NETWORK REPARAMETERIZATION FOR UNSEEN CLASS CATEGORIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Many problems with large-scale labeled training data have been impressively solved by deep learning. However, Unseen Class Categorization (UCC) with minimal information provided about target classes is the most commonly encountered setting in industry, which remains a challenging research problem in machine learning. Previous approaches to UCC either fail to generate a powerful discriminative feature extractor or fail to learn a flexible classifier that can be easily adapted to unseen classes. In this paper, we propose to address these issues through network reparameterization, *i.e.*, reparametrizing the learnable weights of a network as a function of other variables, by which we decouple the feature extraction part and the classification part of a deep classification model to suit the special setting of UCC, securing both strong discriminability and excellent adaptability. Extensive experiments for UCC on several widely-used benchmark datasets in the settings of zero-shot and few-shot learning demonstrate that, our method with network reparameterization achieves state-of-the-art performance.

1 INTRODUCTION

The rich and accessible labeled data has fueled the revolutionary successes of deep learning in various tasks, *e.g.*, visual recognition (He et al. (2016)), object detection (Ren et al. (2015)), machine translation (Bahdanau et al. (2014)), etc. However, requiring numerous annotated data severely limits the applicability of deep learning algorithms to Unseen Class Categorization (UCC) for which we only have access to a limited amount of information, which is frequently encountered in industrial applications. Recently, an increasing number of approaches have been proposed to solve UCC with the help of either attribute descriptions (zero-shot learning (ZSL)) (Kodirov et al. (2017); Zhang et al. (2017)) or one/a few labeled samples for each class (few-shot learning (FSL)) (Snell et al. (2017); Xian et al. (2018)).

Previous approaches to UCC mainly have the following characteristics and limitations: (i) To obtain powerful discriminative feature representation, they often train a deep classification model employing state-of-the-art multi-class classification techniques. However, such models are hard to be adapted to new classes with limited supervision information due to the high volume of model parameters and the gradual updating scheme. (ii) To ensure the consistency of training and test settings and adaptability to new classes, previous methods often train a deep model in an episode fashion (Vinyals et al. (2016)), sometimes along with some specially designed meta-learning updating rules (Finn et al. (2017)). With episode-based training, the model acquires adaptability to new tasks after many training episodes using the knowledge it grasps during the training. However, the episode-based training strategy severely limits the model's capability of extracting discriminative features, because it does not fully exploit the diversity and variance of all classes within the training dataset. The trained model treats the classes in each episode as new classes and attempts to separate them. Therefore, it does not have memory of the competing information of these classes against all the other ones in the whole dataset beyond the current episode. Due to the neglect of this global (dataset-wise rather than episode-wise) discriminative information, the feature extraction capability of the model is suppressed, thus limiting the UCC performance.

To address these issues, we propose to secure both powerful discriminability of feature extraction and strong adaptability of model classification through network reparameterization, *i.e.*, reparametrizing the learnable weights of a network as a function of other variables. We decouple

the feature extraction module and the classification module of a deep classification model, learn the former as a standard multi-class classification task to obtain a discriminative feature extractor, and learn the latter employing a light deep neural network that generates generic classification weights for unseen classes given limited exemplar information. We train the classification weight generator by following the episode-based training scheme to secure the adaptability. Our method can be flexibly applied to both ZSL and FSL, where the exemplar information about unseen classes are provided in the form of either the semantic attributes or one/a few labeled samples. Extensive experiments show that our proposed method achieves state-of-the-art performance on widely-used benchmark datasets for both tasks.

2 Related Work

With regard to the form of the exemplar information provided about unseen classes, UCC can be classified as zero-shot learning and few-shot learning.

2.1 Zero-Shot Learning

ZSL requires recognizing unseen classes based on their semantic descriptions. It is approached by finding an embedding space where visual samples and semantic descriptions of a class are interacted so that the semantic description of an unseen class can be queried by its visual samples. Since the embedding space is often of high dimension, finding the best match of a given vector among many candidates shall inevitably encounter the hubness problem (Radovanović et al. (2010)), *i.e.*, some candidates will be biased to be the best matches for many of the queries. Depending on the chosen embedding space, the severeness of this problem varies. Some approaches select the semantic space as the embedding space and project visual features to the semantic space (Lampert et al. (2014); Frome et al. (2013). Projecting the visual features into a often much lower-dimensional semantic space shrinks the variance of the projected data points and thus aggravates the hubness problem. Alternatively, some methods project both visual and semantic features into a common intermediate space (Akata et al. (2015); Sung et al. (2018); Zhang & Saligrama (2015)). However, due to lacking training samples from unseen classes, these methods are prone to classify test samples into seen classes (Romera-Paredes & Torr (2015)) (for the generalized ZSL setting, seen classes are included when testing). Recently, Zhang et al. (2017) proposed to choose the visual space as the embedding space and learned a mapping from the semantic space to visual space. Benefiting from the abundant data diversity in visual space, this method can mitigate the hubness problem at some extent. However, the limitation of this method is that it strives only to learn a mapping from semantic space to visual space such that the visual samples of a class coincide with the associated semantic description; it however neglects the separation information among visual features of different classes. Our method avoids this problem. We formulate bridging the semantic space and the visual space as a visual feature classification problem conditioned on the semantic features. We learn a deep neural network that generates classification weights for the visual features when fed with the corresponding semantic features. By nature of a classification problem, both intra-class compactness (visual features of the same classes are assigned with the same label) and inter-class separability (visual features of different classes are assigned with different labels) are exploited, hence resulting in a better mapping.

2.2 Few-Shot Learning

FSL aims to recognize unseen classes when provided with one/a few labeled samples of these classes. A number of methods address it from the perspective of deep metric learning by learning deep embedding models that output discriminative feature for any given images (Ren et al. (2018); Vinyals et al. (2016); Snell et al. (2017); Triantafillou et al. (2017); Sung et al. (2018)). The difference lies in the loss functions used. More common approaches are based on meta-learning, also called learning to learn, which is to learn an algorithm (meta-learner) that outputs a model (the learner) that can be applied on a new task when given some information (meta-data) about the new task. Following this line, approaches such as META-LSTM (Ravi & Larochelle (2017)), MAML (Finn et al. (2017)), Meta-SGD (Li et al. (2017)), DEML+Meta-SGD (Li et al. (2017)), Meta-Learn LSTM (Ravi & Larochelle (2017)), Meta-Networks (Munkhdalai & Yu (2017)), and REPTILE (Nichol et al. (2018)) aim to optimize the meta-learned classifiers to be easily fine-tuned on new

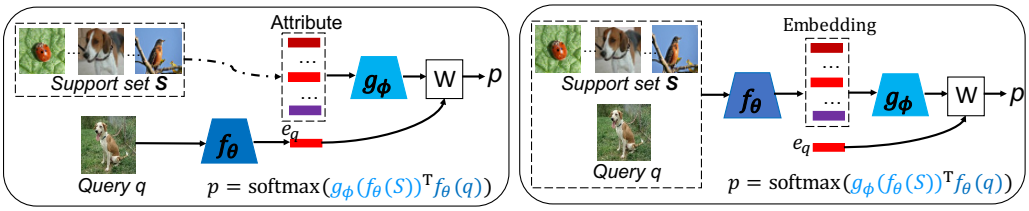

Figure 1: Frameworks for zero-shot (Left) and few-shot (Right) classification.

few-shot tasks using the small-scale support set provided. The common limitation of the above methods is that they adopt the episode-based training scheme to secure adaptability to new classes, which however compromises the capability of discriminative feature extraction due to the forgetting of global (dataset-wise) competing information of among classes beyond individual episodes. Perhaps closest to our approach, Gidaris & Komodakis (2018) proposed the DFSVL algorithm which approaches FSL also in virtue of classification weight generation. The major limitation of DFSVL is that it obtains classification weights for unseen classes simply as a mixture of feature embeddings of support images of novel classes and attended pretrained weights of base (seen) classes, which is too weak to bridge feature embeddings and classification weights. Besides, it cannot bridge information across different domains (due to dimension inconsistency) so that is not applicable for ZSL. We instead learn a network to generate classification weights directly from feature embeddings of support images; it is more powerful and flexible to solve both ZSL and FSL within the same framework.

3 METHOD

We focus on Unseen Class Categorization (UCC), which is to recognize objects of unseen classes given only minimal information (a few labeled samples or the attributes) about the classes. Formally, suppose we have three sets of data $\mathcal{D} = \{\mathcal{D}_t, \mathcal{D}_s, \mathcal{D}_u\}$, where $\mathcal{D}_t = \{\mathcal{X}_t, \mathcal{Y}_t\}$ is the training set and $\mathcal{D}_u = \{\mathcal{X}_u, \mathcal{Y}_u\}$ the test set, with $\mathcal{X}_t$ and $\mathcal{X}_u$ being the images, $\mathcal{Y}_t$ and $\mathcal{Y}_u$ the corresponding labels. There is no overlap between training classes and testing classes, *i.e.*, $\mathcal{Y}_t \cap \mathcal{Y}_t = \emptyset$. The goal of UCC is to learn transferable information from $\mathcal{D}_t$ that can be used to classify unseen classes from $\mathcal{D}_u$, with the help of supporting information from $\mathcal{D}_s$. For ZSL, $\mathcal{D}_s = \mathcal{A}_t \cup \mathcal{A}_u$ is the union of the semantic attribute vectors $\mathcal{A}_t$ for seen classes $\mathcal{Y}_t$ and $\mathcal{A}_u$ for unseen classes $\mathcal{Y}_u$. For FSL, $\mathcal{D}_s$ contains one/a few images and their labels for each class from $\mathcal{D}_u$, *i.e.*, $\mathcal{D}_s = \{\mathcal{X}_s, \mathcal{Y}_s\}$, with $\mathcal{Y}_s \subseteq \mathcal{Y}_u$.

3.1 A NETWORK REPARAMETERIZATION FRAMEWORK FOR UCC

Our main contribution in this paper is the proposed framework that can address both ZSL and FSL with minimal changes. Figure 1 diagrams our framework. Instead of jointly learning the feature extraction network weights and classification weights, which results in a heavy model that is hard to be adjusted for novel classes with limited supervision information, we reparametrize the learnable weights of a classification model as the combination of learnable parameters of a feature extraction model and a weight generation model. In other words, we decouple the feature extraction network f_θ and the classification weight $\mathbf{W}$ of a standard classification network. We train f_θ as a standard multi-class classification task and learn another network g_ϕ to generate the classification weight $\mathbf{W}$. Since f_θ is trained as a standard multi-class classification task to distinguish all classes within the training set, it is supposed to be able to generate more discriminative feature representations for images of unseen classes than that generated by a model trained in episode-based fashion where the model is train to distinguish several classes within mini-batches. Meanwhile, we train g_ϕ in episode-based fashion by constantly sampling new classes and minimizing the classification loss (cross entropy loss on top of Softmax outputs) using the weights generated by g_ϕ. After training, whenever some new classes come, along with supporting information in the form of either attribute vectors (ZLS) or a few-labeled samples (FSL), g_ϕ is supposed to be able to generate generic classification weights that can effectively classify query images that belong to these new classes. Thanks to this network reparameterization strategy, we are able to get a powerful and flexible UCC model.

We adopt the cosine similarity based cross entropy loss to train the weight generator g_ϕ. Traditional multi-layer neural networks use dot product between the output vector of previous layer and the incoming weight vector as the input to activation function. Luo et al. (2017) recently showed that

replacing the dot product with cosine similarity can bound and reduce the variance of the neurons and thus result in models of better generalization. Gidaris & Komodakis (2018) further showed that using the cosine similarity instead of dot product for calculating classification score in the last fully-connected layer of deep neural network brings benefit for classification, with some minor revisions. We adopt this technique to train our weight generator g_ϕ. The classification score of a sample ($\mathbf{e}_x$, y) is calculated as

$$p(y=n|\mathbf{e}_x) = \frac{\exp(s\cos(\mathbf{w}_y, \mathbf{e}_x))}{\sum_{j=1}^{N}\exp(s\cos(\mathbf{w}_j, \mathbf{e}_x))}, \quad (1)$$

$$\mathbf{e}_x = f_\theta(\mathbf{x}), \quad (2)$$

$$\mathbf{w}_j = g_\phi(\mathbf{a}_j), \text{ for ZSL}, \quad (3)$$

$$\mathbf{w}_j = g_\phi(\frac{1}{N_f}\sum_{i=1}^{N_f} f_\theta(\mathbf{x}_{i,j})), \text{ for FSL}, \quad (4)$$

where s is a learnable scalar controlling the peakiness of the probability distribution generated by the softmax operator (Gidaris & Komodakis (2018)), $\mathbf{w}_j$ is the classification weight for class j generated by neural network g_ϕ taking supporting information of the class as input, $\mathbf{x}$ is the input image, $\mathbf{a}_j$ is the attribute vector for class j for ZSL, $\mathbf{x}_{i,j}$ is the i-th input image of class j for FSL, $j = 1, ..., N_f$, and N_f is the number of shots for FSL.

In a typical UCC task $\mathscr{T}$, the loss function is calculated as

$$\mathcal{L}(\theta, \phi) = \sum_{(\mathbf{x},y)\in\mathscr{T}} \big[-s\cos(\mathbf{w}_y, f_\theta(\mathbf{x})) + \log(\sum_{j=1}^{N}\exp(s\cos(\mathbf{w}_j, f_\theta(\mathbf{x}))))\big] + \lambda\|\phi\|_2, \quad (5)$$

where λ is a hyper-parameter weighting the l_2-norm regularization of the learnable parameters of neural network g_ϕ.

3.1.1 Zero-Shot Learning

For ZSL, we are provided with semantic class attributes $\mathcal{S} = \mathcal{A}_t \cup \mathcal{A}_u$ as the assistance for UCC. The basic assumption for existing ZSL algorithms is that the visual-attribute relationship learned from seen classes in a certain embedding space is class-invariant and can be applied to unseen classes. With this assumption, existing methods either project visual features to semantic space or reversely project semantic features to visual space, or alternatively project both visual and semantic features to an intermediate space. In any case, the coincidence of visual and semantic features of a class is utilized to learn the visual-attribute relationship. Zhang et al. (2017) recently showed that it is advantageous to select the visual space as the embedding space because the abundance of data diversity in the visual space can significantly mitigate the so-called "hubness" problem. Their objective function is as follows:

$$\mathcal{L}(\theta, \phi) = \frac{1}{N}\sum_{i=1}^{N}\|f_\theta(\mathbf{x}_i) - h_\psi(\mathbf{a}_{y_i})\|_2 + \lambda\|\psi\|_2, \quad (6)$$

where f_θ is a feature extraction model which outputs a representation vector $f_\theta(\mathbf{x}_i)$ using image $\mathbf{x}_i$ as input. h_ψ is a mapping function which projects attribute vector $\mathbf{a}_{y_i}$ of class y_i to the embedding space where $f_\theta(\mathbf{x}_i)$ lies. Through minimizing the least square embedding loss, the visual-attribute relationship can be established. With this relationship, in the testing stage, the attributes $\mathcal{A}_u$ of unseen classes are mapped to the visual feature embedding space in which the visual feature of an images of any unseen class can find the best class attribute through nearest neighbor searching.

One can observe that this method learns the visual-attribute relationship by only utilizing the coincidence of the visual samples of a class with the associated semantic description. It however neglects to explore the inter-class separation of different classes, which shall be crucial to further avoid the hubness problem. To remedy this, we reformulate the learning of visual-attribute relationship from a regression problem to a visual feature classification problem. We directly learn a network g_ϕ that outputs the classification weights for classifying visual features and use the cross-entropy loss on top of Softmax outputs to guide learning g_ϕ. Through this reformulation, both intra-class compactness and inter-class separability are elegantly exploited for learning the visual-attribute relationship:

Algorithm 1. Network reparam. for ZSL

Input: Training set $\mathcal{D}_t = \{\mathcal{X}_t, \mathcal{Y}_t\}$, attribute set of training classes $\mathcal{A}_t$.

Output: Feature extraction network f_θ and classification weight generation network g_ϕ.

1. Train f_θ as a standard multi-class classification task

while not done **do**

2. Randomly sample from $\mathcal{D}_t$ and $\mathcal{A}_t$ a ZSL task $\mathcal{T} = \{\mathcal{B}_v, \mathcal{B}_a\}$, where $\mathcal{B}_v = \{\{\mathbf{x}_{i,j}\}_{i=1}^{N_z}, y_j\}_{j=1}^{M_z}$ and $\mathcal{B}_a = \{\mathbf{a}_j\}_{j=1}^{M_z}$
3. Calculate loss according to Eq. 5
4. Update g_ϕ through back-propagation.

end while

Algorithm 2. Network reparam. for FSL

Input: Training set $\mathcal{D}_t = \{\mathcal{X}_t, \mathcal{Y}_t\}$

Output: Feature extraction network f_θ and classification weight generation network g_ϕ.

1. Train f_θ as a standard multi-class classification task

while not done **do**

2. Randomly sample from $\mathcal{D}_t$ a FSL task $\mathcal{T} = \{\mathcal{B}_s, \mathcal{B}_q\}$, where $\mathcal{B}_s = \{\{\mathbf{x}_{i,j}\}_{i=1}^{N_f}, y_j\}_{j=1}^{M_f}$ and $\mathcal{B}_q = \{\{\mathbf{x}_{k,j}\}_{k=1}^{Q_f}, y_j\}_{j=1}^{M_f}$
3. Calculate loss according to Eq. 5
4. Update g_ϕ through back-propagation.

end while

Visual features of the same classes should be assigned with the same label (compactness), while visual features of different classes are assigned with different labels (separability).

We follow the network reparameterization scheme by decoupling the feature extraction module f_θ and the classification weight module which is generated by g_ϕ. The feature extraction module f_θ is trained as a standard multi-class classification task to enable us to obtain a discriminative feature representation for any given image. To learn g_ϕ, we adopt the episode based training scheme by continuously exposing g_ϕ with new (randomly sampled) ZSL tasks so as to secure good performance when new real tasks arrive in the testing stage. More specifically, we keep randomly sampling from $\mathcal{D}_t = \{\mathcal{X}_t, \mathcal{Y}_t\}$ and $\mathcal{A}_t$ ZSL tasks and feeding them to the network. Each task consists of M_z classes and the associated M_z attribute vectors. For each class, we randomly sample N_z images. With a batch of $M_z N_z$ images $\mathcal{B}_v$ and M_z attribute vectors $\mathcal{B}_a$, we train g_ϕ by minimizing the loss function defined in Eq. 5. In the testing stage, given attributes of unseen classes $\mathcal{A}_u$, or $\mathcal{S} = \mathcal{A}_t \cup \mathcal{A}_u$ for all (seen and unseen) classes as in generalized ZSL setting, we generate the corresponding classification weights using g_ϕ. The generated classification weights, integrated with the feature extraction network f_θ serve to classify images of unseen classes. **Algorithm 1** outlines the main steps of our method for ZSL.

3.1.2 FEW-SHOT LEARNING

For FSL, one/a few labeled samples $\mathcal{D}_s = \{\mathcal{X}_s, \mathcal{Y}_s\}$ for each unseen class are provided to help recognize objects of these classes. Our novel categorization framework can be easily extended from ZSL to FSL, simply by replacing the semantic attribute vectors with feature embedding vectors as the input to the classification weight generation network g_ϕ. To train g_ϕ, we keep randomly sampling FSL tasks from $\mathcal{D}_t = \{\mathcal{X}_t, \mathcal{Y}_t\}$, each of which consists of a support set and a query set. Images in the both sets are from the same classes. The support set consists of M_f classes and N_f images for each class. With the feature embeddings $\mathcal{B}_e$ of the $M_f N_f$ images as input, g_ϕ generates the classification weights for the M_f classes, which are then used to classify the feature embeddings of images from the query set. Note that if $N_f > 1$, *i.e.*, each class has multiple support samples, we average the embeddings of all images belonging to the same class and feed the averaged embedding to g_ϕ. Similar to ZSL, we learn the resulting model by optimizing the loss function defined in Eq. 5. **Algorithm 2** outlines the main steps for FSL.

One of the most distinct aspects of our method from the existing ones is that we decouple the feature extraction module and the classifier module of the deep classification model, and train each module on the most beneficial tasks. We train the feature extraction module as a standard multi-class classification task. This is motivated by the observation that a simple classifier (*e.g.*, nearest neighbor), when taking as input features obtained by a powerful extractor, can outperform some sophisticated FSL models that use weaker feature extraction models. For example, as shown in Fiugre 2, using nearest neighbor (NN) as the classifier, we can achieve better one-shot classification accuracy than a recent FSL algorithm PROTO NET (Snell et al. (2017)), when using features extracted by ResNet18 (He et al. (2016)) trained on the *Mini-Imagenet* or *CUB* datasets. Moreover, if we train ResNet18 on *ImageNet* where massive data are available for obtaining a even more powerful feature extractor ResNet18*, we can achieve

		AwA1	AwA2	CUB	aPY	SUN
#Class	#Seen	40	40	150	20	645
	#Unseen	10	10	50	12	72
# VisDim		2048	2048	2048	2048	2048
# AttDim		85	85	312	64	102

Table 1: Information of zero-shot classification datasets.

much higher accuracy. (Note that for experiments on *Mini-Imagenet*, we train ResNet18* not on the whole *ImageNet* but on the subset excluding the classes appearing on *Mini-Imagenet*.) The reason for this surprising result is that the episode-based training scheme of existing FSL methods inherently suppresses obtaining a powerful feature extractor: In each episode, the model is fed with a new FSL task that is assumed to have no relationship with the previous ones. The model is trained to separate well the several classes within the task. However, since all training tasks are sampled from the training dataset, one class shall appear in many tasks. The inter-class separation across the whole dataset is neglected by existing FSL methods. Therefore, there is a dilemma for existing FSL algorithms: They need to be trained in an episode-based fashion to ensure flexibility, but which in return compromises feature discriminability. To avoid this awkward situation, our proposed method decoupling the network and training different components in different ways ensures powerful discriminability and strong adaptability.

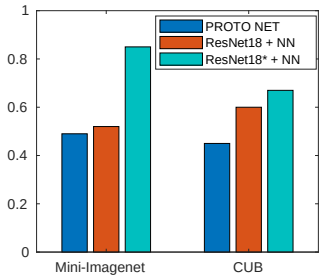

Figure 2: One-shot classification accuracy on two datasets.

4 EXPERIMENTS

We evaluate our framework for both zero-shot learning and few-shot learning tasks.

4.1 ZERO-SHOT LEARNING

Datasets and evaluation settings. We employ the most widely-used zero-shot classification datasets for performance evaluation, namely, AwA1 (Lampert et al. (2014)), AwA2 (Xian et al. (2018)), CUB (Wah et al. (2011)), SUN (Patterson & Hays (2012)) and aPY (Farhadi et al. (2009)). The statistics of the datasets are shown in Table 1. We follow the GBU setting proposed in (Xian et al. (2018)) and evaluate both the conventional ZSL setting and the generalized ZSL (GZSL) setting. In the conventional ZSL, test samples are restricted to the unseen classes, while in the GZSL, they may come from either seen classes or unseen classes.

Implementation details. Following (Xian et al. (2018)), we adopt ResNet101 as our feature extraction model f_θ which results in a 2048-dimension vector for each input image. For the weight generation model g_ϕ, we utilize two FC+ReLU layers to map semantic vectors to visual classification weights. The dimension of the intermediate hidden layer are 1600 for all the five datasets. We train g_ϕ with Adam optimizer and a learning rate 10^{-5} for all datasets by 1,000,000 randomly sample ZSL tasks. Each task consists of 32 randomly sampled classes, 4 samples for each class, *i.e.*, $M_z = 32$ and $N_z = 4$. The hyper-parameters λ is chosen as 10^{-4}, 10^{-3}, 10^{-3}, 10^{-5} and 10^{-4} for AwA1, AwA2, CUB, SUN and aPY, respectively. Our model is implemented with PyTorch.

Experimental results. Table 2 shows the experimental results. For the conventional ZSL setting, our method reaches the best for three out of the five datasets, while being very close to the best for one of the left two. Remarkably, our method consistently outperforms DEM (Zhang et al. (2017)) for all the five datasets, which substantiates the benefit of our method of taking consideration of inter-class separability when learning the mapping from semantic space to visual space. For GZSL setting where seen classes are also included to be the candidates, our method significantly outperforms all competing methods, reaching performance gains over the second best even about 30% in the *AWA1* dataset. We analyze the reason for our dramatic advantage is that our method considers inter-class separation during the training stage so that the resultant classification weights for the seen classes possess good separation property after training. When they are concatenated with the classification weights generated from semantic descriptions of unseen classes in the testing stage, they shall be

	AwA1		*AwA2*		*CUB*		*aPY*		*SUN*	
	ZSL	GZSL	ZSL	GZSL	ZSL	GZSL	ZSL	GZSL	ZSL	GZSL
DAP (Lampert et al. (2014))	44.1	0.0	46.1	0.0	40.0	1.7	33.8	4.8	39.9	4.2
CONSE (Norouzi et al. (2014))	45.6	0.4	44.5	0.5	34.3	1.6	26.9	0.0	38.8	6.8
SSE (Zhang & Saligrama (2015))	60.1	7.0	61.0	8.1	43.9	8.5	34.0	0.2	51.5	2.1
DEVISE (Frome et al. (2013))	54.2	13.4	59.7	17.1	52.0	23.8	**39.8**	4.9	56.5	16.9
SJE (Akata et al. (2015))	65.6	11.3	61.9	8.0	53.9	23.5	32.9	3.7	53.7	14.7
LATEM (Xian et al. (2016))	55.1	7.3	55.8	11.5	49.3	15.2	35.2	0.1	55.3	14.7
ESZSL (Romera-Paredes & Torr (2015))	58.2	6.6	58.6	5.9	53.9	12.6	38.3	2.4	54.5	11
ALE (Akata et al. (2015))	59.9	16.8	62.5	14.0	54.9	23.7	39.7	4.6	58.1	21.8
SYNC (Changpinyo et al. (2016))	54.0	8.9	46.6	10.0	55.6	11.5	23.9	7.4	56.3	7.9
SAE (Kodirov et al. (2017))	53.0	1.8	54.1	1.1	33.3	7.8	8.3	0.4	40.3	8.8
DEM (Zhang et al. (2017))	68.4	32.8	67.1	30.5	51.7	19.6	35.0	11.1	61.9	20.5
RELATION NET (Sung et al. (2018))	68.2	31.4	64.2	30.0	**55.6**	38.1	N/A	N/A	N/A	N/A
Ours	**70.3**	**62.6**	**69.7**	**54.8**	53.6	**45.4**	39.1	**28.4**	**62.5**	**36.8**

Table 2: Zero-shot classification accuracy. The best results are in **bold**.

	Mini-Imagenet		*CUB*	
	1-shot	5-shot	1-shot	5-shot
ResNet18 feat. + NN	52.06±0.78%	65.22±0.71%	60.80±0.86%	78.18±0.56%
ResNet18 feat. + PROTO NET classifier	55.80±0.82%	75.74±0.65%	65.06±0.89%	83.19±0.58%
ResNet18 + PROTO NET	51.72±0.43%	67.45±0.32%	42.98±0.47%	55.94±0.40%
Matching NET (Vinyals et al. (2016))	43.56±0.84%	55.31±0.73%	49.34	59.31
PROTO NET (Snell et al. (2017))	49.42±0.78%	68.20±0.66%	45.27	56.35
MAML (Finn et al. (2017))	48.70±1.84%	63.11±0.92%	38.43	59.15
META-SGD (Li et al. (2017))	50.47±1.87%	64.03±0.94%	N/A	N/A
META-LSTM (Ravi & Larochelle (2017))	43.44±0.77%	60.60±0.71%	40.43	49.65
MACO (Hilliard et al. (2018))	41.09±0.32%	58.32±0.21%	60.76	74.96
DFSVL (Gidaris & Komodakis (2018))	55.95±0.84%	73.00±0.64%	N/A	N/A
RELATION NET (Sung et al. (2018))	57.02±0.92%	71.07±0.69%	N/A	N/A
Ours	**62.95**±1.01%	**79.17**±0.81%	**70.46**±1.09%	**85.21**±0.71%

Table 3: Few-shot classification accuracy. The best results are in **bold**.

quite discriminative to discern that the incoming images do not belong to their classes. From the perspective of hubness problem, since the classification weights for seen class have good separation property, the weight vectors are less likely to be clustered in the embedding space, so that the risk is reduced that some candidates are selected as the nearest neighbors for many query images.

4.2 Few-Shot Learning

Datasets and evaluation settings. We evaluate few-shot classification on two widely-used datasets, *Mini-ImageNet* (Vinyals et al. (2016)) and *CUB* (Wah et al. (2011)). The *Mini-ImageNet* dataset has 60,000 images from 100 classes, 600 images for each class. We follow previous methods and use the splits in Ravi & Larochelle (2017) for evaluation, *i.e.*, 64, 16, 20 classes as training, validation, and testing sets, respectively. The *CUB* dataset is a fine-grained dataset of totally 11,788 images from 200 categories of birds. As the split in Ravi & Larochelle (2017), we use 100, 50, 50 classes for training, validation, and testing, respectively. For both datasets, we resize images to 224×224 to meet the requirement of our adopted feature extraction network. Following the previous methods, we evaluate both 5-way 1-shot and 5-way 5-shot classification tasks where each task instance involves classifying test images from 5 sampled classes with 1 (1-shot) or 5 (5-shot) randomly sampled images for each class as the support set. In order to reduce variance we repeat the evaluation task 600 times and report the mean of the accuracy with a 95% confidence interval.

Implementation details. We use ResNet18 as our feature extraction model f_θ which results in a 512-dimension vector for each input image after average pooling. We train f_θ on the two experimental datasets by following the standard classification learning pipeline: We use Adam optimizer with an initial learning rate 10^{-3} which decays to the half every 10 epochs. The model is trained with 100 epochs. As for g_ϕ, we use two FC+ReLU layers, same as in ZSL. The dimension of the intermediate hidden layer is 512 for both datasets. We train g_ϕ using Adam optimizer with a learning rate 10^{-5} and set the hyper-parameters $\lambda = 10^{-5}$ for both datasets. The model is trained with 60000 randomly sampled FSL tasks, each of which consist of 5 classes, with 1 or 5 samples as the support samples and another 15 as the query samples.

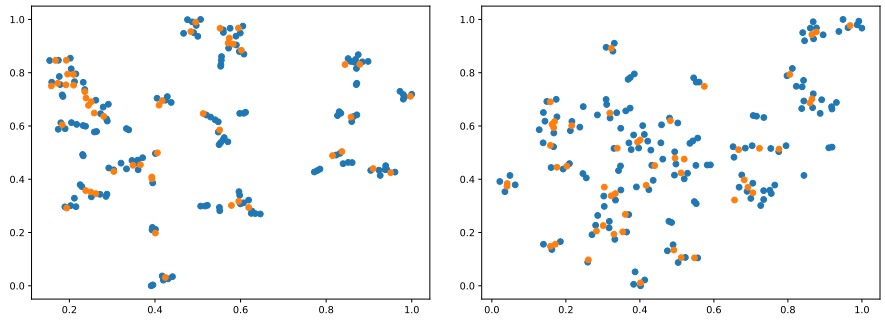

Figure 3: The Barnes-Hut t-SNE (Van Der Maaten (2014)) of the projections of semantic vectors in the visual space obtained by DEM (Zhang et al. (2017)) (Left) and our method (Right). Seen and unseen classes are in orange and blue, respectively.

Experimental results. Table 3 shows the results of the proposed method and the most recent ones. From the table, we can get some interesting observations. First, the baseline method "ResNet18 + NN" beats most competing FSL algorithms where various sophisticated strategies are used. Meanwhile, the accuracy of feeding the classifier of PROTO NET with features obtained by ResNet18 ("ResNet18 feat. + PROTO NET classifier") is much higher than that obtained by training PROTO NET end to end with ResNet18 as the base model ("ResNet18 + PROTO NET"). These results support our analysis that the episode-based training scheme adopted by existing FSL approaches suppresses the discriminability of the feature extraction model. Second, compared with the baseline methods "ResNet18 feat. + NN" and "ResNet18 feat. + PROTO NET classifier", which use the same feature representations as our method, we get obvious improvements. This substantiates the benefit of the proposed weight generation strategy for FSL. Third, compared with the existing methods, our method reaches the best in the both datasets for both 1-shot and 5-shot evaluation settings, often by large margins. This shows the great advantage of our method for handling the FSL problem.

4.3 Further Analysis

As we can see above, our method dramatically outperforms existing methods for the GZSL setting. The advantage is much more significant than that for the ZSL setting. We have analyzed the reason is that the classification weights generated from the attributes of seen classes show good separation property so that the hubness problem is not as severe as that for other methods. The hubness problem refers that in ZSL, some candidate points are prone to be the nearest neighbors of many query points when the dimension is high. So, if the candidate points are more evenly distributed in the space, the less severe of the hubness problem should be. To validate this, we use t-SNE (Van Der Maaten (2014)) to visualize the classification weight vectors generated from all 200 class semantic vectors in the CUB dataset. As a comparison, we do the same thing for DEM (Zhang et al. (2017)) which also learns mapping from semantic space to visual space. The result is shown in Figure 3. We can observe that the points are more evenly distributed for our method than that for DEM. This further validates the benefit of our method in avoiding the hubness problem.

5 Conclusions

In this paper, we propose a flexible framework for unseen class categorization with limited information provided about these classes. We secure two key factors, a powerful feature extractor and a flexible classifier, through network reparameterization. We decouple the feature extraction module and the classification module of a deep model for UCC. The feature extraction module is learned in a standard multi-class classification framework and the classification weight vector is generated by a network from exemplar information of the unseen classes. We train the classification weight generator in an episode-by-episode fashion to enable it flexibility for new tasks. Applying our framework for zero-shot learning (ZSL), we achieve much better results especially for the generalized ZSL setting than the state-of-the-art owing to our incorporation of inter-class separation information for learning the mapping from semantic space to visual space. For few-shot learning (FSL), we also achieve remarkable performance gains relative to existing methods due to the flexible scheme that make it possible a powerful feature extraction model and a flexible weight generation model.

REFERENCES

Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. Synthesized classifiers for zero-shot learning. In *CVPR*, 2016.

Ali Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *CVPR*, 2009.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.

Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Nathan Hilliard, Lawrence Phillips, Scott Howland, Artëm Yankov, Courtney D Corley, and Nathan O Hodas. Few-shot learning with metric-agnostic conditional embeddings. *arXiv preprint arXiv:1802.04376*, 2018.

Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. In *CVPR*, 2017.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2014.

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

Chunjie Luo, Jianfeng Zhan, Lei Wang, and Qiang Yang. Cosine normalization: Using cosine similarity instead of dot product in neural networks. *arXiv preprint arXiv:1702.05870*, 2017.

Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *ICML*, 2017.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *CoRR, abs/1803.02999*, 2018.

Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Greg S Corrado, and Jeffrey Dean. Zero-shot learning by convex combination of semantic embeddings. *ICLR*, 2014.

Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012.

Miloš Radovanović, Alexandros Nanopoulos, and Mirjana Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(Sep):2487–2531, 2010.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

Bernardino Romera-Paredes and Philip Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.

Eleni Triantafillou, Richard Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens. In *NIPS*, 2017.

Laurens Van Der Maaten. Accelerating t-sne using tree-based algorithms. *Journal of Machine Learning Research*, 15(1):3221–3245, 2014.

Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NIPS*, 2016.

Catherine Wah, Steve Branson, Pietro Perona, and Serge Belongie. Multiclass recognition and part localization with humans in the loop. In *ICCV*, 2011.

Yongqin Xian, Zeynep Akata, Gaurav Sharma, Quynh Nguyen, Matthias Hein, and Bernt Schiele. Latent embeddings for zero-shot classification. In *CVPR*, 2016.

Yongqin Xian, Christoph H Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

Li Zhang, Tao Xiang, and Shaogang Gong. Learning a deep embedding model for zero-shot learning. In *CVPR*, 2017.

Ziming Zhang and Venkatesh Saligrama. Zero-shot learning via semantic similarity embedding. In *ICCV*, 2015.