ICLR*, 2014.
[40] Z. Zhang and V. Saligrama, "Zero-shot learning via semantic similarity embedding," in *ICCV*, 2015.
[41] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, *et al.*, "Devise: A deep visual-semantic embedding model," in *NIPS*, 2013.
[42] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele, "Evaluation of output embeddings for fine-grained image classification," in *CVPR*, 2015.
[43] B. Romera-Paredes and P. Torr, "An embarrassingly simple approach to zero-shot learning," in *ICML*, 2015.
[44] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," in *CVPR*, 2016.
[45] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *CVPR*, 2017.
[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.