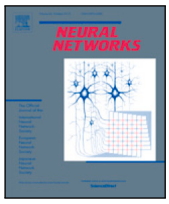
Neural Networks 190 (2025) 107651

Contents lists available at ScienceDirect

Neural Networks

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



Full Length Article

Supervised contrastive learning with prototype distillation for data incremental learning

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

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

ARTICLE INFO

Keywords:
Data incremental learning
Contrastive learning
Prototype distillation

ABSTRACT

The goal of Data Incremental Learning (DIL) is to enable learning from small-scale data batches from non-stationary data streams without clear task divisions. A challenge in this domain is the occurrence of catastrophic forgetting in deep neural networks. To effectively address the challenges inherent to DIL, the trained models must exhibit stability and flexibility, ensuring the retention of information from previously learned classes while adapting to incorporate new ones. Prototypes are particularly effective for classifying separable embeddings within the feature space, as they consolidate embeddings from the same class and push those from different classes further apart. This aligns with the principles of contrastive learning. In this paper, we propose Supervised Contrastive learning with the Prototype Distillation (SCPD) method for the DIL problem. First, we employ supervised contrastive loss (SCL) for model training to enhance the class separability of the extracted model representations and improve the flexibility of the model. To further mitigate the forgetting problem, we propose a prototype distillation loss (PDL), ensuring that feature representations remain close to their corresponding prototypes, enhancing the model's stability. The integration of SCL and PDL within SCPD ensures both the stability and flexibility of the model. Experimental results demonstrate that the SCPD method outperforms prior state-of-the-art approaches across several benchmarks, including those with various imbalanced setups.

1. Introduction

Recently, deep neural networks have shown outstanding performance in tackling complex tasks, primarily due to their remarkable ability to extract meaningful information from large-scale datasets. However, once training is completed, the parameters of feature encoders in these networks are typically frozen. Consequently, such models struggle to adapt to non-identically and independently distributed (non-iid) data without losing most previously learned knowledge in dynamic continual learning (CL) scenarios. As datasets continue to expand in dynamic CL settings, the distribution of input data shifts accordingly (Pan, Zhang, Yu, Zhang, & Gao, 2024). Nonetheless, neural network models trained via gradient descent lack the capacity for lifelong learning. Additionally, due to the normalization effect of the Softmax function on classification results, these models exhibit a preference for newly learned classes in dynamic learning environments (Jodelet, Liu, & Murata, 2022).

Extensive experimental results have revealed that when neural networks are fine-tuned solely on sequential, non-overlapping class data, performance on earlier classes degrades substantially (Song, Chen, & Du, 2024), which is known as the *catastrophic forgetting* problem (Rebuffi, Kolesnikov, Sperl, & Lampert, 2017). The decoupling between the classification layer and the feature extraction layer necessitates retraining the classification layer whenever the feature extraction model is modified. Such retraining processes incur huge computational costs and struggle to accommodate newly introduced classes while preserving knowledge of previously learned ones. Moreover, the minimum of the loss function during different training phases can lead to parameter drifts (Mirzadeh, Farajtabar, Pascanu, & Ghasemzadeh, 2020), underscoring the complexity of achieving seamless lifelong learning.

In light of these challenges, incremental learning techniques offer a more practical solution (Liu, Wang, Li, & Wang, 2024). Recent

* Corresponding author.
E-mail addresses: sryang@smail.nju.edu.cn (S. Yang), DZ1733025@smail.nju.edu.cn (T. Zhang), york_z_xu@smail.nju.edu.cn (Z. Xu), lpj@smail.nju.edu.cn (P. Li), 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.2025.107651
Received 24 December 2024; Received in revised form 6 April 2025; Accepted 19 May 2025
Available online 3 June 2025

advances in this domain have yielded numerous strategies to mitigate forgetting (Wang, Jiang, Hu, Liu, & Ji, 2025; Wang, Zhang, Su, & Zhu, 2024), including data replay, regularization, and parameter isolation (Masana, Liu, Twardowski, Menta, Bagdanov, & van de Weijer, 2020). These techniques contribute to resisting catastrophic forgetting and supporting more robust and efficient continual learning processes. However, they may face some inherent limitations. Specifically, regularization-based methods constrain weight updates to maintain past knowledge, yet they struggle with long-term scalability as they accumulate constraints over time, leading to suboptimal performance (Rahman, Coull, & Wright, 2022; Schwarz et al., 2018). Parameter isolation techniques, which allocate separate model parameters for new tasks, introduce memory overheads and do not scale well for continuous data streams (Jiang & Celiktutan, 2023; Schwarz et al., 2018).

Additionally, despite many efforts in addressing CL problems, the general settings for splitting datasets into subsets often lack the flexibility required to handle real-world data streams, such as social media data streams (Yang, Wei et al., 2024). *Data incremental learning* (DIL) (Lange & Tuytelaars, 2021) is a task-free learning paradigm in which batches of data arrive without predefined task boundaries, necessitating that the model be trained and evaluated continuously on streaming data. To effectively learn in such environments, the model must simultaneously retain previously acquired knowledge and incorporate new information, i.e., achieving a coexistence between stability and flexibility.

To address the limitations above, in this paper, we propose *Supervised Contrastive learning with Prototype Distillation* (SCPD), which integrates exemplars, prototypes, supervised contrastive learning (SCL), and knowledge distillation to address the DIL challenges. SCPD utilizes data replay (De Lange et al., 2022) to safeguard learned knowledge while achieving a balance between stability and adaptability through the synergistic application of SCL and prototype distillation loss (PDL). Specifically, SCPD replaces conventional cross-entropy loss with supervised contrastive learning to promote the extraction of representations possessing higher class separability, thereby enhancing model flexibility by ensuring that embeddings are compact within their respective classes in the feature space. However, when new classes are introduced, SCL alone struggles to maintain stable decision boundaries, as it lacks an explicit mechanism to mitigate representation drift across incremental tasks. Therefore, considering prototypes serve as classifiers in incremental learning, we propose a prototype-specific knowledge distillation mechanism, termed the prototype distillation loss (PDL). This specialized loss imparts minimal impact on the extracted representations, thus preserving model stability without hampering its flexibility. In this way, SCL enhances representation learning by enforcing contrastive constraints, while PDL stabilizes the evolving prototype space by distilling knowledge across incremental tasks. By jointly leveraging SCL for representation learning and PDL for stability preservation, SCPD effectively balances adaptability and knowledge retention in DIL settings.

To demonstrate the superior effectiveness of our proposed framework in DIL setups, particularly with unbalanced data streams, we evaluate the proposed SCPD using various benchmark datasets, including MNIST (LeCun, Bottou, Bengio, & Haffner, 1998), CIFAR-10, CIFAR-100 (Krizhevsky, Hinton, et al., 2009), and Tiny-ImageNet (Vinyals, Blundell, Lillicrap, Wierstra, et al., 2016), under both balanced and imbalanced settings. Compared to prior state-of-the-art baselines, our proposed method demonstrates superior effectiveness across various experimental settings. Moreover, through effect visualization and ablation analysis, we validate that our approach effectively mitigates the forgetting problem and accurately represents the data distribution with the learned prototypes.

In summary, we highlight our contributions as follows:

- We propose a novel supervised contrastive learning with prototype distillation (SCPD) for DIL. By coupling prototype distillation loss and supervised contrastive loss, SCPD obviates the need for cross-entropy loss, simultaneously preventing catastrophic forgetting and bolstering representation learning capabilities.
- The prototype distillation (PD) is proposed for SCPD to construct trainable prototypes. Unlike conventional model distillation, PD distills knowledge among prototypes, effectively mitigating embedding distribution drift and enhancing model stability in incremental learning environments.
- Experimental results on various benchmark datasets, using both balanced and imbalanced data streams, SCPD achieves superior performance compared to existing approaches, thereby establishing its efficacy in dynamic and challenging DIL scenarios.

2. Related works

2.1. Data incremental learning

Incremental Learning is a longstanding challenge even before the rise of deep neural networks (Lopez-Lopez, Pardo, & Regueiro, 2022; Parisi, Kemker, Part, Kanan, & Wermter, 2019), and is also referred to as continual learning (CL) or lifelong learning (Ghunaim et al., 2023; nan Han & wei Liu, 2024; Li, Wang, Sun, & Xu, 2023). The mainstream of supervised CL classification paradigms can be categorized into *task incremental*, *class incremental*, and *domain incremental* learning. The general and ever-changing settings in practical application are adapted to the fixed mode of training and evaluating the models sequentially, and the procedure of train-and-test is named the *task*. The training data batch provided during the phase or task p can be denoted as $\mathcal{D}_p = \{(\boldsymbol{x}_i, y_i)\}_p = (\mathcal{X}_p, \mathcal{Y}_p)$. For each *task*, the input distribution varies, and the output distributions differ in these settings (De Lange et al., 2022). *Task incremental* learning (Jiang & Celiktutan, 2023) focuses on solving different tasks in various phases using partially shared models and different output heads, i.e., $\{\mathcal{Y}_p\} \neq \{\mathcal{Y}_{p+1}\}$, where a task identifier p is required for the model to select the corresponding output head. *Domain incremental* learning (Shi & Wang, 2024) deals with different data distributions with the same labels, such as images of digits rotated at various angles. Thus, the output spaces remain the same during training phases, i.e., $\{\mathcal{Y}_p\} = \{\mathcal{Y}_{p+1}\}$. *Class incremental* learning (Tian, Li et al., 2024; Xu, Yang, Xu, Zhao, & Shen, 2024) models handle the emergence of new data classes in subsequent tasks without relying on a task identifier that classifies within the entire output space, i.e., $\{\mathcal{Y}_p\} \subset \{\mathcal{Y}_{p+1}\}$.

In data incremental learning settings, the training and testing phases are decoupled into two independent models: the *learner* and *evaluator* (Lange & Tuytelaars, 2021). The learner receives data batches $\mathcal{B}_t = \{(\boldsymbol{x}_i, y_i)\}_t$ from the data stream S at episode t, while the evaluator independently tests the model with $\mathcal{B}_{te} = \{(\boldsymbol{x}_j, y_j)\}_{te}$ from the test data stream S_{test} at episode te. The DIL paradigm eliminates the constraint of splitting training data into tasks, thus lacking identification for different learning phases. The learner processes streaming data batches and predicts labels in the entire output space.

Online continual learning (Guo, Liu, & Zhao, 2022) is a similar paradigm that also feeds batches of samples to the model. Yet, the models are still aware of the task boundaries, as in class incremental settings. On the other hand, the models are often required to be trained on the same batch through only a single pass, whereas the learner could be trained on one batch of samples several times.

2.2. Continual learning

Data replay or *rehearsal* is straightforward yet effective for addressing CL problems (Masana et al., 2020; Wang et al., 2025; Yang, Wei et al., 2024). These methods involve preserving a subset of observed

samples in memory and recalling these exemplars from memory to co-train with new input data. Retraining a subset of old data strikes a balance between the performance of retraining all data from scratch and the computational and memory costs involved in fine-tuning solely on the new data. Therefore, the comprehensive training and fine-tuning approaches can be considered as the upper and lower bounds of data replay methods. Besides, constraints on the loss function are often employed with the replay method to mitigate the negative effect of new losses on the performance of previous tasks. For instance, *GEM* (Lopez-Paz & Ranzato, 2017) employs episodic memory and inequality constraints on the loss function to prevent new parameter updates from reducing the loss of old tasks. Similarly, *iCaRL* (Rebuffi et al., 2017) utilizes task-independent exemplar memory combined with *distillation loss* to minimize the distribution drift in old output space. CCIL (Wang et al., 2025) proposes consistency-enhanced data replay with debiased classifiers for class incremental learning by measuring the data consistency quantitatively. Parameter isolation (Mallya & Lazebnik, 2018) is another way to maintain the stability of old tasks by fixing a subset of parameters. However, the parameter capacity for future tasks should be a trade-off between reducing parameters for new tasks and expanding the network's capacity. Recent work (Hu, Tang, Miao, Hua and Zhang, 2021) applies the causal graph to demonstrate that the data replay and distillation methods can effectively combat the forgetting problem by establishing causal paths between old data and the new model.

For the task-free DIL paradigm, rehearsal still enhances the accuracy of prediction. *Reservoir* (Vitter, 1985) simply employs a random replay method to select samples in memory. *MIR* (Aljundi, Belilovsky et al., 2019) chooses a controlled sampling of replay memories, which selects the exemplars most impacted by the parameter updating. *GSS* (Aljundi, Lin et al., 2019) performs constraint sample selection that maximizes the diversity of samples in the memory. The *Continual Prototype Evolution (CoPE)* (Lange & Tuytelaars, 2021) method is proposed as the *learner-evaluator* framework for the DIL problem. The method imports the evolving prototypes and balanced replay with Pseudo-Prototypical Proxy (PPP) loss to train the neural network. Besides, the parameter isolation method *CN-DPM* (Lee, Ha, Zhang, & Kim, 2020) develops a set of neural network experts for subsets of data and extends the scale of parameters for new data classes.

2.3. Supervised contrastive learning

Contrastive learning (Jaiswal, Babu, Zadeh, Banerjee, & Makedon, 2020; Mai, Zeng, & Hu, 2023; Xiong, Wang, Yang, Shen, & Zhao, 2025) has attracted much attention due to its outstanding performance. Different from assigning each class a target vector in cross-entropy loss, contrastive learning aims at bringing embeddings of the same class closer while pushing embeddings of different classes apart. The contrastive loss can be considered an extension of the triplet loss that computes more positive and negative pairs for each input sample (Wang, Han, Wei, Zhang, & Wang, 2021). The supervised contrastive loss (Khosla et al., 2020) proves its outstanding performance and stimulates various further works. For instance, ReGCL (Ji et al., 2024) addresses the conflict between GNNs and contrastive learning by analyzing gradient participation in message passing and introducing gradient-guided structure learning and a gradient-weighted InfoNCE loss. Contrastive learning has been applied in various tasks, including semantic segmentation (Hu, Cui and Wang, 2021), video understanding (Tian, Lu et al., 2024), image enhancement (Liang et al., 2022), and video compression (Tian, Yan, Zhai, Chen, & Gao, 2023), demonstrating its versatility.

Considering the embedding space constructed by deep metric learning (Horiguchi, Ikami, & Aizawa, 2020), the contrastive learning loss function is particularly suitable for continual learning methods using a nearest-class-mean (NCM) classifier, which has already been employed in the class-incremental learning problem (Cha, Lee, & Shin, 2021; Mai, Li, Kim, & Sanner, 2021). Co^{2}L (Cha et al., 2021) employs a modified supervised contrastive loss and instance-wise relation distillation to conduct across all three CL setups. The *Supervised Contrastive Replay* (SCR) method (Mai et al., 2021) employs supervised contrastive loss in online continual problems, while prototypes are recomputed in the evaluation phase and not maintained in the training phase. The method is also suitable for the DIL problem and is evaluated in the following experiments. PASS (Zhu, Zhang, Wang, Yin, & Liu, 2021) leverages prototype augmentation and self-supervised learning to improve feature transferability, while our method incorporates supervised contrastive learning, prototype distillation, and memory replay, ensuring better knowledge retention and feature discriminability.

3. The proposed method

The overview of the SCPD training phase is illustrated in Fig. 1. The SCPD framework comprises four components: prototypes, exemplar memory, supervised contrastive loss, and prototype distillation loss. In training episode t, the model receives a data batch $\mathcal{B}_t$ from the data stream $\mathcal{S}_{train}$. The replay samples $\mathcal{B}_s$ are randomly selected from the exemplar memory to fed to the model together as the input data batch $\mathcal{B}_n$. Subsequently, two random augmentations are applied to generate $\tilde{\mathcal{B}}_n$. The feature extractor $Enc(\cdot)$ encode $\tilde{\mathcal{B}}_n$ into embeddings. The network is then optimized using the supervised contrastive loss $\mathcal{L}_{SC}$ and prototype distillation loss $\mathcal{L}_{PD}$. After optimization, prototypes are either added or updated with the input embeddings, and the exemplar memory will be updated. During the test phase, test embeddings are extracted using $Enc(\cdot)$, and the dot product of embeddings and prototypes is computed to predict the labels. In this section, we provide details of the SCPD module in the subsequent sections.

3.1. Exemplar memory

To mitigate the issue of forgetting when the model is trained on data streams, a prevalent strategy involves retaining a subset of replay samples, known as exemplars, which are subsequently reintroduced during the training process to maintain performance on previously learned classes. In alignment with existing data replay methods (Lange & Tuytelaars, 2021; Rebuffi et al., 2017), we store these exemplars within a constrained memory capacity, denoted as M. The memory is uniformly divided based on the number of currently observed classes. As new classes emerge, the allocation for each class is proportionally reduced, and any samples exceeding the allocated capacity are removed. This approach ensures that the exemplars remain balanced within the memory, thereby providing sufficient replay samples for each class even in imbalanced settings.

In each training episode, the model randomly samples replay samples $\mathcal{B}_s$ from memory without replacement and trains on these samples alongside the new input samples. The sampling range encompasses all data categories in memory, including those currently being trained. This random sampling method significantly reduces network training overhead and sample selection costs, especially when compared to methods that either utilize the entire set of replay samples or select only key samples. In a data incremental setting, where training batches are smaller, the number of training stages t increases proportionally with the same data volume, necessitating more frequent replay operations. Consequently, the efficiency gains from reduced sampling costs of replay samples are more pronounced in comparison to typical class incremental learning environments.

The *memory updating* strategy directly adds input samples to the memory when it is not full. Once the memory is full, a newly arriving sample replaces an existing exemplar with a probability M_c/N_{seen}, where M_c represents the capacity for the class c and N_{seen} is the number of observed samples belonging to the class c. This probabilistic replacement mechanism ensures that the memory maintains a representative and approximately balanced subset of samples across classes, preserving diversity while mitigating class imbalance over time.

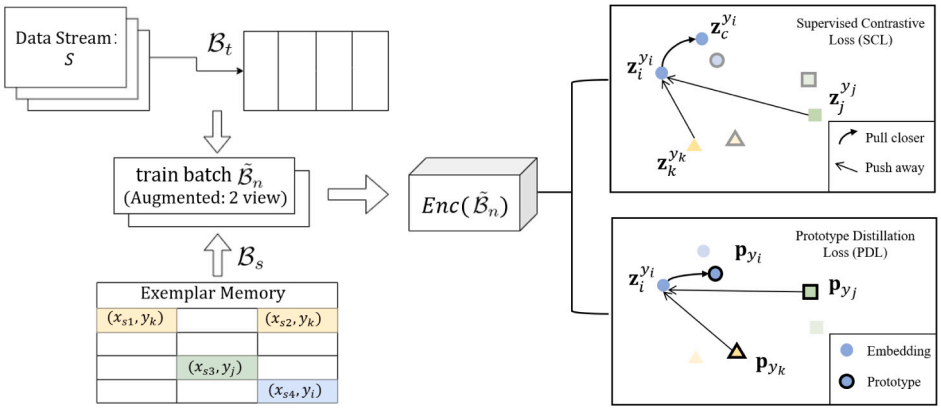


Fig. 1. The overview of the SCPD training process. At training episode t, data batch $\mathcal{B}_t$ is sampled from the data stream S_{train}, and replay exemplars $\mathcal{B}_s$ are retrieved from the memory to alleviate the forgetting problem. The data will be augmented into two views and encoded to compute the supervised contrastive loss and prototype distillation loss.

3.2. Prototypes as the classifier

In contrast to using the softmax linear layer with cross-entropy classification loss, employing prototypes as nearest-class-mean (NCM) classifiers offers enhanced convenience for expansion and adaptation to new class data. Prototypes facilitate label prediction by minimizing the distance or maximizing the similarity between prototypes and embeddings, thereby reducing reliance on trainable parameters. Consequently, we adopt prototypes as the classifier within the SCPD framework.

For classification, test embeddings $\boldsymbol{z}_i = Enc(\boldsymbol{x}_i)$ are compared with all preserved C prototypes $\boldsymbol{p}_k$, where C represents the number of known classes, and each prototype corresponds to a specific class. The dot products between embeddings and prototypes are computed to quantify their similarity. The predicted label is determined by identifying the prototype that yields the highest similarity measure, which is denoted as:

$$\begin{aligned}\hat{y}_i &= \arg\max_{k=1,\dots,C} \frac{\boldsymbol{z}_i \cdot \boldsymbol{p}_k^T}{\sum_j \boldsymbol{z}_i \cdot \boldsymbol{p}_j^T} \\ &= \arg\max_{k=1,\dots,C} \boldsymbol{z}_i \cdot \boldsymbol{p}_k^T.\end{aligned} \tag{1}$$

Thus, the embedding features will be classified into the class corresponding to the prototype with the highest similarity.

When the model receives data from new classes, the old embeddings previously trained through SCL will be pushed away by the emerging embeddings, which may lead to distribution drift. Recalculating prototypes using exemplars stored in memory offers a feasible solution; however, these exemplars, sampled from the training data, may not precisely represent the input distribution, especially when memory capacity is limited. Therefore, prototypes are updated once after the model training in each episode. To prevent the prototypes from being out of date and forgetting old information simultaneously, the prototypes are supposed to update continually with a high momentum factor α for each batch of data (Lange & Tuytelaars, 2021). The momentum α should be large enough to prevent the oscillation of prototypes. The update procedure is conducted as follows:

$$\begin{aligned}\boldsymbol{p}_k^t &= \alpha \boldsymbol{p}_k^{t-1} + (1-\alpha)\frac{1}{|I_k|}\sum_{i\in I_k} \boldsymbol{z}_i, \\ \boldsymbol{p}_k^t &= \boldsymbol{p}_k^t / \parallel \boldsymbol{p}_k^t \parallel_2,\end{aligned} \tag{2}$$

where α is the momentum factor for updating prototypes, I_k denotes the index set of input samples that belong to class k in the new data batch $\mathcal{B}_t$, and the prototypes are normalized after updating. Following previous works (Lange & Tuytelaars, 2021), we apply class-specific prototype momentum updates. Specifically, the high momentum factor α stabilizes prototype updates even when certain classes are underrepresented in recent batches, reducing bias from transient imbalance and stabilizing the impetuous changes in the data stream. At the end of each episode, prototypes are re-estimated using class-representative memory exemplars. These exemplars are sampled to cover all previously seen classes, thus serving as a balancing mechanism.

3.3. Supervised contrastive loss

To accelerate model convergence and ensure that embeddings belonging to the same classes are closely clustered within the feature space (Jaiswal et al., 2020; Khosla et al., 2020), we employ *supervised contrastive learning* (SCL) as the training loss function. Our approach involves two key components to convert input samples into lower-dimensional embeddings: the encoder network $Enc(\cdot)$ and the data augmentation module $Aug(\cdot)$.

We denote the batch size of the training data as N_b and the number of replay exemplars as N_s in each training episode. In episode t, the model processes a training batch $\mathcal{B}_n = \{(\boldsymbol{x}_i, y_i) | i = 1, \dots, N\}$, which consists of new samples $\mathcal{B}_t$ and replay exemplars $\mathcal{B}_s$, with $N = N_b + N_s$. Meanwhile, the data augmentation module applies various transformations – such as cropping, rotation, and flipping – to each input sample $\boldsymbol{x}_i$. Thus, the augmented training batch becomes $\tilde{\mathcal{B}}_n = \{(\tilde{\boldsymbol{x}}_i, \tilde{y}_i) | i = 1, \dots, 2N\}$, where $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_{2i}$ represent two views of $\boldsymbol{x}_i$.

The augmented batches $\tilde{\mathcal{B}}_n$ are then fed into the encoder network $Enc(\cdot)$, which generates the embedding $\boldsymbol{z}_i = Enc(\tilde{\boldsymbol{x}}_i)$ for each $\tilde{\boldsymbol{x}}_i \in \tilde{\mathcal{B}}_n$. These embeddings are normalized to unit vectors using $\boldsymbol{z}_i = \boldsymbol{z}_i / \|\boldsymbol{z}_i\|_2$ (Khosla et al., 2020). The *supervised contrastive loss* is computed on these embeddings as follows:

$$\mathcal{L}_{SC}(\boldsymbol{z}_i) = \sum_{i\in I} \frac{-1}{|C(i)|} \sum_{c\in C(i)} \log \frac{\exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_c / \tau)}{\sum_{a\in A(i)} \exp(\boldsymbol{z}_i \cdot \boldsymbol{z}_a / \tau)}, \tag{3}$$

where $I = \{1, \dots, 2N\}$ represents the index set of the augmented batch $\tilde{\mathcal{B}}_n$, $C(i) = \{c \in A(i) \mid \tilde{y}_c = \tilde{y}_i\}$ denotes the index set of samples from the same category, and $A(i) = I \backslash \{i\}$ represents the set of indices for all samples in the augmented data batches, except for the ith input sample. In this way, the supervised contrastive loss encourages pulling together features from the same class and pushing apart features from different classes. Since the augmentation operations applied are with slight magnitudes, we do not introduce constraints on the consistency between feature representations with and without augmentation.

Unlike prior works (Khosla et al., 2020; Mai et al., 2021), we do not incorporate a projection layer $Proj(\cdot)$ or a linear classification layer into the network architecture. As a continual learning model, there is no phase dedicated to fine-tuning the classification layer using cross-entropy (CE) loss, primarily because prototypes function as classifiers. Besides, the activation of the receptive field in the last layer is sufficient to constitute a meaningful representation of the input data (Mai et al., 2021). In addition, τ is the temperature control parameter; we set it to the recommended value (Rebuffi et al., 2017) of 0.1.

Algorithm 1 The training phase of SCPD in the DIL setup.

Training:
Input: training data continuum $\mathcal{S}_{train}$
Parameter: exemplar memory capacity M, updating momentum for prototypes α, batch size N_b, replay size N_s

Initialize memory $\mathcal{M} \leftarrow \phi$
for $\mathcal{B}_t \leftarrow \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_{N_b}, y_{N_b})\} \sim \mathcal{S}$ **do**
  $\mathcal{B}_s \leftarrow$ randomly sampling N_s exemplars from $\mathcal{M}$
  $\mathcal{B}_n \leftarrow \mathcal{B}_s \cup \mathcal{B}_t$
  $\tilde{\mathcal{B}}_s, \tilde{\mathcal{B}}_t \leftarrow Aug(\mathcal{B}_s), Aug(\mathcal{B}_t)$
  Compute $\mathcal{L}_{SC}$ on $Enc(\tilde{\mathcal{B}}_n)$ according to Eq. (3)
  if $|\mathcal{B}_r| = N_s$ **then**
    Compute $\mathcal{L}_{PD}$ on $Enc(\tilde{\mathcal{B}}_s)$ according to Eq. (4)
  else
    $\mathcal{L}_{PD} = 0$
  end if
  $\mathcal{L} \leftarrow \mathcal{L}_{SC} + \mathcal{L}_{PD}$
  $Enc(\cdot) \leftarrow SGD(\mathcal{L}, Enc(\cdot))$
  Update prototypes according to Eq. (2)
  Employ the memory updating strategy in Section 3.1
end for

In summary, the model trained with SCL enables the extraction of representations with higher class separability, as embeddings are meaningfully and closely clustered in the feature space. This enhanced class divisibility facilitates more effective classification by prototypes, thereby improving the overall performance of the SCPD method in incremental learning scenarios.

3.4. Prototype distillation loss

In the context of incremental learning, catastrophic forgetting can be effectively mitigated by *knowledge distillation* between past and present models. Previous works (Rebuffi et al., 2017; Zhou, Wang, Ye, Ma, Pu, & Zhan, 2022) typically compute the distillation loss using outputs from the past and new network models. While knowledge distillation effectively mitigates forgetting by enhancing model stability, it concurrently impairs the model's ability to assimilate new knowledge, reducing flexibility. This limitation arises because the present model is constrained to remain similar to the old model. In contrast, our approach eschews the preservation of the old model and instead leverages prototypes as representations of the old embeddings. Prototypes are designed to remain stable and are updated with high momentum, ensuring that the learned embeddings remain closely aligned with their corresponding prototypes. This strategy is crucial for maintaining the integrity of learned representations over time. Moreover, the exemplars stored in memory represent only a subsampling from the data stream $\mathcal{S}$ and may not accurately capture the entire feature space distribution. Consequently, it is beneficial to enforce that the embeddings of exemplars remain proximal to their respective prototypes rather than to other prototypes, thereby preventing significant deviations between prototypes and embeddings. To achieve this, we introduce the *Prototype Distillation Loss* (PDL), which is computed for each embedding $\boldsymbol{z}_j = Enc(\tilde{\boldsymbol{x}}_j)$ from the augmented replay samples $\tilde{\mathcal{B}}_s$ as follows:

$$\mathcal{L}_{PD}(\boldsymbol{z}_j) = \log \frac{\exp(\boldsymbol{z}_j \cdot \boldsymbol{p}_c)}{\sum_{k \neq c} \exp(\boldsymbol{z}_j \cdot \boldsymbol{p}_k)}, \tag{4}$$

where c is the index of the prototypes that correspond to the same class as $\boldsymbol{z}_j$. Since embeddings may be randomly distributed at the beginning of training, we will compute prototype distillation loss when the amount of replay exemplars reaches the expected parameter N_s.

Unlike model distillation, prototype distillation facilitates knowledge transfer and targets the prototypes themselves. This approach exerts minimal influence on the representations extracted by the model, thereby preserving stability without compromising the model's flexibility. The overall loss function for SCPD is formulated as a combination of SCL and PDL, denoted as:

$$\mathcal{L}_{SCPD} = \mathcal{L}_{SC} + \mathcal{L}_{PD}. \tag{5}$$

During the model training process, SCL enhances the model's capacity to learn new information by improving the discriminative effectiveness of the extracted representations. Concurrently, PDL mitigates catastrophic forgetting by maintaining the alignment between embeddings and their respective prototypes. The synergistic effect of SCL and PDL ensures that SCPD achieves both flexibility and stability in incremental learning scenarios. Algorithm 1 summarizes the training and testing procedure SCPD employs to address the DIL problem.

Algorithm 2 The testing phase of the SCPD method in the DIL setup

Input: test data continuum $\mathcal{S}_{test}$, test batch size N_{te}

for $\mathcal{B}_{te} \leftarrow \{(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_{N_{te}}, y_{N_{te}})\} \sim \mathcal{S}_{test}$ **do**
  Initialize predicted labels $\hat{\mathcal{Y}}_{te} \leftarrow \phi$
  for $j \leftarrow 1, \dots, N_{te}$ **do**
    Predict $\hat{y}_j$ according to Eq. (1)
    $\hat{\mathcal{Y}}_{te} = \hat{\mathcal{Y}}_{te} \cup \{\hat{y}_j\}$
  end for
end for
return predicted labels $\hat{\mathcal{Y}}_{te}$

4. Experiments

We compare the SCPD method with several baselines in data incremental settings as outlined in Lange and Tuytelaars (2021), utilizing the MNIST, CIFAR10, CIFAR100, and Tiny-ImageNet datasets. The model processes streaming batches of data with a batch size $N_b = 10$ from the training data stream $\mathcal{S}_{train}$. To compare with other baselines, the samples are rearranged following the order of tasks. Thus, after training on all data in the current task, the model trains on data belonging to the new task category. Notably, the DIL models are designed to be agnostic to task boundaries, whereas methods such as GEM and iCaRL still require one epoch to deal with novel classes. All experimental results are conducted over five random seeds to generate random sample sequences and network initialization. Every data batch is trained over five epochs to converge. The SGD optimizer is employed for training as suggested by Mirzadeh et al. (2020), and other DIL methods (Lange & Tuytelaars, 2021; Mai et al., 2021).

4.1. Implementation details

Dataset. *Split-MNIST*, *Split-CIFAR10*, *Split-CIFAR100*, and *Split-TinyImageNet* are used as benchmark datasets in our experiments. The Split-MNIST (S-MNIST) datasets, derived from the MNIST dataset (LeCun et al., 1998), consist of 60k hand-written digits in 10 classes with 50k training samples and 10k samples for testing. The Split-CIFAR10 dataset (S-CIFAR10) from the CIFAR10 dataset and the Split-CIFAR100 (S-CIFAR100) from the CIFAR100 dataset (Krizhevsky et al., 2009), each contains 50k training and 10k testing 32×32 RGB images, with S-CIFAR10 spanning 10 classes and S-CIFAR100 spanning 100 classes. Split-TinyImageNet (S-TinyImageNet), derived from Tiny-ImageNet (Vinyals et al., 2016), contains 64×64 RGB images in 100 classes.

Data stream setups. In the *balanced setup*, the model is trained on the entire training dataset, with samples divided into multiple tasks. Specifically, S-MNIST is divided into 5 tasks, each consisting of approximately 10k samples from 2 disjoint classes. S-CIFAR10 is similarly split into 5 tasks, each containing 10k samples from 2 classes. S-CIFAR100 is divided into 20 tasks, each with 2.5k samples from 5 classes. S-TinyImageNet is divided into 10 tasks, each with 5k samples from 10 classes. After training on the current task, the classification performance of all models will be evaluated on the test data of the learned categories.

The *imbalanced experimental setup* (Lange & Tuytelaars, 2021) exhibits the ability of DIL methods in environments where the number of data samples from different classes is not equivalent. To create this imbalance, we reduce the number of samples in specific tasks. Specifically, the datasets are still split into several tasks, while the amount of training data in each task varies. One task maintains a normal sample size, while the others have reduced sample sizes. The normal task with a normal sample size is denoted as T_i. For S-MNIST, the normal task has 2k samples, while other tasks have 0.2k samples, and $T_i = \{1, 2, 3, 4, 5\}$; For S-CIFAR10, the normal task has 4k samples while others have 2k samples, and $T_i = \{1, 2, 3, 4, 5\}$; For S-CIFAR100, the normal task has 2.5k samples while others have 1k samples, and $T_i = \{1, 5, 10, 15, 20\}$. For S-TinyImageNet, the normal task has 5k samples while others have 1k samples, and $T_i = \{1, 3, 5, 7, 9\}$.

Parameter setting. Closely following the experiment settings from previous works (Lee et al., 2020; Mai et al., 2021), the encoder network $Enc(\cdot)$ is configured as an MLP with two hidden layers of 400 units for the balanced S-MNIST dataset (Lange & Tuytelaars, 2021; Lee et al., 2020), and 100 units for the imbalanced S-MNIST dataset (Aljundi, Lin et al., 2019; Lange & Tuytelaars, 2021). For the S-CIFAR10, S-CIFAR100, and S-TinyImageNet datasets, we utilize the ResNet18 model without the linear classification layer (He, Zhang, Ren, & Sun, 2016). The computation of SCL requires two types of data augmentation (Yang, Li, Xiong, Shen, & Zhao, 2024; Yang, Shen and Zhao, 2024; Yang et al., 2022), and data augmentation strategies from Zhou et al. (2022) are applied to the S-CIFAR10, S-CIFAR100, and S-TinyImageNet datasets. Specifically, we employ the default random cropping function with image size configurations as implemented in PyTorch, ensuring consistency across datasets. Rotation is applied within the range of $[0, 10°]$, and affine transformations are also implemented via Pytorch. Additionally, we adopt a 16-pixel Cutout strategy (He et al., 2016; Zhou et al., 2022). For the balanced setup, the replay memory sizes are set to $|M| = 2000$ for S-MNIST, $|M| = 1000$ for S-CIFAR10, and $|M| = 5000$ for both S-CIFAR100 and S-TinyImageNet. In the imbalanced setup, the memory size is reduced to 300 for the MNIST dataset due to the decreased training sample volume. In the benchmark comparison, we randomly sample $N_s = 100$ exemplars in every training epoch on S-CIFAR10, S-CIFAR100, and S-TinyImageNet, and set $N_s = 300$ for the S-MNIST dataset. Meanwhile, the random rotation and flip strategies should be used carefully to augment samples on digit datasets (Shorten & Khoshgoftaar, 2019). The learning rate of the SGD optimizer on balanced datasets is set to 0.15 for S-MNIST, 0.05 for S-CIFAR10 and S-CIFAR100, and 0.1 for Split-CIFAR10 in imbalanced settings. The prototype updating momentum is set to 0.99 for S-MNIST and S-CIFAR10 and 0.9 for S-CIFAR100 in balanced setups, and 0.9 for S-MNIST in imbalanced setups.

Compared methods. We compare SCPD with many baselines or state-of-the-art methods in balanced setups, with some results adopted from Lange and Tuytelaars (2021), which is marked with "*" in the result tables. Thus, we adhere to the same experimental setups as Lange and Tuytelaars (2021). The *iid-offline* model, which is non-incrementally trained with CE loss, can be regarded as the upper bound of the incremental learning results. Conversely, the *finetune* model, trained sequentially without any anti-forgetting technique, can be treated as the lower bound. Rehearsal methods *iCaRL* and *GEM* are included for comparison, as they address the class incremental problem and require knowledge of task boundaries to accommodate new tasks. Four DIL models are considered, including *Reservoir*, *MIR*, *GSS*, and *CoPE*. Besides, *CN-DPM* is a model expansion method for task-free settings. We have also included two SOTA methods, OnPro (Wei, Ye, Huang, Zhang, & Shan, 2023) and LODE (Liang & Li, 2024), for comparison in the experiment. In imbalanced setups, we also compare the DIL methods CoPE, GSS, MIR Reservoir, and SOTA methods OnPro and LODE.

Notably, our proposed methods are donated as SCPD and SCPD-r. SCPD refers to the label prediction using the updated prototype classifier directly after training is completed, while SCPD-r refers to the label prediction using the mean prototype classifier calculated from all samples in the replay memory after training is completed.

Table 1
The accuracies (average ± std) of all methods on the balanced S-MNIST, S-CIFAR10, S-CIFAR100, and S-TinyImageNet benchmarks. All experiments are conducted across five independent random seeds.

	S-MNIST	S-CIFAR10	S-CIFAR100	S-TinyImageNet
iid-offline[a]	98.44 ± 0.02	83.02 ± 0.60	50.28 ± 0.66	51.33 ± 0.20
finetune[a]	19.75 ± 0.05	18.55 ± 0.34	3.53 ± 0.004	1.22 ± 0.03
GEM[a]	93.25 ± 0.36	24.13 ± 2.46	11.12 ± 2.48	4.46 ± 2.37
iCARL[a]	83.95 ± 0.21	37.32 ± 2.66	10.80 ± 0.37	5.45 ± 0.78
DN-CPM[a]	93.23 ± 0.09	45.21 ± 0.18	20.10 ± 0.12	13.63 ± 0.46
reservoir[a]	92.16 ± 0.75	42.48 ± 3.04	19.57 ± 1.79	13.35 ± 2.07
MIR[a]	93.20 ± 0.36	42.80 ± 2.22	20.00 ± 0.57	15.36 ± 3.73
GSS[a]	92.47 ± 0.92	38.45 ± 1.41	13.10 ± 0.94	10.05 ± 2.96
CoPE[a]	93.94 ± 0.20	48.92 ± 1.32	21.62 ± 0.69	16.75 ± 1.08
PASS	96.33 ± 0.43	70.55 ± 1.05	42.44 ± 0.82	32.58 ± 0.65
CLS-ER	92.55 ± 0.21	69.20 ± 0.71	40.25 ± 0.64	31.33 ± 0.51
OnPro	96.78 ± 0.22	57.80 ± 1.11	30.00 ± 0.60	22.11 ± 0.77
LODE	96.72 ± 0.45	73.13 ± 1.70	37.82 ± 0.35	25.30 ± 1.09
SCPD	**96.92 ± 0.37**	72.13 ± 1.50	46.78 ± 0.57	32.24 ± 0.44
SCPD-r	96.81 ± 0.20	**73.27 ± 0.56**	**47.08 ± 0.56**	**33.10 ± 0.56**

[a] The results are reported from previous works.

4.2. Benchmark comparison

4.2.1. Balanced setup

The results under balanced settings on the S-MNIST, S-CIFAR10, S-CIFAR100, and S-TinyImageNet datasets are summarized in Table 1. To more comprehensively assess the effectiveness of our proposed PDL, we include the recently proposed PASS method (Zhu et al., 2021) in the comparison, which also addresses forgetting via representation regularization. Additionally, given that Exponential Moving Average (EMA) has been demonstrated to enhance knowledge retention in incremental learning scenarios, we incorporate the EMA-based method CLS-ER (Arani, Sarfraz, & Zonooz, 2022) for comparison. As shown in the results, our method consistently achieves better performance across all evaluated datasets, highlighting the effectiveness and generalizability of our approach.

For comparison, the results of the finetune models lead to the same conclusion as other papers, indicating that the accuracy of neural networks will face a sharp decline without anti-forgetting measures. GEM and iCaRL perform poorly in data incremental settings. However, CN-DPM and CoPE show strong results across multiple datasets. CN-DPM leverages an expandable expert system, avoiding contamination of learned knowledge, though data allocation remains a challenge. CoPE applies a similar feature clustering, but does not perform as well as other methods utilizing contrastive loss functions. The SCPD that uses both contrastive loss and prototype distillation has achieved the best results. Moreover, SCPD surpasses SOTA methods OnPro and LODE by up to 10%. These results indicate that SCPD is an effective approach for addressing the DIL problem.

Table 2
The accuracies (average ± std) on the imbalanced S-MNIST, S-CIFAR10, S-CIFAR100, and S-TinyImageNet benchmarks. We report average results across five independent random seeds. The task T_i represents that more samples are observed in the task and fewer samples in other tasks, and the Avg. represents the average accuracies of 5 tasks for these methods.

Dataset	Tasks	SCPD-r	SCPD	OnPro	LODE	CoPE[a]	GSS[a]	MIR[a]	Reservoir[a]
S-MNIST	T_1	80.2 ± 2.3	79.8 ± 3.0	79.1 ± 2.9	80.3 ± 3.0	83.4 ± 2.0	75.9 ± 3.2	64.8 ± 5.1	64.2 ± 2.3
	T_2	85.1 ± 2.2	84.8 ± 1.8	83.5 ± 0.8	84.0 ± 1.0	84.5 ± 1.6	78.5 ± 2.7	67.4 ± 3.2	65.5 ± 4.6
	T_3	87.4 ± 0.9	85.7 ± 2.2	84.3 ± 0.9	86.4 ± 1.0	85.1 ± 0.6	81.5 ± 2.3	72.4 ± 3.0	72.1 ± 4.0
	T_4	89.9 ± 0.8	89.2 ± 1.0	88.4 ± 0.6	87.9 ± 0.7	84.8 ± 1.0	79.5 ± 0.6	72.6 ± 3.1	73.6 ± 2.4
	T_5	91.0 ± 0.6	90.9 ± 1.7	90.1 ± 0.4	84.8 ± 1.0	84.0 ± 1.3	79.1 ± 0.7	77.2 ± 3.4	73.2 ± 4.0
	Avg.	**86.7 ± 4.3**	86.1 ± 4.3	85.1 ± 4.3	84.7 ± 2.8	84.4 ± 0.7	78.9 ± 2.0	70.9 ± 4.9	69.7 ± 4.5
S-CIFAR10	T_1	56.6 ± 3.7	52.9 ± 3.3	39.4 ± 2.3	50.1 ± 3.1	39.0 ± 1.3	32.3 ± 3.0	32.6 ± 3.0	35.5 ± 3.4
	T_2	57.1 ± 2.3	56.1 ± 5.5	35.5 ± 2.4	48.8 ± 4.6	35.3 ± 2.6	28.3 ± 0.4	28.3 ± 0.4	29.3 ± 2.8
	T_3	59.3 ± 1.7	56.1 ± 3.8	41.2 ± 4.8	50.4 ± 2.5	36.2 ± 2.5	29.5 ± 1.5	29.5 ± 1.5	31.4 ± 2.1
	T_4	61.5 ± 3.2	59.7 ± 3.8	46.1 ± 3.2	53.5 ± 3.9	39.1 ± 2.4	34.6 ± 1.3	34.6 ± 1.3	32.1 ± 0.6
	T_5	64.2 ± 1.7	62.8 ± 2.8	54.6 ± 5.4	58.3 ± 2.5	37.3 ± 3.3	28.3 ± 2.4	28.3 ± 2.4	28.8 ± 1.9
	Avg.	**59.7 ± 3.2**	57.5 ± 3.8	43.4 ± 7.6	52.2 ± 4.3	37.4 ± 1.7	30.6 ± 2.8	29.6 ± 2.3	31.4 ± 2.7
S-CIFAR100	T_1	39.6 ± 0.8	40.8 ± 0.9	36.5 ± 0.5	33.9 ± 0.7	18.2 ± 0.6	10.2 ± 3.0	18.4 ± 0.9	11.1 ± 0.6
	T_2	39.9 ± 0.8	40.6 ± 1.2	36.8 ± 1.0	34.2 ± 0.8	18.5 ± 1.3	10.7 ± 0.4	17.6 ± 0.9	11.5 ± 1.4
	T_3	40.0 ± 0.8	41.7 ± 0.3	36.9 ± 0.8	34.5 ± 0.7	19.2 ± 0.9	11.1 ± 1.5	17.8 ± 0.7	11.9 ± 0.7
	T_4	40.1 ± 1.2	40.9 ± 1.2	36.2 ± 1.2	34.1 ± 0.6	18.7 ± 0.6	11.1 ± 1.3	17.8 ± 0.9	12.1 ± 0.8
	T_5	40.9 ± 0.9	41.1 ± 0.9	37.0 ± 0.9	35.1 ± 0.5	18.5 ± 1.5	11.1 ± 2.4	17.6 ± 0.4	12.5 ± 1.1
	Avg.	40.1 ± 0.5	**41.0 ± 0.4**	36.7 ± 0.3	34.4 ± 0.6	18.6 ± 0.4	10.8 ± 0.4	17.8 ± 0.3	11.8 ± 0.5
S-TinyImageNet	T_1	24.3 ± 1.2	23.8 ± 2.3	20.7 ± 0.3	21.8 ± 1.8	13.7 ± 0.8	8.2 ± 2.5	13.2 ± 0.9	9.2 ± 0.3
	T_2	22.7 ± 2.9	23.3 ± 1.3	19.8 ± 1.5	21.5 ± 1.2	13.9 ± 0.3	9.7 ± 2.1	14.3 ± 1.0	10.4 ± 1.1
	T_3	25.5 ± 0.6	23.6 ± 1.6	21.6 ± 0.7	22.9 ± 1.4	12.8 ± 2.2	10.3 ± 0.7	12.4 ± 1.1	10.1 ± 0.7
	T_4	23.8 ± 1.0	22.9 ± 1.3	21.9 ± 0.5	20.4 ± 0.5	13.0 ± 0.5	9.1 ± 1.2	11.5 ± 1.9	11.5 ± 0.8
	T_5	24.8 ± 0.9	24.0 ± 0.8	22.9 ± 0.9	20.8 ± 0.8	11.5 ± 0.9	10.5 ± 2.0	12.8 ± 1.2	10.9 ± 0.9
	Avg.	**24.2 ± 1.0**	23.5 ± 0.4	21.4 ± 1.0	21.5 ± 0.9	13.0 ± 0.9	9.56 ± 0.9	12.8 ± 1.0	10.4 ± 0.9

[a] The results are reported from the previous paper.

4.2.2. Imbalanced setup

The results for imbalanced settings on benchmarks are reported in Table 2. Compared with the performance in the balanced setup, the accuracy decreases, and the imbalanced data problem has an impact on nearly all baselines, including SCPD. This is because the model tends to learn classes containing a larger number of samples and learns a poorer representation of features for classes with fewer samples (Zhang, Kang, Hooi, Yan, & Feng, 2023). Moreover, SCPD uses prototypes, i.e., the mean representation of the training samples for classification. The prototype may produce a positional bias with fewer training samples and cannot reflect the actual representation centroid of the class in the feature space (Lee, Lee, & Kwak, 2022). Despite the accuracy degradation, SCPD significantly surpasses the prior state-of-the-art by about 2% on S-MNIST, 5–7% on S-CIFAR10, 8% on S-CIFAR100, and 3% on S-TinyImageNet. These results demonstrate that combining supervised contrastive loss and prototype distillation loss contributes to performance improvements in imbalanced data scenarios. This is because prototype distillation helps maintain stable representations of previously learned classes, especially minority classes, by preserving historical knowledge and reducing distribution drift. Meanwhile, SCL encourages better semantic separation between classes, enhancing the model's ability to distinguish minority class samples from majority classes. This alleviates the risk of representation collapse and ensures that meaningful features are retained even in imbalanced scenarios.

In addition, the results demonstrate that the overall accuracy can be further improved by using the mean prototype classifier calculated from all samples in the replay memory. Specifically, in balanced settings, SCPD-r brings improvements of approximately 0.2%, 0.3%, and 0.9% on S-CIFAR10, S-CIFAR100, and S-TinyImageNet. Under imbalanced settings, SCPD-r brings improvements of approximately 0.6%, 2.2%, and 0.7% on S-MNIST, S-CIFAR10, and S-TinyImageNet, respectively. These results further validate the effectiveness of prototype distillation.

4.3. Analytical results

4.3.1. Further analysis on classification performance

In Fig. 2, we demonstrate the classification performance on S-CIFAR10 after training on five tasks using the finetune method, the CoPE method, and our proposed SCPD-r method. It can be seen that the finetune method predicts all categories as the two newly learned ones, failing to predict the categories from earlier tasks accurately. The CoPE method shows some improvements, but its correct predictions are mostly concentrated in the last five categories, with earlier learned categories not being correctly predicted. In contrast, our proposed SCPD is capable of predicting data from all previously learned categories, typically achieving higher prediction success rates for more recently learned categories. This advantage can be attributed to the continuously updated prototypes, which play a crucial role in mitigating forgetting and facilitating effective knowledge transfer.

Furthermore, we visualize the test accuracies of SCPD on each task across different training phases using the S-MNIST dataset, as illustrated in Fig. 3. It can be seen that SCPD exhibits a significantly smaller decline in accuracy on earlier tasks as training progresses, demonstrating its strong ability to retain previously acquired knowledge. Overall, these findings validate that our method offers a superior performance in continual learning by effectively balancing stability and plasticity.

4.3.2. Effect visualization using t-SNE

To better understand the effects facilitated by our proposed SCPD, we employ the t-SNE algorithm to visualize the embedding features of the MNIST and CIFAR10 test data in Fig. 4. We employ SCPD to train prototype classifiers and utilize them to embed the entire test set. It can be seen that the embedded features of these datasets are well separated, exhibiting good inter-cluster separation and intra-cluster compactness.

In addition, to investigate the role of the proposed PDL in alleviating forgetting, we visualize the t-SNE clustering of feature embeddings obtained by SCPD with and without PDL on ten classes from the CIFAR-100 dataset, as shown in Fig. 5. It can be observed that without PDL, the model tends to forget earlier-learned classes as training progresses, leading to overlapping or dispersed feature clusters. In contrast, the integration of PDL enables SCPD to better preserve class-specific representations, resulting in improved intra-class compactness and inter-class separation. These results demonstrate the effectiveness of PDL in tackling catastrophic forgetting.

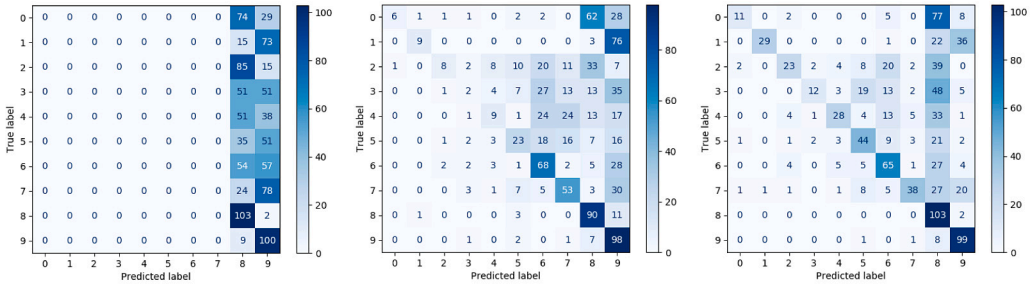


Fig. 2. Confusion matrices showing the classification performance on 10 classes after training on 5 tasks of Split-CIFAR10 using the finetune method, CoPE method, and SCPD-r method. The vertical axis represents the true labels of the test data, the horizontal axis represents the predicted labels of the test data, and darker colors indicate a higher number of test samples that match the current predicted-true label pairs.

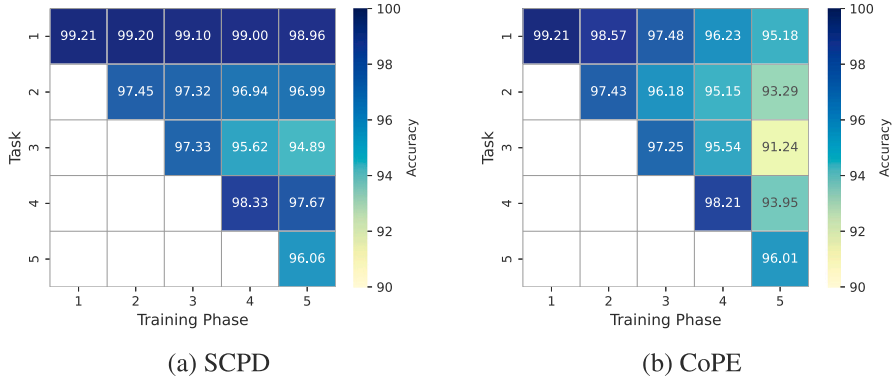


Fig. 3. Heatmap of the SCPD and CoPE results across all training phases on S-MNIST.

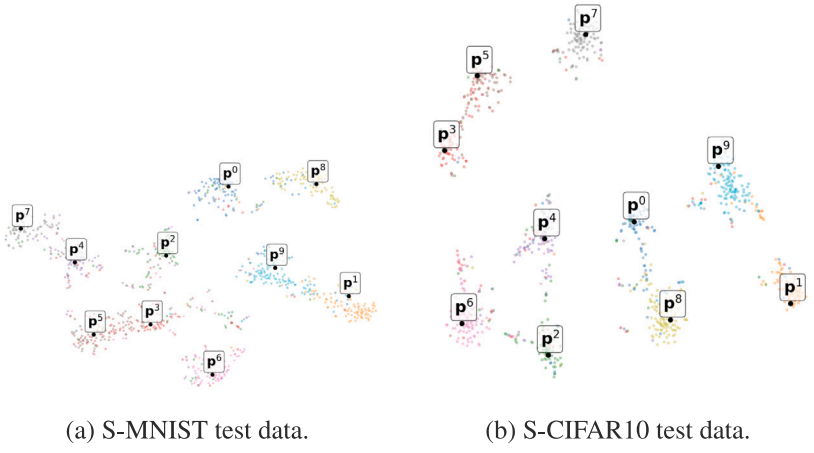


Fig. 4. Visualization of test data embedded features obtained by SCPD learned prototype classifiers on S-MNIST and S-CIFAR10 datasets using t-SNE, respectively.

Fig. 5. t-SNE visualization of embedding features learned by SCPD with and without PDL on CIFAR-100.

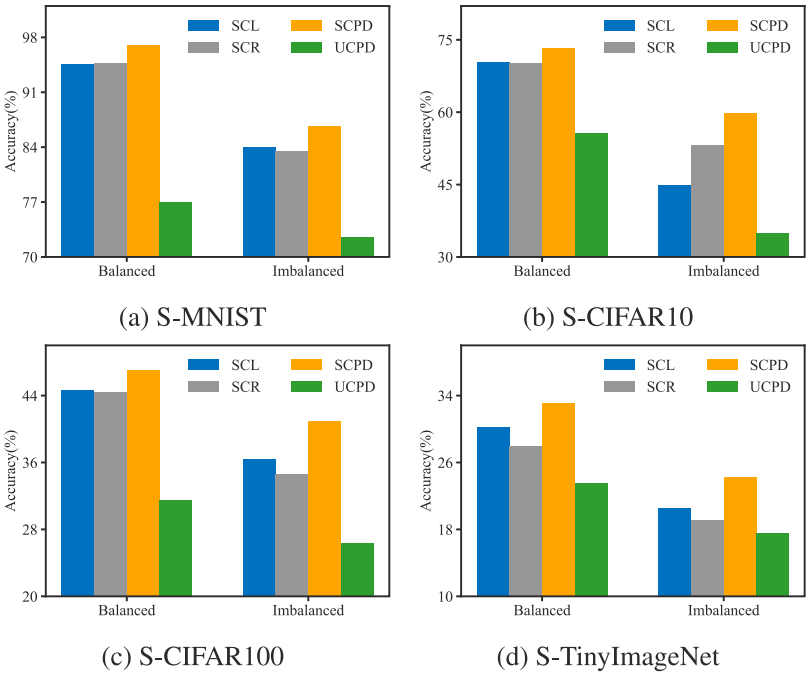


Fig. 6. The results of the ablation study across various benchmark datasets.

4.3.3. Ablation study

To explore the effect of key modules in SCPD, we design the following ablation baselines. The *SCL* variant removes the prototype distillation loss from SCPD, while the *SCR* variant further removes both the prototype distillation loss and the prototype momentum update mechanism. Additionally, to examine the impact of supervised contrastive learning, we introduce the *UCPD* variant, which replaces the supervised contrastive loss with an unsupervised contrastive loss. To ensure fairness, all methods are implemented with the same data augmentation strategies and parameter settings in our experiments.

The results are shown in Fig. 6. By comparing the results of SCL and SCPD, we can analyze the impact of prototype distillation loss. In balanced settings, SCPD outperforms SCL in all datasets by up to 3% in accuracy, and in imbalanced settings, SCPD surpasses SCL in all datasets by up to 15%. The results demonstrate the effectiveness of the prototype distillation loss, especially in the imbalance setting. This is because the model in imbalanced settings is more likely to forget categories containing fewer samples.

Moreover, the accuracy of SCR is generally lower than SCL in most cases, which indicates the importance of prototype momentum updating to improve model accuracy. Therefore, prototype distillation loss has proved beneficial in enhancing the performance of SCPD, and it could work better when combined with the prototype momentum update.

In addition, unlike supervised learning, unsupervised contrastive learning cannot leverage class label information during training. As a result, the learned representations, while robust, offer limited benefits for downstream classification tasks. Consequently, the accuracy of UCPD is significantly lower than SCPD, highlighting the effectiveness of supervised contrastive learning in enhancing the discriminative power and adaptability of SCPD.

4.3.4. Further analysis of parameters

We further investigate the impact of four parameters, memory capacity M, the observed batch size N_b, replay buffer memory size N_s, and momentum factor α on model performance.

Memory Capacity M. Theoretically, a larger memory capacity M stores more exemplars, resulting in a distribution of exemplars more closely resembling the distribution of the data stream S. We compare different replay memory sizes with the results presented in Fig. 7. It can be observed that memory size is positively correlated with final performance, which is consistent with our hypothesis. The results on the S-CIFAR10 dataset are heavily affected by memory capacity, especially on the setups of $M = 100$ and $M = 200$, which indicates that the two setups are not suitable for the dataset. Similarly, $M = 100$ is also slightly small for the S-CIFAR100 dataset.

Batch Size N_b. Previous work (Lange & Tuytelaars, 2021) points out that accuracy decreases with the increase of batch size, and we observe a similar trend in SCPD's performance across different batch sizes. The results are shown in Table 3. Considering the models' training iterations in the setups of $N_b = 100$ and $N_b = 200$ are decreased by a large margin, we increase the training epochs to 10 for each training batch to improve the results. In Table 3, the $N_b = 10$ (online) indicates that the model is trained only once. Our findings show that both online training and larger batch sizes reduce SCPD's performance due to fewer training iterations.

Replay Buffer Memory Size N_s. In the last column of Table 3, we present the test accuracy with $N_s = 300$. Compared with the results with $N_s = 100$ in Table 1, the accuracy experiences a slight variance, indicating high stability of the parameter N_s.

Momentum Factor α. We compare the results of SCPD on CIFAR-100 and S-MNIST datasets at α from 0.3 to 1, and the results are shown in Fig. 8. It can be seen that the appropriate value of α is generally between 0.85 and 0.99; in addition, the results of SCPD under unbalanced settings are more sensitive to α.

4.3.5. Further analysis of loss weights

In our proposed method, the SCL and PDL losses are directly combined without any weighting scheme. To investigate the model's sensitivity to each loss component, we perform an ablation study by varying the weight of either SCL or PDL while keeping the other fixed. As illustrated in Fig. 9, increasing the weight of either loss term results in a gradual decline in overall model accuracy. Specifically, placing greater emphasis on SCL encourages the model to learn more discriminative representations, but this comes at the expense of increased forgetting of previously learned knowledge. In contrast, assigning a higher weight to PDL enhances knowledge retention, yet limits the model's adaptability, impairing its ability to effectively learn new information.

Table 3
The accuracy of SCPD on S-CIFAR10 and S-CIFAR100 datasets using different parameter settings. The $N_b = 10$ (online) represents that the model is solely trained once, i.e., trained online.

Datasets	$N_b = 10$ (online)	$N_b = 20$	$N_b = 50$	$N_b = 100$	$N_b = 200$	$N_s = 300$
S-CIFAR10	65.4 ± 3.2	70.5 ± 2.2	64.2 ± 4.4	67.1 ± 2.4	65.0 ± 2.7	73.1 ± 2.2
S-CIFAR100	31.6 ± 2.1	43.1 ± 1.3	35.3 ± 1.0	28.3 ± 1.4	26.6 ± 1.6	46.7 ± 0.7

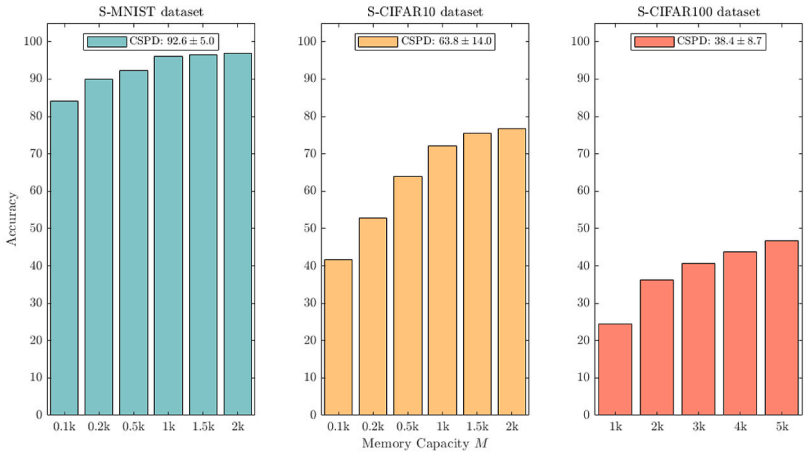


Fig. 7. The results of SCPD over different memory sizes M on S-MNIST, S-CIFAR10, and S-CIFAR100 benchmarks.

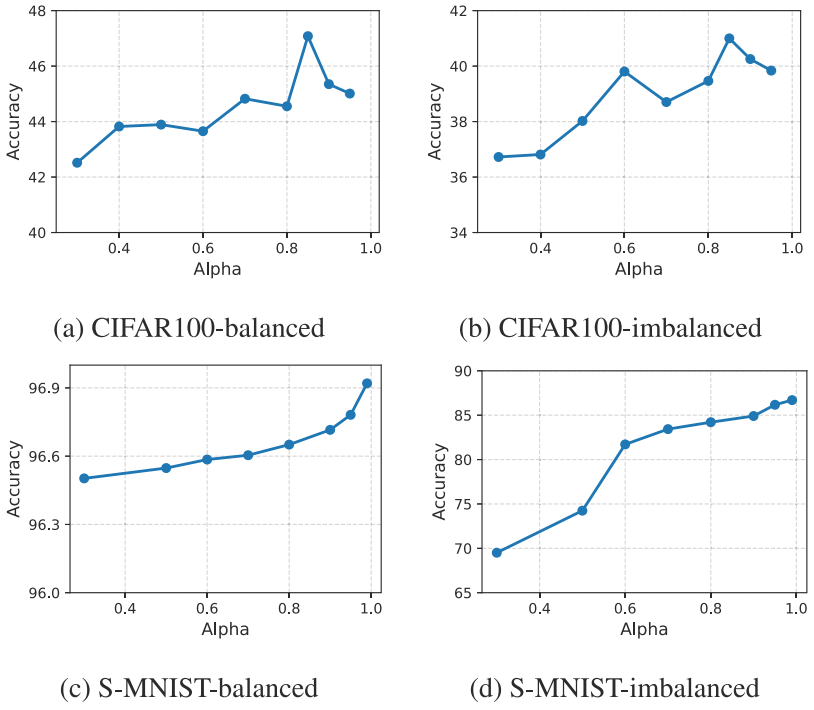


Fig. 8. The results of SCPD over different momentum factor α on the balanced and imbalanced S-MNIST, S-CIFAR100 benchmarks.

4.3.6. Further analysis of training efficiency

To evaluate the training costs associated with each component of SCPD, we conduct a controlled comparison under four experimental settings: (1) the full SCPD method, (2) a variant using only the Supervised Contrastive Loss (SCL), (3) a variant using only the Prototype Distillation Loss (PDL), and (4) a baseline model trained with standard Cross-Entropy (CE) loss in place of our proposed loss functions. All models are trained for the same number of epochs on identical datasets to ensure a fair and consistent comparison.

As shown in Fig. 10, the training costs introduced by PDL are minimal, only a 1% increase compared to the SCL and SCPD settings. This confirms the computational efficiency of the PDL component. On the other hand, the comparison between the SCL and CE groups indicates a notable increase in training time when using SCL. This overhead is primarily attributed to the pairwise similarity computations, where each sample is transformed into two distinct views to enable contrastive learning. These findings suggest that SCL improves representational learning at a modest computational cost.

4.3.7. Further analysis on sampling strategy

A common strategy for addressing data imbalance in machine learning is weighted sampling. To assess the applicability of this approach

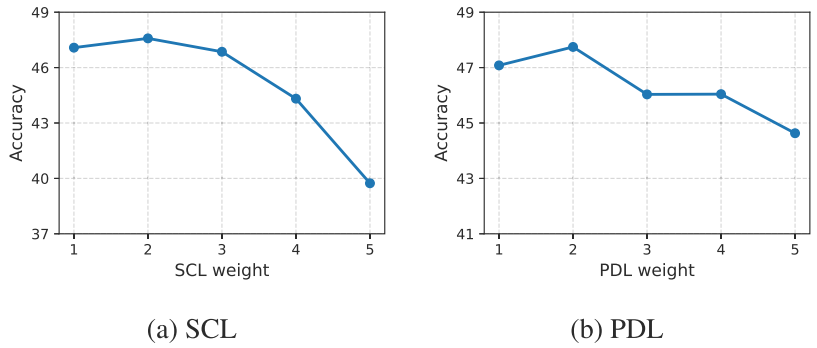


Fig. 9. Performance of SCPD under different combinations of loss weights on the CIFAR-100 dataset.

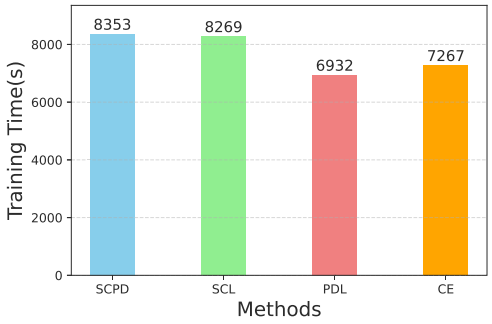


Fig. 10. Training efficiency analysis of SCPD.

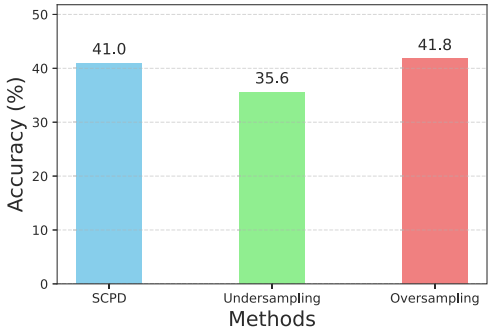


Fig. 11. SCPD performance under different sampling strategies.

within SCPD, we design a set of control experiments on the CIFAR-100 dataset. Specifically, we evaluate three settings: (1) the original SCPD, (2) a variant that undersamples classes with more samples, and (3) a variant that oversamples classes with fewer samples. The results are presented in Fig. 11.

The results indicate that both undersampling and oversampling offer minimal improvement or even lead to a decrease in overall accuracy. In the case of oversampling, the marginal gain is limited due to the lack of genuinely new information being introduced. Conversely, undersampling significantly reduces the number of training examples for majority classes, which hampers the model's ability to learn robust representations for those classes and consequently degrades overall performance. Given that incremental learning tasks typically involve a large number of classes with relatively balanced sample distributions, weighted sampling strategies should be applied with caution to avoid compromising model accuracy across the majority of classes.

4.3.8. Further analysis on replay buffer construction

In SCPD, replay samples are randomly selected. The main advantage of random selection lies in its computational efficiency and its ability to approximate the original data distribution. Nevertheless, many alternative strategies for constructing the replay buffer have been proposed in the incremental learning literature. In this section, we explore the impact of two different approaches using CIFAR-100, i.e., data selection method, Herding (Rebuffi et al., 2017), and data compression method, MRDC (Wang et al., 2022), which enables more replay samples to be stored per class under the same memory budget. The results are shown in Fig. 12. It is indicated that both Herding and MRDC improve the accuracy of SCPD, with MRDC bringing the best results. This improvement can be attributed to its ability to store a larger number of informative samples through compression, thereby enhancing the effectiveness of replay in mitigating forgetting.

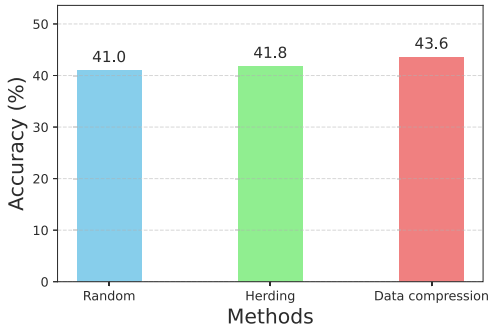


Fig. 12. Performance comparison of SCPD using different replay sample construction strategies.

5. Discussion and future work

In this section, we discuss some potential limitations and future work for our method. Following the standard closed-set assumption commonly adopted in incremental learning literature (Lange et al., 2021), where all incoming data are assumed to belong to one of the previously encountered categories. This may overlook real-world scenarios where ambiguous or unclassifiable samples may arise. Future work could extend our framework to handle such cases by incorporating uncertainty modeling or open-set recognition techniques.

6. Conclusions

In this paper, we propose a novel method named Supervised Contrastive learning with Prototype Distillation (SCPD) to address the data incremental learning. Our approach integrates the supervised contrastive loss and prototype distillation loss to alleviate the issue of catastrophic forgetting and extract more divisible representations. The supervised contrastive loss is effective for assembling embeddings of the same class tightly and, more importantly, for maintaining separation between heterogeneous samples, an aspect not considered by the

conventional cross-entropy loss. Prototype distillation helps the model resist the forgetting problem since it maintains the embeddings of exemplars closer to their corresponding prototypes. Through experiments on different balanced and imbalanced setups across various datasets, SCPD has proven an effective and robust method for the DIL problem.

However, the input features could still exhibit some drift in the embedding space of some datasets. Future research will focus on a better approach to capture the data distribution and prevent embedding drift after incorporating new classes.

CRediT authorship contribution statement

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

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.

Data availability

Data used in our work are all publicly available.

References

Aljundi, Rahaf, Belilovsky, Eugene, Tuytelaars, Tinne, Charlin, Laurent, Caccia, Massimo, Lin, Min, et al. (2019). Online continual learning with maximal interfered retrieval. In *Advances in neural information processing systems 32* (pp. 11849–11860).

Aljundi, Rahaf, Lin, Min, Goujaud, Baptiste, & Bengio, Yoshua (2019). Gradient based sample selection for online continual learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, & Roman Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 11816–11825).

Arani, Elahe, Sarfraz, Fahad, & Zonooz, Bahram (2022). Learning fast, learning slow: A general continual learning method based on complementary learning system. In *International conference on learning representations*.

Cha, Hyuntak, Lee, Jaeho, & Shin, Jinwoo (2021). Co^{2}L: Contrastive continual learning. In *2021 IEEE/CVF international conference on computer vision, ICCV 2021, montreal, QC, Canada, October 10-17, 2021* (pp. 9496–9505). IEEE.

De Lange, Matthias, Aljundi, Rahaf, Masana, Marc, Parisot, Sarah, Jia, Xu, Leonardis, Aleš, et al. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3366–3385.

Ghunaim, Yasir, Bibi, Adel, Alhamoud, Kumail, Alfarra, Motasem, Al Kader Hammoud, Hasan Abed, Prabhu, Ameya, et al. (2023). Real-time evaluation in online continual learning: A new hope. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11888–11897).

Guo, Yiduo, Liu, Bing, & Zhao, Dongyan (2022). Online continual learning through mutual information maximization. In *Proceedings of machine learning research: vol. 162, International conference on machine learning, ICML 2022, 17-23 July 2022, baltimore, maryland, USA* (pp. 8109–8126). PMLR.

nan Han, Ya, & wei Liu, Jian (2024). Adaptive instance similarity embedding for online continual learning. *Pattern Recognition*, *149*, Article 110238.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, & Sun, Jian (2016). Deep residual learning for image recognition. In *2016 IEEE conference on computer vision and pattern recognition, CVPR 2016, las vegas, NV, USA, June 27-30, 2016* (pp. 770–778). IEEE Computer Society.

Horiguchi, Shota, Ikami, Daiki, & Aizawa, Kiyoharu (2020). Significance of softmax-based features in comparison to distance metric learning-based features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(5), 1279–1285.

Hu, Hanzhe, Cui, Jinshi, & Wang, Liwei (2021). Region-aware contrastive learning for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 16291–16301).

Hu, Xinting, Tang, Kaihua, Miao, Chunyan, Hua, Xian-Sheng, & Zhang, Hanwang (2021). Distilling causal effect of data in class-incremental learning. In *IEEE conference on computer vision and pattern recognition, CVPR 2021, virtual, June 19-25, 2021* (pp. 3957–3966).

Jaiswal, Ashish, Babu, Ashwin Ramesh, Zadeh, Mohammad Zaki, Banerjee, Debapriya, & Makedon, Fillia (2020). A survey on contrastive self-supervised learning. CoRR, abs/2011.00362.

Ji, Cheng, Huang, Zixuan, Sun, Qingyun, Peng, Hao, Fu, Xingcheng, Li, Qian, et al. (2024). Regcl: rethinking message passing in graph contrastive learning. *vol. 38*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 8544–8552).

Jiang, Jian, & Celiktutan, Oya (2023). Neural weight search for scalable task incremental learning. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 1390–1399).

Jodelet, Quentin, Liu, Xin, & Murata, Tsuyoshi (2022). Balanced softmax cross-entropy for incremental learning with and without memory. *Computer Vision and Image Understanding*, *225*, Article 103582.

Khosla, Prannay, Teterwak, Piotr, Wang, Chen, Sarna, Aaron, Tian, Yonglong, Isola, Phillip, et al. (2020). Supervised contrastive learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, Hsuan-Tien Lin (Eds.), *Advances in neural information processing systems 33*.

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

Lange, Matthias De, Aljundi, Rahaf, Masana, Marc, Parisot, Sarah, Jia, Xu, Leonardis, Ales, et al. (2022). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*(7), 3366–3385.

Lange, Matthias De, & Tuytelaars, Tinne (2021). Continual prototype evolution: Learning online from non-stationary data streams. In *2021 IEEE/CVF international conference on computer vision, ICCV 2021, montreal, QC, Canada, October 10-17, 2021* (pp. 8230–8239). IEEE.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, & Haffner, Patrick (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, Soochan, Ha, Junsoo, Zhang, Dongsu, & Kim, Gunhee (2020). A neural Dirichlet process mixture model for task-free continual learning. In *8th international conference on learning representations, ICLR 2020, addis ababa, ethiopia, April 26-30, 2020*. OpenReview.net.

Lee, Hojun, Lee, Myunggi, & Kwak, Nojun (2022). Few-shot object detection by attending to per-sample-prototype. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision* (pp. 2445–2454).

Li, Xiaorong, Wang, Shipeng, Sun, Jian, & Xu, Zongben (2023). Memory efficient data-free distillation for continual learning. *Pattern Recognition*, *144*, Article 109875.

Liang, Yan-Shuo, & Li, Wu-Jun (2024). Loss decoupling for task-agnostic continual learning. *Advances in Neural Information Processing Systems*, *36*.

Liang, Dong, Li, Ling, Wei, Mingqiang, Yang, Shuo, Zhang, Liyan, Yang, Wenhan, et al. (2022). Semantically contrastive learning for low-light image enhancement. *Vol. 36*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1555–1563).

Liu, Chong, Wang, Yi, Li, Dong, & Wang, Xizhao (2024). Domain-incremental learning without forgetting based on random vector functional link networks. *Pattern Recognition*, *151*, Article 110430.

Lopez-Lopez, Eric, Pardo, Xose M., & Regueiro, Carlos V. (2022). Incremental learning from low-labelled stream data in open-set video face recognition. *Pattern Recognition*, *131*, Article 108885.

Lopez-Paz, David, & Ranzato, Marc'Aurelio (2017). Gradient episodic memory for continual learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, & Roman Garnett (Eds.), *Advances in neural information processing systems 30* (pp. 6467–6476).

Mai, Zheda, Li, Ruiwen, Kim, Hyunwoo, & Sanner, Scott (2021). Supervised contrastive replay: Revisiting the nearest class mean classifier in online class-incremental continual learning. In *IEEE conference on computer vision and pattern recognition workshops, CVPR workshops 2021, virtual, June 19-25, 2021* (pp. 3589–3599).

Mai, Sijie, Zeng, Ying, & Hu, Haifeng (2023). Learning from the global view: Supervised contrastive learning of multimodal representation. *Information Fusion*, *100*, Article 101920.

Mallya, Arun, & Lazebnik, Svetlana (2018). PackNet: Adding multiple tasks to a single network by iterative pruning. In *2018 IEEE conference on computer vision and pattern recognition, CVPR 2018, salt lake city, UT, USA, June 18-22, 2018* (pp. 7765–7773).

Masana, Marc, Liu, Xialei, Twardowski, Bartlomiej, Menta, Mikel, Bagdanov, Andrew D., & van de Weijer, Joost (2020). Class-incremental learning: survey and performance evaluation. CoRR, abs/2010.15277.

Mirzadeh, Seyed Iman, Farajtabar, Mehrdad, Pascanu, Razvan, & Ghasemzadeh, Hassan (2020). Understanding the role of training regimes in continual learning. *vol. 33*, In *Advances in neural information processing systems* (pp. 7308–7320).

Pan, Zicheng, Zhang, Weichuan, Yu, Xiaohan, Zhang, Miaohua, & Gao, Yongsheng (2024). Pseudo-set frequency refinement architecture for fine-grained few-shot class-incremental learning. *Pattern Recognition*, Article 110686.

Parisi, German Ignacio, Kemker, Ronald, Part, Jose L., Kanan, Christopher, & Wermter, Stefan (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*, *113*, 54–71.

Rahman, Mohammad Saidur, Coull, Scott, & Wright, Matthew (2022). On the limitations of continual learning for malware classification. In *Conference on lifelong learning agents* (pp. 564–582). PMLR.

Rebuffi, Sylvestre-Alvise, Kolesnikov, Alexander, Sperl, Georg, & Lampert, Christoph H. (2017). iCaRL: Incremental classifier and representation learning. In *2017 IEEE conference on computer vision and pattern recognition, CVPR* (pp. 5533–5542). IEEE Computer Society.

Schwarz, Jonathan, Czarnecki, Wojciech, Luketina, Jelena, Grabska-Barwinska, Agnieszka, Teh, Yee Whye, Pascanu, Razvan, et al. (2018). Progress & compress: A scalable framework for continual learning. In *International conference on machine learning* (pp. 4528–4537). PMLR.

Shi, Haizhou, & Wang, Hao (2024). A unified approach to domain incremental learning with memory: Theory and algorithm. *Advances in Neural Information Processing Systems*, *36*.

Shorten, Connor, & Khoshgoftaar, Taghi M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, *6*, 60.

Song, Jialun, Chen, Jian, & Du, Lan (2024). Rebalancing network with knowledge stability for class incremental learning. *Pattern Recognition*, *153*, Article 110506.

Tian, Songsong, Li, Lusi, Li, Weijun, Ran, Hang, Ning, Xin, & Tiwari, Prayag (2024). A survey on few-shot class-incremental learning. *Neural Networks*, *169*, 307–324.

Tian, Yuan, Lu, Guo, Yan, Yichao, Zhai, Guangtao, Chen, Li, & Gao, Zhiyong (2024). A coding framework and benchmark towards low-bitrate video understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Tian, Yuan, Yan, Yichao, Zhai, Guangtao, Chen, Li, & Gao, Zhiyong (2023). Clsa: a contrastive learning framework with selective aggregation for video rescaling. *IEEE Transactions on Image Processing*, *32*, 1300–1314.

Vinyals, Oriol, Blundell, Charles, Lillicrap, Timothy, Wierstra, Daan, et al. (2016). Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, *29*.

Vitter, Jeffrey Scott (1985). Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, *11*(1), 37–57.

Wang, Peng, Han, Kai, Wei, Xiu-Shen, Zhang, Lei, & Wang, Lei (2021). Contrastive learning based hybrid networks for long-tailed image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 943–952).

Wang, Chenyang, Jiang, Junjun, Hu, Xingyu, Liu, Xianming, & Ji, Xiangyang (2025). Enhancing consistency and mitigating bias: A data replay approach for incremental learning. *Neural Networks*, *184*, Article 107053.

Wang, Liyuan, Zhang, Xingxing, Su, Hang, & Zhu, Jun (2024). A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wang, Liyuan, Zhang, Xingxing, Yang, Kuo, Yu, Longhui, Li, Chongxuan, Hong, Lanqing, et al. (2022). Memory replay with data compression for continual learning. arXiv preprint arXiv:2202.06592.

Wei, Yujie, Ye, Jiaxin, Huang, Zhizhong, Zhang, Junping, & Shan, Hongming (2023). Online prototype learning for online continual learning. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 18764–18774).

Xiong, Xin, Wang, Xiangyu, Yang, Suorong, Shen, Furao, & Zhao, Jian (2025). Gmni: achieve good data augmentation in unsupervised graph contrastive learning. *Neural Networks*, *181*, 106804.

Xu, Zhiming, Yang, Suorong, Xu, Baile, Zhao, Jian, & Shen, Furao (2024). Integrating dual prototypes for task-wise adaption in pre-trained model-based class-incremental learning. arXiv preprint arXiv:2411.17766.

Yang, Suorong, Li, Peijia, Xiong, Xin, Shen, Furao, & Zhao, Jian (2024). Adaaugment: a tuning-free and adaptive approach to enhance data augmentation. arXiv preprint arXiv:2405.11467.

Yang, Suorong, Shen, Furao, & Zhao, Jian (2024). EntAugment: Entropy-driven adaptive data augmentation framework for image classification. In *European conference on computer vision* (pp. 197–214). Springer.

Yang, Zhiwei, Wei, Yuecen, Li, Haoran, Li, Qian, Jiang, Lei, Sun, Li, et al. (2024). Adaptive differentially private structural entropy minimization for unsupervised social event detection. In *Proceedings of the 33rd ACM international conference on information and knowledge management* (pp. 2950–2960).

Yang, Suorong, Xiao, Weikang, Zhang, Mengchen, Guo, Suhan, Zhao, Jian, & Shen, Furao (2022). Image data augmentation for deep learning: A survey. arXiv preprint arXiv:2204.08610.

Zhang, Yifan, Kang, Bingyi, Hooi, Bryan, Yan, Shuicheng, & Feng, Jiashi (2023). Deep long-tailed learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(9), 10795–10816.

Zhou, Da-Wei, Wang, Fu-Yun, Ye, Han-Jia, Ma, Liang, Pu, Shiliang, & Zhan, De-Chuan (2022). Forward compatible few-shot class-incremental learning. In *IEEE/CVF conference on computer vision and pattern recognition, CVPR* (pp. 9036–9046). IEEE.

Zhu, Fei, Zhang, Xu-Yao, Wang, Chuang, Yin, Fei, & Liu, Cheng-Lin (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).