
LGM-Net: Learning to Generate Matching Networks for Few-Shot Learning

Huaiyu Li^{1,2} Weiming Dong¹ Xing Mei³ Chongyang Ma⁴ Feiyue Huang⁵ Bao-Gang Hu¹

Abstract

In this work, we propose a novel meta-learning approach for few-shot classification, which learns transferable prior knowledge across tasks and directly produces network parameters for similar unseen tasks with training samples. Our approach, called LGM-Net, includes two key modules, namely, TargetNet and MetaNet. The TargetNet module is a neural network for solving a specific task and the MetaNet module aims at learning to generate functional weights for TargetNet by observing training samples. We also present an intertask normalization strategy for the training process to leverage common information shared across different tasks. The experimental results on Omniglot and *mini*ImageNet datasets demonstrate that LGM-Net can effectively adapt to similar unseen tasks and achieve competitive performance, and the results on synthetic datasets show that transferable prior knowledge is learned by the MetaNet module via mapping training data to functional weights. LGM-Net enables fast learning and adaptation since no further tuning steps are required compared to other meta-learning approaches.

1. Introduction

The ability to rapidly learn and generalize from a small number of examples is a critical characteristic of human intelligence, because humans can leverage the prior knowledge obtained from previous learning experience (Lake et al., 2017). Although current deep learning approaches have achieved significant success in many tasks (Krizhevsky et al., 2012; Ren et al., 2015; Long et al., 2015), massive labeled data and excessive training

time are still required, because each task is independently considered and the model parameters are learned from scratch without incorporating task-specific prior knowledge. How to extract prior knowledge and transfer them to unseen tasks with limited data has become active research areas in machine learning, such as transfer learning (Weiss et al., 2016), metric learning (Koch et al., 2015), and domain adaptation (Motiian et al., 2017).

In this work, we focus on few-shot learning, which aims at learning to recognize unseen categories from few labeled samples. Recently, meta-learning (Thrun & Pratt, 2012) has emerged as a kind of promising approach to solve this problem. A generic meta-learning framework usually contains a meta-level learner and a base-level learner (Andrychowicz et al., 2016; Munkhdalai & Yu, 2017). The base-level learner is designed for specific tasks, such as classification, regression, and neural network policy (Finn et al., 2017). The meta-level learner aims to learn prior knowledge across different tasks. The prior knowledge can be transferred to the base-level learner to help quickly adapt to similar unseen tasks. For instance, (Ravi & Larochelle, 2017) proposed to train the meta-level learner as the weight optimizer of a base-level learner. The prior knowledge was embedded in how to update the base-level learner on a new task. (Finn et al., 2017) proposed to learn a good initialization that can be quickly adapted to new tasks with a few updates. The prior knowledge was learned and embedded in the good initialization.

In this work, we propose a novel meta-learning approach for few-shot learning. In contrast to previous methods (Munkhdalai & Yu, 2017; Ravi & Larochelle, 2017; Andrychowicz et al., 2016), our approach directly generates the functional weights of a network with limited training samples. Our approach contains two key modules, namely, a TargetNet module (the base-level learner) and a MetaNet module¹ (the meta-level learner). The TargetNet module is designed for specific tasks. Traditionally, the parameters of this network are randomly initialized and adjusted by stochastic gradient descent (SGD) algorithms applied on training data. However, the parameters in TargetNet are generated by the MetaNet module conditioned on

¹National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China ²University of Chinese Academy of Sciences, Beijing 100049, China ³Snap Inc. ⁴Kwai Inc. ⁵Youtu Lab, Tencent. Correspondence to: Weiming Dong <weiming.dong@ia.ac.cn>.

¹Our MetaNet module has the same abbreviation with, but is different from, Meta Networks (Munkhdalai & Yu, 2017) method.

training samples. The MetaNet contains two parts, namely, a *task context encoder* designed to learn task context representation and a *weight generator* that learns the conditional distribution of functional weights of TargetNet. We train the task context encoder on many tasks to encode training data of each task. The weight generator is trained to generate the weights of TargetNet for each task conditioned on encoded task representation. The prior knowledge about how to generate functional weights using training data is learned by the MetaNet module, which is a new form of representing transferable prior knowledge. We apply an intertask normalization (ITN) strategy which leverages information shared among different tasks to help training.

After training, MetaNet can generate the weights of TargetNet which can effectively generalize to similar unseen tasks. In the proposed approach, no further fine-tuning steps are necessary when meet unseen tasks because we directly produce functional weights. Accordingly, rapid learning and adaptation on new tasks are achieved. More specifically, we use matching networks (Vinyals et al., 2016) as the TargetNet structure and MetaNet learns to generate the weights for matching networks. Hence, we denote our approach as LGM-Net. Our contributions can be summarized as follows:

- A novel meta-learning algorithm that trains MetaNet to generate the weights of TargetNet on the basis of training samples.
- A simple and effective MetaNet module that encodes training samples and learns conditional distribution of TargetNet parameters.

We demonstrate the effectiveness of our approach on Omniglot and *miniImageNet* datasets. LGM-Net significantly outperforms many other meta-learning methods, especially for *miniImageNet*. We also conduct extensive experiments to show the mechanism of LGM-Net and analyze the generated weights on different tasks for validating our approach.

2. Related Work

Our approach aims at rapidly adapting to similar unseen tasks via generating weights using limited data. Many meta-learning methods are relevant to our work.

Matching networks (Vinyals et al., 2016) use an attention mechanism in the embedding space of training samples to predict classes for testing samples. This model proposes an episodic training scheme where each episode is designed to mimic a few-shot task. Several recent meta-learning approaches extend this episodic training idea. (Ravi & Larochelle, 2017) presented an LSTM-based meta-level

learner to learn the exact learning rules which can be utilized to train a neural network classifier in a few-shot regime. MAML (Finn et al., 2017) learns a good initialization of neural networks, which can be fine-tuned in a few gradient steps and effectively generalized on new unseen few-shot tasks. Other approaches propose to learn a good initialization with an update function (Li et al., 2017) or without any update steps (Nichol & Schulman, 2018). These meta learning methods represent the transferable prior knowledge in the good initialization or learned update functions. In contrast to their approaches, our method represents prior knowledge in encoding the few-shot task samples and producing functional parameters of the TargetNet for each task without fine-tuning.

Another relevant direction involves the exploration of using one neural network to produce the parameters of another. The conceptual framework of fast weights (Hinton & Plaut, 1987; Ba et al., 2016) is proposed to simulate synaptic dynamics at different time scales for storing temporary memories. (Munkhdalai & Yu, 2017) presented a meta-learning model that is equipped with external memory and utilizes meta-information to quickly parameterize both a meta-level learner and a base-level learner. (Qiao et al., 2018) proposed to learn to predict the parameters of a top layer from the activations and adapt the pre-trained neural networks to novel categories. (Gidaris & Komodakis, 2018) used an attention based weight generator to predict the weights for novel categories from pre-trained weights without forgetting previously learned categories. (Rusu et al., 2019) proposed to learn a data-dependent latent representation of model parameters and perform meta-learning in the latent space rather than in the parameter space as (Finn et al., 2017). (Wu et al., 2018) proposed to learn a model code via a meta-recognition model and to construct parameters for a task-specific model by using a meta-generative model.

3. Our Method

3.1. Problem Formulation

We define an N -way K -shot problem using the episodic formulation from (Vinyals et al., 2016). Formally, we have three datasets, i.e., a meta training dataset $D^{meta-train}$, a meta validation dataset $D^{meta-val}$, and a meta test dataset $D^{meta-test}$. Each dataset contains a disjoint set of target classes. For each dataset, we can construct a task distribution $p(\mathcal{T})$ of N -way K -shot tasks. Each task instance $\mathcal{T}_i \sim p(\mathcal{T})$ consists of a training set S_i^{train} and a test set S_i^{test} . The training set S_i^{train} contains N classes randomly selected from a meta dataset and K samples for each class. The test set S_i^{test} contains unseen samples for classes in S_i^{train} and provides an estimation of generalization performance on the N classes for task \mathcal{T}_i .

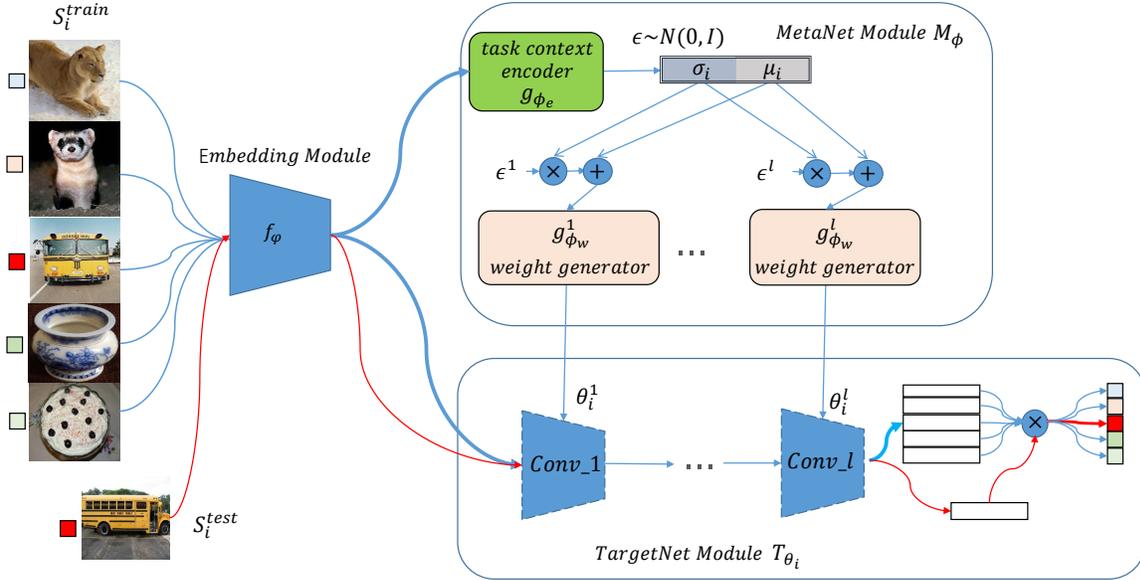


Figure 1. The architecture of our LGM-Net for few-shot learning on 5-way 1-shot classification problems.

We use the meta training dataset to train our model and the meta validation dataset for model selection. The meta test dataset is only used to evaluate the model generalization on unseen tasks.

3.2. Preliminary

The functionality of a deep neural network (DNN) depends on its weights and architecture. When the architecture of a neural network is fixed, the weights determine its functionality. For instance, the AlexNet architecture (Krizhevsky et al., 2012) can be trained to classify 100 types of animals or 100 categories of plants. However, the two AlexNets have different weights. A network has no functionality when the weights are randomly initialized. In this work, we refer to the weights that embody networks abilities as *functional weights*.

When we train a DNN for a specific task, we usually use gradient descent algorithms (Ruder, 2016) to optimize randomly initialized weights on the training data for a finite number of iterations. Since the loss landscape (Li et al., 2018) of neural networks is extremely complicated, the optimization process often converges to different local optima or saddle points. For test samples, these points in the weight space usually present similar generalization ability (Feizi et al., 2017). Therefore, we can consider these functional weights as a distribution over training data. We design the meta-learning approach that can learn the conditional functional weight distribution over training data and directly produce functional weights for DNNs with limited training samples.

Algorithm 1 The training algorithm of LGM-Net for N -way K -shot problems

Required: Meta training dataset $D^{meta-train}$
Required: MetaNet M with parameters ϕ , TargetNet computational structure T with parameter placeholder θ .
 Randomly initialize ϕ
while not converged **do**
 Sample a N -way K -shot task batch \mathcal{T}^{batch} from $D^{meta-train}$
 for all the task instances in a batch **do**
 Divide a task instance as $(S_i^{train}, S_i^{test}) = \mathcal{T}_i$
 Sample a functional weights point $\hat{\theta}$ for TargetNet from $M(S_i^{train})$
 Assign generated weights $\hat{\theta}$ to TargetNet placeholder weights θ
 Compute TargetNet test loss for this task on S_i^{test} as $\mathcal{L}_{\mathcal{T}_i}$
 end for
 Compute batch loss $\mathcal{L}_{\mathcal{T}^{batch}} = \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i}$
 Update ϕ using $\nabla_{\phi} \mathcal{L}_{\mathcal{T}^{batch}}$
end while

3.3. Methodology

As illustrated in Figure 1, our LGM-Net architecture consists of two key modules, namely, a TargetNet module and a MetaNet module. The training procedure is summarized in Algorithm 1. We first get a batch of task \mathcal{T}^{batch} given the meta training dataset. For each task instance \mathcal{T}_i , the MetaNet module generate a functional weight point $\hat{\theta} = M(S_i^{train})$ for the TargetNet conditioned

on training set. Then, the TargetNet assigned with generated weights can infer the matching probability scores for test samples. The classification loss is simultaneously computed. Finally, for each task in a batch, we accumulate the losses and compute the gradient updates for the parameters in MetaNet module. For high dimensional input data, a learnable embedding module f_φ is used to extract low dimensional features as inputs for the two modules. In this way, the amount of parameters of the entire model can be reduced.

3.4. MetaNet Module

Our MetaNet module consists of a task context encoder (Section 3.4.1) and a conditional weight generator (Section 3.4.2).

3.4.1. TASK CONTEXT ENCODER

The task context encoder aims to encode all the training samples of a task and generate a feature representation of the task with fixed size. The task context features should satisfy the following properties to appropriately represent a task: enough distinctions between different tasks, sufficient similarities between similar tasks, and insensitivity to the number and order of samples in a certain task. More importantly, the encoding operation should be differentiable. In Matching networks (Vinyals et al., 2016), BiLSTMs (Hochreiter & Schmidhuber, 1997) are used to extract fully contextual embeddings. However, using BiLSTMs to encode task contexts in our case makes training and converging difficult. We compute the summary statistics of training set of each task without any supervision for simplicity and effectiveness in accordance with the Neural Statistician (Harrison Edwards, 2017). The task context features are reparameterized as a conditional multivariate Gaussian distribution with a diagonal covariance. Specifically, given an N -way K -shot task \mathcal{T}_i with a training set $S_i^{train} = \{(x_i^{n,k}, y_i^{n,k}) | k = 1 \dots K, n = 1 \dots N\}$, we formulate the sampling of the task context as follows:

$$\mu_i, \sigma_i = \frac{1}{NK} \sum_{n=1}^N \sum_{k=1}^K g_{\phi_e}(x_i^{n,k}), \quad (1)$$

$$\mathbf{c}_i \sim q(\mathbf{c}_i | S_i^{train}) = \mathcal{N}(\mu_i, \text{diag}(\sigma_i^2)), \quad (2)$$

where g_{ϕ_e} is the task context encoder, q denotes a conditional probability distribution of task context features and \mathbf{c}_i denotes a sample of task context features for \mathcal{T}_i .

3.4.2. CONDITIONAL WEIGHT GENERATOR

The weight generator is trained to generate the functional weights for TargetNet conditioned on task context features. For each layer of the TargetNet, we construct a conditional

single layer perceptron as the generator to produce weights as follows:

$$\theta_i^l = g_{\phi_w}^l(\mathbf{c}_i), \quad (3)$$

where $g_{\phi_w}^l$ is the weight generator for l -th layer.

We apply weight normalization (WN) to constrain the weight scale for facilitating the training process, but remove learnable parameters compared with the original WN method (Salimans & Kingma, 2016). For the generated weights of a convolution layer, the L2 normalization is applied to each kernel rather than the entire convolution weights. For the generated weights of a fully connected layer, the L2 normalization is applied to each hyperplane weights, which can be formulated as:

$$\theta_{i,j}^l = \frac{\theta_{i,j}^l}{\|\theta_{i,j}^l\|_2}, \quad (4)$$

where $\theta_{i,j}^l$ is the j -th kernel or hyperplane weight of generated weights of l -th layer on i -th task. In practice, the WN helps prevent generating large scale features and stabilize the training process.

3.5. TargetNet Module

We use matching networks as the architecture of TargetNet for few-shot classification problems. The functional weights of TargetNet are generated by MetaNet based on training samples. There are many newly designed parametric layers in neural networks, such as parametric ReLU (He et al., 2015) and batch normalization (Ioffe & Szegedy, 2015). These layers contain learnable parameters and aim to stabilize the training of DNNs. Therefore, we only consider generating convolutional kernels, bias, and fully connected weights. We denote the TargetNet for an N -way K -shot task \mathcal{T}_i as T_{θ_i} and the corresponding test set of \mathcal{T}_i as $S_i^{test} = \{(\hat{x}_i, \hat{y}_i)\}$ in accordance with the previous description. The final embeddings of the training samples $T_{\theta_i}(x_i^{n,k})$ and test samples $T_{\theta_i}(\hat{x}_i)$ are obtained through TargetNet. The probability attention kernel (Vinyals et al., 2016) is computed as follows:

$$a(\hat{x}_i, x_i) = \frac{e^{d(T_{\theta_i}(\hat{x}_i), T_{\theta_i}(x_i^{n,k}))}}{\sum_{n=1}^N \sum_{k=1}^K e^{d(T_{\theta_i}(\hat{x}_i), T_{\theta_i}(x_i^{n,k}))}}, \quad (5)$$

where $d(T_{\theta_i}(\hat{x}_i), T_{\theta_i}(x_i^{n,k}))$ is the cosine distance between test and training sample embeddings. The probability attention is then used to obtain the final probability score of the test sample, which is formulated as:

$$\hat{\mathbf{p}}_i = \sum_{n=1}^N \sum_{k=1}^K a(\hat{x}_i, x_i^{n,k}) \mathbf{y}_i^{n,k}. \quad (6)$$

We adopt cross-entropy loss to construct the final objective function between the predicted probability and the

groundtruth:

$$\mathcal{L}_{\mathcal{T}_i} = H(\hat{y}_i, \hat{p}_i). \quad (7)$$

3.6. Intertask Normalization

Previous meta-learning methods usually consider each task independently. However, similar tasks should share some useful information with each other, which can help meta-level learner to learn additional common prior knowledge. We propose an intertask normalization (ITN) strategy to make the tasks interact with each other in a batch of tasks. In practice, we directly apply batch normalization (Ioffe & Szegedy, 2015) on the embedding module and task context encoder. The normalization is applied to all training samples of a task batch, rather than just to samples of each individual task. The accumulated mean, variance and learned scale and shift parameters in BN incorporate the statistical information shared among tasks. During a testing phase, we independently apply the trained model on each individual unseen task.

4. Experiments

In this section, we conduct several experiments to evaluate and verify our proposed method. In Section 4.1, we apply LGM-Net on four synthetic datasets and visualize the decision boundary of generated TargetNet on unseen tasks. In Section 4.2, we perform few-shot image classification and compare with state-of-the-art methods on Omniglot and *mini*ImageNet datasets. In Section 4.3, we conduct ablation study to evaluate each component of our approach. In Section 4.4, we visualize the distribution of generated weights on different unseen tasks. We implement our algorithm and conduct the experiments using TensorFlow (Abadi et al., 2016). Our source code is available online².

4.1. Results on Synthetic Datasets

To begin with, we consider an experiment on synthetic datasets to show the intuition and testify the effectiveness of our method. As shown in Figure 2, we have designed four 2D synthetic datasets, i.e., *Blobs*, *Lines*, *Spirals*, and *Circles*. To formalize these datasets in the context of few-shot learning, we consider blobs in different positions, lines along different angles, different spirals arms, as well as concentric circles with different radius as different categories in each dataset. The data points of the same color belong to the same category. Each dataset contains 100 categories with 20 samples per category. We randomly select 80 categories as meta training dataset and the remaining 20 categories as meta test dataset.

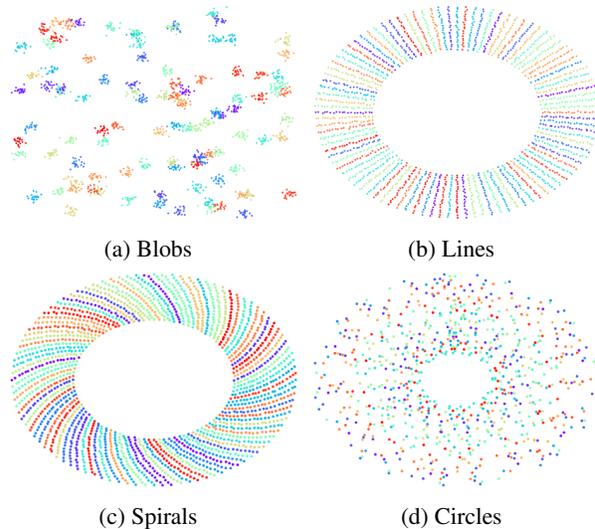


Figure 2. Visualization of four synthetic datasets. The blobs, lines, spirals, and circles in the same color belong to the same category. Each dataset contains 100 categories with 20 samples per category. This figure is best viewed in electronic version.

We can directly feed the training set of a task into MetaNet and generate the weights of TargetNet because the synthetic datasets are 2D. In our experiments, the task context encoder consists of a multilayer perceptron (MLP) with two hidden layers of 8 units and a ReLU activation function. The computational structure of TargetNet includes an MLP with three hidden layers of 16, 12, and 8 units and ReLU activation layers. For better visualization, we train the *Blobs*, *Lines*, and *Spirals* datasets on 5-way 1-shot learning tasks but train the *Circles* dataset on 3-way 1-shot learning tasks.

Figure 3 shows the decision boundaries of TargetNet on the aforementioned four synthetic examples using three different approaches. The learning difficulties gradually increase from top to bottom in the figure. From left to right, in the first column, we plot the decision boundaries of TargetNet with random initialized weights for an unseen task \mathcal{T}_i of each synthetic data. The decision boundaries are in disorder. In the second column, we directly train TargetNet on the training set of task \mathcal{T}_1 using Adam optimization (Kingma & Ba, 2015). The results show that, the directly trained TargetNet cannot effectively generalize to the test samples well even though it can correctly classify the training samples. Most notably, the decision boundaries for the task from *Circles* show that the directly trained TargetNet misclassifies most of the training samples. This fact is due to the traditional gradient descent algorithms that only consider training samples and contain no any prior knowledge about the specific task. These algorithms usually overfit the model with limited training samples. In the third column, we show the decision boundaries of TargetNet

²<https://github.com/likesiwell/LGM-Net/>

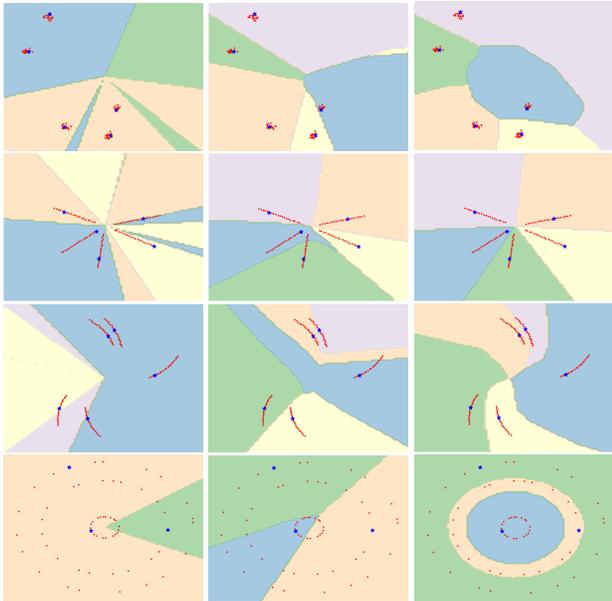


Figure 3. The decision boundaries of TargetNet on four synthetic datasets (from top to bottom: *Blobs*, *Lines*, *Spirals*, and *Circles*). The blue star points denote the training set of a selected task \mathcal{T}_i from the meta testing dataset. The red points are the test samples of \mathcal{T}_i . In each row, from left to right, we show the decision boundaries of TargetNet from randomly initialized weights, directly trained weights using Adam optimization, as well as the weights generated by our MetaNet module.

with generated weights from the learned MetaNet module based on \mathcal{T}_i . The TargetNet using generated weights can effectively generalize on test samples on the selected task. Our approach for all the four datasets can achieve over 99% accuracy on unseen tasks.

The experiments on synthetic examples demonstrate the learning mechanism of LGM-Net. The decision boundaries on different dataset scenarios show that the MetaNet module learns to understand different task scenarios. The MetaNet can directly generate functional weights for the TargetNet for a similar unseen task. However, the TargetNet with directly trained weights fails to generalize well, because the SGD algorithms do not contain any prior knowledge for learning similar scenarios. The comparison results imply that the MetaNet learns the transferable prior knowledge about how to solve tasks for a certain scenario. Even when the unseen task contains few examples, the MetaNet can still generate effective functional weights for TargetNet.

4.2. Few-shot Classification Results

We conduct few-shot classification experiments on two commonly used benchmarks, namely, Omniglot (Lake et al., 2011) and *miniImageNet* (Vinyals et al., 2016). Adam optimization (Kingma & Ba, 2015) is applied for training,

with an initial learning rate of 10^{-3} which is reduced by 10% every 1500 batches. The models are trained end-to-end from scratch without any additional dataset.

4.2.1. RESULTS ON OMNIGLOT DATASET

The Omniglot dataset consists of 1623 characters (classes) with 20 samples for each class from 50 different alphabets. Following (Vinyals et al., 2016; Snell et al., 2017), we randomly select 1200 classes as meta training dataset and use the remaining 423 classes as meta test dataset. The input images are resized to a resolution of 28×28 and augmented by random rotations of 90, 180, or 270 degrees.

In this experiment, we follow the architecture used by (Vinyals et al., 2016) to construct our modules. The embedding module has three 3×3 convolutional layers with 64 filters, followed by batch normalization, a ReLU activation, and 2×2 max-pooling. The TargetNet consists of two 3×3 convolutional layers with 64 filters which are generated by MetaNet. The task context encoder contains two 3×3 convolutional layers with 64 filters. During the test phase, we randomly select 8000 N-way K-shot tasks from meta test dataset for evaluation purpose. As shown in Table 1, we achieve comparable performance against state-of-the-art few-shot learning methods under different experiment settings. Since the performance scores are almost saturated on this dataset, the experimental results can still demonstrate the validity of our approach.

4.2.2. RESULTS ON *miniImageNet* DATASET

The *miniImageNet* dataset, originally proposed by (Vinyals et al., 2016), consists of 60,000 images from 100 selected ImageNet classes, each having 600 examples. We follow the split introduced by (Ravi & Larochelle, 2017), with 64, 16, and 20 classes for training, validation, and test, respectively. All the images are resized to a resolution of 84×84 and are augmented by random rotation of -45 , -22.5 , 22.5 , or 45 degrees.

On the *miniImageNet* benchmark, the results reported by different methods are obtained in different network configurations. Furthermore, different methods may require network configurations. Networks with deeper layers or more learnable parameters usually obtain better performance. However, for some methods such as (Finn et al., 2017; Ravi & Larochelle, 2017), they restrict the number of filters to alleviate overfitting. We construct the network architecture with six convolutional layers from image inputs to outputs, to make a relatively fair comparison with these methods. Hence, the embedding module has four 3×3 convolutional layers with 64 filters, followed by batch normalization, a ReLU nonlinearity, and 2×2 max-pooling. The TargetNet consists of two 3×3 convolutional layers with 64 filters without BN and a global average

Table 1. Mean accuracy of our LGM-Net and state-of-the-art methods on Omniglot dataset.

Model	5-way 1-shot	5-way 5-shot	20-way 1-shot	20-way 5-shot
Siamese Net (Koch et al., 2015)	97.3%	98.4%	88.1%	97.0%
Neural Statistician (Harrison Edwards, 2017)	98.1%	99.5%	93.2%	98.1%
Meta Nets (Munkhdalai & Yu, 2017)	99.0%	-	97.0%	-
Prototypical Nets (Snell et al., 2017)	98.8%	99.7%	96.0%	98.9%
MAML (Finn et al., 2017)	98.7%	99.9%	95.8%	98.9%
Meta-SGD (Li et al., 2017)	99.5%	99.9%	95.9%	99.0%
Relation Net (Sung et al., 2018)	99.6%	99.8%	97.6%	99.1%
Matching networks (Vinyals et al., 2016)	98.1%	98.9%	93.8%	98.5%
LGM-Net (Ours)	99.0%	99.4%	96.5%	98.5%

Table 2. Mean accuracy \pm 95% confidence intervals of our LGM-Net and state-of-the-art methods on *miniImageNet* dataset.

Model	5-way 1-shot	5-way 5-shot	20-way 1-shot
Matching networks (Vinyals et al., 2016)	43.56 \pm 0.84%	55.31 \pm 0.73%	17.31 \pm 0.22%
Meta-LSTM (Ravi & Larochelle, 2017)	43.44 \pm 0.77%	60.60 \pm 0.71%	16.70 \pm 0.23%
MetaNet (Munkhdalai & Yu, 2017)	49.21 \pm 0.96%	-	-
Prototypical Nets (Snell et al., 2017)	49.42 \pm 0.78%	68.20 \pm 0.66%	-
MAML (Finn et al., 2017)	48.70 \pm 1.84%	63.11 \pm 0.92%	16.49 \pm 0.58%
Meta-SGD (Li et al., 2017)	50.47 \pm 1.87%	64.03 \pm 0.94%	17.56 \pm 0.64%
Relation Net (Sung et al., 2018)	51.38 \pm 0.82%	67.07 \pm 0.69%	-
REPTILE (Nichol & Schulman, 2018)	49.97 \pm 0.32%	65.99 \pm 0.58%	-
SNAIL (Mishra et al., 2018)	55.71 \pm 0.99%	65.99 \pm 0.58%	-
(Gidaris & Komodakis, 2018)	56.20 \pm 0.86%	73.00 \pm 0.64%	-
LEO(Rusu et al., 2019)	61.76 \pm 0.08%	77.59\pm 0.12%	-
LGM-Net (Ours)	69.13\pm0.35%	71.18\pm0.68%	26.14\pm0.34%

pooling is appended at last. We compare with other related methods in similar network settings. The results are shown in Table 2. Our LGM-Net achieves state-of-the-art performance and significantly improves by up to 8% on 5-way 1-shot learning problem. The success of our approach lies in the transferrable prior experience learned by the MetaNet. In contrast to the baseline matching networks in which the weights are fixed for unseen tasks, the weights in our TargetNet are dynamically generated and the transferable prior knowledge is quickly adapted to new tasks. Furthermore, compared to alternative methods such as MAML (Finn et al., 2017), Meta-LSTM (Ravi & Larochelle, 2017), and Meta-SGD (Li et al., 2017), our representation of transferable prior knowledge in the way of generating functional weights is more effective than learning a good initialization or a parameter optimizer.

4.3. Ablation Study

We perform an ablation study with detailed results in Table 3 to evaluate the effects of each component in our algorithm. To ensure a fair comparison, we reimplement a matching networks with six 3×3 convolution blocks with 64 filters as the baseline. Additionally, our LGM-Net has the same computational structure, but the weights of the last two layers are generated by MetaNet.

We compare matching networks and LGM-Net both with and without using ITN. As shown in Table 3, using ITN significantly improves the performance. If we train LGM-Net without task context encoder (TCE), then it is equivalent to the matching Network whose last two layers are based on a weight generator of a random prior. No evident difference is found in the test performance. However, the LGM-Nets with TCE show a noticeable improvement. This fact indicates that TCE helps generate better weights for TargetNet, rather than just increasing the learning parameters in the entire model. If we train LGM-Net without WN, then it slightly decreases the performance. We have found that without using WN, the features in TargetNet are usually of large scale which will have negative impact on the training process. If we train LGM-Net directly using task context feature to generate the weights rather than formulating the weight generator as sampling from a reparameterized multivariate Gaussian distribution, then the performance will be worse, which shows that randomness helps improve the performance. Moreover, we can use the weight generator to directly generate several weight points for a single task. These generated weights can directly form an ensemble model with a max voting algorithm. However, this approach leads to no evident improvement and only reduces variance.

Table 3. Ablation Study on the *miniImageNet* dataset.

Model	5-way 1-shot	5-way 5-shot
matching networks(w/o ITN)	44.98±0.88%	55.74±0.75%
matching networks(w ITN)	57.42±0.68%	59.65±0.83%
LGM-Net (w/o TCE)	57.91±0.41%	59.89±0.48%
LGM-Net (w/o ITN)	65.42±0.65%	67.93±0.56%
LGM-Net (w/o randomness)	67.25±0.42%	69.68±0.55%
LGM-Net (w/o WN)	68.85±0.41%	69.97±0.45%
LGM-Net (Plain)	69.13± 0.35%	71.18±0.68%
LGM-Net (Ensemble)	69.15± 0.33%	71.18±0.64%

4.4. Exploring Generated Weight Distribution

In this experiment, the property of the MetaNet module is explored. We select five 5-way 1-shot learning tasks from the meta test dataset. For these tasks, \mathcal{T}_1 , \mathcal{T}_2 , and \mathcal{T}_3 contain the same training samples but in different orders, whereas \mathcal{T}_4 and \mathcal{T}_5 are another two different tasks. Figure 4 shows the t-SNE (Maaten & Hinton, 2008) visualization of the generated functional weight points from the MetaNet conditioned on the selected tasks. For each task, we generate 12 functional weight points from the MetaNet, which are marked in the same color. As shown in Figure 4, the functional weight points from the same task are clustered together. Similar generalization ability is exhibited on the test set for the corresponding task. The weight point clusters of \mathcal{T}_1 , \mathcal{T}_2 , and \mathcal{T}_3 are overlapped together but far away from the clusters of \mathcal{T}_4 and \mathcal{T}_5 . This result indicates that the generated functional weights from MetaNet are not influenced by the order of training data, which is a desired property of our task context encoder. Furthermore, different tasks have distinct distributions of functional weights.

5. Conclusions

Inspired by the ability of human beings to quickly learn from one task and adapt to a similar unseen task, we propose a novel meta learning approach, namely LGM-Net, for few-shot learning. Our key idea is to learn a weight generation network across a large amount of tasks to produce functional weights for TargetNet based on the training data. We choose Matching Nets as the computational structure of TargetNet and design a task context encoder and a weight generator as MetaNet. Compared with recent meta learning algorithms, LGM-Net is simpler and more efficient since it neither contains complicated structures nor needs further fine-tuning. It can learn transferable prior knowledge and enables fast adaptation to new tasks with limited data.

If we leave out the MetaNet module in LGM-Net, the computation pipeline is similar to convolutional siamese nets (Koch et al., 2015) and matching networks (Vinyals et al., 2016). The main difference among the three models lies in the learned parameters of the non-linear mapping

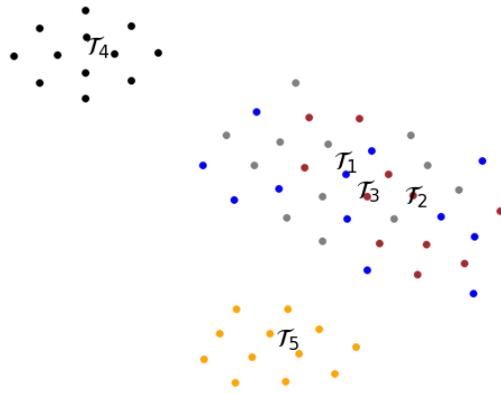


Figure 4. Distributions of generated weights for different tasks.

function from the input to the embeddings. The siamese nets are trained on paired inputs with contrastive loss. The matching networks are trained on few-shot tasks with attentional metric loss. The learned parameters in their nonlinear mapping functions remain unchanged even for different unseen tasks and therefore are difficult to adapt. However, the parameters in our LGM-Net are adaptive to different task samples. Hence, we can get better mapping function for new tasks.

The proposed approach has achieved significant improvement on 5-way 1-shot learning tasks on *miniImageNet* dataset. However, on 5-way 5-shot learning problems, the performance improvement over 5-way 1-shot is not as obvious as other methods. This outcome is due to the straightforward design of task context encoder which computes the statistical mean of training sample features as the task representation. Although the task context encoder is simple and effective on 1-shot learning, it may not provide sufficient information with few-shot samples, thereby leading to limited improvement on few-shot learning tasks. Hence, designing an effective task context encoder will be one of future work.

Finally, LGM-Net, like other meta learning approaches, needs to be advanced towards explainable AI (XAI) (Tickle et al., 1998; Doshi-Velez & Kim, 2017; Murdoch et al., 2019) or transparent AI (Hu et al., 2007). These meta learning methods extract prior knowledge, embed it into meta-level learner, and help base-level learner for solving novel tasks. However, one critical question may arise, i.e., how can we interpret the prior knowledge embedded in meta-level learners and can we believe that it will guarantee better performance than conventional methods? Therefore, it is necessary to design more explainable meta-level learners which can represent prior knowledge in more transparent forms and help us understand how to leverage the embedded prior knowledge to solve new tasks. The related challenges can be another interesting direction for future work.

Acknowledgements

This work was supported by National Key R&D Program of China under no. 2018YFC0807500, National Natural Science Foundation of China under nos. 61832016, 61720106006 and 61672520, as well as CASIA-Tencent Youtu joint research project.

References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.
- Andrychowicz, M., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., and de Freitas, N. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pp. 3981–3989, 2016.
- Ba, J., Hinton, G. E., Mnih, V., Leibo, J. Z., and Ionescu, C. Using fast weights to attend to the recent past. In *Advances in Neural Information Processing Systems*, pp. 4331–4339, 2016.
- Doshi-Velez, F. and Kim, B. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.
- Feizi, S., Javadi, H., Zhang, J., and Tse, D. Porcupine neural networks:(almost) all local optima are global. *arXiv preprint arXiv:1710.02196*, 2017.
- Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 1126–1135, 2017.
- Gidaris, S. and Komodakis, N. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4367–4375, 2018.
- Harrison Edwards, A. S. Towards a neural statistician. In *In International Conference on Learning Representations (ICLR)*, 2017.
- He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of IEEE International Conference on Computer Vision*, pp. 1026–1034, 2015.
- Hinton, G. E. and Plaut, D. C. Using fast weights to deblur old memories. In *Proceedings of the 9th Annual Conference of the Cognitive Science Society*, pp. 177–186, 1987.
- Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- Hu, B.-G., Wang, Y., Yang, S.-H., and Qu, H.-B. How to add transparency to artificial neural networks. *Pattern Recognition and Artificial Intelligence*, 20(1):72–84, 2007.
- Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 448–456, 2015.
- Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.
- Koch, G., Zemel, R., and Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.
- Lake, B., Salakhutdinov, R., Gross, J., and Tenenbaum, J. One shot learning of simple visual concepts. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, pp. 2568–2573, 2011.
- Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.
- Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pp. 6389–6399, 2018.
- Li, Z., Zhou, F., Chen, F., and Li, H. Meta-SGD: Learning to learn quickly for few shot learning. *arXiv preprint arXiv:1707.09835*, 2017.
- Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.
- Maaten, L. v. d. and Hinton, G. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.
- Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. In *International Conference on Learning Representations (ICLR)*, 2018.

- Motiian, S., Jones, Q., Iranmanesh, S., and Doretto, G. Few-shot adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pp. 6670–6680, 2017.
- Munkhdalai, T. and Yu, H. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 2554–2563, 2017.
- Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*, 2019.
- Nichol, A. and Schulman, J. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2018.
- Qiao, S., Liu, C., Shen, W., and Yuille, A. L. Few-shot image recognition by predicting parameters from activations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7229–7238, 2018.
- Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.
- Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pp. 91–99, 2015.
- Ruder, S. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- Rusu, A. A., Rao, D., Sygnowski, J., Vinyals, O., Pascanu, R., Osindero, S., and Hadsell, R. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations (ICLR)*, 2019.
- Salimans, T. and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 901–909, 2016.
- Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pp. 4077–4087, 2017.
- Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. 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.
- Thrun, S. and Pratt, L. *Learning to learn*. Springer Science & Business Media, 2012.
- Tickle, A. B., Andrews, R., Golea, M., and Diederich, J. The truth will come to light: directions and challenges in extracting the knowledge embedded within trained artificial neural networks. *IEEE Transactions on Neural Networks*, 9(6):1057–1068, 1998.
- Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pp. 3630–3638, 2016.
- Weiss, K., Khoshgoftaar, T. M., and Wang, D. A survey of transfer learning. *Journal of Big Data*, 3(1):9, 2016.
- Wu, T., Peurifoy, J., Chuang, I. L., and Tegmark, M. Meta-learning autoencoders for few-shot prediction. *arXiv preprint arXiv:1807.09912*, 2018.