2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

Prototype-based Bilevel Knowledge Distillation for Online Continual Learning

Xiaochen Yang[1][0009-0009-9920-6722]

Harbin Institute of Technology, Weihai 264200, China
yyyxcleo123@gmail.com

Abstract. In Online Continual Learning (OCL), all samples arrive sequentially and are seen only once, posing a challenge in balancing the learning of new tasks with the retention of old task knowledge. Traditional methods often ignore the protection of previously learned knowledge while learning new tasks, leading to catastrophic forgetting. On the other hand, some methods focus on minimizing the forgetting of previous knowledge, which hinders the model's ability to effectively learn new knowledge. To address the balance between learning new tasks and preserving old knowledge, we propose a new framework—Prototype-based Bilevel Knowledge Distillation (PBKD). By incorporating hierarchical prototypes and bilevel distillation mechanisms, PBKD enhances the model's ability to distinguish between classes through personalized feature representations and dynamically adjusts the knowledge transfer between teacher and student models. This approach allows for the effective retention of old task knowledge while improving the model's capacity to learn new tasks. Extensive experimental results demonstrate that PBKD achieves a more favorable combination of accuracy and forgetting rate on three benchmark datasets, validating its effectiveness in addressing the knowledge learning and forgetting issue in OCL.

Keywords: Continual Learning, Knowledge Distillation, Prototype Learning.

1 Introduction

With the rise of deep learning, continual learning (CL) [27] has become essential for adapting models to dynamic tasks. While offline CL [4,12,33] consolidates knowledge through past data, online continual learning (OCL) prohibits data storage, leading to catastrophic forgetting in large-scale models like GPT [3,1]. The key challenge in OCL is balancing new task learning with prior knowledge retention.

Replay-based methods [24,6,21] store past samples but overlook intra-class variations, leading to shortcut learning. Prototype-based approaches [17,28] capture class features but rely on a single prototype, limiting expressiveness. Knowledge distillation [7] transfers knowledge via a static teacher, failing to adapt to new tasks. While intra-layer distillation [29] improves feature learning, it treats all layers equally, ignoring their varying representational capacities.

We propose Prototype-based Bilevel Knowledge Distillation (PBKD), introducing hierarchical prototypes for fine-grained feature learning and reducing intra-class variance. PBKD integrates intra-model and inter-model distillation: intra-model enhances new task learning, while inter-model preserves past knowledge. Unlike fixed-weight approaches, PBKD adaptively weights layers based on their learning contributions, achieving a balance between knowledge retention and adaptation.

Contributions We propose PBKD, a framework integrating intra-model and inter-model distillation to balance learning and forgetting in OCL. We introduce hierarchical prototypes, enabling each layer to capture fine-grained class features and mitigate shortcut learning. We conduct extensive experiments on benchmark datasets, demonstrating that PBKD outperforms state-of-the-art methods across different settings.

2 Related Work

2.1 Continual Learning (CL)

Continual learning updates models incrementally, enabling adaptation to streaming data. Solutions include regularization, dynamic networks, and replay-based methods. Regularization constrains parameters [14] or leverages past exemplars [20]. Dynamic networks expand model capacity [30,25], while replay-based methods [5,13] store samples for simultaneous learning and review.

2.2 Prototype Learning in CL

Prototypes, as class representations, facilitate classification and zero-shot inference. iCaRL [23] aligns features with prototypes, while SCR [21] employs prototype classifiers. OnPro [28] balances prototypes to reduce shortcut learning, and PPE [17] stores class features to mitigate forgetting. However, single-class prototypes fail to capture intra-class variations, limiting adaptability.

2.3 Knowledge Distillation in CL

Knowledge distillation transfers knowledge to reduce forgetting. LwF [18] pioneered distillation in CL, while DMC [31] trains expert models for knowledge retention. COIL [32] enhances alignment via bidirectional distillation. MOSE [29] distills from early to later layers but lacks explicit forgetting mitigation. MKD [22] updates teachers dynamically, reducing interference but without optimizing new task learning.

3 Proposed Methods

3.1 Problem Definition

In line with the widely accepted setup in online continual learning (OCL) tasks [6], we define N tasks $\{T_1, T_2, ..., T_N\}$ on each dataset, where each task corresponds to a distinct set of categories C_i, such that the union of all categories across tasks covers the entire category space, i.e., $\cup C_i = C$, and the categories in different tasks are disjoint, i.e., $C_i \cap C_j = \emptyset$ for any $i \neq j$. This setup is standard in the OCL paradigm and aims to evaluate the model's ability to resist catastrophic forgetting when learning sequential tasks.

Each task's data is denoted as $D_t = \{x_i, y_i\}$, where x_i is the i-th image and \(y_i\) its label. In accordance with the online learning paradigm, each sample is presented to the model exactly once, and no sample is revisited. To mitigate forgetting of previously learned tasks, we adopt a fixed-size replay buffer [5,2], which is a common strategy in OCL research. After learning task T_t, a subset of its samples is stored in the buffer. When learning task T_{t+1}, the model is trained on both the new samples D_{t+1} and the buffered samples M_t from previous tasks. This combination helps to retain knowledge of earlier tasks while allowing the model to learn new tasks.

Table 1. Accuracy of Different Layers on All CIFAR-100 Tasks After the Final Task.

Layer	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9
L1	*26.6*	*17.5*	*32.4*	20.7	*29.1*	33.9	35.4	33.4	43.4	53.3
L2	**26.8**	**19.1**	**33.1**	**23.9**	**31.4**	*35.6*	**39.5**	40.1	48.5	62.1
L3	22.7	16.8	31.0	*21.8*	27.2	**37.6**	*37.5*	*42.5*	**51.1**	*71.6*
L4	22.7	12.9	29.9	18.4	26.2	34.7	36.3	**44.9**	*49.9*	**77.9**

- The best results within four layers are bolded and the second-best results are underlined.

3.2 Weighted Intra-model Distillation

Most online continual learning tasks use ResNet or ViT as the backbone network. In this paper, we adopt ResNet18, which consists of four hierarchical structures f_{li} $(1 \leq i \leq 4)$, each containing BasicBlocks. These BasicBlocks include two 3×3 convolutional layers, batch normalization, and ReLU activation, and are connected through identity mapping. Given an input x_t at stage t, hierarchical features are extracted sequentially as $h_i = f_{li}(h_{i-1})$, where $h_0 = x_t$.

We recognize that shallow layers typically focus on low-level features such as edges, while deeper layers focus on high-level features such as object parts. This motivates us

to introduce the necessity of a distillation loss to transfer knowledge from shallow layers to deep layers. Therefore, we define the intra-model distillation loss, which distills knowledge from the first three layers (shallow layers) into the fourth layer (the deepest layer). Specifically, we define distillation weights $\gamma_i \ (i=1,2,3)$ for each teacher module, and introduce the hierarchical distillation loss:

$$L_{intra-distill} = \sum_{i=1}^{3} \gamma_i \cdot \mathrm{Dist}(q_i, q_4) \tag{1}$$

where q_i represents the classification logits of the i-th teacher module, q_4 represents the classification logits of the student module, and $\mathrm{Dist}(\cdot)$ denotes the $L2$ distance. The weights γ_i are calculated based on the training loss L_{ce}^{i} for each layer:

$$\gamma_i = \frac{\exp(-L_{\mathrm{ce}}^{i})}{\sum_{i=1}^{3} \exp(-L_{\mathrm{ce}}^{i})} \tag{2}$$

The weights γ_i allow dynamic adjustment of the contribution of each teacher module, enabling the student to selectively absorb the most valuable knowledge while reducing interference from redundant or irrelevant information.

Tab. 1 shows that shallow layers have higher accuracy in early tasks, indicating that shallow layers are less prone to forgetting previously learned knowledge than deep layers. This is consistent with the fact that shallow layers primarily focus on simpler features, which are easier to retain. Therefore, during inference, we combine the prediction outputs from all four layers, with each layer contributing based on the knowledge it has learned.

During testing, we compute the weighting coefficient α_i for each layer based on the entropy of its prediction, representing the layer's contribution to the final prediction:

$$\alpha_i = \frac{\exp\left(-H\left(\hat{y}_i\right)\right)}{\sum_{i=1}^{4} \exp\left(-H\left(\hat{y}_i\right)\right)} \tag{3}$$

where $H\left(\hat{y}_i\right)$ denotes the entropy of the i-th layer output $\hat{y}_i$, which measures the certainty of the layer's prediction. A lower entropy indicates a more certain prediction, and the weight becomes larger.

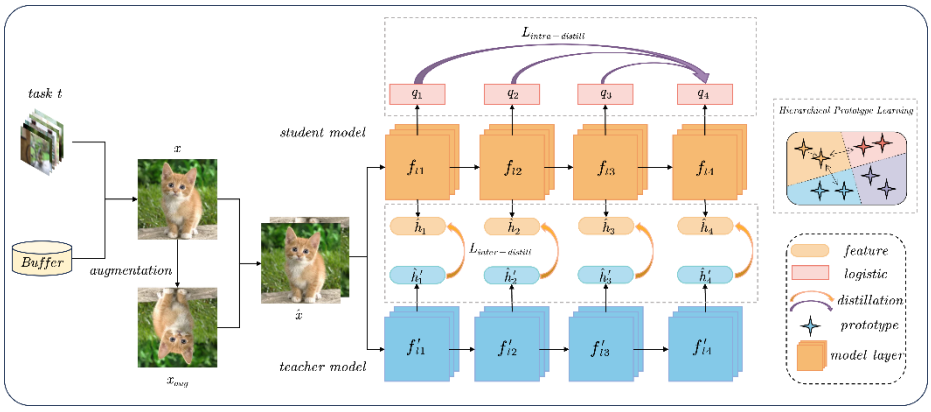

Fig. 1. Illustration of our proposed PBKD framework. In task T_t a batch of images is combined with buffer-sampled and augmented images as the model's input. The distillation process includes intra-model distillation, where the student model's fourth layer logits are aligned with the first three teacher layers' features(frozen during distillation), and inter-model distillation, where the feature tensors from each student layer are aligned with the corresponding teacher layer(frozen during distillation). Each layer's feature output constructs its own prototype for prototype learning to enhance the model's learning ability.

Finally, the final class is computed by weighting the output of each layer $\hat{h}_i$ with the features $\hat{r}_j$ from the replayed samples based on their L2 distance:

$$\hat{y}_{\text{final}} = \arg\min_{c} \sum_{i=1}^{4} \alpha_i \cdot \sum_{j=1}^{N} \text{Dist}\left(\hat{h}_i, \hat{r}_j\right) \tag{4}$$

where c is the class label, $\hat{r}_j$ is the feature representation of the replayed sample j, and $\text{Dist}\left(\hat{h}_i, \hat{r}_j\right)$ is the L2 distance between the i-th layer's output and the replayed sample's feature. Through this weighted distance computation, we can effectively combine the information from shallow and deep layers, and leverage the knowledge from replayed samples to enhance the final performance of the model.

3.3 Inter-model Distillation

Intra-model distillation effectively enhances the learning ability of the current task by continuously distilling knowledge during training, but it overlooks the forgetting of knowledge from previous tasks, resulting in suboptimal performance on older tasks. Fixed-teacher distillation, while providing stable knowledge guidance, is rigid and unable to adapt to the dynamic changes of new tasks. Inspired by [22], we propose a dynamically adaptive teacher model.

Specifically, we utilize a copy of the original model as the teacher model to store the parameters of the student model from previous tasks. The teacher model is updated dynamically using an Exponential Moving Average (EMA). Let the teacher model's parameters for task T_{t-1} be θ'_{t-1} and the student model's parameters for task T_t be θ_t. The update rule is as follows:

$$\theta'_t = \alpha \cdot \theta_t + (1-\alpha) \cdot \theta'_{t-1} \tag{5}$$

where $\alpha \in [0,1]$ is a hyperparameter that controls the trade-off between stability and plasticity. A larger value of α allows the teacher model to adapt more quickly to the current task, while a smaller value of α helps preserve knowledge from previous tasks. This EMA-based dynamic update mechanism allows the teacher model to adjust dynamically to new tasks while ensuring that parameters from previous tasks are not drastically altered, achieving a balance between stability and plasticity.

Based on this, we define the inter-model distillation loss, which aims to guide the student model using the teacher model, thereby reducing the forgetting of previous tasks. The loss function is defined as:

$$L_{inter-distill} = \frac{1}{|B|} \sum_{(x,y) \in B} \mathrm{Dist}\left(f_{\theta_t}, f_{\theta'_{t-1}}\right) \tag{6}$$

where D_t denotes the set of training samples in the Task t, f_{θ_t} and $f_{\theta'_{t-1}}$ represent the feature representations of sample x output by the student and teacher models, respectively, and $\mathrm{Dist}(\cdot)$ denotes the $L2$ distance between features.

3.4 Hierarchical Feature Prototypes

Prototype Definition Previous methods compute contrastive loss using samples from both the current task and the replay buffer to enhance inter-class separation. However, feature distributions within the same class often exhibit high variability, and some samples may contain noise or irrelevant information, leading the model to focus on non-essential details and impeding learning.

To address this, following [17,28], we introduce prototypes as global feature representations for each class. By aggregating features from all samples of the same class, prototypes capture representative characteristics while reducing noise and bias. The prototype for a given class c is defined as:

$$P_c = \frac{1}{N_c} \sum_{k=1}^{N_c} f\left(x_k\right) \tag{7}$$

where N_c is the number of samples in class c, and $f(x_k)$ denotes the feature representation of sample x_k. Prototypes stabilize intra-class clustering while improving inter-class separability, enhancing overall feature discrimination.

Hierarchical Prototype Modeling We define personalized prototypes at each model layer to capture hierarchical feature differences. By constructing layer-specific class prototypes, we leverage low-level semantic details from shallow layers alongside high-level semantics from deeper layers, improving class discriminability.

Class prototypes are computed using both new task samples and replay buffer samples. According to Eq. 7, the layer-wise class prototypes are defined as:

$$P_a = \frac{1}{N_a^{\text{new}}} \sum_{k=1}^{N_a^{\text{new}}} f_{li}(x_{\text{new}}^k), \quad i \in \mathcal{A}^{\text{new}} \tag{8}$$

$$P_b^B = \frac{1}{N_b^{\text{buffer}}} \sum_{k=1}^{N_b^{\text{buffer}}} f_{li}(x_{\text{buffer}}^k), \quad j \in \mathcal{B}^{\text{buffer}} \tag{9}$$

where N_a^{new} and N_b^{buffer} are the sample counts for class a in the new task and class b in the replay buffer, respectively. $f_{li}(x_k^{\text{new}})$ and $f_{li}(x_{\text{buffer}}^k)$ represent their features at layer i. The sets $\mathcal{A}^{\text{new}}$ and $\mathcal{B}^{\text{buffer}}$ denote the class categories in the new task and replay buffer.

To further enhance representation quality, we design a contrastive loss to encourage intra-class consistency and inter-class separation. The prototype loss for new task samples and replay buffer samples is given by:

$$-\frac{1}{|P^a|}\sum_{i=1}^{|P^a|}\log\frac{\exp\left(\frac{P_i^{aT}P_i^{a\prime}}{\tau}\right)}{\sum_{j=1}^{|P^a|}\exp\left(\frac{P_i^{aT}P_j^{a\prime}}{\tau}\right)+\sum_{j\,\text{neq}\,i}\exp\left(\frac{P_i^{aT}P_j^{a}}{\tau}\right)} \tag{10}$$

$$-\frac{1}{|P^b|}\sum_{j=1}^{|P^b|}\log\frac{\exp\left(\frac{P_j^{bT}P_j^{b\prime}}{\tau}\right)}{\sum_{k=1}^{|P^b|}\exp\left(\frac{P_j^{bT}P_k^{b\prime}}{\tau}\right)+\sum_{k\,\text{neq}\,j}\exp\left(\frac{P_j^{bT}P_k^{b}}{\tau}\right)} \tag{11}$$

where P^a and $P^{a\prime}$ are sets of class prototypes and their augmented views, respectively. Similarly, P^b and $P^{b\prime}$ denote buffer prototypes and their augmented versions. The temperature parameter τ controls the sharpness of the similarity distribution. The

hierarchical prototype loss is computed by summing the new sample and buffer losses for each layer:

$$L_{\text{proto}}^{l} = L_{\text{proto}}^{l,\text{new}} + L_{\text{proto}}^{l,\text{buffer}} \tag{12}$$

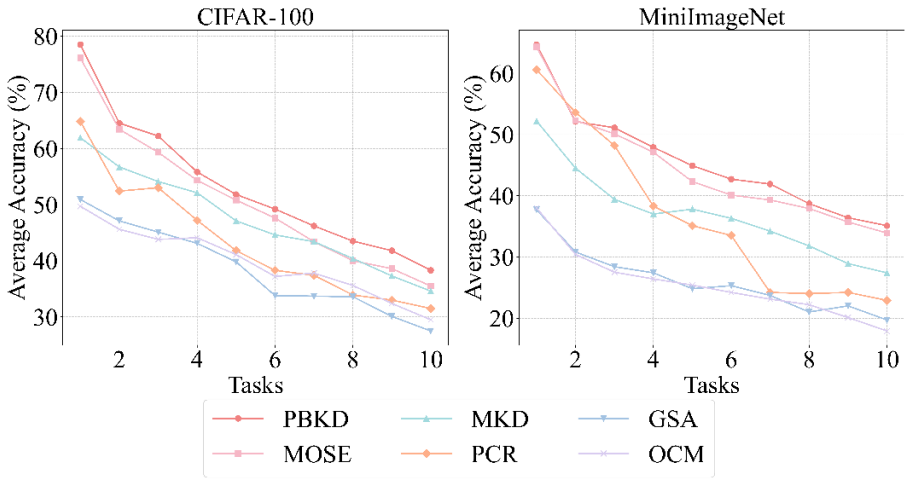

Fig. 2. The average accuracy of six methods (PBKD, MOSE, MKD, PCR, GSA, and OCM) on CIFAR-100 and MiniImageNet with a memory size of 1000 after each task.

Finally, to integrate multi-layer feature information, the total prototype loss is obtained by weighting and summing across layers:

$$L_{\text{proto}} = \sum_{l=1}^{L} \lambda_{l} L_{\text{proto}}^{l} \tag{13}$$

where λ_l balances the contributions of different layers.

3.5 Overall Model Framework

The overall framework is illustrated in Fig. 1. Following [21], we adopt cross-entropy loss and contrastive loss as the base loss L_{base}. Instead of relying solely on the final layer's output, we integrate features from all four layers into the NCM (Nearest Class Mean) module for classification, enhancing accuracy. To improve feature discrimination, we introduce layer-specific prototypes and optimize class separation via L_{proto}. Additionally, intra-layer knowledge distillation $L_{\text{intra-distill}}$ consolidates classification advantages across layers into the final layer, while inter-layer knowledge distillation $L_{\text{inter-distill}}$ mitigates forgetting by constraining parameter shifts using a teacher model. The final loss function is:

$$L = L_{\text{base}} + L_{\text{intra-distill}} + L_{\text{inter-distill}} + L_{\text{proto}} \tag{14}$$

4 Experiment

4.1 Experimental Setup

Datasets. We test our method on three benchmark datasets: CIFAR-100 [15], MiniImagenet [26], and TinyImagenet [16]. CIFAR-100 and MiniImagenet each has 100 classes, while TinyImagenet has 200 classes. Following [8], we divide CIFAR-100 and MiniImagenet into 10 tasks, with 10 classes per task. The buffer size is set to 500/1000/2000. For TinyImagenet, we divide it into 20 tasks, with 10 classes per task, and the buffer size is set to 1000/2000/4000.

Comparison methods. We compare our work with nine OCL methods (ER [6], SCR [21], DVC [9], SSD [8], GSA [11], PCR [19], OCM [10], MOSE [29], MKD [22]), all of which are replay-based methods.

Implementation details. Following the backbone setup of recent works such as MOSE and MKD, we use Full ResNet18 as the baseline model for all methods and apply data augmentation techniques to process the samples, including RandomColorGrayLayer (with probability 0.25), RandomResizedCropLayer (with a scale range of 0.3 to 1.0 and size matching the input dimensions), and RandomFlip. We fix the incoming batch size $B = 10$ for all baselines, and the buffer batch size is set to 64. The parameter α in the EMA function Eq. 5 is set to 0.1. Additionally, we use the Adam optimizer with a learning rate of $\eta = 1 \times 10^{-3}$.For the memory buffer, we utilize a random sampling strategy with a fixed memory size. Specifically, during training, we randomly sample a fixed number of previously seen examples from the buffer to mix with new incoming data.

Evaluation metrics. We evaluate the learning capability and forgetting degree of all models using the Final Average Accuracy and the Final Forgetting Rate. Final Average Accuracy refers to the average accuracy of the model in classifying all test samples from each category after completing the final task. The formula can be expressed as:

$$\text{Final Average Accuracy} = \frac{1}{N}\sum_{t=1}^{N} a_{N,t} \tag{15}$$

where $a_{N,t}$ is the accuracy of the model on task t after learning N tasks.

The Forgetting Rate measures how much the model forgets about previously learned tasks after learning a new task. Specifically, the forgetting rate $f_{T,t}$ can be expressed as:

$$f_{T,t} = \max_{T' \in \{1, \text{dots}, T-1\}} a_{T',t} - a_{T,t} \tag{16}$$

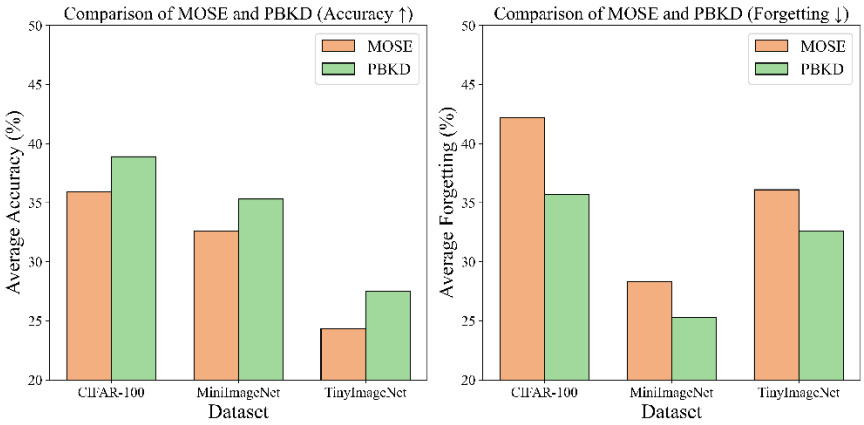

Fig. 3. Comparison of MOSE and PBKD on different datasets (CIFAR-100, MiniImageNet, and TinyImageNet) in terms of average accuracy (higher is better) and average forgetting rate (lower is better).

The Final Forgetting Rate is given by:

$$\text{Final Forgetting Rate} = \frac{1}{T-1} \sum_{t=1}^{T-1} f_{T,t} \tag{17}$$

These two metrics enable fair comparisons in class-incremental settings by evaluating the model's overall performance across all tasks. Final Average Accuracy reflects how well the model retains knowledge, while Final Forgetting Rate measures its resistance to forgetting. Both are computed under consistent settings and task splits, ensuring objective and comparable results across methods.

4.2 Experimental Results and Analysis

The performance of the final average accuracy. The comparison between PBKD and other methods on the average accuracy of different memory buffer sizes across three benchmark datasets is shown in Tab. 2. The results demonstrate that PBKD outperforms most of others in most dataset settings, which is due to our weighted hierarchical knowledge distillation method that improves the model's learning capability.

Moreover, our method also maintains good performance with small sample buffers. For example, in the MiniImageNet dataset, as the memory buffer size decreases from 2000 to 500, the average accuracy of MKD drops sharply from 33.9% to 21.2%,

Table 2. The Average Accuracy (higher is better) of different methods on various datasets with different buffer size.

Data	CIFAR100			MiniImageNet			TinyImageNet		
Buffer	500	1000	2000	500	1000	2000	1000	2000	4000
ER	16.8	24.5	32.4	11.0	15.2	22.0	5.1	6.3	9.2
SCR	13.3	19.2	29.0	10.1	17.2	24.6	6.5	12.5	18.0
DVC	16.5	25.0	25.3	13.0	17.9	21.2	9.2	10.9	8.9
GSA	21.1	27.9	34.8	14.8	18.5	23.2	10.4	15.2	18.2
OCM	22.2	29.2	36.1	12.0	17.8	20.8	11.0	15.7	20.2
PCR	23.7	31.5	37.7	14.2	22.8	29.4	14.4	19.1	25.0
SSD	22.9	28.2	32.6	18.4	23.6	26.2	14.3	17.1	18.0
MOSE	26.1	*35.9*	*45.7*	*24.9*	*32.6*	**43.1**	17.3	24.3	28.4
MKD	*26.3*	34.5	43.4	21.2	27.3	33.9	**20.7**	*26.2*	*32.5*
PBKD	**29.2**	**38.8**	**47.6**	**27.1**	**35.3**	*42.4*	*20.4*	**27.5**	**35.0**

whereas PBKD only decreases from 42.4% to 27.1%. This is because PBKD customizes personalized prototype for each layer's feature, enabling the model to effectively retain various types of knowledge and remember important inter-class features even with a small number of cached samples.Fig. 2 shows the task-wise average accuracy of several models on CIFAR-100 and MiniImageNet, in which the curve of PBKD is much smoother, illustrating its promising performance.

On the MiniImageNet dataset, when the memory size is 2000, PBKD performs slightly worse than MOSE. We speculate that this may be because MOSE is better able to leverage more class information with a larger memory size, while PBKD, with a limited number of prototypes, may not fully capture the complex class features. This suggests that PBKD might require further optimization of prototype representation when larger memory sizes are used.

The performance of the final average forgetting rate. As the forgetting rate is calculated based on the learning rate at each stage, comparing forgetting rates can be misleading if the learning rate is too low during each phase. This work aims to reduce the forgetting rate while maintaining learning accuracy. Therefore, we tested PBKD and the SOTA method MOSE on CIFAR-100, TinyImagenet, and MiniImagenet datasets with memory sizes of 1000, 1000, and 2000, respectively, and evaluated their accuracy and forgetting rate , as shown in Fig. 3. The results show that PBKD not only improves accuracy but also reduces the forgetting rate. This is because, compared to MOSE, PBKD incorporates dynamic teacher distillation between models and hierarchical prototypes, which significantly mitigates forgetting of prior knowledge.

Table 3. Ablation study of different components on CIFAR-100 datasets.

Method	NCM	$L_{intra-distill}$	$L_{inter-distill}$	L_{proto}	CIFAR100
Baseline	-	-	-	-	31.5
	✓	-	-	-	35.1
	✓	✓	-	-	35.7
	✓	✓	✓	-	36.5
PBKD	✓	✓	✓	✓	38.8

- "-" and "✓" represent without or with this component.

Table 4. Impact of adaptive weighting in intra-layer distillation.

Method	Accuracy (%)
w/o weighted distillation	38.3
with adaptive weighting (γ_i)	**38.8**

4.3 Ablation Study

We conduct ablation experiments to analyze the effectiveness of each core component in our framework, including intra-layer distillation, inter-stage distillation, and prototype replay.

Intra-model Effects. Compared with the baseline model trained only with L_{base}, introducing intra-layer distillation enables deeper layers to absorb complementary knowledge from shallower layers. In this stage, we also apply the NCM (Nearest Class Mean) method to combine the outputs of different layers for inference. This method aggregates the logits from each layer, with adaptive weights determined by the loss during training. Compared with uniform weighting, the adaptive weighting based on normalized training loss brings better accuracy, as shown in Tab. 4. This indicates that guiding deeper layers with properly weighted signals leads to more effective feature transfer.

Inter-model Effects. By aligning features and logits of different time stages, the model is encouraged to retain previously acquired knowledge. Experimental results in Tab. 3 show that the introduction of this temporal distillation further improves the performance, especially in later tasks where forgetting is more severe.

Prototypes. We examine the role of prototype replay. By storing a small number of representative features and performing class-mean-based matching during inference, the model achieves more robust prediction. This strategy compensates for memory limitations in continual learning and stabilizes feature distributions across tasks. As shown

in Tab. 3, this component further enhances accuracy, demonstrating its complementary benefit to the distillation modules.

4.4 Sensitivity Analysis of Hyperparameters

We conduct sensitivity analysis on three key hyperparameters: the EMA momentum factor α (Eq. 5), the contrastive temperature τ (Eq. 10, 11), and the weight factor λ_l in Eq. 13. As shown in Fig. 4, PBKD performs best when $\alpha = 0.15$ and $\tau = 0.7$, indicating good stability within reasonable ranges.

Furthermore, for the weight factor λ_l, we set $\lambda_l = 0.7$ for the shallow layers ($l = 0, 1$) and $\lambda_l = 1$ for the deeper layers ($l = 2, 3$). This choice is based on the greater contribution of deeper layers during the learning process, and thus they are given higher weights to optimize model performance.

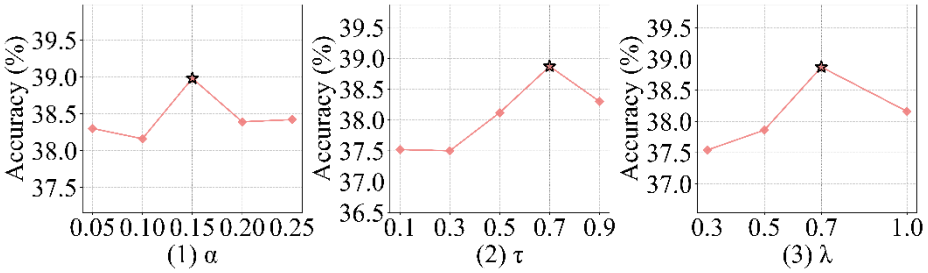

Fig. 4. Sensitivity analysis on CIFAR-100: (Left) EMA momentum α, (Middle) contrastive temperature τ, (Right) weight parameter λ_l.

5 Conclusion

In this paper, we propose the Prototype-based Bilevel Knowledge Distillation (PBKD) framework, aiming to address the problem of catastrophic forgetting in online continual learning (OCL). By combining hierarchical prototypes and dual distillation mechanisms, our model is able to effectively retain old knowledge while learning new tasks. Extensive experimental results demonstrate that PBKD outperforms current state-of-the-art methods on multiple benchmark datasets such as CIFAR-100, MiniImageNet, and TinyImageNet, exhibiting higher average accuracy and lower forgetting rates. By effectively integrating prototypes with bilevel knowledge distillation, PBKD provides a novel and effective solution for the OCL field, offering a new approach to balancing new task learning and old task forgetting.

Moreover, PBKD demonstrates promising scalability. The lightweight hierarchical distillation structure and prototype-guided supervision enable it to extend naturally to longer task sequences and larger-scale models. The intra-model and inter-model distil-

lation mechanisms, along with teacher-guided knowledge transfer, remain effective regardless of model depth or capacity, indicating strong potential for broader deployment in more complex continual learning scenarios.

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient based sample selection for online continual learning. Advances in neural information processing systems 32 (2019)
3. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems 33, 1877–1901 (2020)
4. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. Advances in neural information processing systems 33, 15920–15930 (2020)
5. Chaudhry, A., Dokania, P.K., Ajanthan, T., Torr, P.H.: Riemannian walk for incremental learning: Understanding forgetting and intransigence. In: Proceedings of the European conference on computer vision (ECCV). pp. 532–547 (2018)
6. Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P., Torr, P., Ranzato, M.: Continual learning with tiny episodic memories. In: Workshop on Multi-Task and Lifelong Reinforcement Learning (2019)
7. Gou, J., Yu, B., Maybank, S.J., Tao, D.: Knowledge distillation: A survey. International Journal of Computer Vision 129(6), 1789–1819 (2021)
8. Gu, J., Wang, K., Jiang, W., You, Y.: Summarizing stream data for memory-constrained online continual learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 12217–12225 (2024)
9. Gu, Y., Yang, X., Wei, K., Deng, C.: Not just selection, but exploration: Online class-incremental continual learning via dual view consistency. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7442–7451 (2022)
10. Guo, Y., Liu, B., Zhao, D.: Online continual learning through mutual information maximization. In: International conference on machine learning. pp. 8109–8126. PMLR (2022)
11. Guo, Y., Liu, B., Zhao, D.: Dealing with cross-task class discrimination in online continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11878–11887 (2023)
12. Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 831–839 (2019)
13. Isele, D., Cosgun, A.: Selective experience replay for lifelong learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32 (2018)
14. Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A.A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al.: Overcoming catastrophic forgetting in neural networks. Proceedings of the national academy of sciences 114(13), 3521–3526 (2017)
15. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
16. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. CS 231N 7(7), 3 (2015)

17. Li, Q., Peng, Y., Zhou, J.: Progressive prototype evolving for dual-forgetting mitigation in non-exemplar online continual learning. In: Proceedings of the 32nd ACM International Conference on Multimedia. pp. 2477–2486 (2024)
18. Li, Z., Hoiem, D.: Learning without forgetting. IEEE transactions on pattern analysis and machine intelligence 40(12), 2935–2947 (2017)
19. Lin, H., Zhang, B., Feng, S., Li, X., Ye, Y.: Pcr: Proxy-based contrastive replay for online class-incremental continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 24246–24255 (2023)
20. Lopez-Paz, D., Ranzato, M.: Gradient episodic memory for continual learning. Advances in neural information processing systems 30 (2017)
21. Mai, Z., Li, R., Kim, H., Sanner, S.: Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3589–3599 (2021)
22. Michel, N., Wang, M., Xiao, L., Yamasaki, T.: Rethinking momentum knowledge distillation in online continual learning. arXiv preprint arXiv:2309.02870 (2023)
23. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 2001–2010 (2017)
24. Robins, A.: Catastrophic forgetting in neural networks: the role of rehearsal mechanisms. In: Proceedings 1993 The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems. pp. 65–68. IEEE (1993)
25. Rusu, A.A., Rabinowitz, N.C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., Hadsell, R.: Progressive neural networks. arXiv preprint arXiv:1606.04671 (2016)
26. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. Advances in neural information processing systems 29 (2016)
27. Wang, L., Zhang, X., Su, H., Zhu, J.: A comprehensive survey of continual learning: theory, method and application. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)
28. Wei, Y., Ye, J., Huang, Z., Zhang, J., Shan, H.: Online prototype learning for online continual learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18764–18774 (2023)
29. Yan, H., Wang, L., Ma, K., Zhong, Y.: Orchestrate latent expertise: Advancing online continual learning with multi-level supervision and reverse self-distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23670–23680 (2024)
30. Yoon, J., Yang, E., Lee, J., Hwang, S.J.: Lifelong learning with dynamically expandable networks. arXiv preprint arXiv:1708.01547 (2017)
31. Zhang, J., Zhang, J., Ghosh, S., Li, D., Tasci, S., Heck, L., Zhang, H., Kuo, C.C.J.: Class-incremental learning via deep model consolidation. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 1131–1140 (2020)
32. Zhou, D.W., Ye, H.J., Zhan, D.C.: Co-transport for class-incremental learning. In: Proceedings of the 29th ACM International Conference on Multimedia. pp. 1645–1654 (2021)
33. Zhu, F., Cheng, Z., Zhang, X.Y., Liu, C.l.: Class-incremental learning via dual augmentation. Advances in Neural Information Processing Systems 34, 14306–14318 (2021)