Augmented Meta-Transfer Learning for Few-Shot Learning

Zixuan Chen
120033910005
m13953842591@sjtu.edu.cn

Guandong Lu
120033910017
lugd0525@sjtu.edu.cn

Xing Zhao
120033910157
1033874657@sjtu.edu.cn

1 Introduction

1.1 Few-shot learning

In short, a few-shot learning problem is using a large annotated offline dataset to perform given tasks for novel categories represented by just a few samples each. In usual cases, it is unable to find enough annotated data because of profit issue for companies. Relevant objects are continuously replaced with new ones, which all bring the need for few-shot learning. In addition, few-shot learning involves a bunch of exciting cutting-edge technologies, including Meta-learning, Graph neural networks, Semantic metric spaces, Data synthesizers and GANs.

1.2 High level idea

The high level idea is to combine meta-learning with genenration-based method. Specifically, we would exploit Meta-Transfer learning [1] and optimize it by generating more examples using Δ-encoder [2].

2 Related work

2.1 Meta-learning

Meta-learning is one of the most commonly used methods for few-shot classification problem. [3] presented Model-Agnostic Meta-Learning(MAML), which can be treated as general steps of meta-learning. It consider K-shot classification problems as learning a new task $\mathcal{T}_i$ with K samples. Specifically, a learning task $\mathcal{T}$ can be expressed as $\mathcal{T} = \{\mathcal{L}(x), q(x)\}$, where $q(x)$ is the distribution over observations and $\mathcal{L}(x)$ denote the loss function between observation x and target value y. The meta-objective can be expressed as two steps. First it learn $f(\theta_i')$ over each individual learning task $\mathcal{T}_i$ based on $f(\theta)$. Then it learn $f(\theta)$ by perform one-step gradient descent on sum of all losses of each individual learning task based on $f(\theta_i')$. Most concretely, it can be expressed as

$$\min_{\theta} \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) = \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta} - \alpha \nabla_{\theta} \mathcal{L}_{\mathcal{T}_i}(f_{\theta})) \quad (1)$$

[4] further optimized MAML by introducing First order MAML(FOMAML). It optimized MAML by ignoring second order derivatives. It also introduced Reptile, which is similar to FOMAML as a joint training algorithm but don't need a train-test split for each task. [1] introduced Meta-Transfer Learning(MTL) which combines transfer learning with meta-learning algorthms. It optimized MAML by replacing the typical parameter-level Fine-Tuning (FT) with neuron-level Scaling and Shifting (SS) operation. It only perform FT on the final liner layer, which saved the resources for training models for individual tasks. Meanwhile, it introduced the hard task (HT) meta-batch scheme as an effective learning curriculum for MTL.

2.2 Metric-based method

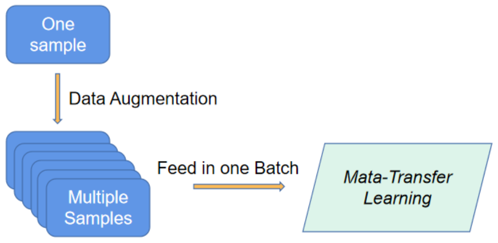
Generally, a metric-based method trains the old classes to achieve good distribution between these classes. Then it exploits a metric to compute the distance from old classes to new classes, which can help the model finding the nearest neighbour in the embedding space. [5] learns a Matching Network that maps a small labelled support set and an unlabelled example to its label, obviating the need for fine-tuning to adapt to new class types. [6] proposed a Relation Network (RN) which can compute relation scores between images, which can be used to classify new classes of images of new classes without further updating the network.

2.3 Generation-based method

Generation-based method solves the few-shot classification problem by augmenting the data instances using synthesizer models. The synthesizer models accept few novel data instances and large normal data instances and produce many data instances. The problem thus can be treated as normal classification problem. So the key to Generation-based method is the proper construction of the synthesizer model. [7] combines meta-learner with "hallucinator" that produces additional training examples, and optimizing both models jointly. [2] presented Δ-encoder based on modified auto-encoder. By exploiting different keys of encoder and decoder, it can generate new instances of novel classes.

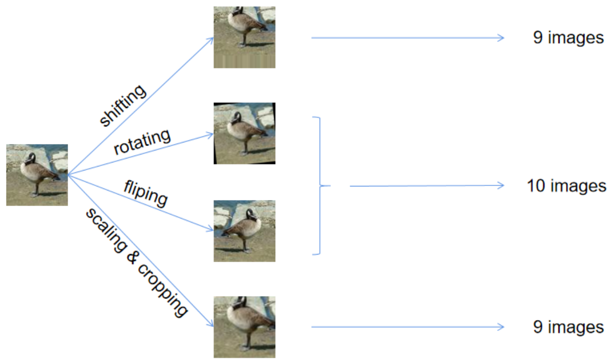
3 Method

We proposed our method, A-MTL training method, as a combination of data augmentation and Meta-transfer learning. We take advantages of the augmented information in generating inputs, and the transfer learning ability of meta-learning. In general, our method can be divided into two parts: Data Augmentation and Meta-Learning. We will describe them in the following distinguished parts. We follow the insight of the more shots means more information to learn. Generally, our idea can be simplified as the following image.



3.1 Data Augmentation

The main propose of data augmentation is to get more samples to train. To achieve this, we will use several ways of data augmentation which will discussed in detail below.
(1)Flipping. Which flips the image from left to right. Because in normal cases, the object in the image cannot be up-side-down, so we do not consider to flip the image from up to down.
(2)Cropping and Scaling. Select a part of the image from the original one. To maintain the size of the input samples, we scale the cropped image into the original image size. To achieve the work above, we change the transformation into an equivalent way: First we scale the original image 1.1, 1.2, 1.3 times larger than the original one, then we pick a window with the same size of the original image. In this way we can get a cropped image with size not changed.
(3)Rotation. Rotate an angle from the original image. We use opencv package to rotate an input image from -6, -3, 3, 6 degree. Since if we crop the rotated image with the original image size, there will be some blanks on the corner. We fill these blanks with (0, 0, 0).
(4)Shifting. Moving the original image several pixels. We move the original image up, down, left, right for 10% and 20%, with the moved part filled with the boundary pixel of the image.
In addition, as the image flipping can only generate one augmented image, we combine the flipping and rotation together. In summary, for each input image, we augment the input as the following image:



3.2 Meta Learning

We will use the method based on MTL-learning [1] to train an Scale-and-Shifting model. After we get a set of augment images, we put them all in one batch and do the MTL-learning.

3.2.1 Network Structure

The architecture of Θ of MTL have two option, ResNet-12 and ResNet-18. $\mathbf{ResNet-12}$ contains 4 residual blocks and each block has 3 CONV layers with 3×3 kernels. At the end of each residual block, a 2×2 max-pooling layer is applied. The number of filters is 64, 160, 320 and 640. Following 4 blocks, there is a mean-pooling layer to compress the output feature maps to a feature embedding. $\mathbf{ResNet-18}$ contains 8 basic blocks and each block has 2 CONV layers with 3×3 kernels. The number of filters starts from 64 and is doubled every next block. The architecture of θ is a single FC layer.

3.2.2 Loss function

For pre-train stage, we use stochastic gradient descent(SGD), with learning rate=0.1, step_size=30, momentum=0.9 and weight_decay=0.0002. For meta-train stage, we use Adam, with learning rate=0.01.

4 Experimental setting

4.1 Dataset

We will use the typical but state-of-the-art dataset such as: miniImagenet, Fewshot-CIFAR100.

miniImageNet proposed by Vinyals et al. for few-shot learning evaluation. Its complexity is high due to the use of ImageNet images but requires fewer resources and infrastructure than running on the full ImageNet dataset. In total, there are 100 classes with 600 samples of 84×84 color images per class. These 100 classes are divided into 64, 16, and 20 classes respectively for sampling tasks for meta-training, meta-validation, and meta-test.

Fewshot-CIFAR100 (FC100) is based on the popular object classification dataset CIFAR100. The splits were proposed by TADAM. It offers a more challenging scenario with lower image resolution and more challenging meta-training/test splits that are separated according to object super-classes. It contains 100 object classes and each class has 600 samples of 32×32 color images. The 100 classes belong to 20 super-classes. Meta-training data are from 60 classes belonging to 12 super-classes. Meta-validation and meta-test sets contain 20 classes belonging to 4 super-classes, respectively.

4.2 Baseline

Our baseline is MTL-learning [1] and another state-of-the-art method MAML [4]. The architecture of MTL is introduced on 3.2.1

The architecture of MAML includes 4 modules with a 3×3 convolutions and 32 filters, followed by batch normalization, a ReLU non-linearity, and 2×2 max-pooling. The last layer is fed into a softmax. The loss function is the cross-entropy error between the predicted and true class.

4.3 Evaluation Metric

We consider the 5-class classification(5-way) and we sample 5-class, 1-shot (5-shot or 10-shot) episodes to contain 1 (5 or 10) samples for train episode, and 15 (uniform) samples for episode test. The result accuracy is presented in the form "mean value + confidence interval", which is calculated by multi-batches of validation records.

4.4 Experimental Result

Table4.4 and Table4.4 present the overall result on miniImageNet and FC100 dataset. We compare our method with MTL and MAML

4.5 Experimental Observation

From the result, we can see that our method outperforms than MTL and MAML in most cases, which can show that our method is useful for enhancing the result of Meta-Transfer Learning by using the data augmentation method. However, it is interesting that in some 5-shot experiments, we can see that our method did not reach the MTL baseline. It might be because that in 5-shot learning, we get an acceptable amount of information to do Meta-Transfer Learning, but when doing data augmentation, we introduce some unexpected noises which is not benefit to the training. However, it is useful when lack of information, such as in 1-shot learning cases. So all of the 1-shot experiments outperforms than the baseline.

For MTL and A-MTL we perform experiment based on ResNet-12 and ResNet-18. We can see from the result that MTL and A-MTL based on ResNet-12 performs better in most scenarios.

5 Conclusion

In this report, we propose a method called A-MTL, which combines the benefits between data augmentation and meta-transfer learning. The experiment result shows that our method can be useful when the information is not much for the Meta-transfer learning.

References

[1] Sun, Q., Liu, Y., Chua, T.-S., and Schiele, B., 2019. "Meta-transfer learning for few-shot learning". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 403–412.

[2] Schwartz, E., Karlinsky, L., Shtok, J., Harary, S., Marder, M., Kumar, A., Feris, R., Giryes, R., and Bronstein, A., 2018. "Delta-encoder: an effective sample synthesis method for few-shot object recognition". In Advances in Neural Information Processing Systems, pp. 2845–2855.

[3] Finn, C., Abbeel, P., and Levine, S., 2017. "Model-agnostic meta-learning for fast adaptation of deep networks". arXiv preprint arXiv:1703.03400.

[4] Nichol, A., Achiam, J., and Schulman, J., 2018. "On first-order meta-learning algorithms". arXiv preprint arXiv:1803.02999.

[5] Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al., 2016. "Matching networks for one shot learning". In Advances in neural information processing systems, pp. 3630–3638.

[6] Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M., 2018. "Learning to compare: Relation network for few-shot learning". In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1199–1208.

[7] Wang, Y.-X., Girshick, R., Hebert, M., and Hariharan, B., 2018. "Low-shot learning from imaginary data". In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7278–7286.

Few-shot learning method	Feature extractor	1-shot	5-shot
MAML	4CONV	$0.4569 + 0.2278$	$0.6273 + 0.1191$
MTL	ResNet-12(pre)	$0.5328 + 0.0080$	$0.6857 + 0.0070$
	ResNet-18(pre)	$0.4632 + 0.0077$	$0.6618 + 0.0069$
A-MTL(ours)	ResNet-12(pre)	$\mathbf{0.5359 + 0.0083}$	$\mathbf{0.6903 + 0.0070}$
	ResNet-18(pre)	$0.4734 + 0.0074$	$0.6597 + 0.0069$

Table 1. The 5-way, 1-shot and 5-shot classification accuracy (%) on miniImageNet dataset. pre means pre-trained for a single classification task using all training datapoints.

Few-shot learning method	Feature extractor	1-shot	5-shot
MAML	4CONV	$0.3630 + 0.2304$	$0.4838 + 0.1037$
MTL	ResNet-12(pre)	$0.3922 + 0.0071$	$0.5153 + 0.0070$
	ResNet-18(pre)	$0.3749 + 0.0078$	$0.5224 + 0.0071$
A-MTL(ours)	ResNet-12(pre)	$\mathbf{0.4007 + 0.0079}$	$0.5119 + 0.0071$
	ResNet-18(pre)	$0.3804 + 0.0069$	$\mathbf{0.5256 + 0.0069}$

Table 2. The 5-way with 1-shot, 5-shot and 10-shot classification accuracy (%) on Fewshot-CIFAR100 (FC100) dataset. pre means pre-trained for a single classification task using all training datapoints