Appendix

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

A. Implementation Details

In this section, we describe experimental details and network configuration details used in our experiments.

A.1. Experimental Details

Three datasets are used in our experiments: CIFAR-100 [19], ImageNet [9], and ImageNet-100 [9]. Specifically, CIFAR-100 consists of 60,000 images covering 100 classes, where all the images are resized to 32×32. ImageNet contains 1.28M training images and 50K validation images from 1,000 classes that are resized to 224×224. ImageNet-100 is a 100-class subset of the full ImageNet, which is sampled as in [14, 17]. For fair comparison, we shuffle the classes of these datasets using seed 1,993 and then split them into multiple tasks as in [14, 17, 22, 27, 29].

Following the common practice [23, 29], we use ResNet-18 [13] as the baseline architecture. Herding [21, 27] is employed to select exemplars after each task. We train the model for 160 epochs using SGD optimizer with batch size 128 on CIFAR-100. The learning rate is initialized to 0.1 and is multiplied by 0.1 after 80 and 120 epochs. For the experiments based on ImageNet and ImageNet-100, we train the model for 90 epochs and the learning rate is multiplied by 0.1 after 30 and 60 epochs.

For evaluation metric, we adopt the average incremental accuracy as the evaluation metric following the exact settings in [14, 21, 22, 29], which is written as

$$\bar{A} = \frac{1}{N+1}\sum_{i=0}^{N} A_i, \tag{1}$$

where $N+1$ denotes the total number of tasks. A_i is the test accuracy in the i-th task.

A.2. Network Configuration Details

Technically, we implement BiMeCo with a Pseudo Siamese network, which has two branches: 1) $f(\cdot;\theta_s)$ for short-term memory and 2) $f(\cdot;\theta_l)$ for long-term memory, and a shared head $h(\cdot;\phi)$. Fig. 2 in the main paper illustrates this modeling. Two branches $f(\cdot;\theta_s)$ and $f(\cdot;\theta_l)$ have identical architectures but different parameters. As a variant of the standard Siamese network [3], it has been widely used in diverse vision tasks [1, 5, 11, 30, 32]. To make the proposed BiMeCo compatible with existing network architectures while enjoying great parameter efficiency, we draw inspiration from the MobileNet series [15, 16, 28], and present the following module replacement strategies: 1) replacing each original $k \times k$ $(k > 1)$ convolution layer with a sequential convolutional layer comprised of 1×1 convolution, $k \times k$ depthwise (DW) convolution [7], 1×1 convolution; 2) reducing group by a factor of 4 for each original 1×1 convolution layer. For the generality of our method, all replaced convolutions do not change the channel dimension. After replacing all convolutional modules, the network is extended into a Pseudo Siamese network, *i.e.*, two identical architectures with different parameters as well as a shared head. In Fig. I and Fig. II, we illustrate the module replacement strategies for $k \times k$ $(k > 1)$ convolution layers and 1×1 convolution layers in the convolutional network.

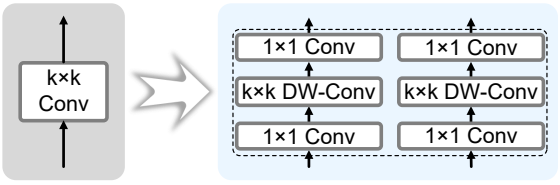

Figure I. Module replacement strategies for each $k \times k$ $(k > 1)$ convolution layer.

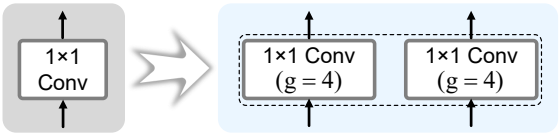

Figure II. Module replacement strategies for each 1×1 convolution layer. "g" denotes the number of group in convolution [15].

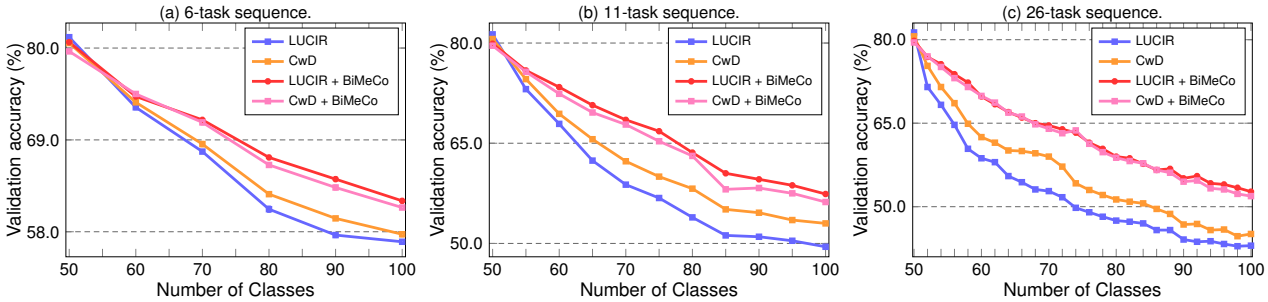

Figure III. Accuracy at each task. In the CIL setting, 50 classes are trained for the first task with later (a) 10 classes per task (6-task sequence), (b) 5 classes per task (11-task sequence), and (c) 2 classes per task (26-task sequence), respectively.

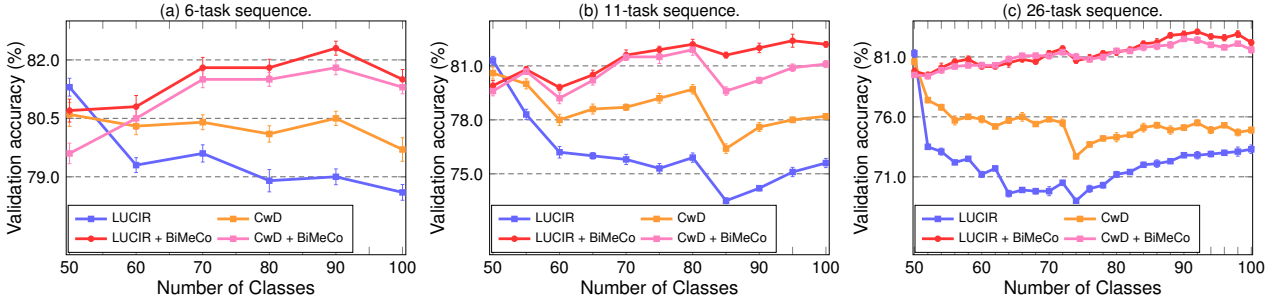

Figure IV. Accuracy at each task. In the TIL setting, 50 classes are trained for the first task with later (a) 10 classes per task (6-task sequence), (b) 5 classes per task (11-task sequence), and (c) 2 classes per task (26-task sequence), respectively.

B. More Comparisons

Following [2, 8, 24, 26], we focus on two typical continual learning setups: Class Incremental Learning (CIL) and Tasks Incremental Learning (TIL), which 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. In this section, we provide more results of analysis of accurate curve, comparison with state of the art in TIL, and analysis of bilateral memory consolidation.

B.1. Analysis of Accuracy Curve

In Fig. III and Fig. IV, we provide more comparisons of the average incremental accuracy curve with LUCIR [14] and CwD [29] on CIFAR-100. First, in the CIL setting (Fig. III), one can observe that the improvement of our proposed BiMeCO is consistently incremental in the remaining tasks after the initial task. As the number of classes increases, BiMeCO owns growing significant advantages. This is reasonable since the limited model size of BiMeCo leads to a slight drop in the first task. Benefiting from the bilateral memory consolidation mechanism, BiMeCO can sufficiently learn long-term and short-term memory representations in a complementary fashion, thus exhibiting strong capacities of preventing forgetting knowledge previously learned. The results in Fig. V also support this claim (the settings are the same as in Tab. 1 and Sec. 4.3). Second, in the TIL setting (Fig. IV), BiMeCo also brings significant accuracy gains, and surprisingly achieves growing performance as the task increases, suggesting that BiMeCo can greatly mitigate forgetting previously learned tasks with the help of access to the task identity.

B.2. Comparison with State of the Art

Here we compare the proposed BiMeCo with some state-of-the-art methods in the TIL setting. Tab. I lists the quantitative results. Particularly, 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. We can observe that BiMeCo consistently brings significant improvements. For example, BiMeCo brings +2.03%∼9.39% accuracy gains on CIFAR-100 over LUCIR [14], while significantly reducing the model parameters by 47%. Consistent with the phenomenon in the CIL setting, as the length of the task sequence increments, our BiMeCo can bring significant performance gains. A vivid example is that BiMeCo brings in gains of +5.71% and +1.07% accuracy over CwD [14] and BiC [31] when $S = 2$ respectively. Furthermore, in the large-scale setting of ImageNet, BiMeCo achieves superior results over BiC [31], which brings +0.92% and +1.04% accuracy gains at $S = 100$ and $S = 50$ respectively with 1.9x fewer model parameters.

B.3. Analysis of Bilateral Memory Consolidation.

In the main paper, we have already analyzed the bilateral memory consolidation in Sec. 4.3. Here we provide more

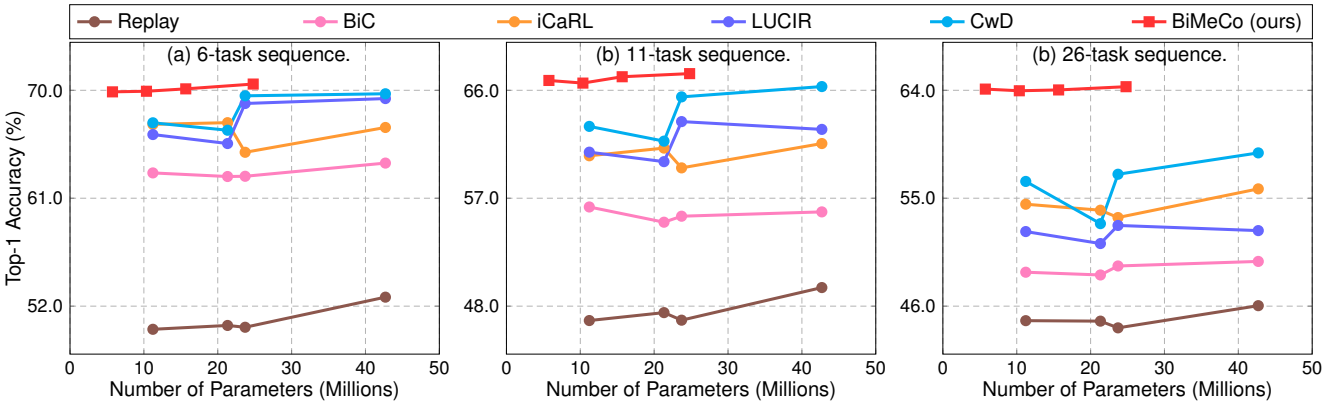

Figure V. Comparison of model size and average incremental accuracy. We use three typical settings on CIFAR-100 [19] in class incremental learning, where 50 classes are learned for the first task with later (a) 10 classes per task (6-task sequence), (b) 5 classes per task (11-task sequence), (c) 2 classes per task (26-task sequence). BiMeCo significantly outperforms other CL methods with 1.51∼2.06x smaller parameters under ResNet-18/34/50/101 [13].

Method	Params (M)	CIFAR-100 (B=50)			ImageNet-100 (B=50)			ImageNet (B=100)	
		S=10	5	2	10	5	2	100	50
LWF [20]	11.7	75.05±0.39	69.50±0.45	67.99±0.36	73.67±0.27	69.73±0.34	66.60±0.38	69.63±0.15	70.52±0.16
iCaRL [27]	11.7	79.59±0.23	76.00±0.30	73.65±0.21	77.42±0.36	76.02±0.29	73.59±0.20	75.95±0.14	77.60±0.12
DualNet [26]	13.8	80.35±0.34	79.30±0.38	79.16±0.31	83.50±0.26	83.63±0.30	82.99±0.24	78.60±0.17	80.32±0.13
AANet [21]	13.1	80.66±0.24	80.31±0.35	80.52±0.27	83.62±0.22	84.51±0.20	83.42±0.28	77.21±0.16	79.89±0.11
Replay	11.7	67.90±0.33	66.11±0.45	66.24±0.38	71.97±0.27	70.15±0.31	69.20±0.29	72.96±0.14	73.72±0.12
+BiMeCo (ours)	6.2	70.25±0.29	67.19±0.35	67.40±0.37	72.68±0.25	70.96±0.23	70.21±0.28	73.73±0.15	74.80±0.11
BiC [31]	11.7	75.10±0.36	71.27±0.28	67.17±0.26	82.83±0.27	80.15±0.24	77.64±0.21	77.38±0.14	79.77±0.15
+BiMeCo (ours)	6.2	75.57±0.32	72.30±0.33	68.24±0.24	83.59±0.24	81.01±0.28	78.68±0.23	78.30±0.12	80.81±0.16
LUCIR [14]	11.7	79.45±0.36	76.11±0.31	71.98±0.30	83.27±0.35	83.71±0.30	81.78±0.26	79.45±0.17	81.78±0.14
+BiMeCo (ours)	6.2	**81.48±0.35**	**81.35±0.34**	**81.37±0.28**	**84.39±0.31**	84.74±0.29	**83.53±0.28**	80.37±0.14	82.96±0.15
CwD [29]	11.7	80.27±0.31	78.64±0.22	75.42±0.27	83.88±0.24	84.18±0.26	82.83±0.30	80.39±0.13	82.77±0.11
+BiMeCo (ours)	6.2	81.03±0.27	80.58±0.23	81.13±0.25	84.30±0.27	**84.82**±0.23	83.49±0.26	**80.88±0.11**	**82.98±0.10**

Table I. Comparison of average incremental accuracy (%) with or without Bilateral Memory Consolidation (BiMeCo) in the TIL setting. 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 [9]. The number of exemplars for each class is set to 20. We report the results that are averaged over 3 runs (mean±std).

results of bilateral memory consolidation under other incremental settings. 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 the following 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 [18]; 4) "+Out.", in which the short-term memory is trained by minimizing its output probabilities with those of the long-term memory, similar to [20].

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) in the main paper. Specifically, we consider the original momentum update that is proposed and fueled by the MoCo series [4,6,12], only updating one branch by back-propagation. Then we introduce our used relaxed version of this momentum-based update rule, which synchronously updates two branches by back-propagation.

Fig. VI and Fig. VII illustrate the results of different regularization techniques under the CIL and TIL settings, respectively. Tab. II-IV and Tab. V-VII list the comparisons of different settings of m in Eq. (3) in the main paper under the CIL and TIL settings, respectively. These results also

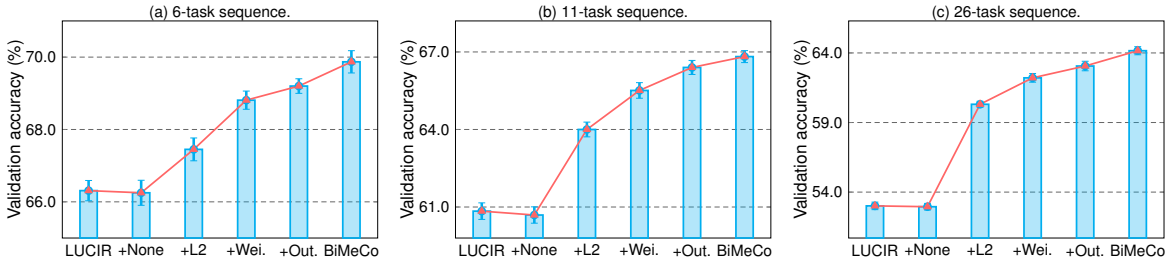

Figure VI. Comparisons of different regularization techniques in the CIL setting. 50 classes are trained for the first task with later (a) 10 classes per task (6-task sequence), (b) 5 classes per task (11-task sequence), and (c) 2 classes per task (26-task sequence), respectively.

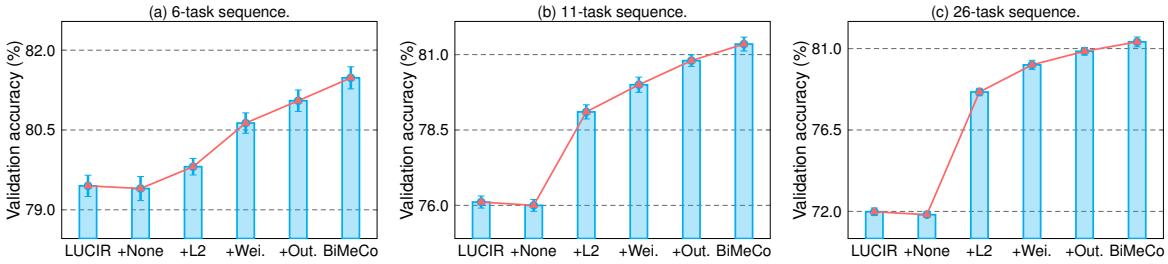

Figure VII. Comparisons of different regularization techniques in the TIL setting. 50 classes are trained for the first task with later (a) 10 classes per task (6-task sequence), (b) 5 classes per task (11-task sequence), and (c) 2 classes per task (26-task sequence), respectively.

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. (%)	65.66	65.86	66.21	66.84	67.07	66.98	66.65	66.42	66.00	65.21	64.33	*fail*

(a) Only updating one branch (short-term memory) by back-propagation as in [12].

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. (%)	69.44	69.61	69.87	69.43	69.10	69.12	69.06	69.00	68.88	68.84	68.81	68.80

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

Table II. Comparison of different settings of m in Eq. (3) in the main paper in the CIL setting. 50 classes are trained for the first task with later 10 classes per task.

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 [12].

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 III. Comparison of different settings of m in Eq. (3) in the main paper in the CIL setting. 50 classes are trained for the first task with later 5 classes per task.

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. (%)	58.09	58.75	59.08	59.47	60.07	60.14	60.42	60.02	59.64	59.01	57.54	*fail*

(a) Only updating one branch (short-term memory) by back-propagation as in [12].

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. (%)	63.60	63.67	64.16	64.02	63.88	63.65	63.41	63.24	63.03	62.99	62.76	62.66

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

Table IV. Comparison of different settings of m in Eq. (3) in the main paper in the CIL setting. 50 classes are trained for the first task with later 2 classes per task.

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. (%)	79.32	79.41	79.54	79.58	79.67	79.66	79.87	80.01	79.71	79.42	79.02	*fail*

(a) Only updating one branch (short-term memory) by back-propagation as in [12].

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. (%)	81.10	81.28	81.48	81.36	81.21	81.05	80.99	80.86	80.61	80.40	80.23	79.62

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

Table V. Comparison of different settings of m in Eq. (3) in the main paper in the TIL setting. 50 classes are trained for the first task with later 10 classes per task.

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. (%)	78.10	78.28	78.53	78.80	79.01	79.34	79.69	79.78	79.42	79.20	78.86	*fail*

(a) Only updating one branch (short-term memory) by back-propagation as in [12].

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. (%)	80.96	81.01	81.35	81.25	81.10	81.02	81.01	80.84	80.75	80.63	80.55	80.36

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

Table VI. Comparison of different settings of m in Eq. (3) in the main paper. in the TIL setting. 50 classes are trained for the first task with later 5 classes per task.

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. (%)	77.10	77.57	77.88	78.01	78.34	78.56	78.11	78.03	77.84	77.68	77.38	*fail*

(a) Only updating one branch (short-term memory) by back-propagation as in [12].

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. (%)	80.98	81.04	81.37	81.30	81.21	81.07	80.97	80.86	80.71	80.66	80.23	80.01

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

Table VII. Comparison of different settings of m in Eq. (3) in the main paper in the TIL setting. 50 classes are trained for the first task with later 2 classes per task.

support our conclusion in the main paper.

C. Selection of Hyperparameter

The hyperparameters λ_1 and λ_2 in Eq. (6) in the main paper control the balance between expressive power and memory consolidation. Here we analyze the effect of different hyperparameter settings on the average incremental accuracy of BiMeCo. The settings of this experiment are the same as in Sec. 4.3, with three CIL protocols that initially learn 50 classes and then learn 10/5/2 classes per task for the rest. As illustrated in Tab. VIII, we can draw the following conclusions. The performance generally improves with the increase of λ_2, then the performance will be saturated when the coefficient $\lambda_2 = 2$ is satisfied. A considerable reason is that the increase of λ_2 typically enhances the ability to defy forgetting previously learned tasks. Nevertheless, an excessively large memory consolidation constraint (*e.g.*, $\lambda_2 = 5$) could deteriorate representation ability for learning new tasks, where a similar phenomenon also occurs in [10, 14, 29]. Moreover, our BiMeCo is not highly sensitive to the particular choice of hyperparameters since different settings can obtain similar performance. Tab. IX-XII provide the chosen hyperparameters in our experiments.

References

[1] 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. 1

[2] M. Boschini, L. Bonicelli, P. Buzzega, A. Porrello, and S. Calderara. Class-incremental continual learning into the extended der-verse. *IEEE TPAMI*, 2022. 2

[3] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah. Signature verification using a" siamese" time delay neural network. *NeurIPS*, 6, 1993. 1

[4] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 3

[5] X. Chen and K. He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021. 1

[6] X. Chen, S. Xie, and K. He. An empirical study of training self-supervised vision transformers. In *ICCV*, pages 9640–9649, 2021. 3

[7] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *CVPR*, pages 1251–1258, 2017. 1

[8] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual

λ_2	λ_1	$S=10$	$S=5$	$S=2$
0.5	1.0	66.04±0.27	63.07±0.33	57.97±0.20
	2.0	66.15±0.24	62.99±0.30	58.21±0.26
	5.0	66.57±0.25	63.40±0.42	58.53±0.31
	8.0	66.92±0.26	63.67±0.27	59.61±0.30
	10.0	66.71±0.21	63.90±0.28	60.02±0.29
	15.0	66.21±0.39	63.96±0.40	60.14±0.36
	20.0	65.95±0.31	63.78±0.32	59.82±0.29
1.0	1.0	67.97±0.38	65.32±0.30	61.50±0.34
	2.0	68.56±0.28	65.85±0.24	61.83±0.27
	5.0	69.08±0.25	65.78±0.34	62.24±0.31
	8.0	69.43±0.30	65.97±0.28	62.87±0.32
	10.0	69.42±0.32	66.19±0.29	63.48±0.25
	15.0	69.25±0.28	66.30±0.28	64.16±0.21
	20.0	68.97±0.41	66.82±0.26	63.57±0.79
2.0	1.0	68.20±0.40	65.46±0.32	61.71±0.34
	2.0	68.62±0.28	65.98±0.37	62.20±0.27
	5.0	69.23±0.28	65.80±0.25	62.37±0.31
	8.0	69.79±0.21	65.84±0.23	63.02±0.40
	10.0	69.30±0.26	66.23±0.26	63.59±0.31
	15.0	69.28±0.34	66.34±0.25	64.10±0.29
	20.0	68.82±0.38	66.75±0.32	63.69±0.27
3.0	1.0	67.52±0.29	65.01±0.36	61.38±0.40
	2.0	68.03±0.34	65.31±0.38	61.82±0.29
	5.0	68.91±0.38	65.52±0.24	62.05±0.25
	8.0	69.12±0.27	65.68±0.23	62.51±0.34
	10.0	68.97±0.40	66.02±0.32	63.28±0.36
	15.0	68.75±0.37	66.17±0.29	63.81±0.26
	20.0	68.40±0.35	66.50±0.32	63.31±0.31
4.0	1.0	67.37±0.36	64.48±0.27	61.02±0.40
	2.0	67.60±0.30	64.70±0.34	61.24±0.25
	5.0	68.23±0.34	64.98±0.30	61.31±0.40
	8.0	68.40±0.27	65.21±0.25	61.70±0.29
	10.0	68.10±0.31	65.42±0.38	62.33±0.26
	15.0	67.84±0.48	65.40±0.31	62.62±0.37
	20.0	67.51±0.29	65.73±0.39	62.95±0.28
5.0	1.0	66.44±0.32	63.38±0.29	59.79±0.38
	2.0	66.39±0.27	63.57±0.28	60.17±0.40
	5.0	67.03±0.35	63.90±0.42	60.38±0.29
	8.0	67.35±0.29	64.04±0.37	60.69±0.32
	10.0	67.02±0.30	64.29±0.25	61.37±0.28
	15.0	66.43±0.38	64.02±0.26	60.32±0.29
	20.0	66.04±0.45	63.62±0.36	59.96±0.31

Table VIII. The effect of different hyperparameter settings in Eq. (6) in the main paper on BiMeCo trained on CIFAR-100. 50 classes are learned in the initial task with later $S = 10/5/2$ classes learned per task. Our proposed BiMeCo is not highly sensitive to the particular choice of hyperparameters since different settings can obtain similar performance. All results are averaged over 3 runs (mean±std).

learning survey: Defying forgetting in classification tasks. *IEEE TPAMI*, 44(7):3366–3385, 2021. 2

[9] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 3, 7

[10] 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. 5

[11] 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. 1

[12] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020. 3, 4, 5

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

Dataset	S=10		S=5		S=2	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
CIFAR-100 (B=50)	5.0	1.0	3.0	1.0	5.0	1.0
ImageNet-100 (B=50)	4.0	1.0	2.0	3.0	1.0	2.0

(a) Hyperparameters for CIFAR-100 [19] and ImageNet-100 [9].

Dataset	S=100		S=50	
	λ_1	λ_2	λ_1	λ_2
ImageNet (B=100)	1.0	2.0	1.0	2.0

(b) Hyperparameters for ImageNet [9].

Table IX. The hyperparameters in Eq. (6) in the main paper used for each of settings for integrating our BiMeCo with for Replay. 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.

Dataset	S=10		S=5		S=2	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
CIFAR-100 (B=50)	2.0	1.0	5.0	1.0	5.0	1.0
ImageNet-100 (B=50)	1.0	2.0	1.0	2.0	1.0	2.0

(a) Hyperparameters for CIFAR-100 [19] and ImageNet-100 [9].

Dataset	S=100		S=50	
	λ_1	λ_2	λ_1	λ_2
ImageNet (B=100)	1.0	2.0	1.0	2.0

(b) Hyperparameters for ImageNet [9].

Table X. The hyperparameters in Eq. (6) in the main paper used for each of settings for integrating our BiMeCo with for BiC [31]. 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.

Dataset	S=10		S=5		S=2	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
CIFAR-100 (B=50)	8.0	0.25	20.0	1.0	15.0	1.0
ImageNet-100 (B=50)	1.0	2.0	1.0	2.0	5.0	1.0

(a) Hyperparameters for CIFAR-100 [19] and ImageNet-100 [9].

Dataset	S=100		S=50	
	λ_1	λ_2	λ_1	λ_2
ImageNet (B=100)	1.0	2.0	1.0	2.0

(b) Hyperparameters for ImageNet [9].

Table XI. The hyperparameters in Eq. (6) in the main paper used for each of settings for integrating our BiMeCo with for LUCIR [14]. 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.

Dataset	S=10		S=5		S=2	
	λ_1	λ_2	λ_1	λ_2	λ_1	λ_2
CIFAR-100 (B=50)	8.0	1.0	8.0	1.0	15.0	1.0
ImageNet-100 (B=50)	3.0	1.0	3.0	1.0	5.0	1.0

(a) Hyperparameters for CIFAR-100 [19] and ImageNet-100 [9].

Dataset	S=100		S=50	
	λ_1	λ_2	λ_1	λ_2
ImageNet (B=100)	1.0	2.0	1.0	2.0

(b) Hyperparameters for ImageNet [9].

Table XII. The hyperparameters in Eq. (6) in the main paper used for each of settings for integrating our BiMeCo with for CwD [29]. 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.

[14] 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, 5, 7

[15] 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. 1

[16] 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. 1

[17] X. Hu, K. Tang, C. Miao, X. Hua, and H. Zhang. Distilling causal effect of data in class-incremental learning. In *CVPR*, pages 3957–3966, 2021. 1

[18] 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

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1, 3, 7

[20] Z. Li and D. Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, 2017. 3

[21] Y. Liu, B. Schiele, and Q. Sun. Adaptive aggregation networks for class-incremental learning. In *CVPR*, pages 2544–2553, 2021. 1, 3

[22] Y. Liu, Y. Su, A. Liu, B. Schiele, and Q. Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *CVPR*, pages 12245–12254, 2020. 1

[23] D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. volume 30, 2017. 1

[24] 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. 2

[25] V. Nair and G.E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[26] Q. Pham, C. Liu, and S. Hoi. Dualnet: Continual learning, fast and slow. volume 34, pages 16131–16144, 2021. 2, 3

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

[28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 1

[29] 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, 3, 5, 7

[30] 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. 1

[31] 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. 2, 3, 7

[32] Z. Zhang and H. Peng. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, pages 4591–4600, 2019. 1