YoooP: You Only Optimize One Prototype per Class for Non-Exemplar Incremental Learning

Anonymous authors
Paper under double-blind review

Abstract

Incremental learning (IL) usually addresses catastrophic forgetting of old tasks when learning new tasks by replaying old tasks' raw data stored in a memory, which can be limited by its size and the risk of privacy leakage. Recent non-exemplar IL methods store class centroids as prototypes and perturb them with high-dimensional Gaussian noise to generate synthetic data for replaying. Unfortunately, this approach has two major limitations. First, the boundary between embedding clusters around prototypes of different classes might be unclear, leading to serious catastrophic forgetting. Second, directly applying high-dimensional Gaussian noise produces nearly identical synthetic samples that fail to preserve the true data distribution, ultimately degrading performance. In this paper, we propose YoooP, a novel exemplar-free IL approach that can greatly outperform previous methods by only storing and replaying one prototype per class even without synthetic data replay. Instead of merely storing class centroids, YoooP optimizes each prototype by (1) shifting it to high-density regions within each class using an attentional mean-shift algorithm, and (2) optimizing its cosine similarity with class-specific embeddings to form compact, well-separated clusters. As a result, replaying only the optimized prototypes effectively reduces inter-class interference and maintains clear decision boundaries. Furthermore, we extend YoooP to YoooP+ by synthesizing replay data preserving the angular distribution between each class prototype and the class's real data in history, which cannot be obtained by high-dimensional Gaussian perturbation. YoooP+ effectively stabilizes and further improves YoooP without storing real data. Extensive experiments demonstrate the superiority of YoooP/YoooP+ over non-exemplar baselines in terms of different metrics. The source code will be released upon acceptance of the paper.

1 Introduction

Catastrophic forgetting McCloskey & Cohen (1989) refers to deep neural networks forgetting the acquired knowledge from the previous tasks disastrously while learning the current task. This stands in stark contrast to human learning, where new knowledge is integrated without erasing prior understanding. To bridge this gap, incremental learning (IL) Gepperth & Hammer (2016); Wu et al. (2019); Douillard et al. (2022); Wang et al. (2022a); Goswami et al. (2024) has emerged as a paradigm that enables AI systems to continuously learn from evolving data.

In the past few years, a variety of methods Roady et al. (2020); Cong et al. (2020); Wang et al. (2021); Xue et al. (2022) have been proposed to mitigate catastrophic forgetting in IL. In this work, we are interested in a very challenging scenario, called class-incremental learning (CIL). CIL is particularly challenging because it requires the model to recognize all learned classes without any task identifier during inference. CIL is especially susceptible to catastrophic forgetting due to overlapping feature representations between old and new tasks Zhu et al. (2021b). To address this issue, many prior studies have adopted *exemplar-based approaches* Rebuffi et al. (2017); Wu et al. (2019); Zhao et al. (2020); Wang et al. (2022b) that store a subset of old class samples in a memory buffer for replay. However, these methods face inherent limitations related to memory capacity and privacy. Thus, *non-exemplar-based methods* Li & Hoiem (2017); Lopez-Paz & Ranzato (2017); Mallya & Lazebnik (2018); Cong et al. (2020); Xue et al. (2022) have been proposed, which avoid storing raw data by relying on regularization, parameter isolation, or generative models to mitigate catastrophic forgetting. Unfortunately, solely applying regularization is often insufficient, parameter isolation increases network size, and generative models can be unstable.

Recently, prototype-based methods Zhu et al. (2021b;a); Petit et al. (2023) have attracted attention in non-exemplar CIL. These approaches store a single *class-mean prototype* as the class centroid for each old class and replay synthetic data augmented from these prototypes in future tasks. Notably, PASS Zhu et al. (2021b) augments stored prototypes via high-dimensional Gaussian noise. However, our observations reveal that such an augmentation strategy may actually degrade prediction accuracy (Sec. 4.1). The underlying issue is twofold. First, simply generating class-mean prototypes without further optimization leads to diffuse clusters of embeddings with unclear decision boundaries between classes. Consequentially, the prototype is less representative, resulting in serious catastrophic forgetting while training future tasks. Second, as shown in Fig. 1(a)-Prototype Augmentation, directly adding high-dimensional Gaussian noise on the prototype yields nearly identical synthetic samples that fail to capture the true distribution of old tasks' class embeddings, ultimately decreasing performance.

Motivated by these limitations, we propose to optimize prototype learning and develop a novel prototype augmentation strategy for CIL. In this work, we introduce YoooP, a new non-exemplar CIL method that stores and replays only one representative prototype per class without relying on synthetic data. The main challenge is obtaining well-separated prototypes that capture the essence of their respective classes. To achieve this, we introduce prototype optimization (Fig. 1 (II)). First, we employ a mini-batch attentional mean shift-based method to shift each class prototype toward high-density regions of its corresponding embeddings (the moving path in Fig. 1 (II)). Next, we optimize the angular distance between the prototype and the class-specific embeddings to form tight, compact clusters (comparing the (I) and (II) in Fig. 1). As a result, each prototype becomes highly representative of its class. Consequently, only replaying the prototype can effectively reduce inter-class interference and maintain clear decision boundaries. Building on YoooP, we further develop YoooP+, which extends our approach with a novel prototype augmentation technique. Instead of directly adding high-dimensional Gaussian noise, YoooP+ synthesizes data by combining a rotation matrix in high-dimensional space with the stored angular distribution between each class's prototype and its corresponding real data. As shown in Fig. 1 (IV), this strategy produces synthetic data that more faithfully reflects the true embedding distribution of old classes and yields higher-quality samples.

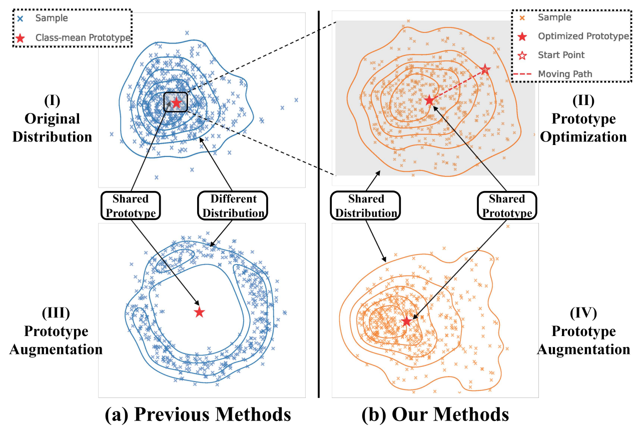


Figure 1: Comparison of previous prototype-based (**left**) methods with our YoooP/YoooP+ (**right**). Previous methods (*e.g.*, PASS) typically store and average all class embeddings to form a class-mean prototype, which is then augmented inappropriately (*e.g.*, by directly adding high-dimensional Gaussian noise). In contrast, YoooP adopts a mini-batch attentional mean-shift method, which constructs representative prototypes from a small batch of embeddings. Prototype optimization further refines these prototypes to form a more compact feature space. YoooP+ further enhances performance by synthesizing data with stored original angular distribution of old tasks. Comparing (I) and (II) demonstrates the benefit of prototype optimization in forming compact feature spaces, while comparing (III) and (IV) shows the advantage of our augmentation strategy in preserving realistic sample distributions.

Our contributions are four-fold: 1) We propose YoooP, a novel non-exemplar CIL algorithm that stores and replays a single prototype per class without synthetic data. 2) To the best of our knowledge, we are the first to explore prototype optimization in CIL, which effectively reduces inter-class interference and maintains clear decision boundaries. 3) We extend YoooP to YoooP+, which develops a new prototype augmentation technique that synthesizes high-quality data reflective of the original distribution. 4) Extensive evaluations on multiple benchmarks demonstrate that both YoooP and YoooP+ significantly outperform non-exemplar baselines in terms of average incremental accuracy, average accuracy, and average forgetting.

2 Related Work

Regularization-based method. This method aims to alleviate catastrophic forgetting by introducing additional regularization terms to correct the gradients and protect the old knowledge learned by the model Li & Hoiem (2017); Rannen et al. (2017); Kirkpatrick et al. (2017); Lee et al. (2017); Liu et al. (2018); Masana et al. (2022). Existing works mainly adopt weight regularization to reduce the impact of learning new knowledge on the weights that are important for old tasks. However, it is very hard to design reasonable and reliable metrics to measure the importance of model parameters. Thus, solely using regularization-based methods is always insufficient.

Parameters isolation-based method. This line of work can be divided into dynamic network expansion and static network expansion. Dynamic network expansion methods adopt individual parameters for each task, so they need a large memory to store the extended network for each previous task during training Yoon et al. (2017); Ostapenko et al. (2019); Yan et al. (2021); Xue et al. (2022). Conversely, static network expansion approaches Serra et al. (2018); Mallya & Lazebnik (2018); Mallya et al. (2018); Zhu et al. (2022) dynamically expand the network if its capacity is not large enough for new tasks, and then adapt the expanded parameters into the original network. Those methods can achieve remarkable performance, but they are not applicable to a large number of tasks.

Data replay-based method. This solution Wu et al. (2018); Rostami et al. (2019); Cong et al. (2020) mainly employs deep generative models to generate synthetic samples of old classes in order to mitigate privacy leakage. Most existing works Shin et al. (2017); Rios & Itti (2018); Ostapenko et al. (2019); Lesort et al. (2019) focus on Variational Autoencoder (VAE) and Generative Adversarial Network (GAN). However, these methods suffer from the instability of generative models and inefficient training for complex datasets.

Prototype-based method. Recent works Zhu et al. (2021b;a); Petit et al. (2023) avoid generating pseudo samples by storing class-representative prototypes and then augmenting them to enhance classifier performance and mitigate catastrophic forgetting. Typical approaches include PASS Zhu et al. (2021b) and its variants, IL2A Zhu et al. (2021a), and FeTrIL Petit et al. (2023). However, these methods rely solely on the class-mean prototype without further optimization, which can result in diffuse embedding clusters and unclear decision boundaries between classes. Consequently, severe catastrophic forgetting may occur during incremental training, and inappropriate augmentation can further degrade performance. A recent study Goswami et al. (2024) has leveraged class-mean prototypes as classifiers, using a Nearest Class Mean (NCM) approach to correct prototype drift during sequential training. Nevertheless, compared to standard gradient-based classifiers, NCM is limited in its ability to learn complex decision boundaries when task distributions shift significantly. In this work, we propose prototype optimization to learn a compact feature space that yields more representative prototypes, and we introduce a novel prototype augmentation strategy to generate high-quality synthetic data to mitigate catastrophic forgetting.

3 Proposed Method

In this section, we first describe YoooP, which optimizes the prototype for each class using an attentional mean-shift method. To further improve prediction accuracy, we extend YoooP to YoooP+ by generating synthetic data from the stored prototypes.

Problem Description. Given a sequence of tasks, each associated with a set of classes C_t and a training set $\mathcal{D}t \triangleq (x_i, y_i)i = 1^{n_t}$ with $y_i \in C_t$, class-incremental learning (CIL) aims to train a model $f(x; [\theta, w]) \triangleq G(F(x;\theta); w)$ that predicts probabilities for all classes $C_{1:t} \triangleq \bigcup_{i=1}^{t} C_i$ without catastrophic forgetting. The model consists of a feature extractor $F(\cdot;\theta)$, which produces compact representations, and a classifier $G(\cdot;w)$, with predictions given by $\mathrm{softmax}(G(F(x;\theta);w))$. Since the parameters θ and w are updated solely using the current task's data, the model typically suffers from catastrophic forgetting. To address this challenge, we propose YoooP and its extension YoooP+ to preserve clear class boundaries as tasks are learned sequentially.

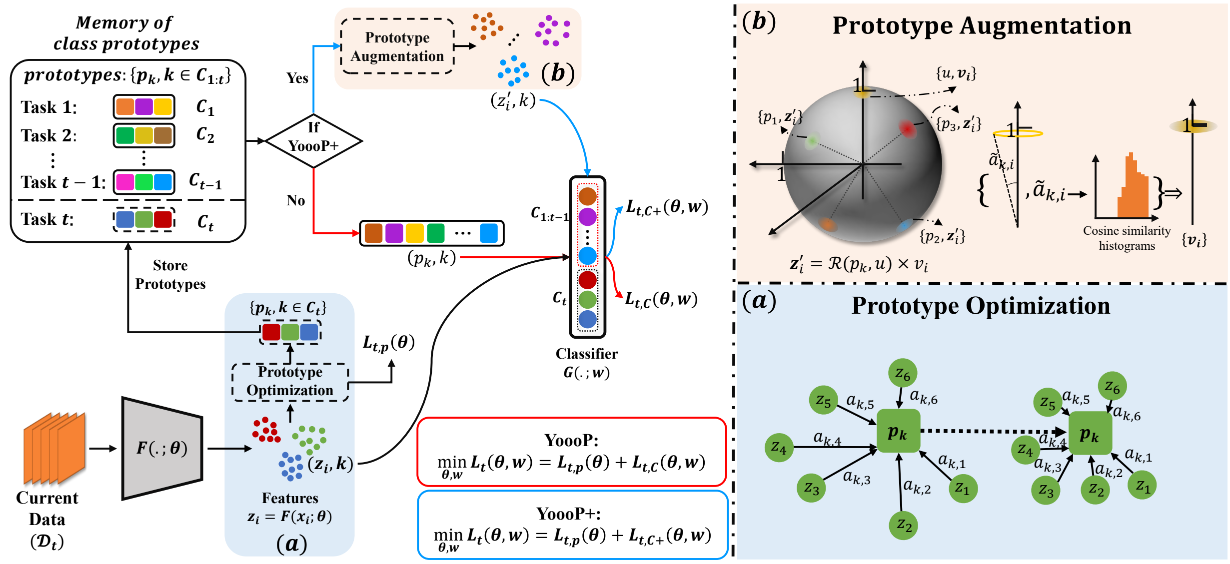


Figure 2: Framework of the proposed YoooP and YoooP+. YoooP only needs to replay one stored prototype for each class while YoooP+ is trained on synthetic data generated from stored prototypes. (a) Prototype optimization aims to learn a compact feature space and obtain a representative prototype for each class. (b) Prototype augmentation aims to generate synthetic data of old classes from stored prototypes and angular distribution using a m-dimensional space rotation matrix.

3.1 YoooP

YoooP comprises two main components: (i) prototype optimization to learn a compact feature space and obtain a representative prototype for each class, and (ii) new task learning with prototype replay.

For (i), we propose a mini-batch attentional mean-shift-based method to shift each class prototype toward high-density regions of its embeddings. Then, we optimize the angular distance between the prototype and the class-specific embeddings to form a tight, compact cluster. As a result, each prototype becomes highly representative. Consequently, replaying only the optimized prototype effectively reduces inter-class interference and maintains clear decision boundaries.

For (ii), when learning a new task, we augment its training set with the stored prototypes from previous tasks. In the YoooP, replaying only the memorized prototypes of old classes can efficiently retain clear boundaries between classes and mitigate catastrophic forgetting.

3.1.1 Prototype Optimization

To obtain a representative prototype without storing all sample embeddings (unlike the traditional class-mean prototype), we propose a mini-batch attentional mean-shift-based method. Specifically, for task-t with classes C_t, and for each class $k \in C_t$, we construct a graph of sample representations $z_i = F(x_i;\theta)$ connected to the prototype p_k. We then shift p_k toward a high-density region by moving it toward a weighted average of the normalized representations of all samples in class-k and subsequently normalizing p_k, *i.e.*,

$$p_k \leftarrow (1-\lambda)p_k + \lambda \sum_{i\in[n_t]:y_i=k} a_{k,i} \cdot \frac{z_i}{\|z_i\|_2}, \quad p_k \leftarrow \frac{p_k}{\|p_k\|_2}, \tag{1}$$

where λ controls the step size of the mean-shift and n_t is the size of the training set for task-t. Unlike the original mean-shift algorithm, the weights $a_{k,i}$ are determined by learnable dot-product attention between each sample z_i and the prototype p_k, *i.e.*,

$$a_k \triangleq \mathrm{softmax}(\bar{a}_k), \quad \bar{a}_k \triangleq [\bar{a}_{k,1}, \cdots, \bar{a}_{k,n_t}], \quad \bar{a}_{k,i} = c(z_i, p_k) \triangleq \frac{\langle z_i, p_k\rangle}{\|z_i\|_2 \cdot \|p_k\|_2}. \tag{2}$$

In practice, when n_t is large, we apply a *mini-batch version* of Eq. 1 over multiple steps, replacing $i \in [n_t]$ with $i \in B$, where B is a mini-batch.

After obtaining a representative prototype for each class, we optimize the similarity between class-specific embeddings and their prototype, to form a compact cluster of each class in the feature space. To achieve that, We train the representation model $F(\cdot;\theta)$ to produce $z_i = F(x_i;\theta)$ such that each sample is close to its class prototype and distant from other classes' prototypes. This is achieved by minimizing the loss as follows,

$$L_{t,P}(\theta) \triangleq \frac{1}{|C_t|}\frac{1}{n_t}\sum_{k\in C_t}\sum_{i\in[n_t]:y_i=k} \ell\left([c(z_i,p_j)]_{j\in C_{1:t}}, k\right), \tag{3}$$

where $\ell(\cdot,\cdot)$ is a cross-entropy loss, in which the first argument is the logits defined by a similarity measure $c(\cdot,\cdot)$, and the second argument k is the ground truth label. Optimizing this loss ensures that embeddings concentrate around their respective prototype, forming compact clusters. Thus, replaying only one prototype per class effectively reduces harmful inter-class interference while training future tasks.

3.1.2 New Task Learning with Prototype Replay

To mitigate catastrophic forgetting of previous tasks' classes $C_{1:t-1}$, YoooP replays stored class prototypes during new task training. Specifically, we augment the training set for task t with prototypes from previous classes $C_{1:t-1}$. The classification objective for all classes $C_{1:t}$ is,

$$L_{t,C}(\theta,w) \triangleq \frac{1}{|C_t|\cdot n_t}\sum_{k\in C_t}\sum_{i\in[n_t]:y_i=k}\ell([c(z_i,w_j)]_{j\in C_{1:t}},k) + \frac{1}{|C_{1:t-1}|}\sum_{k\in C_{1:t-1}}\ell([c(p_k,w_j)]_{j\in C_{1:t}},k), \tag{4}$$

Furthermore, to preserve the performance of learned prototypes while training future tasks, it is necessary to retain the performance of extractor $F(\cdot;\theta)$ on previous tasks. Following previous work Hou et al. (2019); Zhu et al. (2021b), we employ knowledge distillation (KD) Hou et al. (2019) as follows,

$$L_{t,KD}(\theta) \triangleq \frac{1}{n_t}\sum_{i\in[n_t]}\|F(x_i;\theta) - F(x_i;\theta_{t-1})\|_2^2. \tag{5}$$

Thus, the overall training objective at task t is,

$$\text{YoooP}: \min_{\theta,w} L_t(\theta,w) = L_{t,P}(\theta) + L_{t,C}(\theta,w) + \gamma * L_{t,KD}(\theta). \tag{6}$$

where γ is the weight of KD loss.

In summary, $L_{t,P}(\theta)$ shifts the current task's embeddings toward their class prototype, forming compact clusters. $L_{t,C}(\theta,w)$ trains the model using both current task data and the replayed prototypes, while $L_{t,KD}(\theta)$ preserves the extractor's performance on previous tasks. Together, these objectives enable the model to learn new tasks without forgetting previous ones.

3.2 YoooP+

Although prototype-only replay in YoooP is highly effective in mitigating catastrophic forgetting, it is still insufficient to reflect the true embedding distribution of old classes without replaying raw instances. Hence, we propose an extension, YoooP+, which replays synthetic data augmented from the stored prototypes.

3.2.1 Prototype Augmentation.

To generate high-quality synthetic data that matches the real embedding distribution of old classes, we propose a novel prototype augmentation strategy. Our method draws synthetic data for each class from the real angular distribution between the class prototype and its specific embeddings. To simplify the augmentation, we first normalize each prototype to a unit vector, generate synthetic data in the normalized space, and then rotate the synthetic data back. As shown in Fig. 3, we sample cosine similarity values from the stored real angular distribution, $P(\bar{a}_{k,i})$, which is represented by a histogram with N_b bins. These sampled cosine similarities are used to generate synthetic data for each class. Consequently, the angular distribution between each class prototype and its synthetic data faithfully preserves $P(\bar{a}_{k,i})$. In contrast, approaches like PASS add high-dimensional noise to saved prototypes, causing significant divergence from the actual angular distribution.

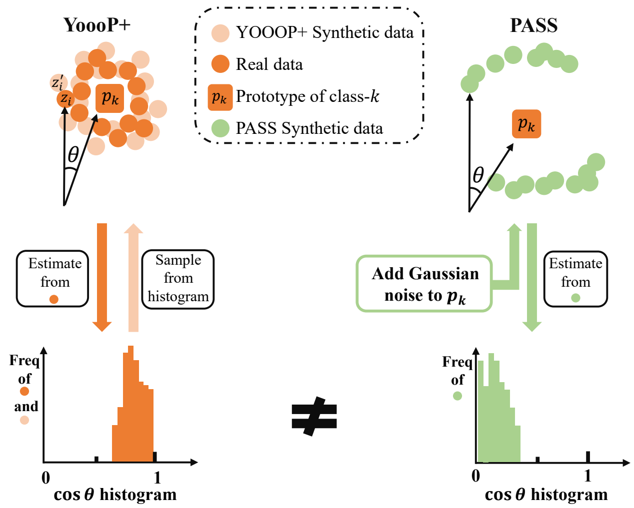


Figure 3: Synthetic data distributions of YoooP+ and PASS with prototype augmentation: **YoooP+ preserves the original angular distribution by a histogram with N_b bins.**

Specifically, by using the stored $P(\bar{a}_{k,i})$, we are able to synthesize a data point z_i' that has a similar angular distance to the prototype p_k as z_i for replaying. This leads to YoooP+ whose replay of each previous class is conducted on multiple synthetic data points instead of a single prototype.

In particular, **we firstly derive a rotation matrix $\mathcal{R}(p_k, \boldsymbol{u})$** that can recover p_k from a unit vector $\boldsymbol{u} = [1, 0, \cdots, 0]$ on an unit m-sphere, *i.e.*, $p_k = \mathcal{R}(p_k, \boldsymbol{u}) \times \boldsymbol{u}$. To synthesize a sample z_i' of class-k as a proxy to z_i (a previously learned sample of class-k), **we then randomly draw $\boldsymbol{v_i}$** in the vicinity of $\boldsymbol{u}$, *i.e.*,

$$\boldsymbol{v_i} = [\tilde{a}_{k,i}, \epsilon_2, \cdots, \epsilon_m], \quad \tilde{a}_{k,i} \sim P(\bar{a}_{k,i}) \tag{7}$$

To ensure $\|\boldsymbol{v_i}\|_2 = 1$, we draw $\epsilon_i \sim \mathcal{N}(0,1)$ for $i \in \{2, \cdots, m\}$ at first and then rescale them by $\epsilon_i \leftarrow \sqrt{1-(\tilde{a}_{k,i}+\epsilon_1)^2 / \sum_{i=2}^{m} \epsilon_i^2} \cdot \epsilon_i$. Thereby, we have $\boldsymbol{u}^T \boldsymbol{v_i} = \tilde{a}_{k,i}$, whose distribution approximates the distribution of cosine similarity $\bar{a}_{k,i}$ between real sample z_i and its associated class prototype p_k.

Next, we create z_i' from $\boldsymbol{v}_i$. As $p_k = \mathcal{R}(p_k, \boldsymbol{u}) \times \boldsymbol{u}$, we apply the same rotation matrix $\mathcal{R}(p_k, \boldsymbol{u})$ to $\boldsymbol{v_i}$ to obtain z_i', *i.e.*,

$$z_i' = \mathcal{R}(p_k, \boldsymbol{u}) \times \boldsymbol{v_i}. \tag{8}$$

This operation preserves the similarity between $\boldsymbol{u}$ and $\boldsymbol{v_i}$ in the transformed space between p_k and z_i'. By sampling a synthetic data point z_i' for each removed real sample z_i, we construct a replay dataset for all seen classes $C_{1:t-1}$.

3.2.2 New Task Learning with Prototype Augmentation

After generating the synthetic data, YoooP+ learns a new task task-t by replaying the synthetic dataset $\mathcal{D}_t'$ as follows,

$$\mathcal{D}_t' \triangleq \left\{(z_i', k) : k \in C_{1:t-1}, z_i' = \mathcal{R}(p_k, \boldsymbol{u}) \times \boldsymbol{v_i}, \boldsymbol{v_i} = [\tilde{a}_{k,i}, \epsilon_2, \cdots, \epsilon_m]\right\}. \tag{9}$$

The training objective for task-t with replayed synthetic data is

$$L_{t,C+}(\theta, w) \triangleq \frac{1}{|C_t| \cdot n_t} \sum_{k \in C_t} \sum_{i \in [n_t]: y_i = k} \ell(c(z_i, w), k) + \frac{1}{|\mathcal{D}_t'|} \sum_{(z,k) \in \mathcal{D}_t'} \ell(c(z, w), k). \tag{10}$$

Overall, the training objective $L_t(\theta, w)$ of YoooP+ at task-t combines the prototype-learning loss, the synthetic-data replay augmented loss, and the KD loss as follows,

$$\text{YoooP+}: \ \min_{\theta, w} L_t(\theta, w) = L_{t,P}(\theta) + L_{t,C+}(\theta, w) + \gamma * L_{t,KD}(\theta). \tag{11}$$

3.3 Practical Improvement to YoooP/YoooP+

Besides, we adopt the following commonly used techniques to mitigate the forgetting.

Model Interpolation. We apply model interpolation to retain the knowledge of the previous model θ_{t-1} and avoid overfitting to the current task. Specifically, after learning task-t, we update the current θ_t by the following interpolation between θ_{t-1} and θ_t, *i.e.*,

$$\theta_t \leftarrow (1-\beta)\theta_{t-1} + \beta\theta_t, \tag{12}$$

where $\beta \in [0,1]$ and we set $\beta = 0.6$ in experiments. Since θ_t is mainly trained on task-t, such simple interpolation between θ_{t-1} and θ_t leads to a more balanced performance on all tasks.

"Partial Freezing" of Classifier. Following Li & Hoiem (2017), instead of completely freezing the classifier parameters for previously learned classes, $w_k, k \in C_{1:t-1}$, we scale down the gradients of these parameters by a small factor α. Specifically, the gradient update is modified as follows,

$$\nabla_{w_k} L_t(\theta, w) \leftarrow \alpha \nabla_{w_k} L_t(\theta, w), \quad \forall k \in C_{1:t-1} \tag{13}$$

This strategy helps prevent significant drift of the classifier parameters for previously learned classes.

We provide the complete procedure of YoooP and YoooP+ in Algorithm 1 in Appendix A.

4 Experiment

In this section, we first compare the proposed YoooP and YoooP+ with non-exemplar-based baselines in three datasets. Then we assess the quality of synthetic data augmented from memorized prototypes. Lastly, we do ablation studies to explore the impact of different key components. We also explore the sensitivity of hyper-parameters in Appendix B.

Datasets. To better evaluate the performance of our proposed methods, we perform on three different scale datasets: CIFAR-100 Krizhevsky et al. (2009), TinyImageNet Yao & Miller (2015), and a subset of ImageNet-1000 Russakovsky et al. (2015) (Sub-ImageNet). CIFAR-100 contains 100 classes of images, which include 50,000 training images and 10,000 test images, and the image size is 32×32. TinyImageNet contains 200 classes of images, which include 100,000 training images and 10,000 test images, and the image size is 64×64. For the Sub-ImageNet, the detailed description and the results are shown in Appendix C.

Experimental settings. We implement all experiments using PyTorch Paszke (2019), and compare our methods with baselines provided by PyCIL Zhou et al. (2021), a popular toolbox for continual incremental learning (CIL). Following prior works Zhu et al. (2021b;a), we adopt ResNet-18 He et al. (2016) as the backbone network for all methods. For YoooP and YoooP+, we train with the SGD optimizer with an initial learning rate of 0.01, which is decayed by a factor of 0.1 every 20 epochs. Models are trained for 60 epochs per task using a batch size of $B = 256$. For knowledge distillation loss (Eq. 6 for YoooP and Eq. 11 for YoooP+), we set a large weight $\gamma = 30$. Additionally, we save the cosine similarity distribution into a histogram consisting of $N_b = 100$ bins within the interval $[0, 1]$ for prototype analysis (Fig. 3). For evaluation, we consider two standard CIL settings: "zero-base," where the total classes are evenly split into multiple incremental phases (*i.e.* 5 or 10 phases) trained sequentially, and "half-base," where half of the classes are learned initially and remaining classes are evenly incremented. Following previous practice Zhu et al. (2021b), all classes in each dataset are arranged in a fixed random order, and reported results are averaged across three runs with a fixed random seed for reproducibility. Additionally, we also perform 20 phases under different settings on TinyImageNet in Appendix D.

Baselines. We compare our proposed YoooP and YoooP+ methods with representative non-exemplar-based methods, including LwF Li & Hoiem (2017), PASS Zhu et al. (2021b), SSRE Zhu et al. (2022), IL2A Zhu et al. (2021a), FeTrIL Petit et al. (2023), and ADC Goswami et al. (2024). Notably, ADC employs an NCM classifier Rebuffi et al. (2017) while YoooP/YoooP+ and other baseline methods perform with a normal classifier with softmax. To further show the impressive performance of our proposed methods, we also compare the performance with some exemplar-based methods on TinyImageNet in Appendix E.

Protocol. We evaluate all methods using three commonly adopted metrics in IL: average incremental accuracy Rebuffi et al. (2017) (AIA), the average accuracy after training the last task Rebuffi et al. (2017) (AA), and average forgetting Chaudhry et al. (2018) (AF).

Table 1: Average incremental accuracy (AIA) and average accuracy after training the last task (AA) of the proposed YoooP/YoooP+ and baselines on CIFAR-100 and TinyImageNet under half-base setting with different phases."half-10" means half-base with 10 phases, "half-5" means half-base with 5 phases. (*) denotes methods using "Nearest Class Mean" (NCM) classifier. **Bold**: the best among non-exemplar methods. Underline: the second best among non-exemplar methods.

Datasets	CIFAR-100				TinyImageNet			
Method	AIA [%]↑		AA [%]↑		AIA [%]↑		AA [%]↑	
	half-5	half-10	half-5	half-10	half-5	half-10	half-5	half-10
LwF Li & Hoiem (2017)	49.00	37.46	27.93	17.09	34.46	23.34	21.34	13.56
SSRE Zhu et al. (2022)	63.38	61.29	52.35	49.27	50.45	47.88	40.48	39.47
IL2A Zhu et al. (2021a)	62.97	52.44	48.89	33.08	45.27	43.34	34.54	33.73
FeTrIL Petit et al. (2023)	**67.57**	**67.43**	**58.38**	**57.45**	51.79	50.10	42.51	41.76
*ADC Goswami et al. (2024)	65.62	61.71	54.31	49.12	52.36	47.12	43.00	37.18
PASS Zhu et al. (2021b)	64.10	57.41	54.80	45.71	48.61	39.99	38.69	30.23
PASS w/o Aug	59.34	55.41	49.64	43.02	46.06	42.84	37.12	33.56
YoooP (Ours)	66.19	58.99	56.61	47.09	62.30	58.66	53.94	49.40
YoooP+ (Ours)	67.40	61.83	58.54	50.81	**65.57**	**61.72**	**58.56**	**52.40**

Table 2: Average incremental accuracy (AIA) and average accuracy after training the last task (AA) of the proposed YoooP/YoooP+ and baselines on CIFAR-100 and TinyImageNet under zero-base setting with different phases."b0-10" means zero-base with 10 phases, "b0-5" means zero-base with 5 phases. (*) denotes methods using "Nearest Class Mean" (NCM) classifier. **Bold**: the best among non-exemplar methods. Underline: the second best among non-exemplar methods.

Datasets	CIFAR-100				TinyImageNet			
Method	AIA [%]↑		AA [%]↑		AIA [%]↑		AA [%]↑	
	b0-5	b0-10	b0-5	b0-10	b0-5	b0-10	b0-5	b0-10
LwF Li & Hoiem (2017)	58.95	47.73	39.87	25.88	46.44	35.00	29.50	17.80
SSRE Zhu et al. (2022)	58.05	46.58	40.99	29.75	47.13	38.54	29.90	22.78
IL2A Zhu et al. (2021a)	59.91	42.92	43.69	27.21	42.16	33.72	25.67	22.46
FeTrIL Petit et al. (2023)	61.41	48.61	44.54	31.07	44.55	36.51	28.23	20.59
*ADC Goswami et al. (2024)	68.60	61.93	57.49	47.00	60.09	53.51	47.46	37.32
PASS Zhu et al. (2021b)	60.33	51.94	43.61	35.81	47.11	40.15	31.05	26.31
PASS w/o Aug	59.02	55.47	45.19	42.41	50.25	42.71	36.27	30.17
YoooP (Ours)	65.24	58.99	52.08	44.78	60.26	53.73	49.04	40.26
YoooP+ (Ours)	**69.61**	**63.30**	**57.56**	**49.60**	**61.92**	**56.47**	**51.32**	**44.25**

4.1 Main Results

In this section, we compare the performance of our proposed methods YoooP and YoooP+ across various datasets under different settings with various baselines and discuss the results.

Evaluation on half-base setting. We first evaluate the proposed methods under half-base settings with 5 and 10 phases. Results shown in Tab. 1 indicate that YoooP/YoooP+ consistently performs well across CIFAR-100 and TinyImageNet datasets. On CIFAR-100, YoooP+ attains performance comparable to FeTrIL, which achieves strong results by training the model only at the initial task while fixed in the future tasks. Although FeTrIL effectively preserves initial task knowledge, this strategy severely restricts the model's capacity to adapt and improve representations for subsequent tasks, particularly when these differ significantly from the initial task. Thus, the FeTrIL fails to achieve great performance if the initial task contains little knowledge and the future tasks have a large gap between the first task. Therefore, the proposed YoooP and YoooP+ beat the FeTrIL on both datasets under zero-base settings and surpass the FeTrIL on TinyImageNet under half-base settings. Specifically, on TinyImageNet, YoooP outperforms FeTrIL by 10.51% and 8.56% (AIA), and 11.43% and 7.64% (AA) under the respective phases. Furthermore, YoooP+ exhibits even greater improvements over FeTrIL, achieving 13.78% and 11.62% (AIA), and 16.05% and 10.64% (AA). YoooP and YoooP+ also clearly surpass ADC by a large margin, reinforcing our methods' capability to dynamically capture richer class representations, particularly on more challenging datasets like TinyImageNet. Similarly, the incremental accuracy progression under the half-base setting is depicted in the upper row of Fig. 4.

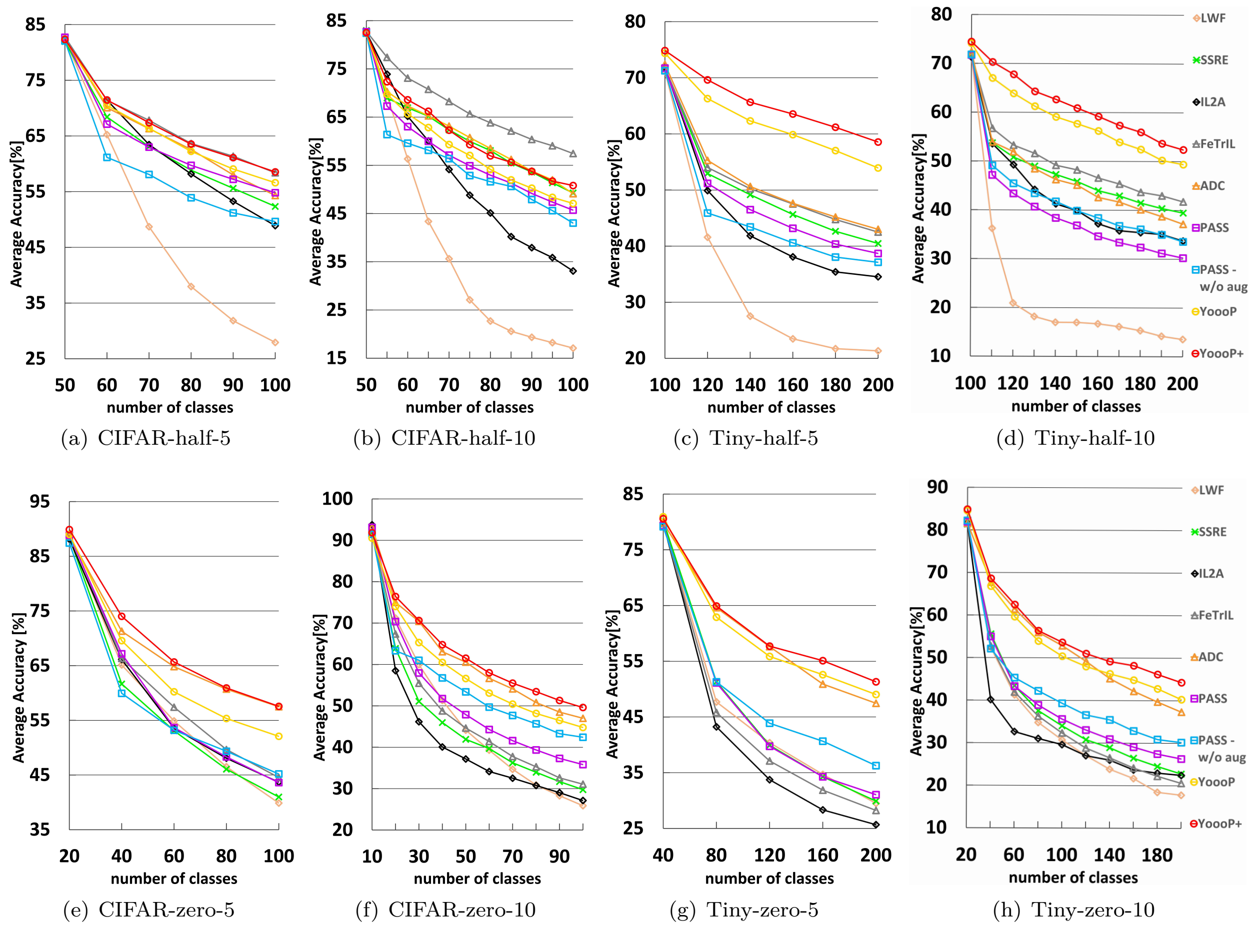
Evaluation on zero-base setting. Next, we evaluate the proposed methods against baseline approaches on CIFAR-100 and TinyImageNet datasets under zero-base settings with 5 and 10 phases. As shown in Tab. 2, our proposed YoooP consistently outperforms non-NCM baselines that rely on the standard classifier layer. Specifically, compared to PASS, which achieves the best results among non-NCM baselines, YoooP achieves accuracy improvements of 4.91% and 7.05% (AIA) and 8.47% and 8.97% (AA) under 5 and 10 phases respectively on CIFAR-100. On TinyImageNet, the advantages of YoooP are even more pronounced, exceeding PASS by 13.15% and 13.58% (AIA), and by 17.99% and 13.95% (AA) under the respective phases. This impressive performance is because YoooP leverages prototype optimization, enabling the model to learn more discriminative and compact representations. To further mitigate catastrophic forgetting, YoooP+ augments the training with synthetic data derived from these optimized prototypes. Consequently, YoooP+ surpasses PASS by 9.28%, 11.36% (AIA), and by 13.95%, 13.79% (AA) on CIFAR-100. On TinyImageNet, the performance gap is even more notable: 14.81%, 16.32% for AIA, and 20.27%, 17.94% for AA under the respective phases. Additionally, YoooP surpasses ADC, which employs the Nearest Class Mean (NCM) classifier, on TinyImageNet under zero-base settings and YoooP+ significantly surpasses ADC on both datasets under zero-base settings. We also show the performance while training the model incrementally on each task in the bottom row in Fig. 4.

Evaluation on Average Forgetting. We also evaluate the average forgetting (AF) under zero-base settings on CIFAR-100 and TinyImageNet. As shown in Tab. 3, our proposed methods, YoooP and YoooP+, achieve relatively low average forgetting compared to most baseline methods. Although some baselines exhibit lower AF, this is primarily due to their sharp accuracy decline in initial incremental phases followed by persistently low performance, as shown in Tab. 2 and clearly illustrated by the incremental accuracy curves in the bottom row of Fig. 4. In contrast, YoooP and YoooP+ effectively maintain higher incremental accuracy (AIA and AA) throughout all phases (Tab. 2), genuinely mitigating catastrophic forgetting while maintaining superior overall performance.

Table 3: Average forgetting (AF) of the proposed YoooP/YoooP+ and baselines on CIFAR-100 and TinyImageNet under zero-base setting with different phases."b0-10" means zero-base with 10 phases, "b0-5" means zero-base with 5 phases. (*) denotes methods using "Nearest Class Mean" (NCM) classifier. **Bold**: the best among non-exemplar methods. <u>Underline</u>: the second best among non-exemplar methods.

AF [%]↓	CIFAR-100		TinyImageNet	
Methods	b0-5	b0-10	b0-5	b0-10
LwF Li & Hoiem (2017)	43.80	51.80	45.79	54.40
SSRE Zhu et al. (2022)	**15.44**	**12.13**	16.31	19.94
IL2A Zhu et al. (2021a)	26.94	25.07	20.89	26.10
FeTrIL Petit et al. (2023)	18.88	16.14	<u>15.13</u>	**15.32**
*ADC Goswami et al. (2024)	19.15	21.21	15.35	23.92
PASS Zhu et al. (2021b)	23.66	18.78	22.00	20.69
PASS w/o Aug	28.11	29.55	24.01	26.00
YoooP (Ours)	21.24	22.69	16.49	25.02
YoooP+ (Ours)	<u>18.75</u>	<u>15.41</u>	**14.28**	<u>19.24</u>

Discussion. Our experimental results yield three main findings: (1). Comparing PASS w/o Aug and PASS in Tab. 2 and Tab. 1, adding high-dimensional Gaussian noise for prototype augmentation can hurt performance, especially under zero-base settings and the half-base setting with 10 phases on TinyImageNet. This is because high-dimensional Gaussian noise produces sparse, nearly equidistant data (the curse of dimensionality). In contrast, YoooP+ generates synthetic data in angular space using a rotation matrix while preserving the real angular distribution, which consistently improves prototype performance. (2). The ADC with NCM shows an advantage only in the CIFAR-100 zero-base setting. As shown in Tab. 2 and Tab. 1, ADC outperforms YoooP on CIFAR-100 under zero-base conditions but does not surpass YoooP+ and fails to outperform YoooP in other settings. Furthermore, in the half-base configuration, several non-NCM



(a) CIFAR-half-5 (b) CIFAR-half-10 (c) Tiny-half-5 (d) Tiny-half-10

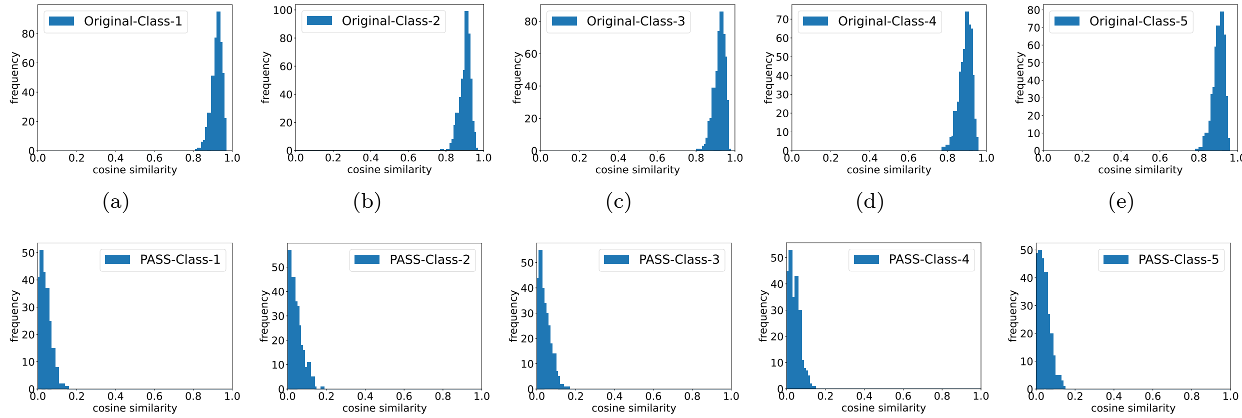
(e) CIFAR-zero-5 (f) CIFAR-zero-10 (g) Tiny-zero-5 (h) Tiny-zero-10

Figure 4: Performance comparison of each task across different methods on CIFAR-100 and TinyImageNet under different settings while training model incrementally. "zero-5,10": "zero-base" with 5 and 10 phases settings. "half-5,10": "half-base" with 5 and 10 phases settings.

methods outperform ADC. This occurs because NCM relies solely on class-mean prototypes and avoids gradient updates to mitigate catastrophic forgetting, limiting its ability to learn complex decision boundaries when task distributions shift significantly. In contrast, a normal classifier with softmax benefits from gradient updates that enable the learning of complex boundaries and better generalization. In the half-base setting, where the initial task has abundant data and later tasks have fewer samples, the pronounced distribution shift leads to a performance drop (comparing the performance of ADC between two tables) and the loss of ADC's initial advantage. (3). The proposed YoooP and YoooP+ consistently outperform other non-exemplar-based methods across different datasets under various settings. They achieve higher AIA and AA while maintaining lower AF, demonstrating robust performance.

4.2 Comparison of Synthetic Data for YoooP+ and PASS

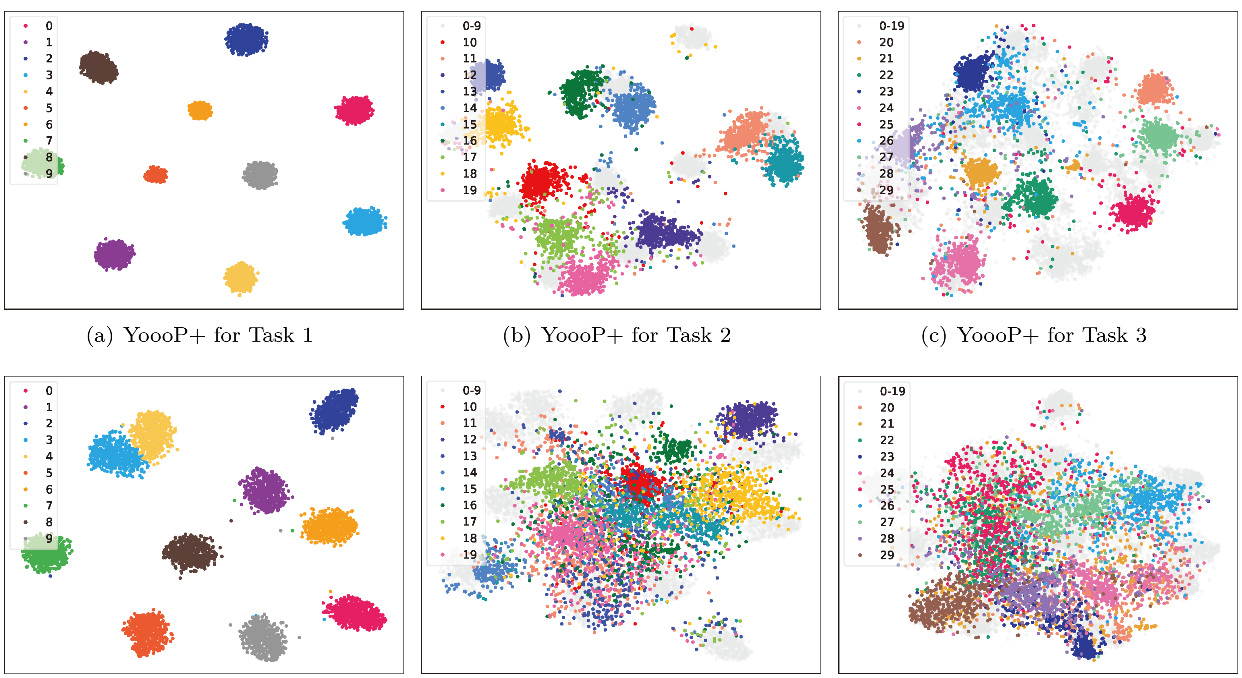
In this experiment, we randomly selected five classes from the CIFAR-100 dataset (task t) under a zero-base setting with 10 phases. We then compared the angular distributions of synthetic data generated from stored prototypes by YoooP+ and PASS, as illustrated in Fig. 5. The upper row of Fig. 5 pertains to the original distribution (also the angular distribution stored in YoooP+), representing the cosine similarities between the representations $F(\cdot;\theta)$ and the stored prototypes for each class. In contrast, the bottom row of Fig. 5 shows the distributions of cosine similarities between the same prototypes and the synthetic data generated by PASS. We observe that PASS produces synthetic samples whose cosine similarities are heavily concentrated



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 5: **Top row**: histograms of the original cosine similarity $\bar{a}_{k,i}$ (Eq. 2) between each class's prototype and the real samples (top). The augmented samples of YoooP+ are drawn from the original histograms. **Bottom row:** histograms of the cosine similarity between each class's prototype and the augmented samples for PASS. PASS fails to preserve the distribution of the original data.



(a) YoooP+ for Task 1 (b) YoooP+ for Task 2 (c) YoooP+ for Task 3

(d) PASS for Task 1 (e) PASS for Task 2 (f) PASS for Task 3

Figure 6: Visualization of the distribution of representations encoded by YoooP+ and PASS on CIFAR-100 base-0 phase 10 setting. The lighter gray points in “Task 2" and “Task 3" represent the distribution of the previous tasks' data.

near 1.0, indicating that these augmented data are nearly identical to the stored prototypes. This narrow distribution leads to reduced diversity and, consequently, diminished performance, as also evidenced in Fig. 4, Tab. 2, and Tab. 1. In comparison, YoooP+ draws its synthetic data based on the original cosine-similarity distribution (top row). This strategy more effectively restores the representations of the original data and helps YoooP+ generate higher-quality synthetic samples than PASS.

4.3 Comparison of the representations for YoooP+ and PASS

In addition, we compare the learned representations produced by YoooP+ and PASS. Fig. 6 shows the distribution of representations on CIFAR-100 under zero-base setting with 10 phases for the first three tasks. Because YoooP+ employs prototype optimization in the first task, it forms more compact class clusters than PASS. In Tasks 2 and 3, YoooP+ continues to encode input data within well-defined boundaries, whereas PASS does not. In Fig. 6 (b), (c), (e), and (f), the light gray points represent data from previous tasks. We observe in (b) and (c) that YoooP+ keeps old and new tasks well separated, while PASS struggles to distinguish between the distributions of old and current tasks. This is because YoooP+ not only creates compact clusters via prototype optimization but also synthesizes high-quality data from the original cosine-similarity distribution, effectively preserving boundaries for old tasks.

4.4 Ablation Studies

We conduct ablation studies to assess the influence of three key components on model performance: prototype optimization (P), synthetic data replay, and model interpolation (MI). Fig. 7 presents the results on CIFAR-100 under the zero-base setting with 10 phases. In particular, YoooP (Class Mean) uses the class-mean for prototype generation (similar to PASS) while still employing the prototype optimization loss in Eq. 3, YoooP (-P) omits prototype optimization, YoooP (-MI) excludes model interpolation, and YoooP (-P-MI) removes both. From Fig. 7, prototype optimization proves crucial, as YoooP (-P) suffers a substantial drop in accuracy compared to the full YoooP. When comparing YoooP and YoooP (Class Mean), YoooP achieves slightly higher accuracy while utilizing a mini-batch attentional mean-shift-based method that requires fewer stored sample embeddings than the conventional class-mean approach (see Appendix F). Moreover, YoooP+ (which incorporates prototype augmentation) further improves prediction accuracy relative to YoooP. Although model interpolation (MI) contributes a modest performance boost by retaining prior knowledge and ensuring current-task performance, the difference between YoooP (-P) and the other variants indicates that prototype optimization remains the most critical factor.

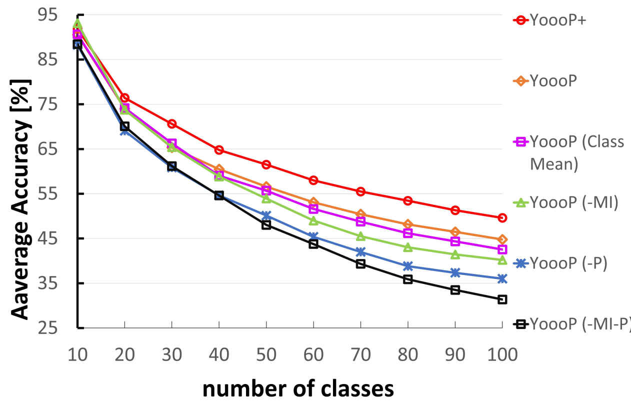


Figure 7: Ablation study of different components in YoooP+. “-M” means without model interpolation, “-P” means without prototype optimization.

5 Conclusion

In this work, we developed two non-exemplar-based methods, YoooP and YoooP+, for class-incremental learning. Specifically, YoooP only needs to store and replay one optimized prototype for each class without generating synthetic data from stored prototypes. As an extension of YoooP, YoooP+ proposed to create synthetic data from the stored prototypes and the stored distribution of cosine similarity with the help of a high-dimensional rotation matrix. The evaluation results on multiple benchmarks demonstrated that both YoooP and YoooP+ can significantly outperform the baselines in terms of accuracy and average forgetting. Importantly, this work offered a new perspective on optimizing class prototypes for exemplar-free CIL. We also show more experimental results in Appendix G.

References

Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 532–547, 2018.

Yizong Cheng. Mean shift, mode seeking, and clustering. *IEEE transactions on pattern analysis and machine intelligence*, 17(8):790–799, 1995.

Yulai Cong, Miaoyun Zhao, Jianqiao Li, Sijia Wang, and Lawrence Carin. Gan memory with no forgetting. *Advances in Neural Information Processing Systems*, 33:16481–16494, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9285–9295, 2022.

Alexander Gepperth and Barbara Hammer. Incremental learning algorithms and applications. In *European symposium on artificial neural networks (ESANN)*, 2016.

Dipam Goswami, Albin Soutif-Cormerais, Yuyang Liu, Sandesh Kamath, Bart Twardowski, Joost Van De Weijer, et al. Resurrecting old classes with new data for exemplar-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28525–28534, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 831–839, 2019.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Sang-Woo Lee, Jin-Hwa Kim, Jaehyun Jun, Jung-Woo Ha, and Byoung-Tak Zhang. Overcoming catastrophic forgetting by incremental moment matching. *Advances in neural information processing systems*, 30, 2017.

Timothée Lesort, Hugo Caselles-Dupré, Michael Garcia-Ortiz, Andrei Stoian, and David Filliat. Generative models from the perspective of continual learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2262–2268. IEEE, 2018.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7765–7773, 2018.

Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 67–82, 2018.

Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D Bagdanov, and Joost Van De Weijer. Class-incremental learning: survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2022.

Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pp. 109–165. Elsevier, 1989.

Oleksiy Ostapenko, Mihai Puscas, Tassilo Klein, Patrick Jahnichen, and Moin Nabi. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11321–11329, 2019.

A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. Fetril: Feature translation for exemplar-free class-incremental learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3911–3920, 2023.

Amal Rannen, Rahaf Aljundi, Matthew B Blaschko, and Tinne Tuytelaars. Encoder based lifelong learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1320–1328, 2017.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Amanda Rios and Laurent Itti. Closed-loop memory gan for continual learning. *arXiv preprint arXiv:1811.01146*, 2018.

Ryne Roady, Tyler L Hayes, Hitesh Vaidya, and Christopher Kanan. Stream-51: Streaming classification and novelty detection from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 228–229, 2020.

Mohammad Rostami, Soheil Kolouri, and Praveen K Pilly. Complementary learning for overcoming catastrophic forgetting using experience replay. *arXiv preprint arXiv:1903.04566*, 2019.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pp. 4548–4557. PMLR, 2018.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017.

Fu-Yun Wang, Da-Wei Zhou, Liu Liu, Han-Jia Ye, Yatao Bian, De-Chuan Zhan, and Peilin Zhao. Beef: Bi-compatible class-incremental learning via energy-based expansion and fusion. In *The eleventh international conference on learning representations*, 2022a.

Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. *arXiv preprint arXiv:2204.04662*, 2022b.

Liyuan Wang, Kuo Yang, Chongxuan Li, Lanqing Hong, Zhenguo Li, and Jun Zhu. Ordisco: Effective and efficient usage of incremental unlabeled data for semi-supervised continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5383–5392, 2021.

Chenshen Wu, Luis Herranz, Xialei Liu, Joost Van De Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Advances in Neural Information Processing Systems*, 31, 2018.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 374–382, 2019.

Mengqi Xue, Haofei Zhang, Jie Song, and Mingli Song. Meta-attention for vit-backed continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 150–159, 2022.

Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3014–3023, 2021.

Leon Yao and John Miller. Tiny imagenet classification with convolutional neural networks. *CS 231N*, 2(5):8, 2015.

Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13208–13217, 2020.

Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, and De-Chuan Zhan. Pycil: A python toolbox for class-incremental learning. *arXiv preprint arXiv:2112.12533*, 2021.

Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-lin Liu. Class-incremental learning via dual augmentation. *Advances in Neural Information Processing Systems*, 34:14306–14318, 2021a.

Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5871–5880, 2021b.

Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9296–9305, 2022.