Efficient Meta-Learning for Continual Learning with Taylor Expansion Approximation

1st Xiaohan Zou
Boston University
zxh@bu.edu

2nd Tong Lin
Key Lab. of Machine Perception (MoE), School of AI,
Center for Data Science, Peking University
lintong@pku.edu.cn

Abstract—**Continual learning aims to alleviate catastrophic forgetting when handling consecutive tasks under non-stationary distributions. Gradient-based meta-learning algorithms have shown the capability to implicitly solve the transfer-interference trade-off problem between different examples. However, they still suffer from the catastrophic forgetting problem in the setting of continual learning, since the past data of previous tasks are no longer available. In this work, we propose a novel efficient meta-learning algorithm for solving the online continual learning problem, where the regularization terms and learning rates are adapted to the Taylor approximation of the parameter's importance to mitigate forgetting. The proposed method expresses the gradient of the meta-loss in closed-form and thus avoid computing second-order derivative which is computationally inhibitable. We also use Proximal Gradient Descent to further improve computational efficiency and accuracy. Experiments on diverse benchmarks show that our method achieves better or on-par performance and much higher efficiency compared to the state-of-the-art approaches.**

Index Terms—**meta-learning, continual learning**

I. INTRODUCTION

Catastrophic forgetting [1], [2] poses a major challenge to artificial intelligence systems: when switching to a new task, the system performance may degrade on the previously trained tasks. Continual learning is proposed to address this challenge, which requires models to be stable enough to prevent forgetting while being flexible to acquire new knowledge.

To alleviate catastrophic forgetting, several categories of continual learning methods have been proposed to penalize neural networks with regularization approaches by calculating the importance of weights [3], [4], to modify the architecture of neural networks [5], [6] and to introduce an episodic memory to store and replay the previously learned samples [7], [8].

The basic idea of rehearsal-based approaches like Gradient Episodic Memory (GEM) [7] is to ensure gradient-alignment across tasks such that the losses of the past tasks in episodic memory will not increase. Interestingly, this objective coincides with the implicit objective of gradient-based meta-learning algorithms [9]–[11]. Further, meta-learning algorithms show promise to generalize better on future tasks

[10], [12]. Meta Experience Replay (MER) [10] integrates the first order meta-learning algorithm Reptile [9] with an experience replay module to reduce interference between old and new tasks. To alleviate the slow training speed of MER, Lookahead-MAML (La-MAML) [11] proposes a more efficient meta-objective for online continual learning. La-MAML then incorporates learnable per-parameter learning rates to further reduce the catastrophic forgetting and achieved state-of-the-art performance.

However, though La-MAML proposes a more efficient objective, it still requires directly computing the Hessian matrix which is computationally inhibitable for large networks. Also, learning a learning rate for each parameter entails more computational overhead and increases memory usage.

To overcome these difficulties, in this paper, we present a novel efficient gradient-based meta-learning algorithm for online continual learning. The proposed method solves the meta-optimization problem without accessing the Hessian information of the empirical risk. Inspired by regularization-based methods, we compute the parameter importance using the first-order Taylor series and assign the learning rates according to the parameter importance. In this way, no extra trainable parameters will be incorporated so that the computational complexity and memory usage can be reduced. We also impose explicit regularization terms in the inner loss to achieve better performance and apply proximal gradient descent to improve efficiency. Our approach performs competitively on four commonly used benchmark datasets, achieving better or on-par performance against La-MAML and other state-of-the-art approaches in a much shorter training time.

II. RELATED WORK

A. Continual Learning

Existing continual learning approaches are mostly broadly classified into *regularization-based*, *expansion-based*, and *rehearsal-based* approaches [13].

Expansion-based methods overcome catastrophic forgetting by dynamically allocating additional task-specific parameters to accommodate new tasks. In [6], [14], [15] a sub-network is trained for each task. Dynamically Expandable Network (DEN) [5] splits or duplicates important neurons on new tasks when expanding the network to reduce such redundancy, whereas [16] shares part of the base network. [17] performs wights pruning after training on each task

to ensure that the model maintains the compactness, which highly increases the computational overhead. Dirichlet process mixture models have also been applied to expand a set of networks [18]. Reinforced Continual Learning (RCL) [19] searches for the best network architecture for arriving tasks using reinforcement learning. By design, these approaches often result in higher model and time complexities.

Regularization-based methods penalize major changes by quantifying parameter importance on previous tasks while using a fixed capacity. Parameter importance could be estimated by Fisher information matrix [3], loss [4] or outputs sensitivity with respect to the parameters [20] and trainable attention masks [21]. Several studies also consider restraining weight updates from Bayesian perspectives [22]–[26].

Rehearsal-based methods maintain a small episodic memory of previously seen samples for replay [7], [8], [27] or train a generative model to produce pseudo-data for past tasks [28]–[30]. Generative models reduce working memory effectively but invoke the complexity of the generative task. In contrast, episodic memory methods are simpler and more effective. Gradient Episodic Memory (GEM) [7] aligns gradients across tasks to avoid interference with the previous tasks. Averaged-GEM (A-GEM) [8] simplifies GEM by replacing all gradients to one gradient of a sampled batch. Experience Replay (ER) [31] considers the online setting and jointly trains the model on the samples from new tasks and episodic memory. A number of methods focus on improving the memory selection process, like MIR [32] that selects most interfered samples for memory rehearsal and HAL [33] that selects the anchor points of past tasks and interleaves them with new tasks for future training.

B. Meta-Learning for Continual Learning

Recently, it has been shown that gradient-based meta-learning algorithms integrated with episodic memory outperform many previous approaches in online settings [10]–[12]. Meta-Experience Replay (MER) [10] aligns gradients between old and new tasks using samples from an experience replay module. However, the training speed of MER is pretty slow so it's impractical to extend it to real-world scenarios. Online-aware Meta-Learning (OML) [12] proposes a meta-objective to learn a sparse representation offline. Lookahead-MAML (La-MAML) [11] introduces a more efficient online objective and incorporates trainable parameter-specific learning rates to reduce the interference. Both La-MAML and MER require the computation of second-order derivatives.

III. PRELIMINARIES

A. Continual Learning

Suppose that a sequence of T tasks $[\tau_1, \tau_2, \ldots, \tau_T]$ is observed sequentially. Each task τ_t is associated with a dataset $\{X^t, Y^t\} = \{(x_m^t, y_m^t)\}_{m=1}^{n_t}$ of n_t example pairs. At any time-step j during online learning, we would like to minimize the loss on all the t tasks seen so far ($\tau_{1:t}$):

$$\theta^j = \arg\min_{\theta^j} \sum_{i=1}^{t} \mathbb{E}_{\tau_i} \left[\ell_i(\theta^j) \right]$$
$$= \arg\min_{\theta^j} \mathbb{E}_{\tau_{1:t}} \left[L_t(\theta^j) \right] \tag{1}$$

where ℓ_i is the loss on τ_i using θ^j, the learnt model parameters at time-step j. $L_t = \sum_{i=1}^{t} \ell_i$ is the sum of all task-wise losses for tasks $\tau_{1:t}$. GEM [7] reformulates this problem as:

$$\min_{\tilde{g}} \frac{1}{2} \|g - \tilde{g}\|_2^2, \ s.t. \ \langle \tilde{g}, g_p \rangle \geq 0, \ \forall p < t, \tag{2}$$

where g and g_p are the gradient vectors computed on the current task and previous tasks k respectively. Such objective can also be treated as maximizing the dot products between gradients of a set of tasks [11]:

$$\theta^j = \arg\min_{\theta^j} \left(\sum_{i=1}^{t} \ell_i(\theta^j) - \alpha \sum_{p,q \leq t} \left(\frac{\partial \ell_p(\theta^j)}{\partial \theta^j} \cdot \frac{\partial \ell_q(\theta^j)}{\partial \theta^j} \right) \right), \tag{3}$$

where α is a trade-off hyper-parameter.

B. Model-Agnostic Meta-Learning

Model Agnostic Meta-Learning (MAML) [34] is an gradient-based meta-learning approach aiming to learn meta-parameters that produce good task specific parameters after adaptation. Meta-parameters are learned in the *meta-update* (outer-loop), while task-specific models are learned in the *inner-update* (inner-loop). In every meta-update, its objective at time-step j can be formulated as below:

$$\underbrace{\min_{\theta_0^j} \mathbb{E}_{\tau_{1:t}} \left[L_{\text{meta}} \left(\overbrace{U_k(\theta_0^j)}^{\text{inner-loop}} \right) \right]}_{\text{outer-loop}} = \min_{\theta_0^j} \mathbb{E}_{\tau_{1:t}} \left[L_{\text{meta}} \left(\theta_k^j \right) \right], \tag{4}$$

where θ_0^j is the meta-parameter at time-step j and $U_k(\theta_0^j) = \theta_k^j$ represents an update function where θ_k^j is the parameter after k steps of stochastic gradient descent.

[9] has proved that MAML and its first-order variation like Reptile approximately optimize for the same objective that gradients are encouraged to align within-task and across-task as well. [10] then showed the equivalence between the objective of GEM (Eq (3)) and Reptile. This implies that the procedure to meta-learn an initialization coincides with learning optimal parameters for continual learning.

C. Meta-Learning for Continual Learning

Although gradient-based meta-learning algorithms implicitly align the gradients, there can still be some interference between the gradients of old tasks $\tau_{1:t-1}$ and new task τ_t. Specifically, when starting training on task τ_t, the gradients are not necessarily aligned with the old ones, since the data of $\tau_{1:t-1}$ is no longer available to us. To ensure meta-updates are

conservative with respect to $\tau_{1:t-1}$, MER modifies the Reptile algorithm to integrate it with an experience replay module, which aligns gradients between old and new tasks during meta-updates. However, MER's algorithm is prohibitively slow in training speed.

La-MAML [11] then introduces two mechanisms as follows. (1) Optimizing for an alternative objective:

$$\min_{\theta_0^j} \sum_{\mathcal{S}_k^j \sim D_t} \left[L_t \left(U_k(\theta_0^j, \mathcal{S}_k^j) \right) \right], \tag{5}$$

where $\mathcal{S}_k^j = \left\{ x_{j+l}^t, x_{j+l}^t \right\}_{l=1}^k$ is a random stream of length k sampled from $\{X^t, Y^t\}$ at time-step j. The meta-loss $L_t = \sum_{i=1}^t \ell_i$ is evaluated on $\theta_k^j = U_k(\theta_0^j, S_k^j)$. Eq (5)'s objective only uses one data point from $\mathcal{S}_k^j$ for one inner-update instead of using the complete batch of data $\mathcal{S}_k^j$ for all inner-updates, which significantly improves the computational and memory efficiency. It appears to be the relationship between SGD and batch-GD. La-MAML then proves that this mechanism coincides with AGEM's objective, i.e., aligning the gradients of τ_t and the *average gradient* of $\tau_{1:t}$ instead of aligning all the pair-wise gradients between tasks $\tau_{1:t}$. (2) Incorporating learnable per-parameter learning rates to mitigate catastrophic forgetting. Learning rates are modulated during each meta-update and used for both inner-update and meta-update.

La-MAML shows impressive performance and drastic speedup when compared to other meta-learning based methods. However, it still relies on directly computing the Hessian matrix during meta-update, which incurs a lot of computational overhead. Also, learning a separate learning rate for each parameter requires increased computational effort and larger memory usage.

IV. PROPOSED METHOD

In this section, based on La-MAML's scheme and inspired by some regularization-based methods, we propose Efficient Meta-Learning for Continual Learning (EMCL), a novel approach that neither requires computing the Hessian matrix nor incorporates any other learnable parameters.

A. Inner-Update with Explicit Regularization

Different from La-MAML where the same meta-update learning rate is also used for the inner-update, we empirically found that explicitly adding regularization terms to the inner loss function achieves better performance:

$$
\begin{aligned}
\theta_k^j &= \arg\min_{\theta_k^j} \ell_i(\theta_k^j) \\
&= \arg\min_{\theta_k^j} \left\{ \mathcal{L}(\theta_k^j) + \frac{\lambda}{2} \sum_m h_m^j \left\| \theta_{k,m}^j - \theta_{0,m}^j \right\|_2^2 \right\} \\
&= \arg\min_{\theta_k^j} \left\{ \mathcal{L}(\theta_k^j) + \frac{\lambda}{2} \left\| \mathbf{H}^j(\theta_k^j - \theta_0^j) \right\|_2^2 \right\}.
\end{aligned}
\tag{6}
$$

$\mathcal{L}$ is the empirical risk function. $\mathbf{H}^j$ is a diagonal matrix with $\mathbf{H}_{m,m}^j = \sqrt{h_m^j}$, where h_m^j is the moving average of the importance of the m-th parameter Ω_m^j at time-step j:

$$h_m^j = \eta h_m^{j-1} + (1-\eta)\Omega_m^j. \tag{7}$$

η is the decay rate and r is a scale factor. The rationale of using the moving average is that even a small change in important parameters may drastically degrade the performance of previous tasks. We will explain the way of computing Ω_m^j later in Section IV-C. This regularization term can be viewed as an importance-weighted ℓ_2 norm. On the one hand, the regularization term in Eq (6) tries to reduce the change in important weights. On the other hand, it also encourages θ_k^j to remain close to θ_0^j, thereby retaining a strong dependence on the initial parameters to avoid over-fitting and vanishing gradients [35]. This point is important especially when multiple inner-loop optimization steps are required.

B. Closed-form Meta-Update with Adaptive Learning Rate

During meta-update, we scale the learning rate for each parameter inversely proportional to the moving average of its importance. Let α_m^j be the learning rate of the m-th meta-parameter at time-step j, we have:

$$\alpha_m^j \leftarrow \frac{r}{h_m^j} \alpha_m^{j-1}, \tag{8}$$

where r is a scale factor. In this way, changes in important parameters can be reduced, while less important parameters allow having larger step sizes in future tasks.

Supposing that θ_k is the unique minimizer of Eq (6), we know that the gradient of the inner-loss should be zero:

$$\nabla L(\theta_k) + \lambda \mathbf{H}^2(\theta_k - \theta_0) = 0. \tag{9}$$

Here we omit the superscript j for convience. Then the gradient of the meta-loss can be expressed in closed-form as:

$$
\begin{aligned}
\nabla_{\theta_0} L_t(\theta_k) &= \left(\frac{\partial \theta_k}{\partial \theta_0} \right)^\top \nabla L(\theta_k) \\
&\quad + \lambda \left(\left(\frac{\partial \theta_k}{\partial \theta_0} \right)^\top - I \right) \mathbf{H}^2(\theta_k - \theta_0) \\
&= \frac{\partial \theta_k}{\partial \theta_0} \left(\nabla L(\theta_k) + \lambda \mathbf{H}^2(\theta_k - \theta_0) \right) - \lambda \mathbf{H}^2(\theta_k - \theta_0) \\
&= \lambda \mathbf{H}^2(\theta_0 - \theta_k).
\end{aligned}
\tag{10}
$$

This algorithm is a first-order method without the need of computing the Hessian matrix. However, unlike other first-order methods like Reptile [9] and FOMAML [34], it can make use of higher-order information of the inner loss beyond gradients to search the optimal hypothesis around θ_0 as well as to offer the theoretical guarantees of convergence and generalization. The proof can be easily extended from [36]'s work.

C. Parameter Importance Estimation via Taylor Expansion

We update the estimation of importance for each parameter in every meta-update. The importance of the m-th meta-parameter can be quantified by the impact on the total loss after zeroing it out:

$$\Omega_m = \left| L_t(\theta_0) - L_t\left(\theta_0\big|_{\theta_{0,m}=0}\right) \right|. \tag{11}$$

Here $\theta_0^j\big|_{\theta_{0,m}^j=0}$ indicates the zeroing out operation. In essence, zeroing out is similar to *leave-one-out* in classification. It is computationally intensive to computing Ω_k for each parameter in every meta-update, so we approximate it using first-order Taylor expansion. The Taylor series of loss function $L(\theta_0^j)$ at $\theta_0\big|_{\theta_{0,m}=0}$ is:

$$L_t\left(\theta_0\big|_{\theta_{0,m}=0}\right) = L_t(\theta_0) + \frac{\partial L_t(\theta_0)}{\partial \theta_{0,m}}(\theta_{0,m} - 0) + o(\theta_{0,m}), \tag{12}$$

where $o(\theta_{0,m})$ represents the terms of higher orders. We omit the higher-order terms in Eq (12) to obtain an approximate parameter importance:

$$\Omega_m = \left| L_t\left(\theta_0\big|_{\theta_{0,m}=0}\right) - L_t(\theta_0) \right| \approx \left| \frac{\partial L_t(\theta_0)}{\partial \theta_{0,m}}\theta_{0,m} \right|. \tag{13}$$

Computing Eq (13) incurs little computational overhead, since the gradient $\partial L_t(\theta_0)/\partial\theta_{0,m}$ is already available after inner-update. It should be noted that, Eq (13) computes the weight importance using the absolute value of the product of the gradient with the *parameter*, which is different from that used by SI [4], where the importance is calculated using the product of the gradient with the *parameter update*.

D. Inner-Update with Proximal Gradient Descent

While Eq (6) can be minimized via applying normal gradient descent methods, we employ the Proximal Gradient Descent (PGD) method. We first denote the proximal operator of a function $f(\cdot)$ with a scalar parameter $\gamma > 0$ as:

$$\text{prox}_{\gamma f}(v) = \arg\min_x \left(f(x) + \frac{1}{2\gamma}\|x - v\|_2^2 \right), \tag{14}$$

where $x \in \mathbb{R}^n, v \in \mathbb{R}^n$ are two n dimensional vectors. In this case, for $\theta_{k,m}$, we have:

$$f(\theta_{k,m}) = \frac{\lambda}{2}h_m\|\theta_{k,m} - \theta_{0,m}\|_2^2. \tag{15}$$

Proximal operators can be interpreted as modified gradient steps:

$$\hat{\theta}_\kappa = \theta_{\kappa-1} - \gamma\nabla L(\theta_{\kappa-1}), \tag{16}$$

$$\theta_\kappa = \text{prox}_{\gamma f}(\hat{\theta}_\kappa), \tag{17}$$

for $\kappa = \{1, \dots, k\}$. We then introduce the following lemma:

Lemma 1: For $f(x) = c\|x - x_0\|_2^2$ with $c > 0$ and any fixed vector x_0, we have:

$$\text{prox}_{\gamma f}(v) = \frac{v + 2\gamma c x_0}{2\gamma c + 1}. \tag{18}$$

Proof 1: $\text{prox}_{\gamma f}$ minimizes the function:

$$l(x) = \|c(x - x_0)\|_2^2 + \frac{1}{2\gamma}\|x - v\|_2^2. \tag{19}$$

We denote x^* as the minimizer of $l(x)$ and we know that $\nabla l(x^*) = 0$ since $l(x)$ is convex. By differentiation, we get:

$$\nabla l(x) = 2c(x^* - x_0) + \frac{1}{\gamma}(x^* - v) = 0,$$
$$\Rightarrow x^* = \frac{v + 2\gamma c x_0}{2\gamma c + 1}. \tag{20}$$

By Lemma 1, we have the following closed-form proximal gradient update rule for Eq (15):

$$\theta_{\kappa,m} = \frac{\hat{\theta}_{\kappa,m} + \gamma\lambda h_m\theta_{0,m}}{\gamma\lambda h_m + 1}. \tag{21}$$

Eq (21) is expected to be more efficient in computation than directly applying vanilla gradient decent methods since it avoids computing gradients for the quadratic regularization term in Eq (6). We also find it achieves higher accuracy (see ablation results in Table IV).

Finally, we summarize our method EMCL in Algorithm 1.

Algorithm 1 EMCL (LR: Learning Rate)

Require: Network weights θ_0^0, inner objective ℓ, meta objective L, inner LR β, meta LRs α, parameter importance decay rate η, LR scale factor r, inner regularization λ
1: **initialize:** $j \leftarrow 0$
2: **for** $t := 1 \rightarrow T$ **do**
3:     **for** batch b in $\{X^t, Y^t\}$ **do**
4:         $k \leftarrow \text{sizeof}(b)$
5:         **for** $\kappa = 1 \rightarrow k$ **do**
6:             $\theta_\kappa^j \leftarrow \theta_{\kappa-1}^j - \beta\nabla_{\theta_\kappa^j}\ell_t(\theta_\kappa^j, b[\kappa], \lambda)$ ▷ optimize for Eq (6) via (16), (17) and (21)
7:         **end for**
8:         $\alpha^{j+1} \leftarrow \text{UpdateMetaLR}(\alpha^j, \eta, r)$ ▷ modulate meta LR accroding to Eq (8)
9:         $\theta_0^{j+1} \leftarrow \theta_0^j - \alpha^j\nabla_{\theta_0^j}L_t(\theta_k^j, b)$ ▷ meta-update via Eq (10)
10:         $j \leftarrow j + 1$
11:     **end for**
12: **end for**

V. EXPERIMENTAL SETUP

In this section, we perform experimental evaluations of our method compared with existing state-of-the-art methods for continual learning on four benchmark datasets.

Datasets: We perform experiments on four benchmark datasets:

TABLE I: Statistics of the benchmark datasets

	MNIST Perm.	Many Perm.	CIFAR100	miniImageNet
Input size (color and image pixels)	$1 \times 28 \times 28$	$1 \times 28 \times 28$	$3 \times 32 \times 32$	$3 \times 84 \times 84$
Num of tasks	20	100	20	20
Num of samples per task	1000	200	2500	2500
Num of classes per task	10	10	5	5

TABLE II: List of hyper-parameters for all approaches

Method	Hyper Parameter	MNIST Perm	Many Perm	CIFAR100	miniImageNet
EWC	lr	-	-	0.03	0.03
	regularization	-	-	100	100
GEM	lr	-	-	0.03	0.03
	memory size	-	-	200	200
A-GEM	lr	-	-	0.03	0.03
	memory size	-	-	200	200
La-MAML	α_0: initial lr	0.3	0.1	0.1	0.1
	η: lr for α_0	0.15	0.1	0.3	0.3
	memory size	200	500	200	200
EMCL	α_0: initial meta lr	0.3	0.15	0.1	0.1
	β: inner lr	0.15	0.03	0.075	0.075
	λ: regularization	10	10	50	50
	γ: scalar for pgd	0.3	0.1	0.1	0.1

- **MNIST Permutations** [3] is a variant of MNIST, where each task is a random permutation of the original MNIST pixels. As such, the input distribution of each task is unrelated. It has 20 tasks, each with 1000 samples from 10 different classes.
- **Many Permutations** [10] is a variant of MNIST Permutations that has 5 times more tasks (100 tasks) and 5 times fewer samples (200 samples) per task.
- **Split CIFAR100** [4] splits the original CIFAR-100 dataset [37] into 20 disjoint tasks, where each consists of 2,500 samples from 5 classes which are not included in other tasks.
- **Split miniImageNet** [38] is constructed by splitting 100 classes of miniImageNet into 20 sequential tasks, where each task has 2,500 samples from 5 classes.

We also summarize the statistics of the benchmark datasets we used in Table I.

Training Details: Unlike most prior works that train each task in multiple epochs, we consider a more challenging setup following [11]: *single-pass* setting formalized by [8] assumes data for every task to be processed for only one epoch. After that, data samples are not accessible anymore unless they were added to a replay buffer. We use this setting for all our experiments.

Single-head setting is used for experiments on MNIST Permutations and Many Permutations, where all tasks share the final classifier layer and inference is performed without task identity. For other experiments, we use *multi-head* setting, where each task has a separate classifier.

For methods that require storing the past samples (GEM, A-GEM, MER, and La-MAML), we use a replay buffer of size 200, 500, 200, and 200 for MNIST Permutations, Many

Permutations, Split CIFAR100, and Split miniImageNet, respectively. We carry out hyper-parameter tuning by performing a grid-search for hyper-parameters related to the learning-rate. See Table II for more details of hyper-parameters.

Model Architecture: We use a fully-connected network with two hidden layers of 100 ReLU units each for MNIST Permutations and Many Permutations. The architecture is the same as in MER [10] and La-MAML [11] to compare the results directly. For Split CIFAR100 and Split miniImageNet, we use a reduced ResNet18 [39] architecture with three times fewer feature maps across all layers by following [7].

For a given dataset, all networks use the same architecture and are optimized via SGD optimizer with a batch size of 10 samples from an online stream [7].

Baselines: We compare our method against the following baselines:

- **EWC**: Elastic Weight Consolidation [3] is an algorithm where the loss is regularized by Fisher Information to avoid catastrophic forgetting.
- **GEM**[1]: Gradient Episodic Memory [7] does constrained optimization by solving a quadratic program on the gradients of new and replay samples, so that they do not interfere with past memories.
- **A-GEM**: Averaged Gradient Episodic Memory [8] improves GEM by replacing all gradients to one gradient of a sampled batch such that the average episodic memory loss will not increase.
- **MER**: Meta Experience Replay [10] samples i.i.d data from a replay memory to meta-learn model parameters

---

[1]The code of GEM and EWC is adopted from https://github.com/facebookresearch/GradientEpisodicMemory

TABLE III: ACC (%) (larger is better), BWT (%) (larger is better) and their standard deviation on MNIST Permutations, Many Permutations, Split CIFAR100, and Split miniImageNet. † denotes the result reported by [11]. Other results are reproduced by us, where each of them is run with 5 random seeds. Due to the slow training speed of MER, we do not report its performance on CIFAR100 and miniImageNet.

| Method	MNIST Perm.		Many Perm.		CIFAR100		miniImageNet	
	ACC	BWT	ACC	BWT	ACC	BWT	ACC	BWT
EWC	62.32 ± 1.34 †	-13.32 ± 2.24 †	33.46 ± 0.46 †	-17.84 ± 1.15 †	39.60 ± 1.11	-23.53 ± 1.19	34.34 ± 2.06	-28.17 ± 1.49
GEM	55.42 ± 1.10 †	-24.42 ± 1.10 †	32.14 ± 0.50 †	-23.52 ± 0.87 †	43.41 ± 2.09	-20.76 ± 1.31	37.02 ± 1.91	-25.29 ± 2.10
A-GEM	56.04 ± 2.36	-24.05 ± 2.47	29.98 ± 1.84	-27.23 ± 1.79	43.87 ± 2.61	-23.38 ± 1.52	36.37 ± 1.56	-25.11 ± 2.92
MER	73.46 ± 0.45 †	-9.96 ± 0.45 †	47.40 ± 0.35 †	-17.78 ± 0.39 †	-	-	-	-
La-MAML	$\mathbf{73.92 \pm 1.05}$	-7.91 ± 0.87	47.69 ± 0.41	-13.24 ± 0.95	61.23 ± 0.94	-19.84 ± 2.20	45.29 ± 1.76	-18.57 ± 2.94
EMCL	73.61 ± 1.12	-10.25 ± 0.73	$\mathbf{48.12 \pm 1.48}$	-14.09 ± 0.74	$\mathbf{61.95 \pm 1.20}$	-16.48 ± 1.96	$\mathbf{46.52 \pm 0.83}$	-17.45 ± 2.38
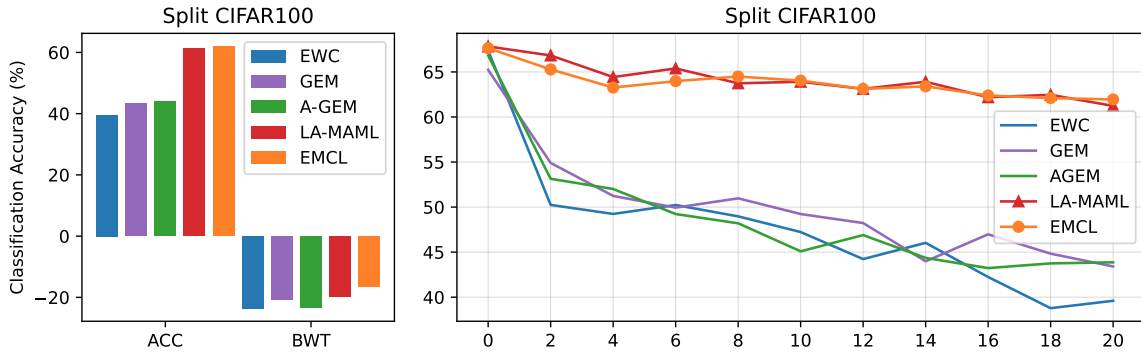
Fig. 1: Left: ACC and BWT for all appraoches on CIFAR100. Right: evolution of the average test accuracy as more tasks are learned.
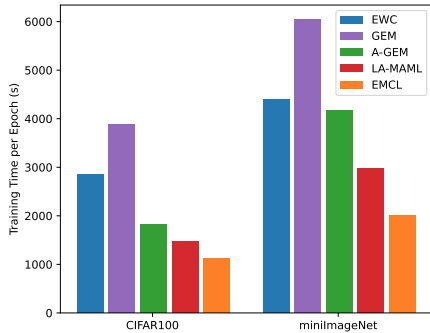

Fig. 2: Training time for all algorithms on Split CIFAR100 and miniImageNet.

that show increased gradient alignment between old and current samples.

- **La-MAML**[2]: Look-ahead MAML [11] uses learnable per-parameter learning rates to mitigate catastrophic forgetting aided by a episodic memory.

---

[2]The code of A-GEM and La-MAML is adopted from https://github.com/montrealrobotics/La-MAML

Performance Metrics: We report the following metrics by evaluating the model on the test set:

- **Accuracy (ACC)**: Average test accuracy of all T tasks after the whole learning is finished:

$$\text{ACC} = \frac{1}{T} \sum_{t=1}^{T} \text{ACC}_{t,T} \quad (22)$$

where $\text{ACC}_{t,T}$ is the accuracy of task t, after finishing the training process on task T.

- **Backward Transfer (BWT)**: The average influence of new learning on past knowledge. For instance, negative BWT indicates catastrophic forgetting. It is formally defined as:

$$\text{BWT} = \frac{1}{T-1} \sum_{t=1}^{T-1} \text{ACC}_{t,T} - \text{ACC}_{t,t} \quad (23)$$

VI. RESULTS AND DISCUSSIONS

Performance: Table III shows the overall experimental results. We also visualize the ACC, BWT, and evolution of the average test accuracy as a function of the number of tasks for CIFAR100 in Figure 1.

In every setting, EMCL outperforms the commonly used baselines like EWC [7], GEM [3] and A-GEM [8] significantly, especially on complex datasets such as Split CIFAR100

TABLE IV: Ablation study on Split miniImageNet

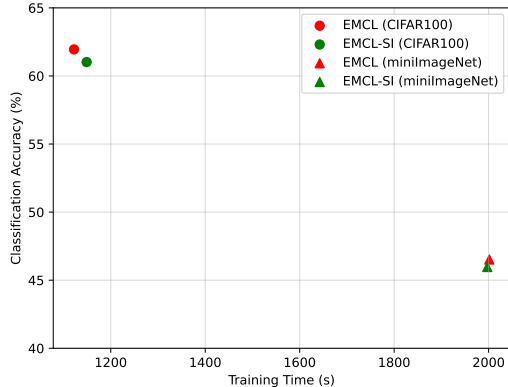| Inner LR modulation	$\times$	$\times$	$\checkmark$
Inner regularization	$\checkmark$	$\checkmark$	$\times$
PGD	$\checkmark$	$\times$	$\times$
ACC (%)	**46.52**	46.17	45.04

Fig. 3: Accuracy vs training time comparison on Split CI-FAR100 and miniImageNet for EMCL and EMCL-SI.

and Split miniImageNet. Even when compared to strong baselines like MER [10] and La-MAML [11], our approach is still able to achieve better or comparable performance while taking shorter training time (Figure 2). It is also worth emphasizing that the number of trainable parameters in La-MAML is twice as much as ours.

Although [8] have mentioned that EWC and similar regularization-based methods perform substantially worse than rehearsal-based methods like GEM and A-GEM on the single-pass setting, they achieve similar accuracy in our experiments. This is possibly due to the size of replay buffers for rehearsal-based methods in our experiments (1-2 samples per class) being much smaller than those in theirs (13-25 samples per class). Such results demonstrate that rehearsal-based methods are not very effective in a low-resource regime. Even so, rehearsal-based methods still require an external episodic memory to store images from past tasks. While EMCL only needs a much smaller memory for storing the latest weighted moving averages of parameter importance and per-parameter learning rates.

Training Time: Figure 2 shows the training time after learning all tasks per epoch for different algorithms. The training time is measured on a single NVIDIA GeForce RTX 2080 GPU, including time spent for memory management for GEM, A-GEM, and La-MAML and weight importance calculation for EWC and EMCL. We see EMCL takes a much shorter training time than EWC, GEM, and A-GEM while achieving much higher accuracy. EMCL is also faster than La-MAML because it doesn't need to compute second-order derivatives and learn per-parameter learning rates.

Ablation Study: We also perform an ablation study and

the results are reported in Table IV. *Inner LR modulation* denotes using the same meta learning rates for the inner loop without adding explicit regularization terms. In this setting, the method performs 1.48% worse, indicating adding explicit regularization terms to inner loss is more effective in alleviating catastrophic forgetting. We also find that removing proximal gradient decent results in a drop in accuracy. This is possibly due to proximal gradient descent can find an analytic solution for part of the inner objective directly.

We then compare our parameter importance estimation method based on first-order Taylor series to that used by SI [4] where parameter importance is calculated using the product of the gradient and the *parameter update*. The results are summarized in Figure 3. We see that the training speed of EMCL-SI is similar to EMCL but it performs worse slightly.

VII. CONCLUSION

In this paper, we propose a novel efficient meta-learning algorithm for continual learning problems. Based on parameter importance estimated using the Taylor series, we modulate the meta-update learning rates and add explicit regularization terms to the inner loss to alleviate catastrophic forgetting. Our method expresses the gradient of meta-updated in closed-form to avoid accessing the Hessian information. We also use proximal gradient descent to solve the inner objective easier and improve the computational efficiency. Experiments on diverse benchmark datasets with different network architectures against strong baselines demonstrate the effectiveness of our approach in achieving high performance in a much shorter time.

REFERENCES

[1] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.

[2] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly, "Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory." *Psychological review*, vol. 102, no. 3, p. 419, 1995.

[3] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[4] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3987–3995.

[5] J. Yoon, E. Yang, J. Lee, and S. J. Hwang, "Lifelong learning with dynamically expandable networks," in *International Conference on Learning Representations*, 2018.

[6] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[7] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6470–6479.

[8] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," in *International Conference on Learning Representations*, 2018.

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

[10] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro, "Learning to learn without forgetting by maximizing transfer and minimizing interference," in *International Conference on Learning Representations*, 2018.

[11] G. Gupta, K. Yadav, and L. Paull, "La-maml: Look-ahead meta learning for continual learning," *arXiv preprint arXiv:2007.13904*, 2020.

[12] K. Javed and M. White, "Meta-learning representations for continual learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 1818–1828.

[13] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[14] P. Singh, V. K. Verma, P. Mazumder, L. Carin, and P. Rai, "Calibrating cnns for lifelong learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 579–15 590, 2020.

[15] J. von Oswald, C. Henning, J. Sacramento, and B. F. Grewe, "Continual learning with hypernetworks," in *8th International Conference on Learning Representations (ICLR 2020)(virtual)*. International Conference on Learning Representations, 2020.

[16] S. S. Sarwar, A. Ankit, and K. Roy, "Incremental learning in deep convolutional neural networks using partial network sharing," *IEEE Access*, vol. 8, pp. 4615–4628, 2019.

[17] S. C. Hung, C.-H. Tu, C.-E. Wu, C.-H. Chen, Y.-M. Chan, and C.-S. Chen, "Compacting, picking and growing for unforgetting continual learning," in *Advances in Neural Information Processing Systems*, 2019, pp. 13 647–13 657.

[18] G. Jerfel, E. Grant, T. L. Griffiths, and K. Heller, "Reconciling meta-learning and continual learning with online mixtures of tasks," in *Advances in Neural Information Processing Systems*, 2019, pp. 9119–9130.

[19] J. Xu and Z. Zhu, "Reinforced continual learning," in *Advances in Neural Information Processing Systems*, 2018, pp. 907–916.

[20] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[21] J. Serra, D. Suris, M. Miron, and A. Karatzoglou, "Overcoming catastrophic forgetting with hard attention to the task," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4548–4557.

[22] H. Ahn, S. Cha, D. Lee, and T. Moon, "Uncertainty-based continual learning with adaptive regularization," in *Advances in Neural Information Processing Systems*, 2019, pp. 4394–4404.

[23] S. Ebrahimi, M. Elhoseiny, T. Darrell, and M. Rohrbach, "Uncertainty-guided continual learning with bayesian neural networks," in *International Conference on Learning Representations*, 2019.

[24] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," in *International Conference on Learning Representations*, 2018.

[25] H. Ritter, A. Botev, and D. Barber, "Online structured laplace approximations for overcoming catastrophic forgetting," in *Advances in Neural Information Processing Systems 31*, 2018, pp. 3742–3752.

[26] M. K. Titsias, J. Schwarz, A. G. d. G. Matthews, R. Pascanu, and Y. W. Teh, "Functional regularisation for continual learning with gaussian processes," in *ICLR*, 2020.

[27] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[28] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," in *Advances in Neural Information Processing Systems 31*, 2017, pp. 2990–2999.

[29] R. Kemker and C. Kanan, "Fearnet: Brain-inspired model for incremental learning," in *International Conference on Learning Representations*, 2018.

[30] O. Ostapenko, M. Puscas, T. Klein, P. Jahnichen, and M. Nabi, "Learning to remember: A synaptic plasticity driven framework for continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 321–11 329.

[31] A. Chaudhry, M. Rohrbach, M. Elhoseiny, T. Ajanthan, P. K. Dokania, P. H. Torr, and M. Ranzato, "Continual learning with tiny episodic memories," *arXiv preprint arXiv:1902.10486*, 2019.

[32] R. Aljundi, E. Belilovsky, T. Tuytelaars, L. Charlin, M. Caccia, M. Lin, and L. Page-Caccia, "Online continual learning with maximal interfered retrieval," *Advances in neural information processing systems*, vol. 32, 2019.

[33] A. Chaudhry, A. Gordo, P. Dokania, P. Torr, and D. Lopez-Paz, "Using hindsight to anchor past knowledge in continual learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 6993–7001.

[34] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1126–1135.

[35] A. Rajeswaran, C. Finn, S. Kakade, and S. Levine, "Meta-learning with implicit gradients," *Advances in neural information processing systems*, 2019.

[36] P. Zhou, X. Yuan, H. Xu, S. Yan, and J. Feng, "Efficient meta learning via minibatch proximal update," *Advances in Neural Information Processing Systems*, vol. 32, pp. 1534–1544, 2019.

[37] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[38] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 3630–3638.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.