Neural Networks 196 (2026) 108389

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Full Length Article

Dual prototypes for adaptive pre-trained model in class-incremental learning

Zhiming Xu [a,b], Suorong Yang [a,c,*], Baile Xu [a,b,*], Furao Shen [a,b], Jian Zhao [d]

[a] State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210023, China
[b] School of Artificial Intelligence, Nanjing University, China
[c] Department of Computer Science and Technology, Nanjing University, China
[d] School of Electronic Science and Engineering, Nanjing University, China,

ARTICLE INFO

Keywords:
Class-incremental learning
Dual prototypes
Pre-trained models
Task-wise adaptation

ABSTRACT

Class-incremental learning (CIL) aims to learn new classes while retaining previous knowledge. Although pre-trained model (PTM) based approaches show strong performance, directly fine-tuning PTMs on incremental task streams often causes renewed catastrophic forgetting. This paper proposes a Dual-Prototype Network with Task-wise Adaptation (DPTA) for PTM-based CIL. For each incremental learning task, an adapter module is built to fine-tune the PTM, where the center-adapt loss forces the representation to be more centrally clustered and class separable. The dual prototype network improves the prediction process by enabling test-time adapter selection, where the raw prototypes deduce several possible task indexes of test samples to select suitable adapter modules for PTM, and the augmented prototypes that could separate confusable classes are utilized to determine the final result. Experiments on multiple benchmarks show that DPTA consistently surpasses recent methods by 1–5%. Notably, on the VTAB dataset, it achieves approximately 3% improvement over state-of-the-art methods. The implementation is available at https://github.com/Yorkxzm/DPTA.

1. Introduction

With the rapid developments of deep learning, deep models have achieved remarkable performances in many scenarios (Bahi et al., 2024; Chen et al., 2022; Liu et al., 2015; Ye et al., 2019). Most of them train models on independently and identically distributed (i.i.d.) data. In real-world settings, however, data often arrives as streams with shifting distributions (Gomes et al., 2017). Training deep models on such non-i.i.d. streams causes previously learned knowledge to be overwritten by new information, a phenomenon known as catastrophic forgetting (French, 1999). This challenge highlights the need for stable incremental learning systems in practical applications.

Among various forms of incremental (continual) learning (De Lange et al., 2021; Roy et al., 2020), class-incremental learning (CIL) receives the most attention due to its closer alignment with real-world applications (Wang et al., 2024; Zhou et al., 2024d). Specifically, CIL builds a model to continually learn new classes from data streams. Previous CIL works are primarily based on sample replay (Rebuffi et al., 2017; Wang et al., 2025), regularization (Nguyen et al., 2017), and distillation (Rolnick et al., 2019). These methods rely on additional replay samples and more trainable parameters to achieve strong performance. More recently, methods based on Pre-Trained Models (PTMs) (Zhou et al., 2024b) have achieved significant progress by leveraging models pre-trained on large-scale corpora (Radford et al., 2021) or image datasets (Barbu et al., 2019; Deng et al., 2009). These approaches keep PTM weights frozen during CIL training, preserving knowledge from pre-training and substantially mitigating catastrophic forgetting.

Despite surpassing previous approaches, massive patterns in downstream incremental tasks are unexposed to PTMs during pre-training. To enhance the model's performance, fine-tuning the PTM with training samples is typically employed. Given that CIL tasks arrive as a continuous stream, sequentially updating the PTM on task streams risks reintroducing catastrophic forgetting. Recently, some approaches (Smith et al., 2023; Zhou et al., 2024a,c) suggest the *task adaptation* strategy for PTM-based methods, which assigns several free-loading and lightweight fine-tuning modules for incremental tasks, such as scale & shift (Lian et al., 2022), adapter (Houlsby et al., 2019), or Visual Prompt Tuning (VPT) (Jia et al., 2022), then load the appropriate module trained in the corresponding task for PTM to extract representations. However, because task identities are unavailable at test time, selecting the correct module

* Corresponding authors.
E-mail addresses: york_z_xu@smail.nju.edu.cn (Z. Xu), sryang@smail.nju.edu.cn (S. Yang), xubaile@nju.edu.cn (B. Xu), frshen@nju.edu.cn (F. Shen), jianzhao@nju.edu.cn (J. Zhao).

https://doi.org/10.1016/j.neunet.2025.108389
Received 1 July 2025; Received in revised form 25 November 2025; Accepted 28 November 2025
Available online 30 November 2025

becomes challenging. Existing solutions rely on complex query matching (Wang et al., 2022d), module combinations (Smith et al., 2023), or ensembles (Zhou et al., 2024c), but these approaches are often limited: key-value matching is unreliable, and ensembling or combining modules does not guarantee that the correct task-specific representation dominates, thereby introducing noise.

Most PTM-based CIL approaches also classify samples by comparing their representations to class prototypes (Bezdek & Kuncheva, 2001). Yet the losses commonly used to fine-tune PTMs are not tailored to prototype-based classification. For example, cross-entropy loss encourages clear decision boundaries but often yields large intra-class variation, causing many samples to become more similar to incorrect prototypes and thus degrading classification accuracy.

To address these challenges, we propose DPTA, a Dual-Prototype Network with task-wise adaptation for PTM-based CIL. We introduce a center-adapt loss that encourages adapted representations to cluster around class centers while enlarging inter-class margins. Similar to existing adaptation strategies (Zhou et al., 2024a,c), each task is equipped with an adapter to fine-tune the PTM. We observe that top-K predictions derived from prototypes remain reliable during CIL, offering valuable cues for selecting appropriate adapters. Building on this insight, DPTA decomposes prediction into two subproblems: top-K candidate label inference and K-class classification. Raw prototypes, derived from the frozen or first-task adapted PTM, estimate the top-K possible labels of a sample, from which task indices and the corresponding adapters are inferred. Augmented prototypes, computed from the task-adapted PTM, are then used to determine the final predicted label. The main contributions of our work are summarized as follows:

- We introduce a center-adapt loss tailored for prototype-based classification, producing more compact and discriminative augmented prototypes.
- We propose the Dual Prototype Network, where raw prototypes estimate top-K candidate labels and augmented prototypes refine the prediction to the top-1 result.
- We conduct extensive experiments across multiple incremental benchmarks, demonstrating that DPTA achieves state-of-the-art performance.

The remainder of the paper is organized as follows. Section 2 reviews related work. Section 3 discusses PTM-based CIL and prototype classifiers. Section 4 presents the proposed method. Section 5 reports experimental results, and Section 6 concludes with discussions on strengths, limitations, and future directions.

2. Related works

2.1. Class-incremental learning and previous methods

A CIL model must continually absorb new class knowledge from sequential tasks during training and make predictions without access to task identities at test time. Existing approaches can be broadly grouped into the following types: Replay-based methods (De Lange & Tuytelaars, 2021; Rolnick et al., 2019; Wang et al., 2025) deposit typical samples into a buffer as exemplars for subsequent training to recover old class knowledge. For instance, iCaRL (Rebuffi et al., 2017) selects exemplars near the class mean representation. Regularization-based methods (Ahn et al., 2019; Zeno et al., 2018) protect the knowledge obtained on the old task by adding regularization to limit the model parameters' updating on the new task. However, they tend to restrict the model's updates on new tasks. To address this limitation, recent flatness-based methods, such as C-flat (Bian et al., 2024) and C-flat++ (Li et al., 2025), mitigate this limitation by encouraging convergence to flatter loss regions. Distillation-based methods (Simon et al., 2021) transfer knowledge from the old model to the new one through feature-level (Zhu et al., 2021), logit-level (Zhao et al., 2020), or correlation-based distillation (Gao et al., 2025c). However, distillation may interfere with learning new classes. For example, logit distillation can conflict with cross-entropy optimization. Gao et al. (2025a) address this by introducing semantic-invariant matching and intra-class distillation. Parameter isolation methods like DER (Yan et al., 2021) and MEMO (Zhou et al., 2022) assign separate parameters to different tasks to prevent forgetting. Additionally, there exist several plug-and-play logit calibration techniques that effectively enhance the model's accuracy in few-shot scenarios (Liu et al., 2024) or mitigate the class imbalance issue induced by the replay buffer (Wang et al., 2022a). These methods typically require extra training of a large neural network.

2.2. Class-incremental learning with pre-trained models

The strong generalization ability of Pre-Trained Models (PTMs) has motivated their adoption in CIL (Tan et al., 2024; Zhang et al., 2023). These methods typically freeze PTM backbones and use them as feature extractors, which naturally encourages the use of prototype-based classifiers (Zhu et al., 2025a,b). SimpleCIL (Zhou et al., 2024a) demonstrates that excellent CIL classification accuracy can be achieved by only building prototypes with raw PTMs. A basic adaptation strategy is *first-task adaptation*, as in APER (Zhou et al., 2024a), which fine-tunes a PTM on the first task using an adapter, then constructs prototypes with the adapted model. Prompt-based approaches (Smith et al., 2023; Wang et al., 2022b,c,d) maintain a pool of prompts trained across tasks, making prompt selection at test time a key challenge. L2P (Wang et al., 2022d) adopts key-query matching; DualPrompt (Wang et al., 2022c) introduces layer-wise prompts divided into expert and shared groups; CODA-Prompt (Smith et al., 2023) applies attention to form weighted prompt combinations. Adapter-based methods have also emerged. EASE (Zhou et al., 2024c) trains a separate adapter for each task and performs subspace prototype ensembling. MOS (Sun et al., 2025) proposes training-free adapter retrieval for efficiency. MoAL (Gao et al., 2025b) performs momentum-based adapter interpolation and introduces a knowledge-rumination mechanism to reinforce old knowledge. In addition, several works have been proposed to accelerate inference speed, like ACmap (Fukuda et al., 2025) with adapter merging and MINGLE (Qiu et al., 2025) for LoRA mixture. Following the structure of prior surveys such as (Bahi et al., 2024), Table 1 summarizes representative PTM-based CIL methods developed in recent years.

3. Preliminaries

In this section, we provide a introduction to PTM-based CIL and introduce prototype-based classification, one of the most prevalent classifiers in CIL .

3.1. Class-incremental learning with PTMs

In class-incremental learning (CIL), a model learns from a sequence of tasks, each introducing new classes, and must make predictions without access to task identities (Wang et al., 2024). Assume that there are T tasks, their training sets are denoted as $D_1, D_2, \dots, D_t$, where $D_t = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n_t}$ is the tth training set that has n_t samples and $D_1, D_2, \dots, D_t$ are non-i.i.d. In the tth task, only D_t can be accessed for training. The learning system is trained to obtain an optimal model $f^*(\boldsymbol{x}) : \mathcal{X} \to \mathcal{Y}$ that minimizes the expected risk for all classes in $D_1 \cup \dots \cup D_t$. The objective function is represented in Eq. (1):

$$f^*(\boldsymbol{x}) = \arg\min_{f \in \mathcal{H}} \mathbb{E}_{(\boldsymbol{x},y)\in D_1 \cup \dots \cup D_t}[\mathbb{I}(f(\boldsymbol{x}) \neq y)]. \tag{1}$$

where $\mathbb{I}$ is the indicator function and $\mathcal{H}$ denotes the hypothesis space of model f.

A widely adopted strategy in CIL is leveraging pre-trained models (PTMs) such as Vision Transformers (Dosovitskiy, 2020) as feature extractors (Zhou et al., 2024b). The decision model of PTM-based methods can be expressed as $f(\boldsymbol{x}) = W^{\mathrm{T}}\phi(\boldsymbol{x})$, where the $\phi(\cdot) : \mathbb{R}^d \to \mathbb{R}^h$ is the PTM

Table 1
A summary of representative PTM-based class-incremental learning methods.

Approach	Year	Method description	Avantages
L2P (Wang et al., 2022d)	2022	Several learnable prompts on sequential tasks and dynamically selecting task-relevant prompts.	Few trainable parameters.
CODA-prompt (Smith et al., 2023)	2023	Use attention mechanism on prompt building and create a weighted combination of prompts.	Do not choose prompts, which can reduce the error.
SimpleCIL (Zhou et al., 2024a)	2023	Only use the original PTM and prototype classifier.	Simple, fast inference speed, short training time.
APER (Zhou et al., 2024a)	2023	Use the fine-tune module to adapt the SimpleCIL on the first task.	Higher but limited accuracy, fast inference speed.
EASE (Zhou et al., 2024c)	2024	Train task-specific adapters with a subspace ensemble process.	Integrated information from all adapters for prediction.
MOS (Sun et al., 2025)	2025	Train task-specific adapters with a training-free self-refined adapter retrieval mechanism.	Less inference time and higher accuracy than EASE.
MoAL (Gao et al., 2025b)	2025	A momentum-updated analytic learner with knowledge rumination mechanism.	High accuracy, fast inference speed, and few adapters.

feature extractor, and W is a customized classifier. Since PTMs encode rich prior knowledge, the extracted representations have both generalizability and adaptivity (Zhou et al., 2024a). Freezing the PTM during incremental training further prevents catastrophic forgetting.

3.2. Prototype classifier in CIL

Although fully connected classifiers with softmax activation (Krizhevsky et al., 2012) are common in deep learning, their need for full re-training makes them less suited for the growing label space in CIL. A widely used classifier in CIL is the prototype-based Nearest Class Mean (NCM) (Xu et al., 2020). For a new class k with M_k training samples, its prototype $\boldsymbol{p_k}$ is computed in Eq. (2):

$$\boldsymbol{p_k} = \frac{1}{M_k}\sum_{j=1}^{M_k} \mathbb{I}(y_i = k)\phi(\boldsymbol{x_i}), \tag{2}$$

where y_i is the label corresponding to sample $\boldsymbol{x}_i$. During inference, the similarity between a test representation and each prototype is measured using cosine similarity (Yang et al., 2018). With N learned classes, the prediction is denoted in Eq. (3):

$$\hat{y}_i = \underset{k=1,2,\dots N}{\arg\max} (Sim(\phi(\boldsymbol{x_i}), \boldsymbol{p_k})_{\cos}). \tag{3}$$

Unlike linear classifiers, prototypes can be updated through simple class-wise additions without retraining, making them highly suitable for CIL.

4. Methodology

Freezing PTMs and using one prototype per class often fails to separate many classes in downstream tasks, while incrementally fine-tuning PTMs typically causes catastrophic forgetting. As illustrated in Fig. 1, we address this by assigning a lightweight adapter to each task and constructing dual prototypes for each class. During inference, we adopt a two-step procedure. First, we obtain top-K candidate labels from raw prototypes and infer the underlying task index. Then we load the corresponding task adapters and obtain the final prediction from augmented prototypes.

In the following, we first introduce task adaptation with the proposed Center-Adapt loss, which improves the separability of PTM representations for prototype-based classification. Then, present the dual prototype network (DPN), which defines the classifier of DPTA. Finally, we describe the inference process.

4.1. Task adaptation with center-adapt loss

Fig. 2(a) and (b) show that training with standard cross-entropy (CE) loss enforces decision boundaries between classes but still produces regions where different classes overlap in the feature space. In such regions, samples of one class can be close to the centroid of another class. This is problematic for prototype-based classification, which relies on distances or similarities in the feature space. Large intra-class dispersion and overlapping clusters lead to ambiguous prototype boundaries and reduced accuracy.

To mitigate this, we propose the Center-Adapt (CA) loss to fine-tune the PTM. Specifically, we adopt Center Loss (CL) (Wen et al., 2016) as an auxiliary objective that pulls samples of the same class towards their class prototype. Formally, the CL is defined in Eq. (4):

$$\mathcal{L}_C = \frac{1}{2}\sum_{i=1}^{M} ||\boldsymbol{x}_i^{(k)} - \boldsymbol{p_k}||_2^2, \tag{4}$$

where the $\boldsymbol{x}_i^{(k)}$ is the ith training sample of class k, $\boldsymbol{p_k}$ is the prototype of class k. This objective encourages representations of each class to contract towards their prototype, thereby reducing similarity to prototypes of other classes. Since the feature representation changes as the adapter parameters are updated, the centers $\boldsymbol{p_k}$ must be refreshed at each training step.

However, prototype-based methods depend on well-separated class centers. When the objective includes only attractive forces, such as CL, all samples of a class will be pulled toward a single point. Consequently, prototypes from different classes may drift in similar directions, resulting in prototype mode collapse. It could weaken the discriminative structure of the embedding space. To mitigate this effect, we combine CL with the repulsive influence of Cross-Entropy (CE) (Zhang & Sabuncu, 2018), whose classification objective naturally pushes class logits apart. The two components are complementary. CE preserves inter-class margins, while CL tightens intra-class clusters. The resulting CA loss is defined in Eq. (5).

$$\mathcal{L}_{CA} = \mathcal{L}_{CE} + \lambda\mathcal{L}_C, \tag{5}$$

where λ is a constant scalar weight. As shown in Fig. 2(c), fine-tuning with CA loss yields compact, approximately isotropic clusters around class prototypes and clearly enlarged inter-cluster margins. This improves prototype-based classification by reducing overlap between classes in the embedding space.

Following prior work (Zhou et al., 2024a,c), we use the adapter (Houlsby et al., 2019) as the fine-tuning module due to its strong accuracy-efficiency trade-off. Each adapter inserts a bottleneck structure in every Transformer block, as denoted in Eq. (6):

$$\boldsymbol{z}_i' = \boldsymbol{z}_i + ReLU(\boldsymbol{z}_i W_{dp})W_{up}, \tag{6}$$

where $\boldsymbol{z}_i$ and $\boldsymbol{z}_i'$ are feature vectors of input and output respectively. When task t arrives, we attach a temporary linear classification head, fine-tune only the adapter t using CA loss on D_t. The PTM and those adapters from other tasks are frozen. This preserves PTM knowledge while enabling task-specific specialization.

4.2. Dual prototype network

Prototype-based classification with cosine similarity often misclassifies ambiguous samples. However, our analysis reveals that top-K

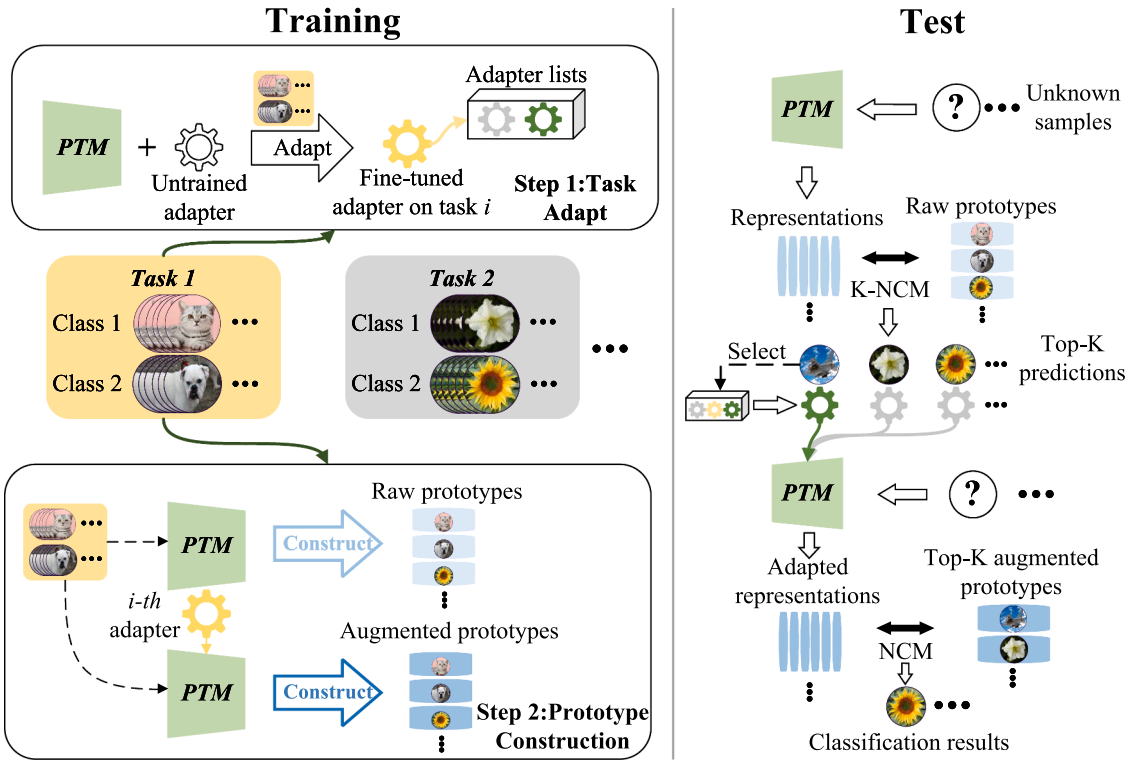


Fig. 1. Overview of the proposed DPTA. **Left:** Training. When a new task *i* arrives, a task-specific adapter is fine-tuned and saved. Then, using the raw and task-adapted PTM to construct raw and augmented prototypes. **Right:** Test. Raw prototypes produce top-K candidate labels, from which the relevant task adapters are identified. The augmented prototypes of the selected adapters then predict the final label.

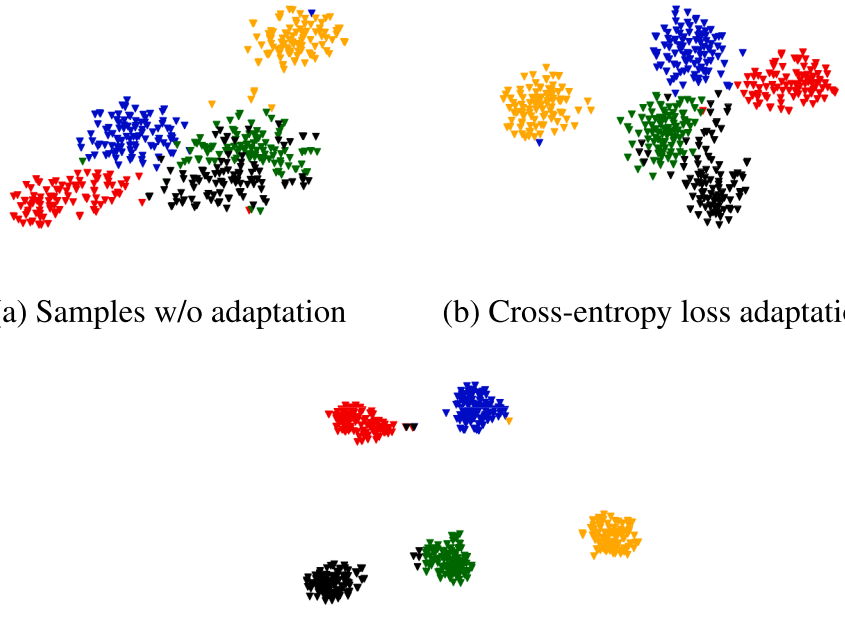


Fig. 2. t-SNE (Van der Maaten & Hinton, 2008) visualizations of original space and task-adapted subspace trained with CE loss and CA loss.

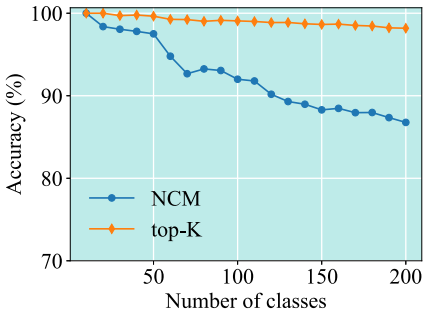


Fig. 3. The comparison of NCM top-1 and top-5 predictions accuracy, where the accuracy of the top-5 group ranged from 98 % to 100 %. The prototypes were built with a pre-trained VIT-B/16-IN21K without fine-tuning.

predictions using raw PTM features remain highly stable, even when top-1 accuracy drops. As shown in Fig. 3, top-5 accuracy stays near 100 %. This suggests that top-K raw predictions can reliably identify a small candidate label set, which carries useful task and class information.

We therefore introduce the Dual Prototype Network, which maintains two prototype sets per class. The first set, raw prototypes, is computed using Eq. (2) with the original PTM (or the first-task-adapted PTM) and is used to produce reliable top-K candidate labels for each test sample. These labels further reveal up to K possible task indices when the task label correspondence established during training is preserved. The second set arises from representations extracted by the CA-adapted PTM, which exhibit enhanced inter-class separability and are therefore better suited for prototype-based prediction. These features are referred to as augmented representations, and their corresponding class means form the augmented prototypes, computed as in Eq. (7).

$$\boldsymbol{p}_k^{aug} = \frac{1}{M_k}\sum_{j=1}^{M} \mathbb{I}(y_j = k)\phi^{A_{t_k}}(\boldsymbol{x}_j). \tag{7}$$

where the $\phi^{A_{t_k}}$ is the PTM loaded with kth adapter. The augmented prototype leverages the sample's augmented representations, assisted by complementary information provided by the raw prototype, to infer the sample's true class label.

After completing task adaptation on D_i, the classes' raw and augmented prototypes in task i are computed using Eqs. (2) and (7), respectively. Once all n tasks have been learned, the model retains n adapters, $\sum_{i=1}^{n} N_i$ raw prototypes and augmented prototypes, where N_i denotes the number of classes in task i.

4.3. Prediction process of DPTA

Augmented prototypes cannot be directly used because the task ID is unknown during inference. One naive solution is to load all adapters and compute similarities in every subspace, but this induces large computational overhead and noise from irrelevant subspaces.

Neural networks generally assume independently and identically distributed inputs, which makes them sensitive to out-of-distribution (OOD) data. For transformers, the attention matrix $softmax(\frac{QK^T}{\sqrt{d}})$ assigns low weights to features that deviate from the learned distribution. As a result, the representations produced by a PTM equipped with an incorrect adapter tend to fall far from the corresponding task's feature region. This property allows augmented prototypes to naturally distinguish representations generated under mismatched adapters.

Fig. 4 illustrates this effect. For a given task t_n, samples from other tasks do not participate in its adaptation, so their augmented representations are projected away from all class clusters of t_n (light blue 'x' markers). Since prototypes act as class centroids, these representations show consistently lower similarity to all augmented prototypes of task

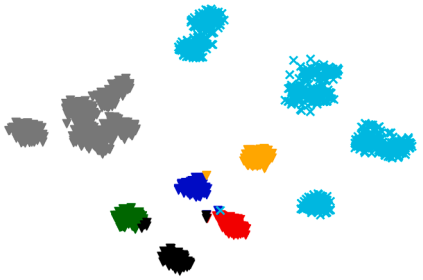


Fig. 4. t-SNE visualizations in a task-adapted subspace trained with CA loss.

t_n, so it has:

$$Sim(\phi^{A_{t_n}}(\boldsymbol{x}_j), \boldsymbol{p}_k^{aug})_{\cos} < Sim(\phi^{A_{t_n}}(\boldsymbol{x}_i), \boldsymbol{p}_k^{aug})_{\cos}. \tag{8}$$

Meanwhile, if a sample $\boldsymbol{x}_i$ from t_n is processed with a wrong adapter, its representation is projected into a different subspace (gray 'v' markers), yielding:

$$Sim(\phi^{A_i \neq A_{t_n}}(\boldsymbol{x}_i), \boldsymbol{p}_k^{aug})_{\cos} < Sim(\phi^{A_{t_n}}(\boldsymbol{x}_i), \boldsymbol{p}_k^{aug})_{\cos}. \tag{9}$$

Analogously, samples of t_n behave as OOD inputs for other tasks:

$$Sim(\phi^{A_{t(v)}}(\boldsymbol{x}_i), \boldsymbol{p}_v^{aug})_{\cos} < Sim(\phi^{A_{t_n}}(\boldsymbol{x}_i), \boldsymbol{p}_k^{aug})_{\cos}, \tag{10}$$

where v represents classes from other tasks, $t(v)$ denotes the task ID of the class v. The relationships in Eqs. (8)–(10) indicate that augmented representations generated with an incorrect adapter become clear outliers. They lie far from every augmented prototype and exhibit consistently lower similarity, which could be naturally suppressed. In other words, the augmented prototypes could implicitly filter out mismatched tasks and perform automatic cross-task inference.

However, directly loading all adapters to compute augmented representations in inference is computationally prohibitive. Therefore, DPTA uses raw prototypes to obtain a small candidate set of labels and tasks, which only requires K times adapter loading. More importantly, the raw prototypes offer an additional source of robustness. Task-specific fine-tuning may introduce overfitting, which can cause the augmented prototypes to yield incorrect predictions. In contrast, the raw prototypes rely on representations from the original PTM, which retain stronger generalization. The resulting high-accuracy top-K predictions effectively eliminate most incorrect classes and tasks before augmented prototypes.

The prediction process of DPTA can be divided into two steps. First, raw prototypes are utilized to predict the top-K class labels, as represented in Eq. (11):

$$Y_i^{topK} = \underset{k=1,2,\ldots N}{K\text{-}\arg\max}(Sim(\phi(\boldsymbol{x}_i), \boldsymbol{p}_k^{raw})_{\cos}), \tag{11}$$

then the corresponding task IDs are obtained based on the top-K labels. If the preset K exceeds the number of raw prototypes n, it can be temporarily set to n until sufficient raw prototypes are attained. The final label is determined by augmented prototypes, as denoted in Eq. (12):

$$\hat{y}_i = \underset{k \in Y_i^{topK}}{\arg\max}(Sim(\phi^{A_{t(k)}}(\boldsymbol{x}_i), \boldsymbol{p}_k^{aug})_{\cos}). \tag{12}$$

In summary, the raw prototypes contribute robust class & task localization, while the augmented prototypes complete the final classification. We summarize the training and testing pipeline of DPTA in Algorithms 1 and 2.

The expected performance of DPN follows from conditional probability, as denoted in Eq. (13):

$$\mathop{\mathbb{E}}_{\boldsymbol{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}}[\mathbb{I}(y_i = f(\boldsymbol{x}_i))] = \mathbb{E}[\mathbb{I}(y_i = \hat{y}_i)|\mathbb{I}(y_i \in Y_i^{topK})] * \mathbb{E}[\mathbb{I}(y_i \in Y_i^{topK})]. \tag{13}$$

It shows that DPN improves when both prototype sets become more accurate. Since raw top-K accuracy is inherently high, the augmented prototypes govern the achievable performance ceiling.

Algorithm 1 Training process of DPTA.

Input: Incremental datasets: $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_T\}$. Pre-trained embedding: $\phi(\mathbf{x})$.

1: **for** $t = 1, 2, \ldots, T$ **do**
2: Get the incremental training set $\mathcal{D}_t$ of $t-$th task
3: Initialize a new fine-tune module $\mathcal{A}_t$
4: Train $\mathcal{A}_t$ in $\mathcal{D}_t$ via Eq. (5)
5: Construct the raw and augmented prototypes in task t via Eq. (2) and Eq. (7)
6: **end for**

Output: Trained modules $\{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_T\}$. Raw and augmented prototypes $\boldsymbol{p}^{raw}$, $\boldsymbol{p}^{aug}$.

Algorithm 2 Inference process of DPTA.

Input: Test dataset: $\mathcal{D}_{test}$ with N samples. Class-task query $q(\cdot)$. Pre-trained embedding: $\phi(\mathbf{x})$. Trained modules: $\{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_T\}$. Raw and augmented prototypes $\boldsymbol{p}^{raw}$, $\boldsymbol{p}^{aug}$.

1: **for** $i = 1, 2, \ldots, N$ **do**
2: Get the sample i in $\mathcal{D}_{test}$
3: Using $\boldsymbol{p}^{raw}$ to predict the top-K labels Y_i^{topK} via Eq. (11)
4: Using $q(\cdot)$ to predict the task indexes corresponding to Y_i^{topK}
5: Using $\boldsymbol{p}^{aug}$, task indexes and $\{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_T\}$ to predict $\hat{y}_i$ via Eq. (12)
6: **end for**

Output: Predicted labels $\hat{\boldsymbol{y}}_{test}$ of $\mathcal{D}_{test}$

4.4. Complexity analysis

This subsection will analyze the inference and storage complexity of dual prototypes. Let T denote the number of tasks, N_t the number of classes introduced at task t, and $C = \sum_{t=1}^{T} N_t$ the total number of classes. We denote the feature space dimensionality of the prototypes as d, and by F the computational cost of a single forward pass through the PTM equipped with one task-specific adapter. Methods like EASE maintain a complete prototype set in each task-specific subspace. As new tasks arrive, the prototype set in the subspace in task t must cover all classes observed, i.e., $\sum_{i=1}^{t} N_i$ classes. Therefore, the total number of stored prototypes in EASE is $\sum_{t=1}^{T}\left(\sum_{i=1}^{t} N_i\right)$. Under the common assumption that each tasks introduce an equal number of classes ($N_t = C/T$), it scales as $\mathcal{O}(TC)$. Consequently, the overall storage complexity is:

$$\mathcal{O}(TCd). \tag{14}$$

In contrast, each class in DPTA is associated with a raw and augmented prototype, so the number of stored prototypes is $2C$. The storage complexity is :

$$\mathcal{O}(Cd), \tag{15}$$

where the cost is independent of the task number T.

In inference, EASE will load all adapters to perform T forward passes through the PTM, yielding a computational cost of $\mathcal{O}(TF)$. In each subspace, the sample representation is compared against the corresponding prototype set to compute similarities and obtain an ensemble prediction. If each subspace maintains prototypes for all C classes, the total number of similarity computations per sample is on the order of TC, leading to an additional cost of $\mathcal{O}(TCd)$. Overall, the per-sample inference complexity of EASE is:

$$\mathcal{O}(TF + TCd), \tag{16}$$

which scales linearly with the number of tasks, both in terms of forward passes and similarity evaluations.

In DPTA, inference is decomposed into two stages. In the first stage, a single forward pass through the shared PTM (without loading any task-specific adapter beyond the base or first-task configuration) produces an embedding in the raw prototype space. This embedding is compared to all raw prototypes to obtain top-K candidate class labels. The cost of this stage is$\mathcal{O}(F + Cd)$, where $\mathcal{O}(F)$ accounts for the forward pass and $\mathcal{O}(Cd)$ for computing cosine similarities to the C raw prototypes. In the second stage, the top-K labels are mapped to their corresponding task IDs. Only the adapters associated with these candidate tasks are then loaded, leading to at most K additional forward passes through the PTM, with cost $\mathcal{O}(KF)$. The augmented representations are compared against the augmented prototypes of these top-K candidate classes, requiring $\mathcal{O}(Kd)$ similarity computations. Overall, the inference complexity of DPTA per sample is:

$$\mathcal{O}\big(F + Cd + KF + Kd\big) = \mathcal{O}\big(KF + (K + C)d\big), \tag{17}$$

where K is a constant and typically set to 5. It generally satisfies $K \ll T$, so the complexity of Eq. (17) can be further simplified to $\mathcal{O}\big(F + Cd\big)$.

Overall, DPTA achieves lower prototype storage and inference complexity. Both scale linearly with class numbers rather than tasks, making DPTA substantially more efficient in continual learning scenarios.

5. Experiments

This section empirically evaluates the proposed DPTA framework by addressing three key questions: [**RQ1**] Does DPTA outperform state-of-the-art (SOTA) methods on benchmark datasets? [**RQ2**] How much do the dual-prototype mechanism and the center-adapt loss contribute to the final performance? and [**RQ3**] How do the key hyperparameters influence DPTA's behavior?

5.1. Implementation details

Dataset and split. Most PTM-based CIL methods use ViT-B/16-IN21K (Dosovitskiy, 2020). Because it is pre-trained on ImageNet-21K, we select benchmark datasets with notable domain gaps, including CIFAR-100 (Krizhevsky et al., 2009), Stanford Cars (Kramberger & Potočnik, 2020), ImageNet-A (Hendrycks et al., 2021b), ImageNet-R (Hendrycks et al., 2021a), and VTAB (Zhai et al., 2019). Following the "B/Base-m, Inc-n" rule protocol (Zhou et al., 2024a), datasets are split into CIFAR B0 Inc10, CARS B16 Inc10, ImageNet-A B0 Inc20, ImageNet-R B0 Inc20, and VTAB B0 Inc10, consistent with (Zhou et al., 2024c). Here, m is the number of classes in the first task and n is that of each subsequent task;. $m = 0$ indicates uniform division. All competing methods use identical training/testing splits.

Baselines. We compare DPTA with a wide range of baseline and SOTA approaches, including SDC (Yu et al., 2020), L2P (Wang et al., 2022d), Dual-Prompt (Wang et al., 2022c), CODA-Prompt (Smith et al., 2023), SimpleCIL and APER (Zhou et al., 2024a), EASE (Zhou et al., 2024c), MOS (Sun et al., 2025), and MoAL (Gao et al., 2025b). For reference, we also report sequential finetuning (Finetune). All PTM-based baselines adopt ViT-B/16-IN21K to ensure fairness.

Programming and hyperparameters. All experiments are implemented in PyTorch 2.4.1 and conducted on NVIDIA A4000 GPUs. DPTA is implemented using the Pilot toolbox (Sun et al., 2023). Baseline settings follow the recommended configurations in their original papers or Pilot. Learning rates use cosine annealing. For DPTA, the K value is fixed at five. The combined loss weight λ is set to 0.001 on the VTAB dataset and 0.0001 on the other datasets. All methods are trained under the same random seed.

Evaluation metrics. Following Rebuffi et al. (2017), we use A_b to denote the b-stage accuracy after learning tasks D_b, $\overline{A} = \frac{1}{T}\sum_{b=1}^{T} A_b$ is average stage accuracy over T tasks, and A_F is the final accuracy on the overall test set.

Table 2
The comparison of average accuracy $\overline{A}$ and final accuracy A_F results on benchmark datasets, where the best results achieved on all benchmarks are in bold. All methods used here are exemplar-free methods in which no replay samples are required.

Methods	CIFAR		CARS		ImageNet-A		ImageNet-R		VTAB	
	$\overline{A}$	A_F	$\overline{A}$	A_F	$\overline{A}$	A_F	$\overline{A}$	A_F	$\overline{A}$	A_F
Finetune	63.51	52.10	42.12	40.64	46.42	42.20	48.56	47.28	50.72	49.65
SDC	68.45	64.02	42.12	40.64	29.23	27.72	53.18	50.05	48.03	26.21
L2P	85.95	79.96	47.95	43.21	47.12	38.49	69.51	75.46	69.77	77.05
DualPrompt	87.89	81.17	52.72	47.62	53.75	41.64	73.10	67.18	83.23	81.20
CODA-Prompt	89.15	81.94	55.95	48.27	53.56	42.92	77.97	72.27	83.95	83.01
SimpleCIL	87.57	81.26	65.54	54.78	59.77	48.91	62.55	54.52	86.01	84.43
APER(Adapter)	90.55	85.10	66.76	56.25	60.49	49.75	75.82	67.95	86.04	84.46
APER(VPT-S)	90.52	85.21	66.73	56.22	59.43	47.62	68.83	62.03	87.25	85.37
EASE	92.45	87.05	78.45	67.01	65.35	55.04	81.73	76.17	93.62	93.54
MOS	93.29	89.25	79.58	68.56	67.07	56.22	82.75	77.83	92.62	92.79
MoAL	**94.10**	**90.09**	80.65	70.23	69.45	58.44	83.88	78.00	92.02	91.10
DPTA(ours)	92.90	88.60	**81.64**	**71.02**	**69.56**	**58.67**	**84.90**	**78.20**	**94.52**	**94.09**

Table 3
The comparison of accuracy with SOTA replay-based methods. All methods use ViT-B/16-IN21K.

Methods	CIFAR		ImageNet-A		ImageNet-R	
	$\overline{A}$	A_F	$\overline{A}$	A_F	$\overline{A}$	A_F
iCaRL	82.36	73.67	29.13	16.15	72.35	60.54
DER	86.11	77.52	33.72	22.13	80.36	74.26
FOSTER	89.76	84.54	34.55	23.34	81.24	74.43
MEMO	84.33	75.56	36.54	24.43	74.12	66.45
DPTA(ours)	**92.90**	**88.60**	**69.56**	**58.67**	**84.90**	**78.20**

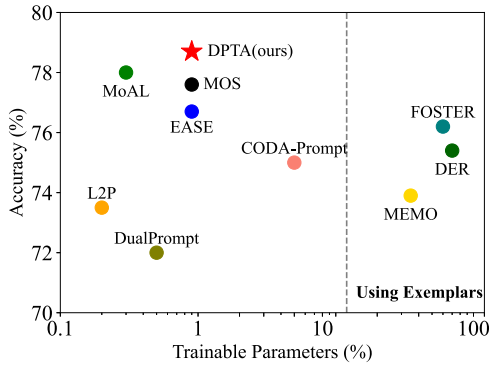


Fig. 5. The comparison of accuracy and trainable parameter sizes on the ImageNet-R dataset.

5.2. Benchmark comparison

We first evaluate DPTA against baseline and SOTA methods on the selected datasets, with results summarized in Table 2. DPTA achieves the highest classification accuracy on most datasets and consistently surpasses previous SOTA approaches.

To further assess its competitiveness, we compare DPTA with several milestone exemplar-based methods, including iCaRL (Rebuffi et al., 2017), DER (Yan et al., 2021), FOSTER (Wang et al., 2022a), and MEMO (Zhou et al., 2022), as shown in Table 3. All of them using ViT-B/16-IN21K with a fixed 2000 exemplar size. DPTA remains highly competitive without exemplars and achieves the best accuracy among all compared methods.

We also analyze the parameter-accuracy trade-off in Fig. 5. DPTA attains the best accuracy with a parameter scale similar to other exemplar-free methods. To quantify efficiency, we report average inference time $\bar{t}_{\text{inf}}$ of representative methods for 10,000 samples in Fig. 6. DPTA demonstrates moderate latency, while EASE incurs the highest cost due to loading all adapters and integrating prototypes across tasks.

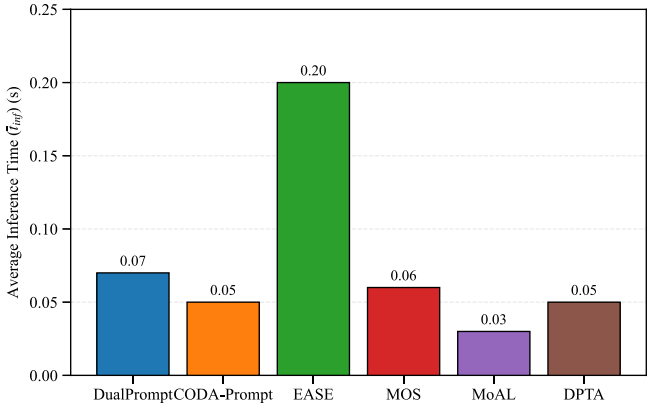


Fig. 6. The comparison of average inference time.

5.3. Ablation study

An ablation study is proposed to investigate the effectiveness of the DPTA's components. The accuracy A_b of the control group is reported in Fig. 7. Adapter-CA and Adapter-EA remove the dual-prototype network (DPN), making it impossible to identify tasks; thus, only the first-task adapter can be used, corresponding to first-task adaptation. Adapter-CA uses the center-adapt loss (Eq. (5)), while Adapter-EA uses cross-entropy loss. DPTA significantly outperforms Adapter-CA, confirming the effectiveness of DPN. Comparing Adapter-CA and Adapter-EA shows that the center-adapt loss improves prototype classification.

To assess whether the CA loss causes collapse of augmented prototypes, we measure the average inter-class cosine similarity S_{inter}, the maximum pairwise similarity $S_{\max}$, and the concentration ratio R, denoted as follows:

$$S_{\text{inter}} = \frac{1}{K(K-1)} \sum_{i \neq j} sim(\mathbf{p}_i^{aug}, \mathbf{p}_j^{aug})_{\cos}, \tag{18}$$

$$S_{\max} = \max_{i \neq j} sim(\mathbf{p}_i^{aug}, \mathbf{p}_j^{aug})_{\cos}, \tag{19}$$

$$R = \left\| \frac{1}{K} \sum_{i=1}^{K} \mathbf{p}_i^{aug} \right\|_2. \tag{20}$$

where high values of S_{inter} or $S_{\max}$ mean that prototypes belonging to different classes are similar to each other, which indicates prototype collapse, while a higher R shows that the prototypes become aligned in a similar direction, revealing a global prototype mode collapse.

We evaluate three configurations: (i) CE-only, (ii) Center-only, and (iii) Ours (CA loss). As shown in Table 4, Center-only training produces high S_{inter}, $S_{\max}$ and R, confirming severe prototype collapse. In contrast, our method produces much lower values, which are close to or

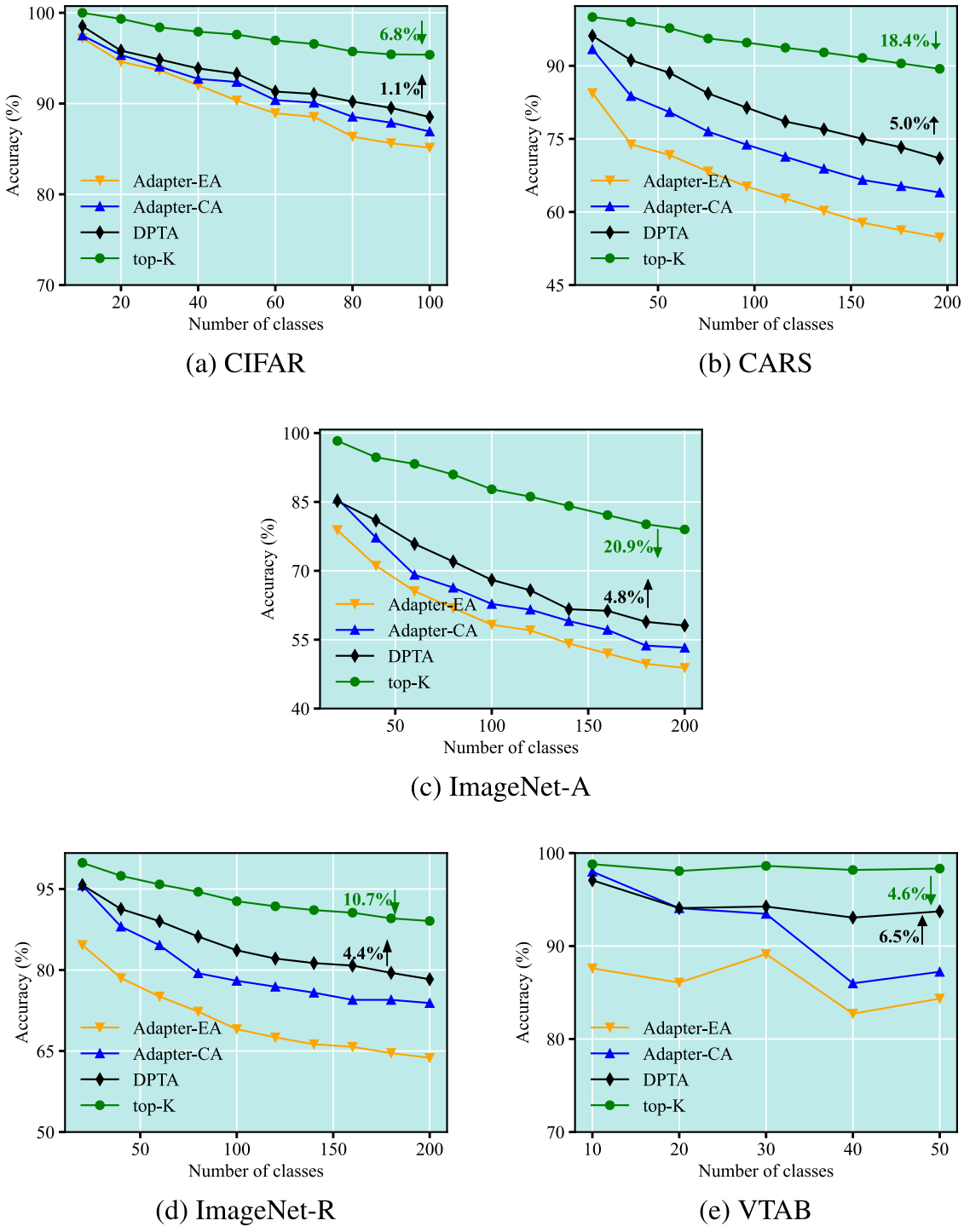


Fig. 7. Ablation study results on benchmark datasets. The green annotation indicates the decrease in the final accuracy A_F of DPTA over top-K prediction. The black annotation denotes the improvement of DPTA over Adapter-CA.

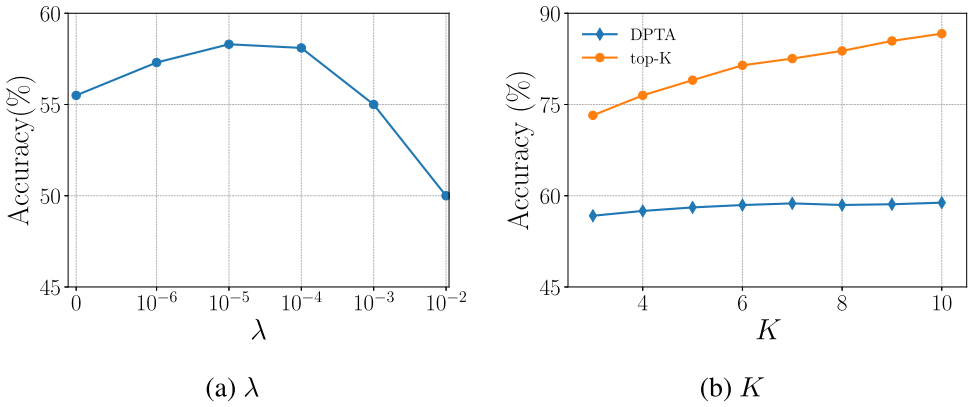
Table 4
Prototype dispersion metrics comparison on ImageNet-A and VTAB.

Loss	ImageNet-A			VTAB		
	S_{inter}	S_{max}	R	S_{inter}	S_{max}	R
CE only	0.34	0.90	0.59	0.14	0.83	0.42
Center only	0.33	0.99	0.58	0.32	0.99	0.57
Ours	0.12	0.82	0.35	0.18	0.77	0.45

smaller than CE-only. These findings show that our CA loss produces prototype structures that are as stable as CE. In some cases, such as on ImageNet-A, the prototypes are even more robust, since all three metrics are significantly lower than those of CE-only.

5.4. Hyperparameter setting study

We study the effect of the two key hyperparameters, λ and K of DPTA, as shown in Fig. 8. For λ, performance degrades when its



(a) λ (b) K

Fig. 8. The relationship of hyperparameters λ and Kvalue with the A_F on IN-A dataset.

Table 5
The comparison of DPTA accuracy with different fine-tune modules.

Methods	CIFAR		ImageNet-R		VTAB	
	$\overline{A}$	A_F	$\overline{A}$	A_F	$\overline{A}$	A_F
DPTA(VPT)	91.88	87.55	82.06	77.95	94.81	94.20
DPTA(SSF)	91.50	86.34	81.27	77.57	93.68	92.86
DPTA	92.90	88.60	84.90	78.20	94.52	94.09

Table 6
The comparison of DPTA accuracy with different PTM backbones.

PTM backbones	ImageNet-A		VTAB	
	$\overline{A}$	A_F	$\overline{A}$	A_F
ViT-B/16-IN21K	69.56	58.67	94.52	94.09
ViT-B/16-IN1K	68.36	58.20	93.72	93.50
ViT-B/16-IN21K-ft1K	72.82	64.40	94.30	93.62
VIT-MoCo-v3	72.02	63.01	94.01	93.51

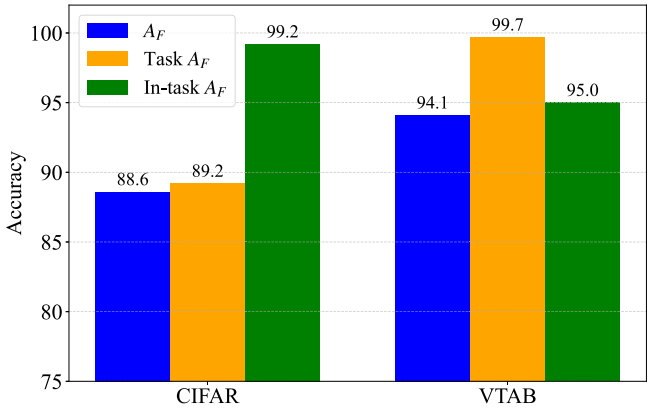


Fig. 9. Three types of accuracy results on CIFAR-100 and VTAB dataset.

value departs from an appropriate range. Excessively high λ overemphasizes the center loss, hindering the model's ability to learn class-discriminative features. Conversely, an insufficiently low λ fails to activate the center-adapt loss effectively, causing the model to degenerate into the case trained solely with cross-entropy loss. For K, increasing the top-K range substantially improves top-K prediction accuracy. Expanding k from 3 to 7 yields roughly a 3 % gain in A_F. Larger K values offer no further improvement because the resulting top-K set already covers the most ambiguous classes.

5.5. Generalization ability and plug-and-play flexibility of DPTA

The center-adapt loss and dual-prototype mechanism in DPTA are not tied to adapters. They can be combined with other parameter-efficient tuning modules. Table 5 summarizes the results of integrating DPTA with VPT and SSF. DPTA preserves strong performance across all datasets. This confirms that the dual-prototype mechanism and center-adapt loss function operate effectively in different modular configurations.

Moreover, since the task adaptation-based CIL methods rely on the knowledge embedded in the PTM, we further evaluate DPTA using different ViT backbones trained under diverse procedures, including supervised ImageNet-21K, ImageNet-1K, ImageNet-21K with extra ImageNet-1K fine-tuning, and ViT with MoCo-v3. As shown in Table 6, DPTA delivers excellent performance across all backbones. These results indicate that DPTA is robust to changes in backbone training strategy and does not rely on a specific type of PTM backbone.

In summary, DPTA demonstrates broad generalization in both dimensions: flexibility across fine-tuning modules and stability across heterogeneous pre-trained backbones, supporting its plug-and-play applicability in PTM adaptation.

5.6. Further result interpretation

We now provide deeper insight into the empirical behaviors of DPTA. In Section 5.2, compared with L2P, Dual-Prompt, CODA-Prompt, and EASE, DPTA's task-wise adaptation yields more reliable task identification than prompt matching, weighted prompt combinations, or prototype ensembling. While MOS and MoAL are strong competitors, DPTA achieves better accuracy on most datasets, especially VTAB, whose tasks contain severely (even fewer than 10 samples in fifth task) imbalanced and small-sample training sets. These results demonstrate DPTA's robustness in constrained or imbalanced training data regimes. However, DPTA underperforms MOS and MoAL on CIFAR-100. To investigate this, we examine three metrics on both CIFAR-100 and VTAB: A_F, final task prediction accuracy (Task A_F), and final in-task prediction accuracy, defined as accuracy conditioned on correct task identification (In-Task A_F). The results are illustrated in Fig. 9. In CIFAR-100, In-Task A_F is nearly 100 %, while Task A_F closely matches A_F, indicating that the performance bottleneck lies in cross-task reasoning rather than in-task classification.

This limitation stems from the characteristics of CIFAR-100: images are originally 32×32 and must be upsampled to 224×224 for ViT, introducing interpolation artifacts and blurred details. These artifacts reduce inter-task feature disparities, making tasks more visually homogeneous. Since the dual-prototype mechanism relies on natural task-level distribution differences to support cross-task prediction, reduced inter-task separability makes the mechanism less effective, thus lowering accuracy. In contrast, datasets such as VTAB contain high-resolution images with larger inter-task variation, enabling DPTA to achieve high task prediction accuracy and A_F. Despite this limitation, DPTA still surpasses most baselines on CIFAR-100, demonstrating competitive performance even in low-resolution scenarios.

In summary, while DPTA provides strong accuracy, it has two limitations. First, although it uses fewer adapters and prototypes than EASE, loading multiple adapters during inference introduces higher latency than single-adapter methods such as MoAL. Second, the dual-prototype mechanism depends on inter-task feature disparities, but these disparities are weak on low-resolution or highly homogeneous datasets. This suggests that future work may benefit from additional training objectives that explicitly enlarge feature-space separation between tasks.

6. Conclusion

In real-world applications, we expect machine learning models to learn from streaming data without forgetting. This work introduces DPTA, a dual-prototype framework with task-wise adaptation for PTM-based CIL. Task-specific adapters are trained with a center-adapt loss to produce more discriminative representations. During inference, raw prototypes identify suitable adapters for each test sample, and task-wise augmented prototypes further refine prediction. Extensive experiments verify the effectiveness of DPTA. Nevertheless, DPTA depends on intrinsic inter-task distribution differences; its performance lags behind state-of-the-art methods on low-resolution datasets such as CIFAR-100. Future work will explore improved training objectives that enhance intra-class compactness and inter-class separability.

CRediT authorship contribution statement

Zhiming Xu: Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Formal analysis, Data curation; **Suorong Yang:** Writing – review & editing, Writing – original draft, Resources, Formal analysis; **Baile Xu:** Writing – review & editing, Project administration, Methodology, Investigation, Conceptualization; **Furao Shen:** Writing – review & editing, Supervision, Funding acquisition; **Jian Zhao:** Writing – review & editing, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the STI 2030-Major Projects of China under Grant 2021ZD0201300 and by the National Natural Science Foundation of China under Grant 62276127.

References

Ahn, H., Cha, S., Lee, D., & Moon, T. (2019). Uncertainty-based continual learning with adaptive regularization. *Advances in Neural Information Processing Systems*, *32* 4394–4404.

Bahi, A., Gasmi, I., Bentrad, S., & Khantouchi, R. (2024). MycGNN: Enhancing recommendation diversity in e-commerce through mycelium-inspired graph neural network. *Electronic Commerce Research*, (pp. 1–31).

Barbu, A., Mayo, D., Alverio, J., Luo, W., Wang, C., Gutfreund, D., Tenenbaum, J., & Katz, B. (2019). Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in Neural Information Processing Systems*, *32* 9453–9463.

Bezdek, J. C., & Kuncheva, L. I. (2001). Nearest prototype classifier designs: An experimental study. *International Journal of Intelligent Systems*, *16*(12), 1445–1473.

Bian, A., Li, W., Yuan, H., Wang, M., Zhao, Z., Lu, A., Ji, P., Feng, T. et al. (2024). Make continual learning stronger via c-flat. *Advances in Neural Information Processing Systems*, *37*, 7608–7630.

Chen, S., Gong, C., Li, J., Yang, J., Niu, G., & Sugiyama, M. (2022). Learning contrastive embedding in low-dimensional space. *Advances in Neural Information Processing Systems*, *35*, 6345–6357.

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., & Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3366–3385.

De Lange, M., & Tuytelaars, T. (2021). Continual prototype evolution: Learning online from non-stationary data streams. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 8250–8259).

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255). IEEE.

Dosovitskiy, A. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929

French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, *3*(4), 128–135.

Fukuda, T., Kera, H., & Kawamoto, K. (2025). Adapter merging with centroid prototype mapping for scalable class-incremental learning. In *Proceedings of the computer vision and pattern recognition conference* (pp. 4884–4893).

Gao, Z., Han, S., Zhang, X., Xu, K., Zhou, D., Mao, X., Dou, Y., & Wang, H. (2025a). Maintaining fairness in logit-based knowledge distillation for class-incremental learning. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 16763–16771). (*vol. 39*).

Gao, Z., Jia, W., Zhang, X., Zhou, D., Xu, K., Dawei, F., Dou, Y., Mao, X., & Wang, H. (2025b). Knowledge memorization and rumination for pre-trained model-based class-incremental learning. In *Proceedings of the computer vision and pattern recognition conference* (pp. 20523–20533).

Gao, Z., Liu, B., Xu, K., Mao, X., & Wang, H. (2025c). Correlation-based knowledge distillation in exemplar-free class-incremental learning. *IEEE Open Journal of the Computer Society*, *39*(16), 16763–16771.

Gomes, H. M., Barddal, J. P., Enembreck, F., & Bifet, A. (2017). A survey on ensemble learning for data stream classification. *ACM Computing Surveys (CSUR)*, *50*(2), 1–36.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M. et al. (2021a). The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 8340–8349).

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., & Song, D. (2021b). Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 15262–15271).

Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., De Laroussilhe, Q., Gesmundo, A., Attariyan, M., & Gelly, S. (2019). Parameter-efficient transfer learning for NLP. In *International conference on machine learning* (pp. 2790–2799). PMLR.

Jia, M., Tang, L., Chen, B.-C., Cardie, C., Belongie, S., Hariharan, B., & Lim, S.-N. (2022). Visual prompt tuning. In *European conference on computer vision* (pp. 709–727). Springer.

Kramberger, T., & Potočnik, B. (2020). Lsun-stanford car dataset: Enhancing large-scale car image datasets using deep learning for usage in gan training. *Applied Sciences*, *10*(14), 4913.

Krizhevsky, A., Hinton, G. et al. (2009). Learning multiple layers of features from tiny images. Technical report.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, *25* 1106–1114.

Li, W., Yuan, H., Zhao, Z., Zhu, Y., Lu, A., Feng, T., & Sun, Y. (2025). C-flat++: Towards a more efficient and powerful framework for continual learning. arXiv preprint arXiv:2508.18860

Lian, D., Zhou, D., Feng, J., & Wang, X. (2022). Scaling & shifting your features: A new baseline for efficient model tuning. *Advances in Neural Information Processing Systems*, *35*, 109–123.

Liu, C., Zhao, L., Lyu, F., Du, K., Hu, F., & Zhou, T. (2024). Cala: A class-aware logit adapter for few-shot class-incremental learning. arXiv preprint arXiv:2412.12654

Liu, Z., Luo, P., Wang, X., & Tang, X. (2015). Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision* (pp. 3730–3738).

Van der Maaten, L., & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(11).

Nguyen, C. V., Li, Y., Bui, T. D., & Turner, R. E. (2017). Variational continual learning. arXiv preprint arXiv:1710.10628

Qiu, Z., Xu, Y., He, C., Meng, F., Xu, L., Wu, Q., & Li, H. (2025). Mingle: Mixtures of null-space gated low-rank experts for test-time continual model merging. arXiv preprint arXiv:2505.11883

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J. et al. (2021). Learning transferable visual models from natural language supervision. In *International conference on machine learning* (pp. 8748–8763). PMLR.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., & Lampert, C. H. (2017). Icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2001–2010).

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., & Wayne, G. (2019). Experience replay for continual learning. *Advances in Neural Information Processing Systems*, *32* 2579–2605.

Roy, D., Panda, P., & Roy, K. (2020). Tree-CNN: A hierarchical deep convolutional neural network for incremental learning. *Neural Networks*, *121*, 148–160.

Simon, C., Koniusz, P., & Harandi, M. (2021). On learning the geodesic path for incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 1591–1600).

Smith, J. S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., & Kira, Z. (2023). Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11909–11919).

Sun, H.-L., Zhou, D.-W., Ye, H.-J., & Zhan, D.-C. (2023). Pilot: A pre-trained model-based continual learning toolbox. arXiv preprint arXiv:2309.07117

Sun, H.-L., Zhou, D.-W., Zhao, H., Gan, L., Zhan, D.-C., & Ye, H.-J. (2025). Mos: Model surgery for pre-trained model-based class-incremental learning. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 20699–20707). (*vol. 39*).

Tan, Y., Zhou, Q., Xiang, X., Wang, K., Wu, Y., & Li, Y. (2024). Semantically-shifted incremental adapter-tuning is a continual vitransformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 23252–23262).

Wang, C., Jiang, J., Hu, X., Liu, X., & Ji, X. (2025). Enhancing consistency and mitigating bias: A data replay approach for incremental learning. *Neural Networks*, *184*, 107053.

Wang, F.-Y., Zhou, D.-W., Ye, H.-J., & Zhan, D.-C. (2022a). Foster: Feature boosting and compression for class-incremental learning. In *European conference on computer vision* (pp. 398–414). Springer.

Wang, L., Zhang, X., Su, H., & Zhu, J. (2024). A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *46*(8), 5362–5383.

Wang, Y., Huang, Z., & Hong, X. (2022b). S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *Advances in Neural Information Processing Systems*, *35*, 5682–5695.

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.-Y., Ren, X., Su, G., Perot, V., Dy, J. et al. (2022c). Dualprompt: Complementary prompting for rehearsal-free continual learning. In *European conference on computer vision* (pp. 631–648). Springer.

Wang, Z., Zhang, Z., Lee, C.-Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., & Pfister, T. (2022d). Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 139–149).

Wen, Y., Zhang, K., Li, Z., & Qiao, Y. (2016). A discriminative feature learning approach for deep face recognition. In *Computer vision–ECCV 2016: 14th european conference, amsterdam, the netherlands, october 11–14, 2016, proceedings, part VII 14* (pp. 499–515). Springer.

Xu, W., Xian, Y., Wang, J., Schiele, B., & Akata, Z. (2020). Attribute prototype network for zero-shot learning. *Advances in Neural Information Processing Systems*, *33*, 21969–21980.

Yan, S., Xie, J., & He, X. (2021). Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3014–3023).

Yang, H.-M., Zhang, X.-Y., Yin, F., & Liu, C.-L. (2018). Robust classification with convolutional prototype learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3474–3482).

Ye, H.-J., Zhan, D.-C., Li, N., & Jiang, Y. (2019). Learning multiple local metrics: Global consideration helps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(7), 1698–1712.

Yu, L., Twardowski, B., Liu, X., Herranz, L., Wang, K., Cheng, Y., Jui, S., & van de, W. J. (2020). Semantic drift compensation for class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 6982–6991).

Zeno, C., Golan, I., Hoffer, E., & Soudry, D. (2018). Task agnostic continual learning using online variational bayes. arXiv preprint arXiv:1803.10123

Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A. S., Neumann, M., Dosovitskiy, A. et al. (2019). A large-scale study of representation learning with the visual task adaptation benchmark. arXiv preprint arXiv:1910.04867

Zhang, G., Wang, L., Kang, G., Chen, L., & Wei, Y. (2023). Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 19148–19158).

Zhang, Z., & Sabuncu, M. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in Neural Information Processing Systems*, *31* 8792–8802.

Zhao, B., Xiao, X., Gan, G., Zhang, B., & Xia, S.-T. (2020). Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13208–13217).

Zhou, D.-W., Cai, Z.-W., Ye, H.-J., Zhan, D.-C., & Liu, Z. (2024a). Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, (pp. 1–21).

Zhou, D.-W., Sun, H.-L., Ning, J., Ye, H.-J., & Zhan, D.-C. (2024b). Continual learning with pre-trained models: A survey. arXiv preprint arXiv:2401.16386

Zhou, D.-W., Sun, H.-L., Ye, H.-J., & Zhan, D.-C. (2024c). Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 23554–23564).

Zhou, D.-W., Wang, Q.-W., Qi, Z.-H., Ye, H.-J., Zhan, D.-C., & Liu, Z. (2024d). Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *46*(12), 9851–9873.

Zhou, D.-W., Wang, Q.-W., Ye, H.-J., & Zhan, D.-C. (2022). A model or 603 exemplars: Towards memory-efficient class-incremental learning. arXiv preprint arXiv:2205.13218

Zhu, F., Zhang, X.-Y., Cheng, Z., & Liu, C.-L. (2025a). Pass++: A dual bias reduction framework for non-exemplar class-incremental learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *47*(8), 7123–7139.

Zhu, F., Zhang, X.-Y., Wang, C., Yin, F., & Liu, C.-L. (2021). Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5871–5880).

Zhu, G., Wu, D., Gao, C., Wang, R., Yang, W., & Sang, N. (2025b). Adaptive prototype replay for class incremental semantic segmentation. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 10932–10940). (*vol. 39*).