Boosting Transferable Meta-learning via Unsupervised Visual Representation

Fengqi Liu
020033910040

Qizhi Li
120033910139

Mingjie Li
517030910344

Abstract

Deep neural network have been widely used in image classification, but it requires a lot of labeled data to train to obtain a higher accuracy rate. However, extensive data acquisition and manual label annotation are expensive. Therefore, few-shot classification is of crucial significance which aims to recognize novel categories with only few labeled data. Many existing few-shot classification algorithms predict categories by comparing the feature embeddings of query images which are not good enough with those from a few labeled images (support examples) or using full connected layer to classify. In this paper, we propose a learning method to obtain better features by fine-tuning Deep neural network using transferable meta-learning. In addition, we exploit the complementarity of few-shot learning and self-supervision learning and use self-supervision as an auxiliary task in a few-shot learning pipeline, enabling feature extractors to learn richer and more transferable visual representations while still using few annotated samples. We conduct extensive experiments using two challenging few-shot learning benchmarks: Mini-Imagenet and Fewshot-CIFAR100. Experimental results have proved the effectiveness of our method and perfect performance on benchmarks.

1 Introduction

Deep learning has achieved great success in various tasks, especially in computer vision tasks (He et al., 2016; Szegedy et al., 2015). However, due to the fact that deep neural networks usually have a large number of trainable parameters (Krizhevsky et al., 2012; Simonyan & Zisserman, 2014), people usually needs to collect large amounts of labeled data before hand, which entails considerable human labour. Till now, deep learning is mainly based on fully-supervised and big-data regime. This may cause a problem when there is a limited amount of carefully labeled data. Under such circumstance, many algorithms emerged to tackle low supervision cases, *e.g.* semi-supervised learning (Berthelot et al., 2019), unsupervised learning (Caron et al., 2018), few-shot learning (Snell et al., 2017), or even one-shot learning (Vinyals et al., 2016), *etc*. Among them, few-shot learning (FSL) has attracted many researchers in recent years.

In few-shot learning, the major aim is to "fully" exploit the information in the scarce training data, and learn a powerful learning scheme (*e.g.* a good initialization, a reasonable decision boundary). State-of-the-art works have made lots of attempts into the field of FSL. We summarize these methods as: (i) embedding learning to learn better representations; (ii) data augmentation or (iii) introducing prior knowledge to "enlarge" the dataset; (iv) meta-learning to learn better hyper-parameters for learning; (v) multi-task learning to extract richer information in the small amount of training data. However, none of the previous work considered to combine two or more of these methods.

Generally speaking, few-shot learning is related to self-supervised representation learning, which is a form of unsupervised visual learning that trains a model via non-labeled pretext task. Similar to few-shot learning, self-supervised learning aims to learn rich and generic representations of images that can be transferred to other downstream visual understanding tasks. Inspired by meta-transfer

Statistical Learning and Inference (CS7335, 2020 Fall) Final Project, Instructor: Li Niu

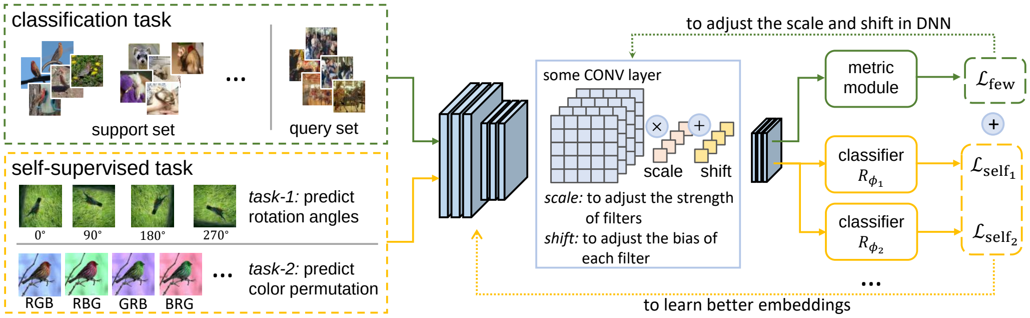

classification task
support set
query set
self-supervised task
0°
90°
180°
270°
task-1: predict rotation angles
RGB
RBG
GRB
BRG
task-2: predict color permutation
to adjust the scale and shift in DNN
some CONV layer
scale
shift
scale: to adjust the strength of filters
shift: to adjust the bias of each filter
metric module
classifier R_{ϕ_1}
classifier R_{ϕ_2}
$\mathcal{L}_{\text{few}}$
$\mathcal{L}_{\text{self}_1}$
$\mathcal{L}_{\text{self}_2}$
to learn better embeddings

Figure 1: Overview of the proposed FSL framework. Inspired by meta-transfer learning and multi-task learning, we combine meta-learning and self-supervised learning to further improve the performance for few-shot classification. Specifically, meta-leraning aims to finetune the scale and shift in convolutional layers, while self-supervised learning aims at providing more knowledge-rich representations for classification.

learning and multi-task learning, we combine pre-train process and self-supervised tasks to further improve the performance for few-shot classification. In the pre-training part, we trained with based class without few shot setting to initialize the parameters of the feature filters. To avoid over-fitting with the few shot setting after the phase, we constrain the number of trainable parameters with the similar structure of Meta-transfer learning. In the meta-learning part, we introduce self-supervised learning to help the feature extraction of the meta-training. Not only the original meta-learning training method is retrained and the generalized ability trained by the process is guaranteed, but the more features can be extracted from the auxiliary tasks.

In summary, the contribution of our work are: a) Extend the form of transfer from two relative simple operations: Scaling and Shifting to the extensive feature transformation methods. Convolutional neural network has been trained with large scale data and have a great initialization. Control the number of trainable parameters, only allow the transfer operation : scale, shift and other transfer operation to modify the model. b) In the meta training part, the self-supervised mechanism is applied to train the support set and fine-tune the transfer parameters in the model. To be precise, self-supervised mechanism are used for data augmentation to assist in training the model features transfer.

2 Related Work

In this section, we first provide a formal definition of the few-shot learning (FSL) problem, with notations provided. Then, we summarize recent literature aiming to or useful to tackle the FSL problem.

Few-shot learning. Few-shot learning refers to understanding new concepts from only a few examples (Ravi & Larochelle, 2016; Wang et al., 2020). In FSL, there is a terminology: 'N-way K-shot' classification problem. In an 'N-way K-shot' problem, there are N classes, with each containing K samples in the training data. In other words, there are only $N \times K$ samples are known. The main part in solving FSL problems is to overcome the over-fitting problem, due to the severely small number of training samples. This will lead to performance collapse on testing data. To tackle the FSL problems, people usually form several FSL tasks (*training tasks*) from the base data to help the model obtain some prior knowledge (*e.g.* a good initialization, a reasonable optimization method, *etc.*). The training and testing sets in these tasks are called *support sets* and *query sets*. To evaluate the performance, another bunch of tasks are formed, which are termed *testing tasks*. There have been many methods to tackle the FSL problems, as follows.

• **FSL via embedding learning.** Embedding learning embeds each sample $x_i \in \mathbb{R}^N$ to a low dimensional representation space, *i.e.* $z_i = f(x_i) \in \mathbb{R}^D$. The aim is to let similar samples (in the same class) to be closer in the embedding space, in order to differentiate them with dissimilar samples (in different classes). The matching network proposed by Vinyals et al. (2016) was the first attempt on this. Based on a carefully designed LSTM architecture, the samples are first projected onto the embedding space and then match the embedding of the unseen test sample via a cosine similarity function. The relation network (Sung et al., 2018) extended the matching network by introducing a relation module. Another classical algorithm is the prototyical network proposed in (Snell et al., 2017), which define the prototype of each class as the average embedding of samples in this class. Attempts on designing distance metrics include DeepEMD (Zhang et al., 2020), which used the EMD distance to determine image relevance; DSN (Simon et al., 2020) calculated a subspace for each class and then measured similarity.

• **FSL via data augmentation.** The most direct idea is to add synthesized new data to these data-limited classes, so that it will fit the big-data deep learning scheme. As a first work, Hariharan & Girshick (2017) provided techniques to hallucinate additional training examples for data-starved classes. New samples are generated by adding the learned variations to x_i. Similarly, Δ-encoder (Schwartz et al., 2018) learned to both extract transferable intra-class deformations, or "deltas", between same-class pairs of training examples, and to apply those deltas to the few provided examples of a novel class (unseen during training) in order to efficiently synthesize samples from that new class. FATTEN (Liu et al., 2018) used a set of attribute strength regressors learned from a large set of images, to generate new samples.

• **FSL via multi-task learning.** Due to the scarcity of training samples in the settings of FSL, the major aim is to “fully” exploit the data. One of the choice is to design auxiliary tasks to learn. For example, SSF-CNN (Keshari et al., 2018) used the dictionary learning method to initialize the parameters in the model, which contains rich information about the training data. Self-supervision tasks are also considered during training, *e.g.* another task aiming to predict the rotation angle of the image, as is proposed by Gidaris et al. (2019).

• **FSL via meta-learning.** As the first work of meta-learning, MAML (Finn et al., 2017) proposed a new mechanism called meta-learner for fast fine tune for the new task. Meta-learner could use the error of on the query set with gradient descent. With the guidance of meta-learner the base-learner can fit the style of learning new tasks faster. ANIL (Raghu et al., 2019) prove the fine tuning of the fully connected part in the last layer could achieve the equivalent performance of the original MAML model. Due to the limitation of the amount of data, under the MAML framework, only smaller-scale model can be effectively trained. In response to this drawback, Meta-transfer learning (Sun et al., 2019) has been proposed for model pre-training. It constrain the trainable weight as Strength and Shift with similar structure as SSF-CNN (Keshari et al., 2018) to reduce the number of parameters that can be trained in the convolutional layers. Similarly, the transformation layer in (Tseng et al., 2020) with same structure get great efforts. Apart from build model for K shot setting, cnaps (Requeima et al., 2019) built a conditional neural processes and add task-specific parameter to adapt to task in setting more than K shot. Simple-cnaps (Bateni et al., 2020) improve the method with simple class-covariance-based distance metric, namely the Mahalanobis distance and get a significant performance with fewer trainable parameters.

• **FSL via introducing prior knowledge.** Since the settings in FSL restricted the amount of information while learning, there were also people aiming to maximize the useful information encoded, *e.g.* introducing prior knowledge during learning. The first attempt is in natural language processing (NLP). Tsai & Salakhutdinov (2017) extracts the aggregation weight from an auxiliary text corpus. Recently, in visual FSL tasks, adaptive margin loss was proposed by (Li et al., 2020) that leveraged the word embedding of each class to formulate a more reasonable margin between the embedding spaces of classes.

3 Methodology

Meta-learning is composed of two phases: meta-train and meta-test. During the phase of meta-train, we pre-trained the feature extractor and classifier with self-supervised learning and then retrain the meta learner and classifier in meta learning stage. During the next phase of meta-test, we fine-tune the classifier using novel classes dataset.

As explained above, few-shot learning paradigms have two learning stages and two corresponding sets of classes: base classes and novel classes. Here, we define as $D_b = \{(x, y) \mid x \in I_b, y \in Y_b\}$ the dataset of base classes used in the meta-train phase, where $x \in I_b$ is an image with label y in label set Y_b of size N_b. Similarly, $D_n = \{(x, y) \mid x \in I_n, y \in Y_n\}$ is defined as the meta-test dataset of size N_n. One talks about N_n way K shot learning, where each class in meta-test dataset has K samples.

In this section, first, we describe the DNN structure compatible with few-shot setting after pre-training. Second, we consider self-supervised task as an auxiliary loss to help learn the visual representation. Last part, we explained in detail how to use mata-loss and auxiliary task loss to update in the meta-learning phase.

3.1 DNN structure compatible with pre-training.

In few shot setting, there are only support set with few data providing the information of new class. The mainstream deep neural network always have many parameters that need to be trained. If We directly use the pre-trained model to fine tuning on the support set, it may hardly modify parameters in it to learn some new information, but mislead some original features extractor. As Meta-Transfer Learning does, we need to simplify the structure or control the number of trainable parameters in the fine tuning, to reduce the complexity of the model.

In this part, we use the structure of SSF-CNN (Structure and Strength Filtered CNN) to reduce the parameter complexity of our model. In SSF-CNN model, it abandoned the fine tune the feature filter extraction, since the basic feature extract from the image should be similar. More than just fine tune the fully connected network, it separate the parameters in extractor into 2 kinds: 1) the Structure of the filter W and 2) the strength of the filters$t$. The strength could be interpreted as some parameter to modify the filters for the few shot task. In the pre-train stage, W and t should be initialized with the big data set. In the fine tune processing, we only train the Strength of filter t.

3.2 Auxiliary task of self-supervised learning

A major challenge in few-shot learning is encountered during the first stage of learning. How to make the feature extractor learn image features that can be readily exploited for novel classes with few training data during the second stage? With this goal in mind, we propose to leverage the recent progress in self-supervised feature learning to further improve current few-shot learning approaches.

Through solving a non-trivial pretext task that can be trivially supervised, such as recovering the colors of images from their intensities, a network is encouraged to learn rich and generic image features that are transferable to other downstream tasks such as image classification. In the first learning stage, we propose to extend the training of the feature extractor by including such a self-supervised task besides the main task of recognizing base classes.

3.2.1 Self-supervised pretext task

In recent study, predictive task has been shown to achieve state-of-art performance within existing unsupervised visual learning works. To learn more diverse feature in meta-learning, three auxiliary tasks are involved where use the nature of picture to make prediction.

Image rotations. This pretext task predicts the rotation of an input image, which is effective and simply incorporated into our few-shot learning paradigm.In the image rotations predictive task, the convnet must recognize among four possible 2D rotations in $\mathcal{R} = \{0°, 90°, 180°, 270°\}$, the one applied to an image. Specifically, given an image x, we first create its four rotated copies$\{\mathbf{x}^\mathbf{r} | \mathbf{r} \in \mathcal{R}\}$,here $\mathbf{x}^\mathbf{r}$ is the image $\mathbf{x}$ rotated by $\mathbf{r}$ degrees. Based the features $F_\theta(\mathbf{x}^\mathbf{r})$ extracted from such a rotated image, the new network R^r_ϕ attempts to predict the rotation class $\mathbf{r}$. Accordingly, the self-supervised loss of this task is defined as:

$$L_{\text{self}}(\theta, \phi; X_b) = \mathbb{E}\left[\sum_{\forall r \in \mathcal{R}} - \log R^r_\phi\left(F_\theta\left(\mathbf{x}^r\right)\right)\right] \tag{1}$$

where X denotes the original meta-train dataset consisting of non-rotated images and the negative log-likelihood loss is used to optimize the feature extractor $F_\theta(\cdot)$

Color permutation. Image's color are influenced with the pixel values on RGB channels. Permutation of 3 channels (Noroozi & Favaro, 2016) constructs $3! = 6$ different images where wrong swapping will makes new images in a weird style. We first create 6 permuted copies. Base on the Cosine Classifier, features extractor could learn the feature in the color permutation task.

method	**backbone**	*mini*		*CIFAR*	
		5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
MatchingNets (Vinyals et al., 2016)	4CONV	43.56	55.31	-	-
	ResNet-12	63.08	75.99	-	-
MAML (Finn et al., 2017)	4CONV	48.70	63.11	38.10	50.40
ProtoNet (Snell et al., 2017)	4CONV	49.42	68.20	-	-
	ResNet-12	60.37	78.02	41.54	57.08
BF3S (Gidaris et al., 2019)	4CONV	54.83	71.86	-	-
	WRN-28-10	62.93	79.87	-	-
MTL (Sun et al., 2019)	ResNet-12	61.20	75.50	45.10	57.60
ours (*w. the best SS-task*)	ResNet-12	60.01	**80.97**	39.69	55.40

Table 1: Comparative results for FSL on the miniImageNet dataset and the Fewshot CIFAR-100 dataset. The averaged accuracy (%) on 600 test episodes is reported.

Relative patch location. In this task, patches are created randomly from an images as (Doersch et al., 2015) done. Among the 9 positions in 3×3 grid. Specifically, we divide one image into 9 patches on 3×3 grid and then randomly sample a patch within this region. Binary Cross Entropy are applied to measure the distance of patch labels and prediction.

3.3 Meta-learning with pre-training model

The meta learning frame is follow the MTL(meta-transfer learning). It was divided into 2 parts. In this two parts, only the Strength t of Scaling and shifting and the fully connected network will be fine-tuned. That enable faster convergence in meta-learning processing.

To illustrate the meta-learning process of our model, we can first standardize the setting of meta-learning. Each task $\mathcal{T}$ can be denoted as follows:

$$\mathcal{T}=\left\{\mathcal{L}\left(\mathbf{x}_1, \mathbf{a}_1, \ldots, \mathbf{x}_H, \mathbf{a}_H\right), q\left(\mathbf{x}_1\right), q\left(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{a}_t\right), H\right\} \tag{2}$$

In the K-shot learning, the model is learn from a support set with size K which have known label. Then the model should be test on the query set. Since the the label of the query set on the training dataset is known, while in the test set the query is need to be predicted, in the meta-learning it can be train as a meta-learner which have 2 parts.

$$L_{\text{few}}\left(\theta, W_b; D_b\right)=\underset{(\mathbf{x}, y) \sim D_b}{\mathbb{E}}\left[-\log C^y\left(F_\theta(\mathbf{x}); W_b\right)\right] \tag{3}$$

Combined with the self-supervised learning auxiliary task, our goal can be written as:

$$\min_{\theta,\left[W_b\right], \phi} L_{\text{few}}\left(\theta,\left[W_b\right]; D_b\right)+\alpha L_{\text{self}}\left(\theta, \phi; X_b\right) \tag{4}$$

Meta-training part: This stage is concerned about fine tuning the parameters with 2 gradient descents. The first optimization is on the support set. It calculate loss of each task and optimized with gradient descent. The second optimization is after test on the query set. This model makes predictions on all query sets. After the prediction is over the loss of all tasks are summed, and then a gradient descent is performed.

Meta-testing part: The final prediction will only have few shots for training. Corresponding with the setting of maml (Finn et al., 2017), with support set of the test data the base-learner could be trained to fine tuning the parameters. In this way, we could evaluate new tasks on the query set of the model.

4 Experiments and Discussions

4.1 Datasets

We perform extensive experiments on two popular few-shot learning benchmarks, Mini-Imagenet (Snell et al., 2017), and CIFAR-FS (Bertinetto et al., 2018). Mini-Imagenet is a mainstream data set widely used in recent studies (Finn et al., 2017), which is randomly sampled from ImageNet benchmark. And CIFAR-FS benchmark is much more chanllenging than Mini-Imagenet due to its more constrained splits between training set and test set and lower image resolution.

Mini-Imagenet. Mini-Imagenet consists of 100 classes randomly picked from the ImageNet dataset, including 64 base classes, 16 validation classes, and 20 novel test classes. In each class, there are 600 images. Due to the diversity of ImageNet, it is more complex than other mainstream datasets but requires less memory and computation resource than directly training on the entire ImageNet.

Fewshot-CIFAR100. Fewshot-CIFAR100 is based on the popular object classification data set CIFAR100 (Krizhevsky et al., 2009). It provides a more challenging solution with lower image resolution and more challenging meta-training/testing splits, which are separated according to object superclasses. It contains 100 object classes, each with 600 32×32 color image samples. These 100 categories belong to 20 super categories. The meta-training data is expanded from 60 categories to 12 super categories. The meta-verification and meta-test sets each contain 20 categories, which belong to 4 super categories. These splits conform to the superclass, thereby minimizing the overlap of information between training and evaluation/testing tasks.

4.2 Implementation Details

We consider the task of a) 5 class classification with 1-shot and 15 query, b) 5 class classification with 5-shot and 15 query. The sample strategy is following the related work Meta-transfer learning (Sun et al., 2019) as uniform sampling. The pre-training parameters are the optimal selected with the validation set in 110 epochs of training. Learning rate of pre-training stage is 0.1, while learning rates on support set (query set) in the meta-learning stage is 0.01 (0.0001 on encoder or 0.001 on classifier).

Backbones. In our experiments, the network architecture of few shot classification task is same as ResNet-12 (Vinyals et al., 2016)on the two datasets with 3 auxiliary task. ResNet-12 is a popular backbones of fewshot setting. The feature extractor4 contains residual blocks and 3 CONV layers with 3×3 kernels where the number of filters in first layer is 64 and is doubled every next block. At each end of residual block, there are 2×2 max-pooling. After the blocks, there is a mean-pooling layer to pool the feature map. The architecture of classifier is one layer fully connected network which correspondingly predict the features extracted into 5 classes.

In order to improve the performance of extractor, the auxiliary task share feature extraction layer. The parameter of classifier of auxiliary task is Cosine Classifier for simplicity and flexibility. Since the different of three auxiliary task is only the output dimension, the only hyper parameter in the network structure is the weights of auxiliary tasks where the few shot task and self-supervised task have a weight of 0.5 each.

Baseline. MAML MTL SSL Compared with three existing algorithms: MAML (Finn et al., 2017), MTL (Sun et al., 2019), BF3S (Gidaris et al., 2019), perform the experiments with backbones mentioned above indicating with self-supervised and pre-train part, our model is better than any of the self-supervised learning, MAML and MTL. For our approach, we use rotation prediction as the self-supervised learning task.

Evaluation Metrics. Few-shot classification algorithms are evaluated based on the classification accuracy in the meta-test stage. Specifically, a large number of N_n-way K-shot tasks are sampled from the novel classes dataset. Each task is created by randomly sampling N_n novel classes from the available test (validation) classes and then within the selected classes, we randomly select K samples as support set and M samples as query set per class. The classification accuracy is measured on the $N_n \times M$ test images and is averaged over all the sampled few-shot tasks. Except otherwise stated, we set $N_n = 5$, $M = 15$, and $K = 1$ or $K = 5$ (1-shot and 5-shot settings respectively)for all experiments.

Rotation		*Color*		*Patch*	
1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
60.01	**80.97**	57.00	72.77	21.25	21.92

Table 2: The testing accuracy of novel tasks using 3 different self-supervised tasks (measured in %)

4.3 Experimental Results

In our experiments, we used one NVIDIA GTX 2080 Ti GPU (with 11GB CUDA memory) for training. Table 1 shows the results of five previous methods and the results of ours on the mini-ImageNet and the FC100 datasets. Note that for our proposed method, Table 1 reports the best result among three self-supervised methods we tried, *i.e.* we reported the accuracy when using the rotation auxiliary task for the experiment. For results obtained using different auxiliary tasks, please refer to Section 4.4.

For mini-ImageNet, our model with self-supervised task has beat 80.97% accuracy on 5 shot setting which exceeds all listed base lines with same back bone. This result exceeds BF3S by 1% (even if it used the more powerful WRN-28-10 backbone), and exceeds MTL by more than 5%. It implies that combining meta-learning and self-supervised learning improve the results of the model.

On the more challenging Fewshot-CIFAR100 dataset, our model still exceeds the basic MAML model by more than 1% on the 1 shot task and more than 5% on the 5 shot task. However, the performance does not exceed the MTL model proposed in 2019. Maybe because there are too many parameters, which is difficult to adapt to more difficult tasks.

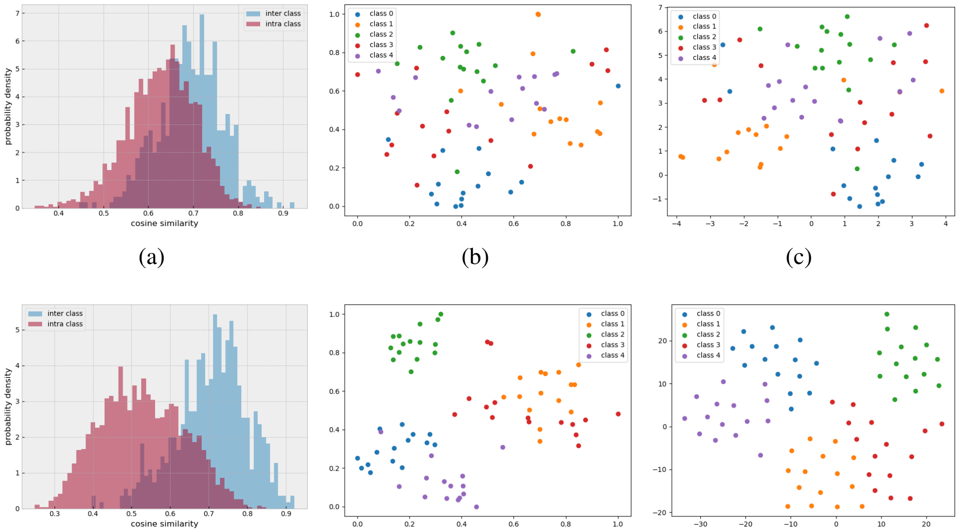
• **Analysis** on results that are not so good. We admit that the performance of our method is far from satisfactory. We think there are two possible reasons. (1) Adding auxiliary tasks would increase the computation in training. However, due to the cuda memory limit, we used mixed precision when training, in order to reduce the computational cost. This may result in insufficient accuracy during gradient descent, thus affecting the training results. (2) The 1-shot-learning task may require more elaborate tunning of the hyper-parameters in our experiments. However, due to the time and computational resource limits, we have not found a set of parameters that works for the 1-shot task. For the above reasons, our experimental results are still comparable to the baseline, and some of the results even exceed the BF3S and MTL models we rely on.

4.4 Comparison Between Different Self-supervised Tasks

In Section 4.3, we have seen that our method achieved competitive performance. In this section, we compared different self-supervised tasks, which were key components of our approach. Besides the rotation prediction task, we also tried the color channel permutation task and the relative patch location task.

Table 2 shows that the rotation task is the best, while the relative patch location task is the worst, and the color channel permutation task is of the middle performance. We thought simpler self-supervised task would lead to better performance than more difficult self-supervised task. This might due to the fact that more difficult tasks would force the network to learn embeddings related to more complex knowledge. The theoretical analysis on this conclusion will be left to future work.

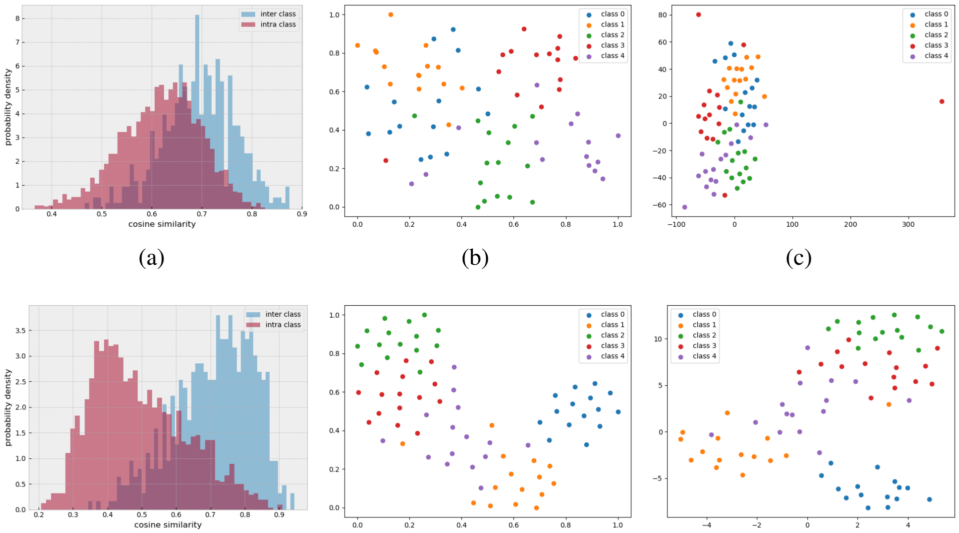
Furthermore, we have conducted some visualizations to make clear the properties of features learned using different self-supervised tasks. Let $\mathbf{f}_i^{(t)} = F_\theta^{(t)}(\mathbf{x}_i)$ be the feature of sample $\mathbf{x}_i$ learned with self-supervised task t. For 5-way 1-shot tasks, Figure 2(a) shows the normalized histogram the inter-class cosine similarity $\left\{cos\left(\mathbf{f}_i^{(patch)}, \mathbf{f}_j^{(patch)}\right) : i < j, y_i = y_j\right\}$ (red) and the intra-class cosine similarity $\left\{cos\left(\mathbf{f}_i^{(patch)}, \mathbf{f}_j^{(patch)}\right) |i < j, y_i \neq y_j\right\}$ (blue). Figure 2(b) and Figure 2(c) shows the t-SNE visualization of embeddings using Euclidean and cosine distance. The second row in Figure 2 shows the visualization corresponding to the rotation task. Figure 3 shows the visualization corresponding to the 5-way 5-shot task. Note that for the same set of comparison, we visualized the sample embeddings in the query set of same test task. Please see Appendix A for more visualization results.



(a) (b) (c)

(d) (e) (f)

Figure 2: Visualization of the sample embeddings in the query set of a certain 1-shot task, using the patch task and the rotation task.



(a) (b) (c)

(d) (e) (f)

Figure 3: Visualization of the sample embeddings in the query set of a certain 5-shot task, using the patch task and the rotation task.

5 Conclusion

In this paper, we tried three kinds of self-supervised task as auxiliary tasks during the training of few-shot recognition models. And we combine transfer learning and self-supervised learning to tackling few-shot learning problems. The annotation-free nature of the self-supervised loss allows us to achieve richer and more transferable visual representations. Transfer learning allows us to maintain the learned common characteristics. The key operations of our method on pre-trained phase and meta-train phase efficiently adapting learning experience to the unseen task. In terms of learning scheme, out method with rotate auxiliary task have good performance on Mini-Imagenet and FC100 benchmark. Finally, we visualize features to clearly present the superiority of our method.

References

Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14493–14502, 2020.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pp. 5049–5059, 2019.

Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 132–149, 2018.

Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pp. 1422–1430, 2015.

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

Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Boosting few-shot visual learning with self-supervision. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 8059–8068, 2019.

Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3018–3027, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Rohit Keshari, Mayank Vatsa, Richa Singh, and Afzel Noore. Learning structure and strength of cnn filters for small sample size training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9349–9358, 2018.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Aoxue Li, Weiran Huang, Xu Lan, Jiashi Feng, Zhenguo Li, and Liwei Wang. Boosting few-shot learning with adaptive margin loss. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12576–12584, 2020.

Bo Liu, Xudong Wang, Mandar Dixit, Roland Kwitt, and Nuno Vasconcelos. Feature space transfer for data augmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9090–9098, 2018.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pp. 69–84. Springer, 2016.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of maml. *arXiv preprint arXiv:1909.09157*, 2019.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *Advances in Neural Information Processing Systems*, pp. 7959–7970, 2019.

Eli Schwartz, Leonid Karlinsky, Joseph Shtok, Sivan Harary, Mattias Marder, Abhishek Kumar, Rogerio Feris, Raja Giryes, and Alex Bronstein. Delta-encoder: an effective sample synthesis method for few-shot object recognition. In *Advances in Neural Information Processing Systems*, pp. 2845–2855, 2018.

Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4136–4145, 2020.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pp. 4077–4087, 2017.

Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 403–412, 2019.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1199–1208, 2018.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Yao-Hung Hubert Tsai and Ruslan Salakhutdinov. Improving one-shot learning through fusing side information. *arXiv preprint arXiv:1710.08347*, 2017.

Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. *arXiv preprint arXiv:2001.08735*, 2020.

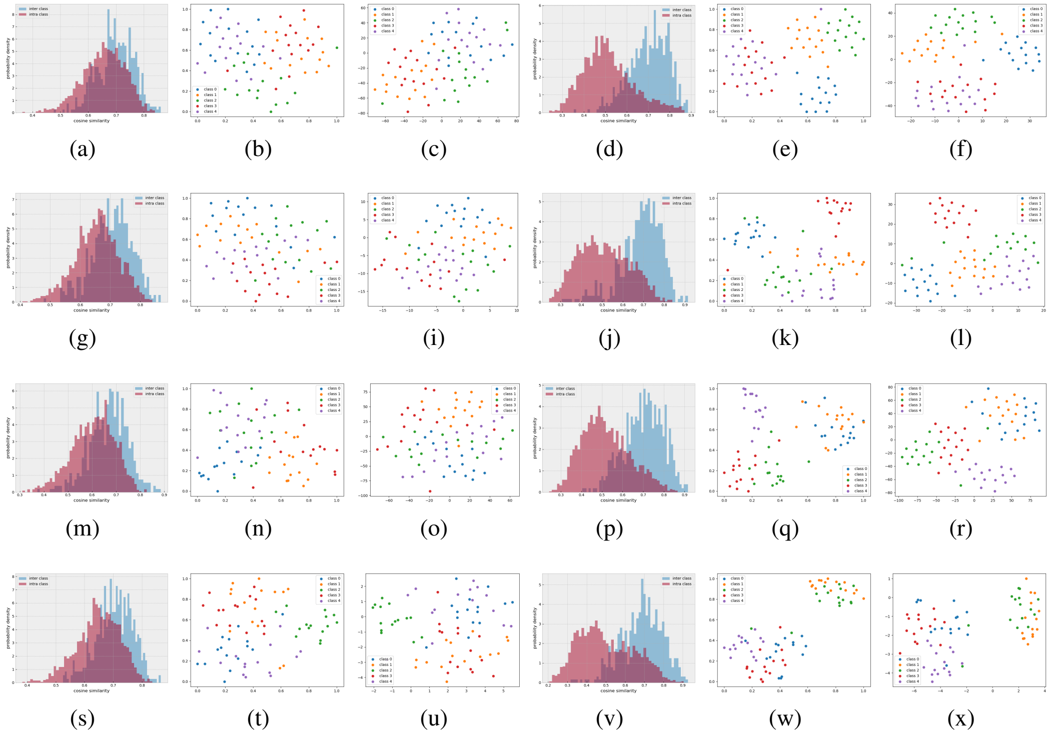
Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys (CSUR)*, 53(3):1–34, 2020.

Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12203–12213, 2020.

A More Visualization Results

This section provides more visualization results of Figure 2 and Figure 3 in Section 4.4.



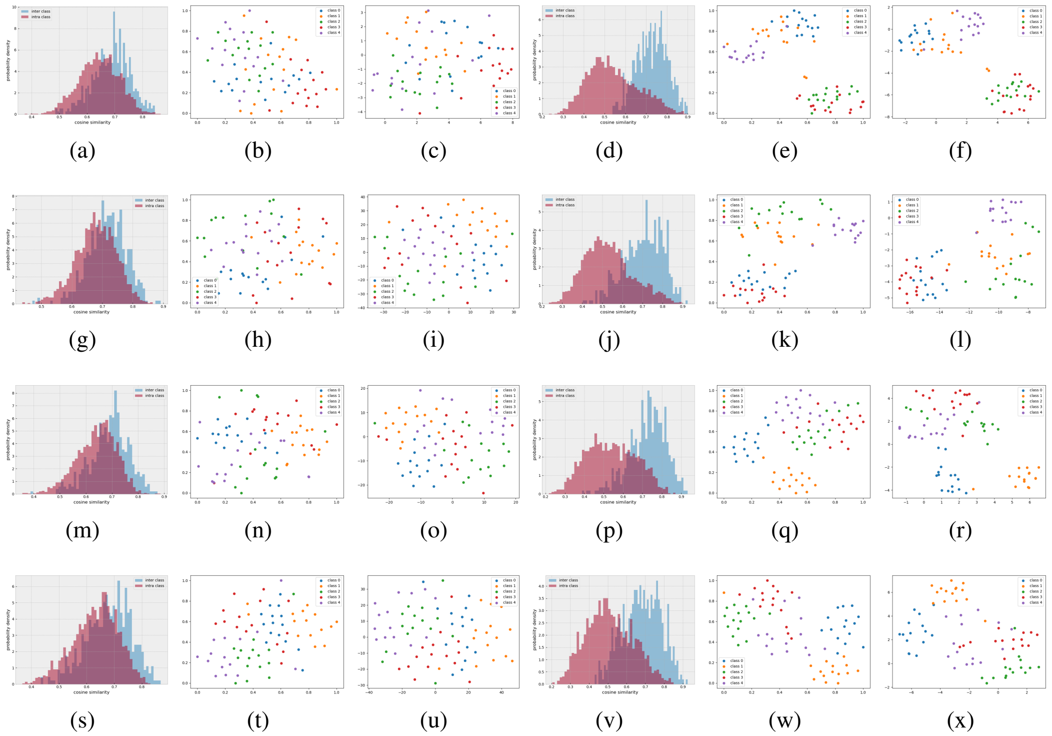
(a) (b) (c) (d) (e) (f)

(g) (i) (j) (k) (l)

(m) (n) (o) (p) (q) (r)

(s) (t) (u) (v) (w) (x)

Figure 4: Visualization of the sample embeddings in the query set of some 1-shot tasks, using the patch task and the rotation task. Each row corresponds to one set of comparison.



(a) (b) (c) (d) (e) (f)

(g) (h) (i) (j) (k) (l)

(m) (n) (o) (p) (q) (r)

(s) (t) (u) (v) (w) (x)

Figure 5: Visualization of the sample embeddings in the query set of some 5-shot tasks, using the patch task and the rotation task. Each row corresponds to one set of comparison.