Bilateral Memory Consolidation for Continual Learning

Xing Nie[1,2], Shixiong Xu[1,2], Xiyan Liu[3], Gaofeng Meng[1,2,4],
Chunlei Huo[1,2], Shiming Xiang[1,2]

[1] State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences.
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences. [3] Baidu Inc., China.
[4] Centre for Artificial Intelligence and Robotics, HK Institute of Science & Innovation, CAS.
{niexing2019, xushixiong2020}@ia.ac.cn

Abstract

Humans are proficient at continuously acquiring and integrating new knowledge. By contrast, deep models forget catastrophically, especially when tackling highly long task sequences. Inspired by the way our brains constantly rewrite and consolidate past recollections, we propose a novel Bilateral Memory Consolidation (BiMeCo) framework that focuses on enhancing memory interaction capabilities. Specifically, BiMeCo explicitly decouples model parameters into short-term memory module and long-term memory module, responsible for representation ability of the model and generalization over all learned tasks, respectively. BiMeCo encourages dynamic interactions between two memory modules by knowledge distillation and momentum-based updating for forming generic knowledge to prevent forgetting. The proposed BiMeCo is parameter-efficient and can be integrated into existing methods seamlessly. Extensive experiments on challenging benchmarks show that BiMeCo significantly improves the performance of existing continual learning methods. For example, combined with the state-of-the-art method CwD [55], BiMeCo brings in significant gains of around 2% to 6% while using 2x fewer parameters on CIFAR-100 under ResNet-18.

1. Introduction

Acquiring knowledge in a continuous fashion is a key capability for a learning system to achieve human intelligence. To this end, *Continual Learning* (CL) is introduced to foster the network to learn a sequence of tasks incrementally with the aim of exploiting existing knowledge to adapt quickly to new tasks. In CL, the training data of different classes comes in a phase-by-phase manner, where the model is trained on new class data at each task and then evaluated on all learned old and new classes. During training, a small number of exemplars of old classes are allowed to be saved in the memory buffer with a strict budget restriction. These limited exemplars mitigate forgetting of previously learned tasks to some extent, but lead to another pressing issue – the extreme data imbalance between new and old classes, making the model biased towards recently learned tasks.

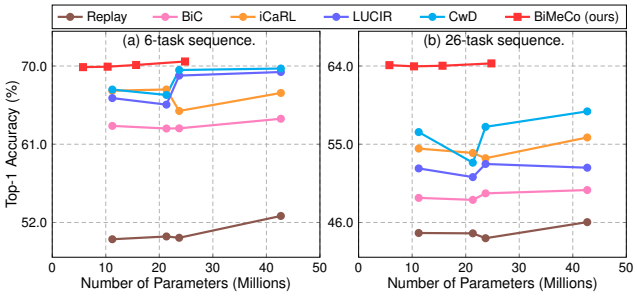

Figure 1. **Model Size vs. Average Incremental Accuracy.** Taking class incremental learning [45] as an example, we use two typical settings on CIFAR-100 [39], where 50 classes are for the first task with later (a) 10 classes per task (6-task sequence) and (b) 2 classes per task (26-task sequence). BiMeCo significantly outperforms other CL methods with 1.51∼2.06x smaller parameters under **ResNet-18/34/50/101** [26]. More details are in Appendix.

A wide variety of learning algorithms have been proposed to tackle the above shortcomings, including approximating exemplars per class for model inference [51], rectifying the network predictions to alleviate the task bias [5, 61], and preserving the original model on old classes to provide soft labels for resisting to activating drifting [29, 42]. Recent studies such as CwD [55] also show that merely improving the learned representations of the initial task can facilitate mitigating forgetting greatly.

Although the achievements in the literature are brilliant, these methods still struggle to prevent forgetting when tackling long task sequences. In Fig. 1, we provide an empirical evaluation of representative CL methods on CIFAR-100 [39]. Though achieving decent accuracy in the incremental setting of 6 tasks, surprisingly they encounter evident damages when handling 26 tasks. A vivid exam-

ple is CwD [55], compared with fine-tuning with exemplars (Replay), it boosts accuracy by more than 17% on the 6-task sequence under various ResNet-type networks [26], but merely brings around 10% accuracy gains on the more challenging 26-task sequence.

As pointed out in [18, 45], increasing memory buffer sizes or model parameters typically can alleviate forgetting for tackling long task sequences. However, we argue these strategies could be sub-optimal, since a desirable continual learning system should be able to scale to a wide variety of incremental scenarios without the need for extra overheads.

In this work, we approach this problem from a rather different perspective, and we study how to sufficiently mine memory samples to form structured knowledge for all learned tasks. To this end, we present *Bilateral Memory Consolidation* (BiMeCo) as a simple framework for continual learning. Specifically, BiMeCo explicitly disentangles model parameters into two complementary and parallel parts: short-term memory module and long-term memory module. Short-term memory module focuses on the representation ability of the model and rapid adaptation to recent tasks to form working memory, while being guided by the long-term memory module to prevent forgetting by knowledge distillation. Concurrently, long-term memory module learns a few task-balance samples online to consolidate working memory, while inheriting strong expressiveness from the short-term memory module by momentum-based updating. The key to BiMeCo is encouraging dynamic interactions between different degrees of memories to produce rich feature representations. This way, BiMeCo enables continually evoking previous knowledge while learning from changing data streams. In implementation, we present a simple module replacement to integrate BiMeCo with existing models in a parameter-efficient manner.

We demonstrate that the proposed BiMeCo is significantly superior to other CL methods in a variety of aspects with great parameter efficiency. Fig. 1 illustrates the quantitative results of our BiMeCo on CIFAR-100 [39] of two incremental settings under four ResNet-type networks [26]. Specifically, integrated with LUCIR [29], BiMeCo brings in a gain of at least +2.58% average incremental accuracy while significantly reducing parameters over other methods on the 6-task sequence under ResNet-18. BiMeCo also achieves a superior result compared with CwD [55], which brings +5.52% accuracy gains with the reduction of parameters by 41.9% on the 26-task sequence under ResNet-101. Notably, BiMeCo is flexible and can be compatible with existing methods seamlessly with further performance gains.

The key contributions can be summarized as follow:

- A bilateral memory consolidation framework, *i.e.*, BiMeCo, is proposed. This generic framework is parameter-efficient and can be seamlessly integrated into existing CL methods to further boost performance.
- The proposed BiMeCo decouples model parameters into two complementary memories, encouraging dynamic interactions between different degrees of memories to prevent forgetting for continual learning.
- Extensive experiments as well as comprehensive ablation studies demonstrate the effectiveness of our BiMeCo on three challenging benchmarks under a wide variety of incremental settings.

2. Related Work

2.1. Continual Learning

Recently, a large body of work in continual learning (CL) has been developed to alleviate catastrophic forgetting [46]. Broadly, CL methods can be grouped into four categories: rehearsal, regularization, task-recency bias correction, and network expansion. Rehearsal-based methods keep a small set of exemplars in a raw format or generate pseudo-samples for preventing forgetting. iCaRL [51] stores a subset of exemplars and selects the best approximate class means in the learned feature space. Subsequent work explores new update rules using exemplar samples [13, 44], new sampling strategies [4, 12, 37, 60], and using generative models to construct pseudo-examples [35, 62]. Regularization-based methods use regularization terms in the loss function to mitigate forgetting. Some studies regularize the weights and perform the importance metric estimation for each parameter [1, 41, 48], while others are devoted to the importance of remembering feature representations [33, 40, 50, 65]. Another line of CL is bias-correction methods, which aim to alleviate the tendency of the network to be biased toward the recently learned tasks. BiC [61] learns a bias correction layer for correcting task bias of the network with two-stage training. LUCIR [29] replaces the standard softmax layer with a cosine normalization layer to enforce balanced magnitudes across all classes. Recently, expansion-based methods are proposed to dedicate different model parameters to each task for mitigating forgetting, either freezing previously learned parameters [43, 53, 63], or allocating a model copy to each task [2]. In this work, the proposed method falls into the expansion-based category. But it differs from previous studies in a key aspect: instead of freezing model parameters for generalization over old tasks, we encourage dynamic interactions between model parameters of generality over all learned tasks and scalability over learning new tasks. This way, our method can effectively prevent forgetting with great parameter efficiency.

2.2. Knowledge Distillation

Knowledge distillation is originally proposed to learn a more compact student network from a larger teacher network [9, 27]. As a pioneer, LwF [42] employs this technique to keep the representations of previous data from

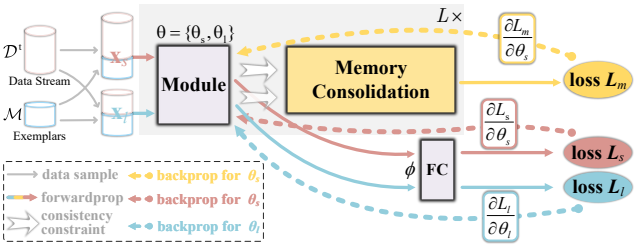

Figure 2. Overview of the proposed BiMeCo. For clarity, a model comprised of L modules (*e.g.*, ResNet [26] with L stacked residual blocks) is illustrated as an example. Here we use the shorthand $\mathcal{M}$ for exemplars $\mathcal{M}^{1:t-1}$. The skip connection remains unchanged (omitted for simplicity). Best viewed in color.

excessively drifting while learning new tasks. Subsequently, numerous pieces of research are devoted to applying knowledge distillation for mitigating forgetting of previous tasks by constraining the model, including directly on the weights [12, 38, 64], the gradients [13, 44], the network outputs [11, 51], intermediate features [21, 23, 29, 69], and combinations thereof. In these methods, when learning a new task, the model of previous tasks is used as the teacher, and the network is regularized to constrain their feature drift. In this way, the knowledge of previously learned classes can be preserved. However, these studies typically rely on a fixed teacher trained on previous tasks. Our work extends this route by dynamically updating the teacher network, which is trained on online balance class distribution of old and new tasks. This way, instead of only containing the constant and static task biases that are previously learned, the teacher network provides a generic input-conditional activation trained on class-balanced data for memory consolidation. Finally, our work is also analogous to DINO [10] in self-supervised learning, where the student and teacher have the same architecture and use distillation during training. However, the teacher in DINO is updated with an average of the student, while it is updated separately in a class-balanced manner in our work.

3. Methodology

In this section, we describe the proposed bilateral memory consolidation framework for continual learning (CL), which can be readily integrated with existing CL methods.

3.1. Preliminary

In CL, a model is trained on a sequence of tasks with a distribution shift. Suppose there are totally T tasks, the t-th task contains a set of $\{\mathbf{x}^t, \mathbf{y}^t\}_{t=1}^{|\mathcal{D}^t|}$, representing the training data pairs from the t-th task comprised of c^t new classes, where $\mathbf{x}^t$ denotes the input images and $\mathbf{y}^t$ the corresponding ground truth labels. While all $\mathbf{x}^t$ are *i.i.d* within $\mathcal{D}^t$, the overall training procedure does not abide by the *i.i.d* assumption, *i.e.*, $\mathcal{D}^i \cap \mathcal{D}^j = \varnothing$ s.t. $i \neq j$, since the input distribution shifts between tasks and labels change. When finishing the training on one task, its training data would be discarded, except that only a tiny number of samples can be stored in a memory buffer as exemplars. At the t-th task, we denote the exemplars as $\mathcal{M}^{1:t-1}$, which are comprised of $c^{1:t-1}$ old classes in total.

The popular pipeline of existing CL methods is to directly load the union of new class data $\mathcal{D}^t$ and a few exemplars $\mathcal{M}^{1:t-1}$ to train the model $f(\cdot;\theta)$. However, there are mainly two limitations of this pipeline when applied to continual learning: 1) due to the memory constraint, the performance is typically dependent on selecting representative exemplars [45]; 2) naive uniting a large amount of new data and a few exemplars causes high bias to recently learned tasks and quick forgetting of old tasks as the number of tasks grows [49]. These issues can be partly alleviated by some techniques like exemplar sampling strategies [12, 19, 32, 51, 52] or increasing the model capacity for learning new tasks [34, 43, 56, 57, 59], yet they could be sub-optimal because of requiring carefully designing the selecting of exemplars or growing network parameters. A desirable continual learning system should scale to a large number of tasks without excessive human prior and increasing parameter overheads.

3.2. Bilateral Memory Consolidation

We argue that the above limitations can be mitigated by sufficiently mining structured knowledge from memory samples. To this end, we develop *Bilateral Memory Consolidation* called BiMeCo. BiMeCo explicitly decouples the memory mechanism into two parts: short-term phase and long-term phase, with the goal of consolidating the structured knowledge across all learned tasks by training on online balanced data while improving the representational capacity. Specifically, short-term phase focuses on expressiveness with a large amount of training samples and rapid adaptation to recent tasks to form working memory. Concurrently, long-term phase learns a few task-balance samples online to consolidate working memory for guiding the short-term phase to prevent forgetting, while inheriting strong expressiveness from the short-term phase. In this way, BiMeCo encourages dynamic interactions between different degrees of memories to produce rich representations to preserve scalability over learning new tasks and generality over the learned tasks synchronously.

Modeling. Formally, given the model $f(\cdot;\theta)$, we disentangle the model parameters θ as short-term memory module θ_s, long-term memory module θ_l, as well as a shared fully-connected layer ϕ. The left part of Fig. 2 illustrates this modeling. Our underlying motivation is that, although short-term memory module and long-term memory module are responsible for learning the task sequence from different aspects, *i.e.*, representational capacity and mem-

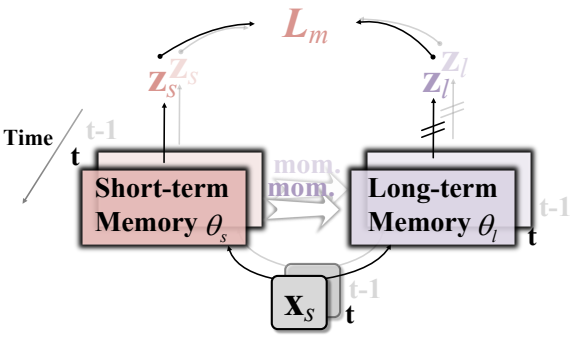

Figure 3. Sketch of bilateral memory consolidation. "mom.": the momentum-based update rule from short-term memory module to long-term memory module after each task in Eq. (3).

ory consolidation, they nevertheless have a uniform goal of performing classification on the learned task sequence. This makes cross-memory knowledge transferring feasible. Meanwhile, a shared classification head as a unified interface can provide a category-consistency constraint for encouraging their high-level information interaction for boosting the final classification.

In Fig. 2, we illustrate this process with a model comprised of L modules (*e.g.*, ResNet with L stacked residual blocks [26]) as an example. At each task, the short-term memory $f(\cdot;\theta_s)$ receives $\mathbf{x}_s$ as input, which is the union of a training batch $\mathcal{B}^t$ from the data stream $\mathcal{D}^t$ and a random batch $\mathcal{B}_{M_1}$ from a few exemplars $\mathcal{M}^{1:t-1}$, *i.e.*, $\mathbf{x}_s = \mathcal{B}^t \cup \mathcal{B}_{M_1}$. In the meantime, the long-term memory $f(\cdot;\theta_l)$ samples a small portion of samples $\mathcal{B}_b^t$ from $\mathcal{D}^t$ (line 5 in Alg. 1), then unites these samples with a random batch $\mathcal{B}_{M_2}$ from $\mathcal{M}^{1:t-1}$ to construct the class-balanced $\mathbf{x}_l$ between old tasks and new tasks as input, *i.e*, $\mathbf{x}_l = \mathcal{B}_b^t \cup \mathcal{B}_{M_2}$. Finally, a shared fully-connected layer $h(\cdot;\phi)$ is used to output the network predictions. To encourage knowledge interactions of two types of memories, the bilateral memory consolidation mechanism is introduced, as described next.

Memory consolidation. We first employ a distillation loss to realize the guidance from long-term memory (as a teacher) to short-term ones (as a student) for preventing forgetting. Specifically, taking $\mathbf{x}_s$ as input, the output embeddings of each module $f_\ell(\mathbf{x}_s;\theta_l)$ and $f_\ell(\mathbf{x}_s;\theta_s)$ are used to compute the constraints by minimizing their L2 distance, where $\ell = \{1,\dots,L\}$ and L denotes the number of modules. Motivated by recent studies [22, 23, 28], we employ the pooling operation over their output embeddings along horizontal and vertical dimensions respectively for better generality across tasks. Given an embedding tensor[1] $\mathbf{z} \in \mathbb{R}^{H\times W\times C}$, the extracted embedding Ψ by horizontal and vertical pooling can be defined as

$$\Psi(\mathbf{z}) = \left[\frac{1}{W}\sum_{w=1}^{W}\mathbf{z}[:,w,:] \middle\| \frac{1}{H}\sum_{h=1}^{H}\mathbf{z}[h,:,:]\right], \quad (1)$$

[1]For simplicity, we omit the mini-batch dimension in this notation.

Algorithm 1: BiMeCo (in the t-th task)

Input: Data stream $\mathcal{D}^t$; buffer $\mathcal{M}^{1:t-1}$; the number of new and old classes c^t, $c^{1:t-1}$; model parameters $\theta = \{\theta_s, \theta_l, \phi\}$; momentum coefficient m.

Output: Buffer $\mathcal{M}^t$, model parameters θ.

1 **for** mini-batches in $\mathcal{D}^t \cup \mathcal{M}^{1:t-1}$ **do**
2 $\quad\mathcal{B}^t \subseteq \mathcal{D}^t$ *// $\mathcal{B}_t$ is one sampled batch from $\mathcal{D}^t$.*;
3 $\quad\{\mathcal{B}_{M_1}, \mathcal{B}_{M_2}\} \subseteq \mathcal{M}^{1:t-1}$ *// $\mathcal{B}_{M_1}$, $\mathcal{B}_{M_2}$ denote different sampled batches from $\mathcal{M}^{1:t-1}$.* ;
4 $\quad$Uniting new samples $\mathcal{B}^t$ and limited exemplers $\mathcal{B}_{M_1}$ to obtain $\mathcal{B}^t \cup \mathcal{B}_{M_1}$ for training θ_s;
5 $\quad$Randomly sampling $c^t/(c^t + c^{1:t-1}) \cdot |\mathcal{B}^t|$ new samples $\mathcal{B}_b^t$ from $\mathcal{D}^t$ to form class-balanced $\mathcal{B}_b^t \cup \mathcal{B}_{M_2}$ for trianing θ_l;
6 $\quad\mathbf{z}_s, \mathbf{z}_l \leftarrow f(\mathcal{B}^t \cup \mathcal{B}_{M_1};\theta_s), f(\mathcal{B}_b^t \cup \mathcal{B}_{M_2};\theta_l)$ *// z_s, z_l denote the concatenated outputs of L modules over the channel axis of θ_s, θ_l.*;
7 $\quad$Update model parameters $[\theta_s, \theta_l, \phi]$ on $\{\mathcal{B}^t \cup \mathcal{B}_{M_1}, \mathcal{B}_b^t \cup \mathcal{B}_{M_2}, \mathbf{z}_s, \mathbf{z}_l\}$ by Eq. 6;
8 **end**
9 $\theta_l \leftarrow m\theta_l + (1-m)\theta_s$;
10 Update buffer $\mathcal{M}^t$, *e.g.*, by herding [29, 51];
11 Replace $\mathcal{M}^{1:t-1}$ with $\mathcal{M}^{1:t-1} \cup \mathcal{M}^t$ in the memory.

where $[\cdot \| \cdot]$ denotes concatenation over the channel axis. In our BiMeCo, this output embedding is computed at each module, for both long-term memory and short-term memory. Interestingly, consistent with self-supervised methods [10, 68], we observe that applying a stop-gradient (sg) operator on the long-term memory (teacher) to propagate gradients only through the short-term memory (student) can further boost the performance (Sec. 4.3). Formally, this distillation loss can be described as

$$\mathcal{L}_{\mathrm{m}}(\theta_s) = \frac{1}{L}\sum_{\ell=1}^{L}\left\|\Psi\left(f_\ell(\mathbf{x}_s;\theta_l)\right) - \Psi\left(f_\ell(\mathbf{x}_s;\theta_s)\right)\right\|^2. \quad (2)$$

Second, the short-term phase transfers its strong expressiveness to the long-term ones, in order to endow the long-term phase with providing better guidance. We draw inspiration from the momentum encoder [25] and develop a momentum-based update rule – after each task, the long-term memory weights are updated by the following form

$$\theta_l \leftarrow m\theta_l + (1-m)\theta_s, \quad (3)$$

where $m \in [0,1]$ is the momentum coefficient. Particularly, different from the momentum encoder, where only the parameters θ_s are updated by the back-propagation, we further relax this constraint to support the synchronous gradient update of two types of memory, *i.e.*, θ_s and θ_l. One underlying assumption is that, by back-propagating through the

Method	Params (M)	CIFAR-100 (B=50)			ImageNet-100 (B=50)			ImageNet (B=100)	
		S=10	5	2	10	5	2	100	50
LWF [42]	11.7	54.01±0.46	48.40±0.65	45.49±0.34	53.62†	47.64†	44.32†	41.42±0.11	28.31±0.10
iCaRL [51]	11.7	67.16±0.20	60.54±0.23	54.50±0.36	65.44†	59.88†	52.97†	49.88±0.15	42.52±0.13
DualNet [48]	13.8	68.01±0.21	63.42±0.33	63.22±0.40	71.36±0.30	67.21±0.35	66.35±0.29	55.80±0.11	51.96±0.19
AANet [43]	13.1	68.85±0.28	65.68±0.30	63.56±0.35	72.08±0.20	69.25±0.41	67.41±0.38	51.88±0.13	47.91±0.14
Replay	11.7	50.07±0.31	46.80±0.35	44.79±0.28	51.80±0.30	47.71±0.27	43.82±0.25	42.77±0.15	39.93±0.13
+BiMeCo (ours)	6.2	51.36±0.25	48.25±0.31	46.99±0.30	52.81±0.21	48.86±0.18	45.51±0.20	43.86±0.16	41.61±0.14
BiC [61]	11.7	63.11±0.20	56.27±0.25	48.83±0.16	70.09±0.34	64.88±0.26	57.82±0.22	52.46±0.17	47.30±0.12
+BiMeCo (ours)	6.2	64.17±0.13	58.64±0.32	52.66±0.22	71.01±0.30	65.97±0.21	59.35±0.19	53.55±0.14	49.41±0.19
LUCIR [29]	11.7	66.31±0.31	60.84±0.27	53.01±0.24	70.86±0.40	67.85±0.32	63.01±0.23	56.47±0.11	52.55±0.16
+BiMeCo (ours)	6.2	**69.87±0.19**	**66.82±0.21**	**64.16±0.22**	**72.87±0.25**	69.91±0.16	**67.85**±0.18	57.36±0.10	53.84±0.12
CwD [55]	11.7	67.29±0.18	62.99±0.12	56.84±0.20	71.93±0.23	69.42±0.27	65.06±0.46	57.49±0.09	53.51±0.15
+BiMeCo (ours)	6.2	69.23±0.20	65.78±0.25	62.73±0.17	72.81±0.20	**70.64±0.22**	66.20±0.41	**58.30±0.16**	**54.91±0.14**

Table 1. Comparison of average incremental accuracy (%) with or without Bilateral Memory Consolidation (BiMeCo). B denotes the number of classes learned in the initial task and S denotes the number of classes learned per task after the initial one. For fair comparison, Params denotes the model parameters on ImageNet [20]. Note that the number of exemplars for each class is set to 20, as is common practice [29, 55]. The results denoted by † are taken from [55]. We report the results that are averaged over 3 runs (mean±std).

long-term memory trained on online class-balanced data, the long-term memory can obtain powerful generalization ability across all learned tasks. We study different updating rules in Sec. 4.3 and show updating two memory branches synchronously by back-propagation works better than only updating the short-term memory, indicating that the learned generalization across tasks of long-term memory is the key to providing guidance for preventing forgetting. Fig. 3 highlights the process of our bilateral memory consolidation.

3.3. BiMeCo for Continual Learning

Architecture. Technically, we implement BiMeCo with a Pseudo Siamese network, which has two branches $f(\cdot;\theta_s)$ and $f(\cdot;\theta_l)$ with identical architectures but different weights as well as a shared head $h(\cdot;\phi)$. As a variant of the standard Siamese network [8], it has been widely used in diverse vision tasks [6, 15, 24, 58, 67]. To make BiMeCo compatible with existing network architectures while enjoying great parameter efficiency, we draw inspiration from the MobileNet series [30, 31, 54], and present the following module replacement strategies: 1) replacing each original $k \times k$ ($k > 1$) convolution with a sequential convolutional layer comprised of 1×1 convolution, $k \times k$ depthwise convolution [17], 1×1 convolution; 2) reducing group by a factor of 4 for each original 1×1 convolution. For the generality of our method, all replaced convolutions do not change the channel dimension. Then the network is extended into a Pseudo Siamese network. Details are in Appendix.

Training. In Alg. 1, we summarize the overall training process of the proposed BiMeCo in the t-th task (where $t \in [1, \ldots, T]$). Formally, the object of the proposed BiMeCo can be formulated as

$$\mathcal{L}_l(\theta_l) = -\mathbf{y}_l \log(h(f(\mathbf{x}_l;\theta_l);\phi)), \tag{4}$$

$$\mathcal{L}_s(\theta_s) = -\mathbf{y}_s \log(h(f(\mathbf{x}_s;\theta_s);\phi)), \tag{5}$$

$$(\theta_l^*, \theta_s^*, \phi^*) = \arg\min_{\theta_l,\theta_s,\phi} [\mathcal{L}_l + \lambda_1 * \mathcal{L}_s + \lambda_2 * \mathcal{L}_m], \tag{6}$$

where $\mathbf{y}_l$, $\mathbf{y}_s$ denote the ground truth labels for $\mathbf{x}_l$, $\mathbf{x}_s$ respectively, λ_1, λ_2 are the hyper-parameters to balance the trade-off between three terms.

4. Experiments

In this section, we first introduce the experimental settings and implementation details in Sec. 4.1. Then, we integrate our BiMeCo into some state-of-the-art methods on three challenging benchmarks to evaluate its effectiveness in Sec. 4.2. Finally, comprehensive ablation studies are provided to analyze BiMeCo thoroughly in Sec. 4.3.

4.1. Experimental Settings

Following [7, 18, 45, 48], we focus on two typical continual learning setups: Class Incremental Learning (CIL) and Tasks Incremental Learning (TIL), which typically unfold a base classification problem in successively learned tasks. In two setups, all training classes are split into multiple tasks and learned sequentially. The difference is that TIL has access to the task identity of test samples at inference time, but CIL is not allowed. Due to the space restriction, here we report the results of the CIL setting, the results of the TIL setting are in Appendix.

Datasets. Three datasets are used in our experiments: CIFAR-100 [39], ImageNet [20], and ImageNet-100 [20]. Details of these datasets are in Appendix.

m	0	0.01	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Acc. (%)	61.41	63.10	63.42	63.62	63.47	63.58	63.75	62.53	61.38	60.72	58.65	*fail*

(a) Only updating one branch (short-term memory) by back-propagation as in [25].

m	0	0.01	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Acc. (%)	66.25	66.29	66.82	66.61	66.38	66.25	66.15	66.06	65.86	65.75	65.67	65.44

(b) Synchronously updating two branches (two types of memory) by back-propagation.

Table 2. Comparison of different settings of m in Eq. (3) under ResNet-18 on CIFAR-100.

Baselines. We combine the proposed BiMeCo with the following advanced baselines: BiC [61], LUCIR [29], CwD [55], as well as a naive baseline fine-tuning with exemplars (Replay). Additionally, we compare BiMeCo with some classic methods (LWF [42] and iCaRL [51]) and strong expansion-based methods (DualNet [48] and AANet [43]). For CwD [55] and AANet [43], we adopt the default settings in the original paper such as using their versions based on LUCIR [29] for fair comparison.

Implementation details. Following the common practice [44, 55], we use ResNet-18 [26] as the baseline architecture. Herding [43, 51] is employed to select exemplars after each task. More details are in Appendix.

4.2. Results

Extensive experiments are conducted on CIFAR-100, ImageNet-100, and ImageNet. Specifically, we add our BiMeCo to several representative state-of-the-art methods to evaluate its effectiveness. Notably, B denotes the number of classes learned in the initial task, and S denotes the number of new classes learned per task in the rest.

The quantitative comparisons are summarized in Tab. 1. From the results, we observe that BiMeCo consistently brings significant improvements. Taking LUCIR [29] as an example, BiMeCo achieves better performance with +3.56%$\sim$11.15% validation accuracy gains on CIFAR-100, while significantly reducing the model parameters by 47%. Particularly, as the length of the task sequence increments, our BiMeCo can bring significant performance gains. For example, BiMeCo brings +5.89% and +3.83% validation accuracy gains over CwD [29] and BiC [61] when $S = 2$, which demonstrates that our method does better work with longer incremental learning sequences. Our method also shows powerful performance for scenarios with a large number of classes such as ImageNet. BiMeCo surpasses the baseline method fine-tuning with exemplars (Replay) by a large margin ($+1.09\%$ when $S = 100$ and $+1.68\%$ when $S = 50$) using 1.9x fewer trainable parameters.

Moreover, we illustrate the comparisons of the accuracy curve with LUCIR [29] and CwD [55] on CIFAR-100 in Fig. 4(h). It can be seen that, in the remaining tasks after the initial task, the improvement of our method is consistently incremental. As the number of classes increases, our method owns growing significant advantages. This is reasonable since the limited model size of our BiMeCo leads to a slight drop in the first task. Benefiting from the bilateral memory consolidation mechanism, our model can sufficiently learn long-term and short-term memory representations in a complementary fashion, thus exhibiting strong capacities of preventing forgetting knowledge previously learned. More results of the accuracy curve on other incremental settings are in Appendix.

4.3. Ablation Study

In this subsection, extensive ablation studies are conducted to analyze our BiMeCo systematically. Specifically, experiments are based on LUCIR [29] under ResNet-18 on CIFAR-100, learning with 50 classes in the initial task and 5 classes per task for the rest, unless otherwise specified.

Analysis of bilateral memory consolidation. First, to reveal the effect of different strategies of the guidance from long-term memory to short-term ones for preventing forgetting, we compare BiMeCo with four additional and important baselines: 1) "+None", in which the two types of memory are trained without the regularization constraint; 2) "+L2", in which the short-term memory is trained by adding an L2 regularization constraint with the long-term memory on each weight; 3) "+Wei.", in which the short-term memory is trained by estimating important parameters with the long-term memory, similar to [38]; 4) "+Out.", in which the short-term memory is trained by minimizing its output probabilities with those of the long-term memory, similar to [42]. As shown in Fig. 4(a), we can observe that 1) compared with the L2 regularization, estimating important parameters obtains better performance (64.00% *vs.* 65.51%), where a similar phenomenon also occurs in [38]; 2) the data distillation manners (+Out. and BiMeCo) work better than weight regularization (+L2. and +Wei.), implying that data distillation is more suitable for our proposed framework.

Second, to transfer the strong expressiveness of the short-term memory to the long-term ones, we further investigate the effectiveness of our used momentum-based update rule in Eq. (3). Specifically, we consider the original momentum update that is proposed and fueled by the MoCo series [14, 16, 25], only updating one branch by back-propagation. Then we introduce the relaxed version of this momentum-based update rule, which synchronously updates two branches by back-propagation. The results in Tab. 2 shows that the relaxed momentum update is particularly well-suited for BiMeCo. Under the relaxed case, when

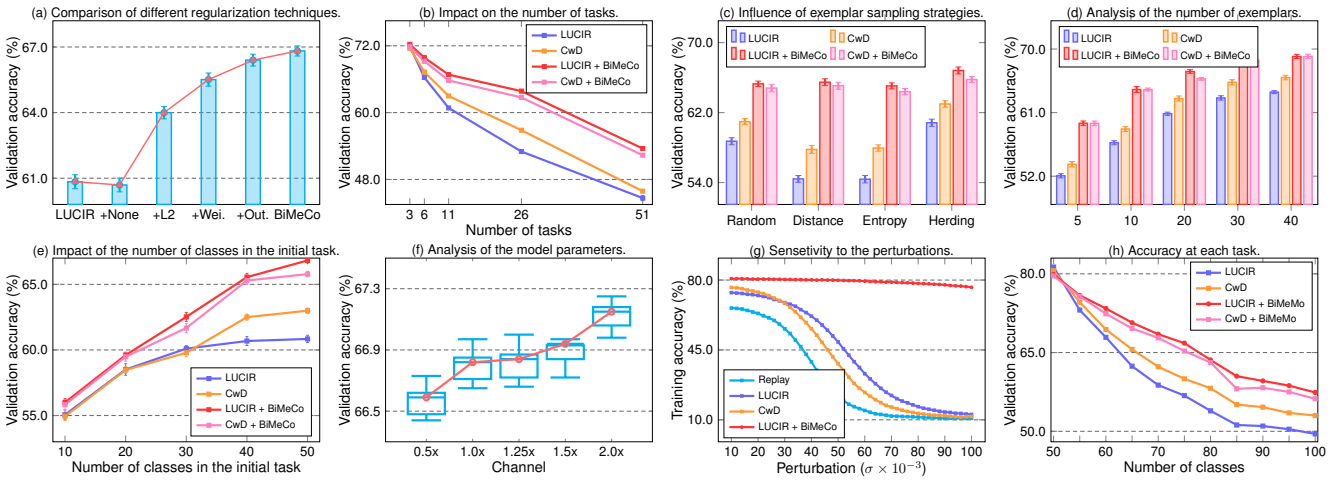

Figure 4. Further analysis of the proposed BiMeCo. (a) Comparisons of different regularization techniques. (b) The impact of the number of tasks that vary from 3, 6, 11, 26, and 51. Note that the X-axis represents the total number of tasks, where the first task covers 50 classes and the rest tasks equally divide the rest 50 classes. (c) Influence of exemplar sampling strategies. (d) Influence of the number of exemplars, varying from 5, 10, 20, 30, and 40 per class. (e) Impact of the number of classes in the initial task. (f) Ablation on the model parameters of BiMeCo. (g) The training accuracy under varying Gaussian noise added to the weights of each layer of the trained model. (h) Accuracy at each task. With 50 classes in the initial task and 5 classes per task for the rest. Best viewed in color.

the long-term memory is not updated at all ($m = 1$), the model fails to converge. If there is no momentum ($m = 0$), it leads to a drop in accuracy. At a relatively small momentum ($m = 0.1$), BiMeCo achieves the best performance. These results support our motivation for designing the relaxed momentum update rule for inheriting the representation ability from short-term memory to long-term ones. The effectiveness of bilateral memory consolidation is also verified under other incremental settings in Appendix.

Analysis of two types of memory. Tab. 3 lists the average incremental accuracy and average forgetting[2] for thoroughly analyzing long-term memory and short-term memory. Compared with the baseline method LUCIR [29], one can observe that the variant of LUCIR (denoted by LUCIR†), which naively extends the backbone to the Pseudo Siamese network as described in Sec. 3.3, exhibits slightly inferior performance due to the limited model capacity. Additionally, we add two important baselines, BiMeCo$^{\text{L}}$ and BiMeCo$^{\text{S}}$, which denote the variants of BiMeCo without short-term memory and without long-term memory, respectively. Of particular interest, we observe that BiMeCo$^{\text{S}}$ achieves decent accuracy but performs poorly in denying forgetting, while BiMeCo$^{\text{L}}$ can prevent forgetting well. This result supports our hypothesis in Sec. 3.2, *i.e.*, long-term memory can obtain strong generalization ability across all learned tasks through training on online class-balanced data by back-propagating. Besides, in diverse incremental settings, one can observe that BiMeCo is superior to its variant BiMeCo*, where only the samples

[2]For convenience of analysis, we show the average forgetting, which is calculated as the difference between the peak accuracy and the final accuracy of each task [47, 48].

Method	#Params	S=6		S=11		S=26	
		Acc. ↑	For. ↓	Acc. ↑	For. ↓	Acc. ↑	For. ↓
LUCIR [29]	11.2M	66.31	19.02	60.84	22.95	53.01	27.65
LUCIR†	5.8M	66.00	19.71	60.40	23.48	52.07	29.28
BiMeCo$^{\text{S}}$	3.1M	65.76	20.64	60.21	24.09	51.65	29.99
BiMeCo$^{\text{L}}$	3.1M	64.66	**13.30**	56.69	**12.19**	45.58	**10.01**
BiMeCo*	5.8M	68.74	14.32	65.62	18.81	62.90	24.07
BiMeCo	5.8M	**69.87**	13.78	**66.82**	17.15	**64.16**	22.56

Table 3. Ablation study of BiMeCo (%). #Params denotes the model parameters on CIFAR-100. For each metric, ↑ (↓) indicates that the larger (the smaller) values, the better results are.

of old tasks are used to train long-term memory. This shows that the effectiveness of our dynamically updated teacher network, as stated in Sec. 2.2.

Impact on the number of tasks. We vary the number of tasks and implement LUCIR [29] and CwD [55] with or without our BiMeCo. As shown in Fig. 4(b), the X-axis represents the total number of tasks (denoted as n), in which 50 classes are trained in the initial task, and the remaining classes are equally divided into $n - 1$ tasks. We can observe that as the length of the task sequence increases, all methods are prone to forgetting old tasks, resulting in a drop in overall performance. However, our method drops more smoothly and outperforms the baselines in each case, especially when the number of tasks is larger. For example, when $n = 51$, our BiMeCo achieves superior results over LUCIR [29] and CwD [55], which brings +8.90% and +6.46% validation accuracy improvements, respectively. This convincingly demonstrates that the proposed BiMeCo does better work in continual learning, especially when tackling the highly long task sequence.

Influence of exemplar sampling strategies. Previous

methods [29, 55] have proven that reserving a few samples is helpful for maintaining model performance on old classes. However, as illustrated in Fig. 4(c), different sampling strategies perform unstably on LUCIR [29] and CwD [55]. For example, LUCIR [29] and CwD [55] perform unsatisfying in the distance and entropy sampling. On the contrary, our method outperforms the baselines comprehensively. Taking LUCIR [29] as an example, BiMeCo brings +6.56%, +11.06%, +10.67% and +5.98% validation accuracy gains under four sampling strategies, respectively. The dramatic improvements demonstrate the capability of our BiMeCo to be insensitive to sampling strategies.

Analysis of the number of exemplars. We investigate the impact of reserving a few samples of old classes on model performance. The results in Fig. 4(d) show that all methods can obtain performance gains as the number of exemplars increases. Specifically, our proposed BiMeCo outperforms the baselines in each case, especially when the number of reserved samples is quite small. Notably, integrated with BiMeCo, the baselines with reserving 20 exemplars per class suffices to surpass those of their vanilla version by utilizing 40 exemplars per class, which further verifies the power of our proposed method.

Impact of the number of classes in the initial task. In this ablation, we discuss the effect of the number of classes learned in the initial task on the average incremental accuracy. As shown in Fig. 4(e), the X-axis represents the number of classes in the initial task, and the remaining classes are incremented with 5 classes per task. We can find that all methods can achieve better performance when the number of classes in the initial task increase. Particularly, our BiMeCo consistently outperforms other methods, with validation accuracy gains of +0.89%∼5.98%.

Analysis of the model parameters. By varying the number of intermediate channels, we investigate the effect of model size on performance. For example, 0.5x represents reducing the intermediate channel dimension of modules (*i.e.*, the residual blocks in ResNet-18 [26]) by a factor of 0.5. From Fig. 4(f), we can see that BiMeCo achieves better performance with the increased parameters. Additionally, compared with prior methods, our method shows significant advantages in accuracy with great parameter efficiency. A vivid example is that BiMeCo with reducing channels dimension by 0.5x still achieves a superior result over LUCIR [29], which brings in a gain of +5.75% accuracy with a dramatic reduction of parameters by 73.5%.

Sensitivity to the perturbation. Following the common practice [3, 66], we add the Gaussian noise to the weights of each layer of the model to explore the model generalization in Fig. 4(g). Compared with Replay, LUCIR [29], and CwD [55], our method is less sensitive to perturbations, which reflects the characteristic of converging to flatter minima [36, 66]. Specifically, the accuracy of three baselines encounters devastating damages (-55.2%∼-64.6%), while our BiMeCo shows only a slight drop (at most -4.3%). This demonstrates that our method can facilitate the optimization converges to a sharp minimum.

Model	Distillation	Pooling	SG	Mom.	acc.
A					60.69
B	✓				64.53
C	✓	✓			65.09
D	✓	✓	✓		66.25
E				✓	61.74
F	✓			✓	65.75
G	✓	✓		✓	66.07
H	✓	✓	✓	✓	**66.82**

Table 4. Ablation study of each component of BiMeCo (%). "Mom." denotes using momentum update in our method. "SG" denotes the stop gradient operation.

Effect of different components. Tab. 4 shows the results of the ablation on knowledge distillation (Eq. (2)), pooling (Eq. (1)), stop gradient (Eq. (2)), and momentum update (Eq. (3)). From the results, we can observe that: 1) based on model A, all operations bring positive gains. For example, momentum update improves accuracy gain by +1.05%; 2) distillation-based guidance from short-term memory to long-term memory is a key component in our method, which brings a gain of +3.84% accuracy.

5. Conclusion

In this work, we present BiMeCo, a simple yet efficient framework for continual learning (CL). From the perspective of sufficiently mining memory samples to form generic knowledge, BiMeCo is designed to explicitly decouple model parameters into long-term memory module and short-term memory module. By encouraging their dynamic interactions, BiMeCo can preserve scalability over learning new tasks and generality over all learned tasks synchronously. Extensive experiments show that BiMeCo is lightweight and can be integrated into existing CL methods seamlessly with further performance gains.

Limitation. Similar to most CL methods, our BiMeCo focuses on image classification and works well on it, but how to extend it to other tasks such as object detection remains an open question in this field. Our work provides a feasible solution to prevent forgetting while reducing parameter overheads. However, there are still unexplored issues, such as entirely removing the reliance on exemplars.

6. Acknowledgements

This research was supported by the National Key Research and Development Program of China under Grant No. 2018AAA0100400, the National Natural Science Foundation of China under Grants 62071466, 62076242, 61976208, and the InnoHK project.

References

[1] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, pages 139–154, 2018. 2

[2] R. Aljundi, P. Chakravarty, and T. Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *CVPR*, pages 3366–3375, 2017. 2

[3] E. Arani, F. Sarfraz, and B. Zonooz. Learning fast, learning slow: A general continual learning method based on complementary learning system. In *ICLR*, 2022. 8

[4] J. Bang, H. Kim, Y. Yoo, J. Ha, and J. Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, pages 8218–8227, 2021. 2

[5] E. Belouadah and A. Popescu. Il2m: Class incremental learning with dual memory. In *ICCV*, pages 583–592, 2019. 1

[6] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, and P.HS. Torr. Fully-convolutional siamese networks for object tracking. In *ECCV*, pages 850–865, 2016. 5

[7] M. Boschini, L. Bonicelli, P. Buzzega, A. Porrello, and S. Calderara. Class-incremental continual learning into the extended der-verse. *IEEE TPAMI*, 2022. 5

[8] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a" siamese" time delay neural network. *NeurIPS*, 6, 1993. 5

[9] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *SIGKDD*, pages 535–541, 2006. 2

[10] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, pages 9650–9660, 2021. 3, 4

[11] F.M. Castro, M.J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari. End-to-end incremental learning. In *ECCV*, pages 233–248, 2018. 3

[12] A. Chaudhry, P.K. Dokania, T. Ajanthan, and P.H. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, pages 532–547, 2018. 2, 3

[13] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny. Efficient lifelong learning with a-gem. In *ICLR*, 2018. 2, 3

[14] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 6

[15] X. Chen and K. He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021. 5

[16] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *ICCV*, pages 9640–9649, 2021. 6

[17] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, pages 1251–1258, 2017. 5

[18] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE TPAMI*, 44(7):3366–3385, 2021. 2, 5

[19] M. De Lange and T. Tuytelaars. Continual prototype evolution: Learning online from non-stationary data streams. In *ICCV*, pages 8250–8259, 2021. 3

[20] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[21] P. Dhar, R.V. Singh, K.C. Peng, Z. Wu, and R. Chellappa. Learning without memorizing. In *CVPR*, pages 5138–5146, 2019. 3

[22] A. Douillard, Y. Chen, A. Dapogny, and M. Cord. Plop: Learning without forgetting for continual semantic segmentation. In *CVPR*, pages 4040–4050, 2021. 4

[23] A. Douillard, M. Cord, C. Ollion, T. Robert, and E. Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pages 86–102, 2020. 3, 4

[24] K. Fu, D. Fan, G. Ji, Q. Zhao, J. Shen, and C. Zhu. Siamese network for rgb-d salient object detection and beyond. *IEEE TPAMI*, 2021. 5

[25] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020. 4, 6

[26] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1, 2, 3, 4, 6, 8

[27] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *preprint arXiv:1503.02531*, 2015. 2

[28] Q. Hou, L. Zhang, M. Cheng, and J. Feng. Strip pooling: Rethinking spatial pooling for scene parsing. In *CVPR*, pages 4003–4012, 2020. 4

[29] S. Hou, X. Pan, C.C. Loy, Z. Wang, and D. Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pages 831–839, 2019. 1, 2, 3, 4, 5, 6, 7, 8

[30] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, pages 1314–1324, 2019. 5

[31] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 5

[32] D. Isele and A. Cosgun. Selective experience replay for lifelong learning. In *AAAI*, volume 32, 2018. 3

[33] Heechul Jung, Jeongwoo Ju, Minju Jung, and Junmo Kim. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*, 2016. 2

[34] M. Kanakis, D. Bruggemann, S. Saha, S. Georgoulis, A. Obukhov, and L.V. Gool. Reparameterizing convolutions for incremental multi-task learning without task interference. In *ECCV*, pages 689–707, 2020. 3

[35] R. Kemker and C. Kanan. Fearnet: Brain-inspired model for incremental learning. In *ICLR*, 2018. 2

[36] N.S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P.T.P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017. 8

[37] C.D. Kim, J. Jeong, and G. Kim. Imbalanced continual learning with partitioning reservoir sampling. In *ECCV*, pages 411–428, 2020. 2

[38] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A.A. Rusu, K. Milan, J. Quan, T. Ramalho, Grabska-B.A., et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526, 2017. 3, 6

[39] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1, 2, 5

[40] K. Lee, K. Lee, J. Shin, and H. Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *ICCV*, pages 312–321, 2019. 2

[41] S.W. Lee, J.H. Kim, J. Jun, J.W. Ha, and B.T. Zhang. Overcoming catastrophic forgetting by incremental moment matching. volume 30, 2017. 2

[42] Z. Li and D. Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017. 1, 2, 5, 6

[43] Y. Liu, B. Schiele, and Q. Sun. Adaptive aggregation networks for class-incremental learning. In *CVPR*, pages 2544–2553, 2021. 2, 3, 5, 6

[44] D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. volume 30, 2017. 2, 3, 6

[45] M. Masana, X. Liu, B. Twardowski, M. Menta, A.D. Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE TPAMI*, 2022. 1, 2, 3, 5

[46] M. McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychol. Learn. Motiv.*, volume 24, pages 109–165. 1989. 2

[47] S.I. Mirzadeh, A. Chaudhry, D. Yin, T. Nguyen, R. Pascanu, D. Gorur, and M. Farajtabar. Architecture matters in continual learning. *arXiv preprint arXiv:2202.00275*, 2022. 7

[48] Q. Pham, C. Liu, and S. Hoi. Dualnet: Continual learning, fast and slow. volume 34, pages 16131–16144, 2021. 2, 5, 6, 7

[49] J. Rajasegaran, S. Khan, M. Hayat, F.S. Khan, and M. Shah. itaml: An incremental task-agnostic meta-learning approach. In *CVPR*, pages 13588–13597, 2020. 3

[50] A. Rannen, R. Aljundi, M. B. Blaschko, and T. Tuytelaars. Encoder based lifelong learning. In *ICCV*, pages 1320–1328, 2017. 2

[51] S.A. Rebuffi, A. Kolesnikov, G. Sperl, and C.H. Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 1, 2, 3, 4, 5, 6

[52] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne. Experience replay for continual learning. *NeueIPS*, 32, 2019. 3

[53] 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. 2

[54] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 5

[55] Y. Shi, K. Zhou, J. Liang, Z. Jiang, J. Feng, P.H. Torr, S. Bai, and V.Y. Tan. Mimicking the oracle: An initial phase decorrelation approach for class incremental learning. In *CVPR*, pages 16722–16731, 2022. 1, 2, 5, 6, 7, 8

[56] P. Singh, P. Mazumder, P. Rai, and V.P. Namboodiri. Rectification-based knowledge retention for continual learning. In *CVPR*, pages 15282–15291, 2021. 3

[57] P. Singh, V.K. Verma, P. Mazumder, L. Carin, and P. Rai. Calibrating cnns for lifelong learning. *NeurIPS*, 33:15579–15590, 2020. 3

[58] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708, 2014. 5

[59] V.K. Verma, K.J. Liang, N. Mehta, P. Rai, and L. Carin. Efficient feature transformations for discriminative and generative continual learning. In *CVPR*, pages 13865–13875, 2021. 3

[60] L. Wang, X. Zhang, K. Yang, L. Yu, C. Li, L. Hong, S. Zhang, Z. Li, Y. Zhong, and J. Zhu. Memory replay with data compression for continual learning. In *ICLR*, 2022. 2

[61] Y. Wu, Y. Chen, L. Wang, Y. Ye, Z. Liu, Y. Guo, and Y. Fu. Large scale incremental learning. In *CVPR*, pages 374–382, 2019. 1, 2, 5, 6

[62] Y. Xiang, Y. Fu, P. Ji, and H. Huang. Incremental learning using conditional adversarial networks. In *ICCV*, pages 6619–6628, 2019. 2

[63] J. Xu and Z. Zhu. Reinforced continual learning. volume 31, 2018. 2

[64] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *ICML*, pages 3987–3995, 2017. 3

[65] J. Zhang, J. Zhang, S. Ghosh, D. Li, S. Tasci, L. Heck, H. Zhang, and C.C. Kuo. Class-incremental learning via deep model consolidation. In *WACV*, pages 1131–1140, 2020. 2

[66] Y. Zhang, T. Xiang, T.M. Hospedales, and H. Lu. Deep mutual learning. In *CVPR*, pages 4320–4328, 2018. 8

[67] Z. Zhang and H. Peng. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, pages 4591–4600, 2019. 5

[68] J Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. Yuille, and T. Kong. Image bert pre-training with online tokenizer. In *ICLR*, 2022. 4

[69] P. Zhou, L. Mai, J. Zhang, N. Xu, Z. Wu, and L.S. Davis. M2kd: Multi-model and multi-level knowledge distillation for incremental learning. *arXiv preprint arXiv:1904.01769*, 2019. 3