Calibrating CNNs for Few-Shot Meta Learning

Peng Yang, Shaogang Ren, Yang Zhao, Ping Li
Cognitive Computing Lab
Baidu Research
10900 NE 8th St. Bellevue, WA 98004, USA
{pengyang5612, renshaogang, yangzhao.eric, pingli98}@gmail.com

Abstract

Although few-shot meta learning has been extensively studied in machine learning community, the fast adaptation towards new tasks remains a challenge in the few-shot learning scenario. The neuroscience research reveals that the capability of evolving neural network formulation is essential for task adaptation, which has been broadly studied in recent meta-learning researches. In this paper, we present a novel forward-backward meta-learning framework (FBM) to facilitate the model generalization in few-shot learning from a new perspective, i.e., neuron calibration. In particular, FBM models the neurons in deep neural network-based model as calibrated units under a general formulation, where neuron calibration could empower fast adaptation capability to the neural network-based models through influencing both their forward inference path and backward propagation path. The proposed calibration scheme is lightweight and applicable to various feed-forward neural network architectures. Extensive empirical experiments on the challenging few-shot learning benchmarks validate that our approach training with neuron calibration achieves a promising performance, which demonstrates that neuron calibration plays a vital role in improving the few-shot learning performance.

1. Introduction

Learning to fast adaptation is an essential ability of human intelligence, with which humans can draw inferences about new tasks by effectively leveraging their previously acquired knowledge and prior learned skills. In stark contrast, it is a challenging task for artificial learning systems to effectively deal with new tasks with limited amounts of inputs. For example, the training of deep neural network models relies heavily on large-scale labeled training data in order to generalize well on the targets. To address this challenge, transfer learning [29] has been proposed exploiting source tasks to train a base-learner and then fine-tune the model to learn target task through knowledge transfer. Specifically, multi-task learning [5] is regarded as a form of inductive transferring learner. The inductive bias is distilled by the auxiliary tasks and then derives the hypothesis of the targets. Nonetheless, aforementioned techniques require strong assumptions over the training model. For instance, a common form of inductive bias in multi-task learning is ℓ_1 regularization that results in a sparse solution. While these approaches can partially facilitate knowledge adaptation, the biased learner toward a specific hypothesis degrades model generalization. This motivates us to empower generalization capability to the model from a different perspective, such as learning an effective strategy of fast adaptation in meta-learning schemes.

Recent meta-learning algorithms [10, 9] have studied how deep neural network-based models distill the knowledge of previous learned tasks and then adapt to new tasks with only a few examples [41, 8]. Recent studies have adopted distinct perspectives of neural network training process to facilitate fast adaptation in few-shot learning scenario. For example, structure-based models [22, 38] generally learn the update direction and step-size for individual tasks via operating task-specific parameterization of neural model. In contrast, context-based models [25, 32] typically guide the learning process via exploiting context input of specifical tasks. Despite their promising properties under certain circumstances, these algorithms suffer from two drawbacks: a) learning framework requires a bunch of similar tasks to maximize the generalization of multiple tasks; b) training input is typically optimized by the fix-structured feature extractor, which might be prone to over-fitting in few-shot learning scenarios.

This work tackles the few-shot learning problem from a novel view that is distinct to all the aforementioned attempts, i.e., seeking a better model generalization with neuron calibration. Specifically, we refer to neuron calibration as a process of mathematically adjusting the transformation functions in the layers of neural networks [11]. Considering that task adaptation in contemporary deep learning-based model is closely related to the plasticity of deep neural networks, our proposed neural calibration approach aims to regularize the learning process to facilitate task adaptation via posing

a trainable soft mask on both the latent feature and model parameter, which thence influences the feed-forward inference pass and the backward propagation pass simultaneously. This work is inspired by the earlier work that seeks optimal model generalization through calibrating neural network parameters or labels [36], as well as a recent continual learning work that learns feature calibration via retaining task-specific embedding [30]. Compared to those works, we attempt to solve few-shot learning problem with a general calibration formulation in meta-learning schema [4]. Instead of reserving task-specific parameters for preserving task knowledge, we learn the task-agnostic calibration formulation that could be applied to various feed-forward neural network architectures. The contributions of our work are three-fold:

- We introduce a general and light-weight neuron calibration approach to tackle few-shot learning problems with feed-forward deep neural network-based function approximation.
- A novel meta-learning paradigm is formulated to train the calibrated model with a bi-level optimization scheme to achieve a better generalization in few-shot learning scenarios.
- Extensive empirical experiments on the challenging few-shot learning benchmarks demonstrate the effectiveness of the proposed algorithms with significant margins on all the evaluation datasets.

2. Related Work

It is a challenging task to design an effective model in few-shot learning scenario [25], since the distributions of the new tasks are unknown in advance and there are only few-shot instances available in each task. Meta-learning has been introduced to tackle this issue from two distinct perspectives of deep neural network learning: 1) *Metric-based* approaches aim to learn a suitable embedding space with a neural network so that nearest neighbor classification works well given a metric in this space [41, 37, 40, 28, 33]. 2) *Gradient-based* approaches [8, 25, 31, 45], e.g., MAML, search for the optimal model parameterization across tasks such that fine-tuning from the base learner leads to fast adaptation to new tasks. The two research have critical differences: 1) Gradient-based methods usually exploit a linear model with classification loss objectives (e.g., cross entropy), while metric-based methods aim to optimizing the representation for each class and utilize a nearest neighbor classifier for label prediction according to the metric similarity between an input and class representation. 2) Metric-based methods are known to be less prone to over-fitting and perform better than gradient-based methods (due to biased classifier [15]). Although gradient-based approaches are more challenging to study, optimization-based technique is more widely applicable to different scenarios than metric-based, e.g., reinforcement learning [42, 24].

Our study focuses on the optimization-based meta-learning, where there are generally two research categories: 1) context-based approaches train a recurrent neural network with the temporal context and adjust parameter optimization with meta-learning schema [23, 1, 32]; 2) structure-based approaches [22, 38] introduce the task-specific parameterization into the transformation function of neural networks to facilitate knowledge transfer. It is worth mentioning two inspiring related approaches: *Meta Transfer Learning* [38], which calibrates the transformation function in the *forward* inference path of neural network, and *Meta-SGD* [25], which calibrates the per-parameter learning rate in the *backward* learning pass. Nonetheless, either one of the algorithms devises a promising strategy from a different and isolated perspective of neuron calibration, and thus may perform insufficiently in few-shot learning problems.

In this paper, we introduce a general and light-weight neuron calibration formulation to tackle few-shot learning problems with deep neural network-based function approximation. To improve model generalization, we develop a novel forward-backward training schema, where neuron calibration is performed simultaneously on the forward inference pass and backward propagation pass. We formulate a meta-learning framework that alternates the learning of the calibrated model under an interleaved optimization scheme, which potentially generalizes well with greater effectiveness than many conventional approaches.

3. Elements of Meta Learning

We introduce the problem setup and notations of meta-learning, following three neural network adaptation skeletons on meta-learning [41, 32, 8, 28].

3.1. Meta-Learning for Few-shot Learning

The goal of few-shot meta-learning is to leverage the knowledge from a batch of tasks for fast adaptation onto the target tasks [32]. To achieve this goal, the training, validation and testing dataset have disjoint label spaces during the learning process. Follow the conventional setting of MAML [8], we define a batch of tasks for training $\{\mathcal{T}_t\}_{t=1}^{B}$. For each task $\mathcal{T}_t$, we acquire two separate data splits $\mathcal{D}_t^{tr}$ and $\mathcal{D}_t^{ts}$ for the purpose of the interleaved optimization in *meta-train phase*, which is formally given by

$$\begin{aligned} &\text{Outer Loop:} \quad \theta \leftarrow \theta - \alpha_{out} \nabla_\theta \sum_{t=1}^{B} \mathcal{L}(\widehat{\theta}_{(t)}, \mathcal{D}_t^{ts}) \\ &\text{Inner Loop:} \quad \text{s.t.} \quad \widehat{\theta}_{(t)} \leftarrow \theta - \alpha_{in} \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_t^{tr}) \end{aligned} \tag{1}$$

where α_{in} and α_{out} are learning rates for the inner-loop update and outer-loop update, respectively, and $\mathcal{L}(\theta, \mathcal{D})$ denotes the empirical loss function occurred by model θ on

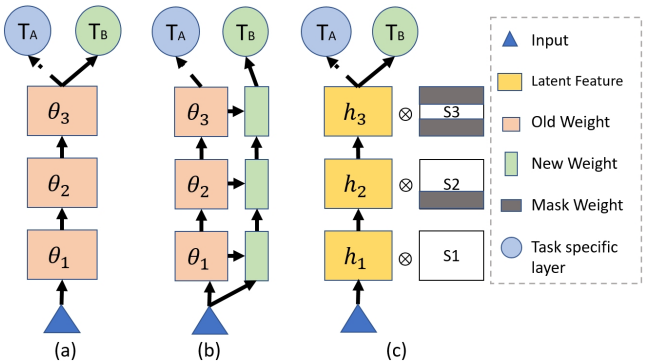

Figure 1. Illustration of three meta-learning approaches: (a) All tasks share a single set of model parameters that are optimized with single learning rules. (b) Each task will add additional task-specific parameters while original weights are frozen. (c) Our approach, where fast adaptation is achieved via calibrating a subset of model parameter and feature map in each layer of neural network. The figure shows that meta-learner decides to *re-use* the first layer of network and *re-scale* some parts of the following two layers.

the dataset $\mathcal{D}$. In Eq. (1), the goal of inner-loop update is to optimize the task-specific base-learner using data split $\mathcal{D}_t^{tr}$, and then temporal model $\widehat{\theta}_{(t)}$ learned on specific tasks is used to guide the optimization of the base-learner on disjoint data $\mathcal{D}_t^{ts}$ in the outer-loop update. Specifically, the base-learner is fine-tuned on the data split $\mathcal{D}_t^{tr}$ and its corresponding loss $\mathcal{L}(\theta, \mathcal{D}_t^{tr})$ is computed with model θ to derive an intermediate parameter $\widehat{\theta}_{(t)}$ for each task t. After that, the outer loop exploits temporal parameter $\widehat{\theta}_{(t)}$ to optimize the base-model θ upon the loss function $\mathcal{L}(\widehat{\theta}_{(t)}, \mathcal{D}_t^{ts})$ on test data $\mathcal{D}_t^{ts}$. In other word, the gradient to update θ is actually estimated by $\widehat{\theta}_{(t)}$. Given new tasks sampled from the unseen testing dataset, *meta-test phase* aims to evaluate the model adaptation capability via a limited number of the update on the trained model. So we can use the meta-test phase to implement the fine-tuning over the "pretrained" model learned on the meta-train phase.

3.2. Model Adaptation

As illustrated in Figure 1(a), one straightforward solution of meta-learning is to utilize a single set of model parameters θ for all tasks and optimize θ using gradient descent over a collection of tasks. Following this direction, the performance of new tasks is affected by the knowledge acquired from the previous learned tasks. Therefore, well-learned initial model parameters are essential for ensuring reasonable performance for new tasks [8]. Specifically, promising result would be guaranteed only if the previous learned experience was well transferred to new tasks. Nonetheless, there is no guarantee for training effectiveness on the initial model, since dissimilar tasks may disturb the training process.

Figure 1(b) presents an alternative way of few-shot learning via augmenting neural networks with external memory [41]. Generally, this method keeps previous learned model θ fixed when extending model parameter for new tasks, i.e., $\theta_{(t)} = \theta_{(t-1)} \cup \psi_t$, which memorizes the new knowledge with the new parameter ψ_t allocated for new task at time t. As introducing new ψ_t for each task is usually hand-crafted, current work often adopts simple heuristics, e.g., by adding extra channels on top of current network structure or adding extra support vectors into the dictionary. By doing this, the learner will suffer from expensive memory overhead and poor scalability. In addition, the isolated parameters will fail to exploit the relationship between the related tasks for achieving better generalization.

Our proposed technique is related to gradient-based meta-learning [8], the primary goal of which is to improve the generalization capability through posing efficient optimization strategies throughout the meta-learning process. Unlike prior related work [8] that exploits a shared model across all tasks, our work aims to seeking optimal calibration for neural-network based models through approximating neuron transformation function [13, 19, 36], as shown in Figure 1(c), so that the calibrated model could generalize well across different tasks. Motivated by recent approaches that successfully exploit neural calibration schema to address machine learning problem [19, 36], we attempt to tackle few-shot learning problem with neuron calibration. Instead of reserving task-specific parameters for preserving the knowledge of each task, we propose a task-agnostic calibration formulation, which boosts model generalization through influencing forward pass and backward pass of deep neural networks.

4. Calibrating CNNs

We introduce a general neural calibration framework for solving the few-shot learning problem on image classification, where the models are parameterized by feed-forward neural networks. By applying neural calibration, we aim to adapt the transformation functions in the deep neural network layers, with the hope that our proposed learning paradigm with neuron calibration could effectively achieve a stable consolidation of knowledge from multiple tasks and thus enable a fast adaptation towards new tasks. Specifically, in this work, we formalize the calibration of two common transformation operations in feed-forward deep neural networks: feed-forward inference and backward propagation optimization. Figure 2 provides an illustrative example of the forward-backward neural calibration procedure.

4.1. Forward Inference Calibration

We introduce a general forward inference calibration module on the neural network layers, i.e., feature calibration module. Before applying feature calibration module, the transformation function at the i-th layer $\mathcal{F}_{\theta_i}$ is parameterized by θ_i and produces the i-th layer feature map $h_i = \mathcal{F}_{\theta_i}(h_{i-1})$

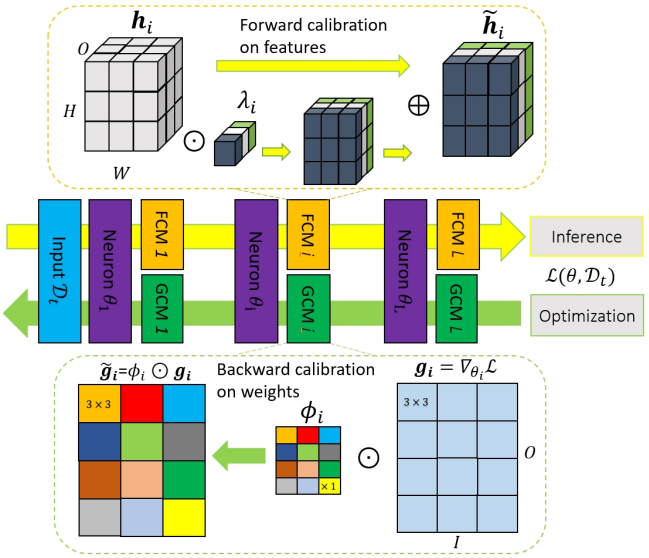

Figure 2. Overview of our proposed Forward-Backward Meta Calibration for Few-shot Learning framework. FBM consists of two types of calibration modules: feature calibration module (FCM) and gradient calibration module (GCM), which are sequentially applied to the layers in the base model (as shown in the figure) to calibrate the feature maps and model weights, respectively.

from the network transformation function. The neural network function before calibration is denoted as the *base* network. Then, the feature calibration module learns to scale the feature maps predicted by transformation function on each network layer. Let $\Omega_{\lambda_i}(\cdot)$ denote the feature calibration function employed by the i-th layer, parameterized by λ_i. The output features of the neuron transformation at each layer are processed by the feature calibration module to generate the calibrated feature map for that layer. Overall, the feature calibration unit is modularized by an element-wise multiplication operation, which is applied between the feature maps and the calibrator parameters. When calibrating the i-th layer of a neural network, we use $\widetilde{h}_i$ to denote the feature maps after applying feature calibration, i.e., $\widetilde{h}_i = \Omega_{\lambda_i}(h_i)$. Formally, the feature calibration function is defined in the following manner,

$$\Omega_{\lambda_i}(h_i) = \begin{cases} \text{tile}(\lambda_i) \odot h_i, \psi_i \in \mathbb{R}^O \text{ and } h_i \in \mathbb{R}^{H\times W\times O} \text{(Conv Layer)} \\ \lambda_i \odot h_i, \psi_i \in \mathbb{R}^O \text{ and } h_i \in \mathbb{R}^O \quad \text{(FC Layer)} \end{cases} \tag{2}$$

where $\odot$ denotes element-wise multiplication, O denotes the number of channel, and (H, W) the resolution of output feature maps. To reduce the size of the calibrator parameters for efficient computation, we specify the calibrator size to be much smaller than the original size of h_i. The calibration for the *convolutional* layers and *fully connected* layers are specified to work at per-channel-level and per-feature-level, respectively. To scale up the shape of the calibrator parameters λ_i to match that for h_i, a tile$(\cdot)$ function is applied on the calibrator parameter λ_i. With the aforementioned feature calibration approach, the calibrator module plays a crucial role during the model training process: at the forward inference path, it scales the value of the feature maps from the base network to regularize the transformation function; at the backward optimization path: it serves as a prioritized weight to regularize the model update (e.g., $\bigtriangledown_{\theta_i}\mathcal{L}$ is derived in the form of $\bigtriangledown_{\widetilde{h}_i}\mathcal{L} \cdot \bigtriangledown_{\theta_i}(h_i \odot \text{tile}(\lambda_i))$, scaled by λ_i).

In the end, the original feature map and calibrated one from (2) get added up in an element-wise manner by a residual connection. This is followed by normalization and activation operations to produce a final output for that layer. In summary, the overall calibration process for the i-th layer could be formulated as follows,

$$h_i = \sigma\left(\mathcal{BN}\left(\Omega_{\lambda_i}\left(\mathcal{F}_{\theta_i}\left(h_{i-1}\right)\right) \oplus \mathcal{F}_{\theta_i}\left(h_{i-1}\right)\right)\right), \tag{3}$$

where $\mathcal{BN}(\cdot)$ denotes the batch normalization, $\oplus$ denotes an element-wise addition operator, and $\sigma(\cdot)$ is an activation function. After that, h_i is sent as input to the $i+1$-th layer in the feed-forward neural network. All the aforementioned calibrator parameters are initialized with a value of 1.0 at the start of training. We demonstrate an example case of applying feature calibration on a CNN-based model in Figure 2.

Discussion: For the neural network model, the role the calibration module performs on the depth adapts to the needs of the feature transformation. In the early layers, it learns to activate informative features in a label-agnostic manner, strengthening the quality of the basic-level feature maps. In the following layers, the feature map becomes abstract and specialized for the specific tasks. The feature calibration module poses a trainable soft mask on the feature maps, which then influences the model inference process through the forward inference path. In addition, the proposed forward calibration module is model-agnostic, and could be deployed into various convectional neural network, e.g., VGG [35], ResNet-12 [14], Inception-V3 [39], etc.

4.2. Backward Optimization Calibration

We introduce a general module of backward optimization calibration, i.e., gradient calibration module, which aims to scale the gradients of the parameters in back-propagation step. Specifically, we denotes the gradient as $\mathbf{g}_i = \nabla_{\theta_i}\mathcal{L}$, where $\mathcal{L}$ is the loss function and θ_i is the model parameter at layer i. Let $\Omega_{\phi_i}(\cdot)$ denote the gradient calibration function applied on the i-th layer of the network, which is parameterized by ϕ_i. Similar with forward inference calibration, the gradient calibration operation is formalized by the element-wise multiplication between the gradients and the calibrator

parameters, which is defined as,

$$\Omega_{\phi_i}(\mathbf{g}_i) = \begin{cases} \text{tile}(\phi_i) \odot \mathbf{g}_i, & \phi_i \in \mathbb{R}^{O \times I} \quad \text{(Conv Layer)}, \\ \text{tile}(\phi_i) \odot \mathbf{g}_i, & \phi_i \in \mathbb{R}^{O} \quad \text{(FC Layer)}, \end{cases}$$

where $\odot$ is the element-wise product between matrices of the same shape. The soft mask ϕ_i proposes to carry out adaptive optimization through calibrating gradient descent of parameters on each layer of transformation function. Similarly, calibration for the convolutional layers and fully connected layers is specified to work at per-channel-level and per-feature-level, respectively.

By applying the backward optimization calibration on backward propagation path, the gradient descent rule at the i-th layer could be formulated as follows,

$$\theta_i \leftarrow \theta_i - \alpha \cdot \widetilde{\mathbf{g}}_i, \quad \text{s.t.} \quad \widetilde{\mathbf{g}}_i = \Omega_{\phi_i}(\mathbf{g}_i)$$

where $\alpha > 0$ is the learning rate, $\mathbf{g}_i$ and $\widetilde{\mathbf{g}}_i$ are denoted as the gradients before and after applying gradient calibration, respectively. The calibrator parameters in gradient calibration module are initialized with binary value of 1 at the beginning of training. Specifically, the probability of each ϕ_i being 1 is parameterized based on $\phi_i \sim \sigma(m_i) = \mathcal{B}\left(\frac{\exp(m_i)}{\sum_i \exp(m_i)}\right)$ where $\mathcal{B}$ denotes the Bernoulli sampling. We can approximately differentiate the Bernoulli sampling of activation via the Gumbel-Softmax estimator [16]:

$$\phi_i \leftarrow \left(\frac{\exp(\frac{m_i+g_i}{\tau})}{\sum_i \exp(\frac{m_i+g_i}{\tau})}\right), \quad g_i \sim \mathsf{Gumbel}(0,1), \quad (4)$$

where $\tau > 0$ is a temperature hyperparameter. This reparameterization allows us to backpropagate through binary activation directly. At the limit of $\tau \approx 0$, Eq. (4) follows the behavior of one-hot vector. Each logit determines whether to activate the gate of a filter or not. The final output is obtained by blocking the original gradient with the activation. We illustrate an example case of applying backward calibration operation on a convolutional layer in Figure 2.

Discussion: Without losing generalization, backward optimization calibration could be applied for any convolutional layers of interest. Given a network with any depth of layers, backward calibration enables the adaptive update on each layer of neural network. The candidate choices for each layer include the *freeze* and *adapt*. The *freeze* choice (i.e., $\phi_i \to 0$) will freeze the parameters and prevent the gradient change of the weights, while the *adapt* choice (i.e., $\phi_i > 0$) will scale the parameter change.

5. Bi-level Learning Framework

We formulate the meta-learning procedure to train the calibrated model under an interleaved optimization scheme [10], with the parameters from the base model and those from the calibration module being optimized by their respective update rules. As shown in Algorithm 1, there are two loops:

Algorithm 1: Forward-Backward Meta Adaptive Learning (FBM)

Input: task distribution $p(\mathcal{T})$, model parameter θ, calibration parameters ϕ and λ, hyperparameters α_{in} and α_{out};
Initialize model θ and calibrator ϕ and λ;
while *not done* **do**
  Sample a batch of tasks $\mathcal{T}_t$ from $p(\mathcal{T})$;
  for *all* $\mathcal{T}_t$ **do**
    $f_{\theta_t}(\mathbf{x}, \lambda_t) \leftarrow \text{Softmax}(\Omega_{\lambda_t}(\mathcal{F}_{\theta_t}(\mathbf{x})))$;
    $\mathcal{L}_{tr(\mathcal{T}_t)} \leftarrow \frac{1}{|\mathcal{D}_t^{tr}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_t^{tr}} \ell(f_{\theta_t}(\mathbf{x}, \lambda_t), \mathbf{y})$;
    $\widehat{\theta}_t \leftarrow \theta_t - \alpha_{in} \cdot \Omega_{\phi_t}(\nabla_\theta \mathcal{L}_{tr(\mathcal{T}_t)})$;
    $\mathcal{L}_{te(\mathcal{T}_t)} \leftarrow \frac{1}{|\mathcal{D}_t^{te}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}_t^{te}} \ell(f_{\widehat{\theta}_t}(\mathbf{x}, \lambda_t), \mathbf{y})$;
  end
  $(\theta_{t+1}, \phi_{t+1}, \lambda_{t+1}) \leftarrow (\theta_t, \phi_t, \lambda_t) - \alpha_{out} \nabla_{(\theta,\phi,\lambda)} \mathcal{L}_{te(\mathcal{T}_t)}$;
end

$$\begin{aligned} &\text{Outer Loop: } (\theta_t^*, \phi_t^*, \lambda_t^*) = \arg\min_{(\theta,\phi,\lambda)} \mathcal{L}(\widehat{\theta}_t, (\phi_t, \lambda_t), \mathcal{D}_t^{ts}) \\ &\text{Inner Loop: s.t.} \quad \widehat{\theta}_t = \arg\min_{\theta} \mathcal{L}(\theta_t, (\phi_t, \lambda_t), \mathcal{D}_t^{tr}) \end{aligned} \quad (5)$$

where (ϕ^*, λ^*) denote the optimal calibrator parameters given the task-specific base parameter $\widehat{\theta}_t$. In the interleaved optimization, base model θ is used to initialize the learner for any new tasks, ϕ and λ are two calibration parameters that operate on the forward inference and backward optimization, respectively. At the inner loop, the base model is learned on the training data $\mathcal{D}_t^{tr}$ to fine-tune a task-specific parameter $\widehat{\theta}_t$. At the outer loop, we leverage $\widehat{\theta}_t$ to optimize the calibrator and base model via employing the loss function on test data $\mathcal{D}_t^{ts}$ in order to generalize well over multiple tasks. In both loops of optimization, $\mathcal{L}(\cdot)$ denotes the cross-entropy loss for image classification problem.

Optimizing the objective (5) involves explicit optimization of both model parameters and calibration parameters, which are task-agnostic. The learning process is explicitly taken into consideration in two distinct aspects of neuron operations: *forward inference* and *backward optimization*. The former aims to achieve the adaptive inference by calibrating feature map from each layer of transformation, while the latter proposes to carry out the adaptive optimization by scaling gradient descent on each layer of model parameter. During the interleaved optimization process, we fix the (ϕ, λ) and then take gradient with regard to θ as follows:

$$\widehat{\theta}_t \leftarrow \theta_t - \alpha_{in} \cdot \nabla_\theta \mathcal{L}(\theta_t, (\phi_t, \lambda_t), \mathcal{D}_t^{tr})$$

After that, we continue to optimize the calibrator and base model when the inference takes place with the intermediate

model parameters,

$$\phi_{t+1} \leftarrow \phi_t - \alpha_{out} \cdot \nabla_\phi \mathcal{L}(\widehat{\theta}_t, (\phi_t, \lambda_t), \mathcal{D}_t^{te})$$
$$\lambda_{t+1} \leftarrow \lambda_t - \alpha_{out} \cdot \nabla_\lambda \mathcal{L}(\widehat{\theta}_t, (\phi_t, \lambda_t), \mathcal{D}_t^{te})$$
$$\theta_{t+1} \leftarrow \theta_t - \alpha_{out} \cdot \nabla_\theta \mathcal{L}(\widehat{\theta}_t, (\phi_t, \lambda_t), \mathcal{D}_t^{te})$$

where α_{in} and α_{out} are learning rates for the inner loop update and outer loop update, respectively. By employing the calibrated parameterization of the neural network-based model and optimizing it with the interleaved learning scheme, our method could potentially improve the generalization capability in the few-shot learning problem with greater effectiveness than many conventional approaches. We present the detailed algorithm in Algorithm 1.

6. Experiments

We evaluate the proposed algorithm in terms of few-shot classification accuracy and model robustness. Below we describe the datasets and detailed settings, followed by a comparison to state-of-the-art methods and ablation study.

6.1. Datasets and Setting

The proposed method is evaluated on three datasets:

FC100 dataset [28], derived from CIFAR-100 [18], includes 100 classes of images with 600 images for each class. These classes are randomly partitioned into 60 classes for meta-training, 20 classes for meta-validation, and 20 classes for meta-testing.

Mini-Imagenet dataset [41] is a subset of data sampled from the Imagenet [6]. It includes 100 classes with 600 examples for each class. All image examples are cropped to 84×84 pixels and the whole dataset is randomly split into 64 training classes, 16 validation classes and 20 test classes.

Omniglot is a character recognition dataset [20]. It spans 1623 characters from 50 different alphabets with 20 images for each class. Following the setting in [34], 1200 characters for training and the remain ones for testing, and augment the dataset by randomly rotating multiple times of 90 degrees.

Architecture: Motivated by the related work [8, 38], we utilize two feature extractors: Conv-4 (4-layer CNNs) and ResNet-12, and both extractors exploit a FC-layer for the last layer's output. For Conv-4, the network structure, objective function and learning strategies follow the setting of MAML [8]; for ResNet-12, the learning setting and objective function follow the setting of MTL [38]. We show the model capacity in terms of trainable parameter sizes for different models in Table 1. The results demonstrate that our proposed calibration approach leads to moderate increases in parameter size compared to its corresponding backbone models for all the architectures. The parameter increase is approximately 10% $\sim$$11\%$ for the CNN-based models, and specifically that for the last FC-layer is much more smaller, i.e., much less than 1%.

Architecture	Backbone # Params	FBM # Params	Increase
Conv-4 (Ominiglot)	119,429	132,042	10.56%
Conv-4 (ImageNet)	32,645	35,946	10.11%
ResNet-12 (ImageNet)	7,994,565	8,866,501	10.91%
ResNet-12 (FC100)	7,994,565	8,866,501	10.91%

Table 1. Light-weight model complexity w.r.t parameter size for various network architectures. FBM results in moderate parameter increase over its backbone architectures in all testified datasets.

We utilize meta-training and meta-validation data to learn meta learner, and exploit meta-testing data to evaluate its performance. In N-way K-shot setting, we first randomly sample *N* classes from the training classes, and then randomly sample *K* instances for each class to build a task. Therefore, each iteration of meta-train uses *NK* labeled instances to do the classification. We tune the learning rate from $\{10^{-5}, 10^{-4} \cdots, 10^{-2}\}$ for both the inner loop α_{in} and outer loop α_{out}. Meanwhile, the inner update step is set to be $\{10, 15, 20\}$ on all datasets. All training and evaluation experiments are performed using TITAN X GPUs. The whole learning process takes around 1 GPU day.

6.2. Comparison Results

The experimental results on Omniglot are shown in Table 2, where the proposed technique achieves state-of-the-art performance. Specifically, FBM outperforms most of baselines on all evaluation metrics, which validates the efficiency of the proposed adaptive learning algorithm. In particular, FBM achieves a better performance than its backbone MAML with calibrated transformation function (e.g., 99.76% vs. 98.70%, and 96.45 vs. 95.8% in 5-way and 20-way 1-shot setting), which demonstrate that the proposed neuron calibration can improve model generalization in few-shot learning scenario.

The experimental results on MiniImagenet and FC100 are shown in Table 3 and Table 4, respectively. We deploy FBM on two state-of-the-art backbones, i.e., MAML and MTL. The results show that FBM outperforms original backbones on all benchmark datasets, achieving the new state-of-the-art performance on FC100 (i.e., 44.8% and 57.0% on the setting of 5-way 1-shot and 5-shot) and Omniglot (i.e., 99.76% and 99.93% on 5-way 1-shot and 5-shot). In specific, the FBM improves MTL by a large margin on FC100 (e.g., 44.8% vs 43.7% on 5-way 1-shot, 57.0% vs 55.4% on 5-way 5-shot). For MiniImagenet, FBM obtains a better or comparable accuracy to the baselines on same feature extractor (e.g., 61.38% vs 59.74% and 57.48% with ALFA and MTL on ResNet-12).

In this paper, we aim to design an effective optimization-based meta-learning technique. From the comparison result,

Models	Backbone	5-way		20-way	
		1-shot (%)	5-shot (%)	1-shot (%)	5-shot (%)
Neural Statistician [7]	Statistic	98.1	99.5	93.2	98.1
Memory Mod. [17]	LSTM	98.4	99.6	95.0	98.6
Hyper-representation [10]	Conv-5	98.6	99.5	95.5	98.4
MT-net [22]	Conv-5	99.10 ± 0.30	-	96.20 ± 0.40	-
SNAIL [26]	ResNet-12	99.07 ± 0.16	99.78 ± 0.09	97.64 ± 0.30	99.26 ± 0.18
Matching Networks [41]	Conv-4	98.1	99.7	93.8	98.5
ProtoNet [37]	Conv-4	97.4	99.7	96.0	98.9
mAP-SSVM [40]	Conv-4	98.6	99.6	95.2	98.6
MetaGAN + MAML [43]	Conv-4	99.10 ± 0.3	99.70 ± 0.21	96.40 ± 0.27	98.90 ± 0.18
Meta-SGD [25]	Conv-4	99.53 ± 0.26	99.92 ± 0.09	95.93 ± 0.38	$\mathbf{98.97 \pm 0.20}$
MAML [8]	Conv-4	98.70 ± 0.40	99.90 ± 0.10	95.80 ± 0.30	98.90 ± 0.20
FBM + MAML (Ours)	Conv-4	$\mathbf{99.76 \pm 0.25}$	$\mathbf{99.93 \pm 0.05}$	$\mathbf{96.45 \pm 0.36}$	98.96 ± 0.14

Table 2. Results on Omniglot data set on 5-way and 20-way with 1-shot and 5-shot accuracy (%).

Models	Backbone	5-way	
		1-shot (%)	5-shot (%)
Match Networks [41]	Conv-4	43.56 ± 0.84	55.31 ± 0.73
Meta-LSTM[32]	Conv-4	43.44 ± 0.77	60.60 ± 0.71
mAP-SSVM [40]	Conv-4	50.32 ± 0.80	63.94 ± 0.72
REPTILE [27]†	Conv-4	49.70 ± 1.83	65.91 ± 0.84
FLATIPUS [9]	Conv-4	50.13 ± 1.86	–
Hier-MAML [12]	Conv-4	49.40 ± 1.83	–
Meta-SGD [25]∗	Conv-4	49.10 ± 1.87	64.05 ± 0.84
MAML [8]	Conv-4	48.70 ± 1.84	63.11 ± 0.92
FBM + MAML(Ours)	Conv-4	50.62 ± 1.79	64.78 ± 0.35
SNAIL [26]	ResNet-12	55.71 ± 0.99	68.88 ± 0.92
L2F [3]+	ResNet-12	57.48 ± 0.49	74.68 ± 0.43
TADAM [28]	ResNet-12	58.50 ± 0.30	76.70 ± 0.30
ALFA [2]	ResNet-12	59.74 ± 0.49	$\mathbf{77.96 \pm 0.41}$
MTL [38]	ResNet-12	60.20 ± 1.80	74.30 ± 0.90
FBM + MTL(Ours)	ResNet-12	$\mathbf{61.41 \pm 1.87}$	76.11 ± 0.92

Table 3. Results on MiniImagenet data set on 5-way 1-shot and 5-shot accuracy (%). "*" denotes re-run results on the same training/test split used in our setting. "+" and "†" indicate the MiniImagenet result from [2] and [44], respectively.

Models	Backbone	5-way	
		1-shot (%)	5-shot (%)
ProtoNet [37]∗	Conv-4	35.3 ± 0.6	48.6 ± 0.6
MAML [8]	Conv-4	38.1 ± 1.7	50.4 ± 1.0
FOMAML [8]	Conv-4	37.7 ± 1.9	49.1 ± 1.0
iMAML [31]	Conv-4	38.4 ± 1.7	49.4 ± 0.8
REPTILE [27]	Conv-4	38.4 ± 1.9	50.5 ± 0.9
META-RKHS [44]	Conv-4	41.2 ± 2.2	51.5 ± 0.9
ProtoNet [37]†	ResNet-12	37.5 ± 0.6	52.5 ± 0.6
TADAM [28]	ResNet-12	40.1 ± 0.4	56.1 ± 0.4
MetaOptNet-SVM [21]	ResNet-12◇	41.1 ± 0.6	55.5 ± 0.6
MTL [38]	ResNet-12	43.7 ± 1.8	55.4 ± 0.9
FBM + MTL(Ours)	ResNet-12	$\mathbf{44.8 \pm 1.9}$	$\mathbf{57.0 \pm 1.0}$

Table 4. Results on FC100 dataset on 5-way with 1-shot and 5-shot accuracy (%). "∗" and "†" indicate results from [28] and [21], respectively. "ResNet-12◇" is a variant of ResNet with number of filters changed from (64,128,256,512) to (64,160,320,640) is trained with data augmentation [21].

Model	MiniImagenet	Omniglot
MAML	7.31	29.33
FBM + MAML	7.50	31.67

Table 5. Running time (GPU hours) comparison on two datasets

FBM outperforms gradient-based techniques significantly in both Conv-4 and ResNet-12 backbones. In addition, we conduct the running-time comparison on two benchmark datasets. The results in Table 5 show that FBM+MAML actually runs slightly more time than MAML on two datasets. Nonetheless, additional time overhead could be compensated for the performance improved by FBM.

6.3. Ablation Study

Controlling the Depth of Calibration: We start with analyzing the effectiveness of the calibration on different layers of neural network. We conduct the ablation study experiment with ResNet-12 and evaluate the model on MiniImageNet and FC100. Specifically, we froze the calibrator in specific layers of backbone and disabled their scaling functions, e.g., $\theta_{[1,2,3]}$ denotes that the calibrators from the layer 1 to layer 3 are frozen. The results in Table 6 show that the more layers with neuron calibration, the better performance the model achieves, which demonstrates the effectiveness of neuron

	MTL	FBM + MTL			
MiniImageNet	$\theta_{[1,2,3,4]}$	$\theta_{[1,2,3]}$	$\theta_{[1,2]}$	$\theta_{[1]}$	θ (Ours)
	60.2	60.4	60.6	61.0	**61.4**
	MTL	FBM + MTL			
FC100	$\theta_{[1,2,3,4]}$	$\theta_{[1,2,3]}$	$\theta_{[1,2]}$	$\theta_{[1]}$	θ (Ours)
	43.7	44.2	44.5	44.6	**44.8**

Table 6. Accuracy comparison with respect to the layers of adaptor on MiniImagenet and FC100 in the setting of 5-way 1-shot.

calibration in the few-shot learning. Based on this observation, we apply the neuron calibration to all layers of neural network in the following empirical experiment.

Effect of Calibration Components: We conduct ablation study to explore the impact of forward calibration module and backward calibration module, separately. Specifically, we evaluate the model equipped with either forward calibration (i.e., FM + MTL) or backward calibration (i.e., BM + MTL) and present the result in Table 7. We observe that equipped with forward inference calibration, the performance is apparently better than the original model where the update rule is essentially identical to the baselines except their difference in model inference architecture. In contrast, the model with backward calibration that updates partial parameters following the original proposed inference architecture results in the accuracy scores that are close to original model. Overall, the results demonstrate that calibrating on latent feature and model parameter is a promising direction.

	MTL	FM+MTL	BM+MTL	FBM+MTL
MiniImageNet	60.2	61.1	60.5	**61.4**
FC100	43.7	44.5	44.0	**44.8**

Table 7. Ablation study results on exploring calibration module properties on MiniImagenet and FC100 in the setting of 5-way 1-shot.

Effect of Inner-Loop Step: We make analysis on the fast adaptation capability of our method by varying the number of inner update steps. Specifically, we measure the model performance when trained for a specified number of inner-loop steps and report the results in Table 8. Regardless of the number of steps, the calibrated model consistently outperforms the original model that performs fast adaptation with various inner update steps.

Inner-loop Steps	5	10	15	20
MTL	38.23	40.50	41.21	43.69
FBM + MTL	**39.51**	**41.35**	**43.81**	**44.80**

Table 8. Accuracy (%) comparison between MTL and the proposed FBM on FC100 5-way 1-shot.

Effect of Activation Sparseness: In the Gumbel-softmax estimator, the hyperparameter τ determines the sparseness of activation. When $\tau \to 0$, the Gumbel-softmax computation smoothly approaches the operator $\arg\max$, and the activation ϕ_i approaches one-hot; when $\tau \to \infty$, the Gumbel-softmax distribution becomes identical to the categorical input distribution $\mathbf{m}$. Specifically, we study the impact of activation sparseness and set τ to be $\{0.1, 0.5, 1, 5, 10\}$, and run the algorithm under each value of τ. We show the comparison result in Figure 3.

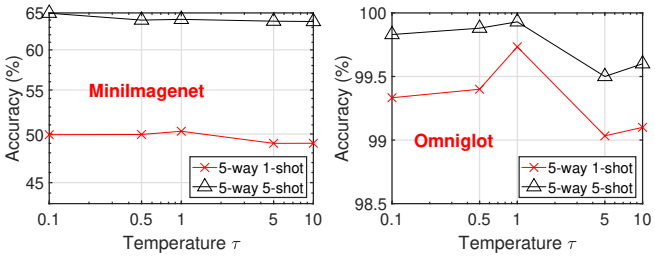

Figure 3. Ablation study on activation sparseness

We observe that the calibrated model maintains a better or comparable performance under various values of τ. We also observe that the performance on Omniglot has a slight drop with τ increased. The reason is that the calibrator with an increased τ allows the learner to update more model parameters. This may cause over-fitting in few-shot learning and harm the model generalization. The performance shows that the sparse activation on the neural structure is able to enhance the generalization with greater effectiveness.

7. Conclusion

This paper shows that the proposed framework trained with forward-backward meta optimization achieves promising performance for tackling few-shot learning problems. The calibration operation on the DNN neurons proves highly efficient for adapting the new task's learning experience. The superiority is mainly achieved in the extreme one-shot cases on three challenging benchmarks. In terms of learning schemes, the calibration learning shows consistently good performance in baseline comparison and ablation study. On the more challenging benchmark, it leads to being incredibly helpful for boosting generalization capability. This design is light-weighted and can be applied to various CNN-based model architectures. Moreover, the calibration formulation is task-agnostic and could be generalized well whenever the task's complexity could be to evaluate.

References

[1] Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3981–3989, Barcelona, Spain, 2016.

[2] Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. Meta-learning with adaptive hyperparameters. In *Advances in Neural Information Processing Systems (NeurIPS)*, virtual, 2020.

[3] Sungyong Baik, Seokil Hong, and Kyoung Mu Lee. Learning to forget for meta-learning. In *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2376–2384, Seattle, WA, 2020.

[4] Ondrej Bohdal, Yongxin Yang, and Timothy Hospedales. Meta-calibration: Meta-learning of model calibration using differentiable expected calibration error. *arXiv preprint arXiv:2106.09613*, 2021.

[5] Rich Caruana. Multitask learning. *Mach. Learn.*, 28(1):41–75, 1997.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, Miami, FL, 2009.

[7] Harrison Edwards and Amos J. Storkey. Towards a neural statistician. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1126–1135, Sydney, Australia, 2017.

[9] Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9537–9548, Montréal, Canada, 2018.

[10] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 1568–1577, Stockholmsmässan, Stockholm, Sweden, 2018.

[11] Charles D Gilbert and Wu Li. Top-down influences on visual processing. *Nature Reviews Neuroscience*, 14(5):350–363, 2013.

[12] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas L. Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[13] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, (ICML)*, pages 1321–1330, Sydney, Australia, 2017.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, 2016.

[15] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 831–839, Long Beach, CA, 2019.

[16] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[17] Lukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[18] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[19] Ananya Kumar, Percy Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3787–3798, Vancouver, Canada, 2019.

[20] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[21] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10657–10665, Long Beach, CA, 2019.

[22] Yoonho Lee and Seungjin Choi. Gradient-based meta-learning with learned layerwise metric and subspace. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pages 2933–2942, Stockholmsmässan, Stockholm, Sweden, 2018.

[23] Ke Li and Jitendra Malik. Learning to optimize. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, 2017.

[24] Yuxi Li. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

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

[26] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. A simple neural attentive meta-learner. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, Vancouver, Canada, 2018.

[27] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[28] Boris N. Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. TADAM: task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 719–729, Montréal, Canada, 2018.

[29] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2010.

[30] Quang Pham, Chenghao Liu, Doyen Sahoo, and Steven C. H. Hoi. Contextual transformation networks for online continual learning. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria, 2021.

[31] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. Meta-learning with implicit gradients. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 113–124, Vancouver, Canada, 2019.

[32] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the 5th International Conference on Learning Representations, (ICLR)*, Toulon, France, 2016.

[33] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, New Orleans, LA, 2019.

[34] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy P. Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of the 33nd International Conference on Machine Learning (ICML)*, pages 1842–1850, New York City, NY, 2016.

[35] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015.

[36] Pravendra Singh, Vinay Kumar Verma, Pratik Mazumder, Lawrence Carin, and Piyush Rai. Calibrating cnns for lifelong learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, virtual, 2020.

[37] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems (NIPS)*, pages 4077–4087, Long Beach, CA, 2017.

[38] 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 (CVPR)*, pages 403–412, Long Beach, CA, 2019.

[39] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Las Vegas, NV, 2016.

[40] Eleni Triantafillou, Richard Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens. In *In Advances in Neural Information Processing Systems (NIPS)*, pages 2255–2265, Long Beach, CA, 2017.

[41] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Advances in neural information processing systems (NIPS)*, pages 3630–3638, Barcelona, Spain, 2016.

[42] Zhongwen Xu, Hado P van Hasselt, and David Silver. Meta-gradient reinforcement learning. In *Advances in neural information processing systems (NeurIPS)*, pages 2402–2413, Montréal, Canada, 2018.

[43] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2371–2380, Montréal, Canada, 2018.

[44] Yufan Zhou, Zhenyi Wang, Jiayi Xian, Changyou Chen, and Jinhui Xu. Meta-learning with neural tangent kernels. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, Virtual Event, Austria, 2021.

[45] Luisa M. Zintgraf, Kyriacos Shiarlis, Vitaly Kurin, Katja Hofmann, and Shimon Whiteson. Fast context adaptation via meta-learning. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, pages 7693–7702, Long Beach, CA, 2019.