Neural Networks 190 (2025) 107634

Contents lists available at ScienceDirect

Neural Networks

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

Full Length Article

Prototypes as Anchors: Tackling *Unseen Noise* for online continual learning

Shao-Yuan Li [a,b,c], Yu-Xiang Zheng [a], Sheng-Jun Huang [a], Songcan Chen [a], Kangkan Wang [d,*]

[a] *MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China*

[b] *State Key Lab. for Novel Software Technology, Nanjing University, Nanjing, 211106, PR China*

[c] *Joint Laboratory of Spatial Intelligent Perception and Large Model Application, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, PR China*

[d] *School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, 210094, PR China*

ARTICLE INFO

Keywords:
Continual learning
Noisy label
Open-set noise

ABSTRACT

In the context of online class-incremental continual learning (CIL), adapting to label noise becomes paramount for model success in evolving domains. While some continual learning (CL) methods have begun to address noisy data streams, most assume that the noise strictly belongs to closed-set noise—i.e., they follow the assumption that noise in the current task originates classes within the same task. This assumption is clearly unrealistic in real-world scenarios. In this paper, we first formulate and analyze the concepts of *closed-set* and *open-set* noise, showing that both types can introduce *unseen classes* for the current training classifier. Then, to effectively handle noisy labels and unknown classes, we present an innovative replay-based method Prototypes as Anchors (PAA), which learns representative and discriminative prototypes for each class, and conducts a similarity-based denoising schema in the representation space to distinguish and eliminate the negative impact of unseen classes. By implementing a dual-classifier architecture, PAA conducts consistency checks between the classifiers to ensure robustness. Extensive experimental results on diverse datasets demonstrate a significant improvement in model performance and robustness compared to existing approaches, offering a promising avenue for continual learning in dynamic, real-world environments.

1. Introduction

Class-incremental continual learning (CIL) (Zhou et al., 2023) is a specialized area within the broader domain of continual learning (CL) (Lange et al., 2021). It focuses on the dynamic acquisition of knowledge where new classes are incrementally introduced over time. In some real-world applications like recommender systems and search engine optimization, necessitated by privacy protocols or big data the challenges, new classes of data typically arrive in an *online data streams* manner, leading to the online CIL research field (Aljundi & Lucas, 2019; Chaudhry et al., 2019; Shim et al., 2021).

One primary challenge of CIL is avoiding catastrophic forgetting, where new class knowledge overwrites previously learned old class information. Maintaining a delicate equilibrium between old and new knowledge is crucial. Besides, the online data processing constraints require the model to adapt to the current task's data stream efficiently. To solve the dual requirements of rapid adaptability and memory retention, various CIL studies have been proposed (Wu et al., 2019; Yan, Xie, & He, 2021; Zhu, Cheng, yao Zhang, & lin Liu, 2021), focusing on recovering the knowledge forgotten by the model to the greatest extent through data augmentation, knowledge distillation, or dynamic architectures. For example, Yan et al. (2021) fixed the features for the previous stage and applied new feature extractors for the incoming tasks. Zhu et al. (2021) generated 'mixed classes' by mixing between samples, expecting the representation to adapt to more unseen classes. However, they largely assumed pristine, error-free training data, a simplification that can limit the utility of CL in complex real-world scenarios.

In dynamic environments, models are often exposed to noisy data where instances may be incorrectly labeled or belong to unknown classes. As shown in Fig. 1, we formulate online CIL in three scenarios with clean and noisy labels. Assume we have in total T sequential tasks, with task t at timestamp t concerning a specific class set C_t, different tasks have no overlap in their classes. Take the training data D_t for task t as an example, it concerns classes $C_t = \{`cat', `dog'\}$. The top subfigure shows CIL with error-free training data within each task, i.e., each sample in D_t either belongs to class 'cat' or 'dog', and is correctly labeled. This is the focus of most existing works. The medium subfigure illustrates the case we call CIL with *closed-set* noise, i.e., some

* Corresponding author.
E-mail addresses: lisy@nuaa.edu.cn (S.-Y. Li), zhengyx@nuaa.edu.cn (Y.-X. Zheng), huangsj@nuaa.edu.cn (S.-J. Huang), s.chen@nuaa.edu.cn (S. Chen), wangkangkan@njust.edu.cn (K. Wang).

https://doi.org/10.1016/j.neunet.2025.107634
Received 29 July 2024; Received in revised form 5 May 2025; Accepted 14 May 2025
Available online 19 June 2025

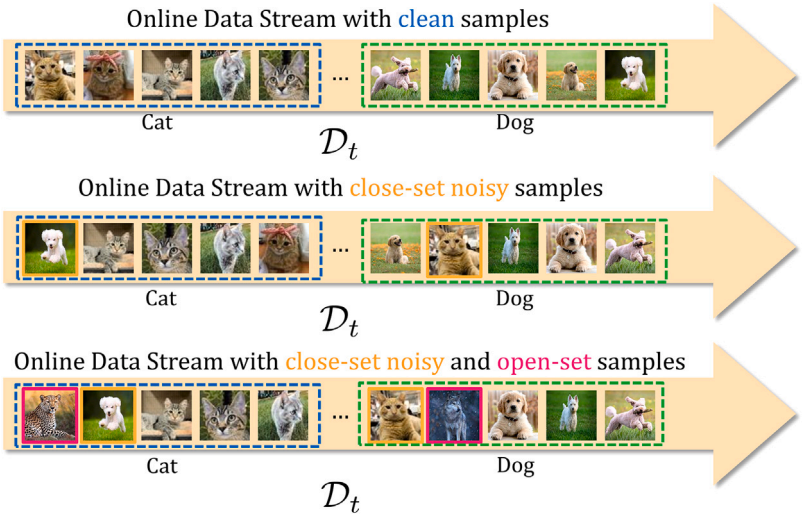

Fig. 1. Comparison of three online CIL scenarios: (Top) Each task consists of correctly labeled samples. (Mid) Each task contains*closed-set* noisy samples. (Bottom) Each task contains both *closed-set* and *open-set* noisy samples.

sample $x \in D_t$ is wrongly labeled as 'cat' or 'dog', with its true class belonging to the union classes of all tasks $\cup_{t=1}^{T} C_t$. The bottom subfigure shows the *mixed noise* case, where *open-set* noise also occurs, i.e., some sample $x \in D_t$ is wrongly labeled as 'cat' or 'dog', and its true class is out of the union classes of all tasks $\cup_{t=1}^{T} C_t$. Such *open-set* noise is inevitable in many real-world scenarios, e.g., when crawling data from the web, searching for "cat" images as training samples but coming up with the open-set samples of "leopard".

It is noteworthy that in the context of online CIL, the *closed-set* and *open-set* concepts are defined by whether the sample falls within the union class range of all tasks $\cup_{t=1}^{T} C_t$. Thus the *closed-set* noisy sample $x \in D_t$ can be either *seen closed-set* noise with its true class in $\cup_{ti=1}^{t} C_{ti}$, or *unseen closed-set* noise from future unseen classes $\cup_{ti=t+1}^{T} C_{ti}$.

To our knowledge, very few studies (Bang et al., 2022; Karim, Khalid, Esmaeili, & Rahnavard, 2022; Kim, Jeong, Moon, & Kim, 2021) have considered the issue of CIL under contaminated data streams. Kim et al. (2021) employed self-supervised learning to create a purified buffer and discarded the potentially noisy samples. To ensure the participation of all available samples, Karim et al. (2022) proposed a masking-based distance metric to detect the noisy samples, and generated pseudo-labels for them through the model's prediction. Bang et al. (2022) considered online CIL from the blurry corrupted data stream, proposing sampling diverse exemplars with a label purifying scheme, which splits the noisy samples based on the small-loss trick and utilizes them by semi-supervised learning. Without explicitly claiming the noise type they address, the above works' implementations are limited to the traditional *seen closed-set* noise. When encountering *unseen closed-set* or *open-set* noisy samples, they will predict them as seen classes and be harmful. Effective management of *unseen* noise is paramount for ensuring model success within the real-world online CIL.

To tackle the noisy labels and unknown classes for the online CIL scenario, we propose an innovative replay-based approach named Prototypes as Anchors (PAA) to effectively reduce catastrophic forgetting and distinguish between known and unknown data instances. Specifically, PAA learns robust representative and discriminative online prototype descriptions for each class. Treating them as anchors, we compute the similarities between the samples to them and use them to guide differentiating between the clean and erroneous samples. To avoid the negative influence of noisy samples with *unseen* classes, we further propose a KNN score-based scheme that cleans up the high-confidence samples in the erroneous samples and filters out the low-confidence ones as unseen samples. Later, PAA implements a dual-classifier architecture and imposes consistency regularization between them for better robustness. The robust representation learning with similarity-based denoising dynamically enhances the resilience of the model.

Our contributions are summarized as follows:

- We first formally define the concept of *closed-set* and *open-set* noise in the context of online CIL, for both *unseen* classes can occur and necessitate careful handling. Previous work has completely ignored this aspect.
- We propose a replay-based method PAA, which learns representative and discriminative prototypes for each class and conducts a similarity-based denoising schema to avoid the negative influence of *unseen* classes.
- We conduct experiments on multiple synthetic and real noise benchmarks, showing that PAA significantly outperforms current SOTA and many combinations of continual learning and noisy label learning methods.

2. Related works

2.1. Continual learning

Continual learning methods are primarily categorized into three groups: regularization-based, parameter-isolation-based, and replay-based. (1) The regularization-based approaches, as expounded in Aljundi, Babiloni, Elhoseiny, Rohrbach, and Tuytelaars (2018), Chaudhry, Dokania, Ajanthan, and Torr (2018) and He and Jaeger (2018), implement additional regularization constraints on network parameters to alleviate forgetting. (2) The parameter-isolation-based strategies, detailed in Rusu et al. (2016), Yan et al. (2021) and Zhou, Wang, Ye, and Zhan (2022), counteract forgetting through dynamic parameter allocation or network architecture modifications. (3) Replay-based methods, referenced in Aljundi and Lucas (2019), Buzzega, Boschini, Porrello, Abati, and Calderara (2020), Chaudhry et al. (2019) and Prabhu, Torr, and Dokania (2020), consistently update a memory buffer that archives exemplars from prior tasks. Under online CIL setup, replay-based methods have gained prominence owing to their simplicity and effectiveness. Experience Replay (Chaudhry et al., 2019), for instance, involves random sampling from the buffer. MIR (Aljundi & Lucas, 2019) selects buffer samples by evaluating the interference in loss values. Shim et al. (2021) introduces a novel buffer management concept based on the Shapley value. In recent years, research focus has gradually shifted from designing heuristic memory buffer sampling strategies to creating designs that enable models to rapidly adapt to samples within data streams. OCM (Guo, Liu, & Zhao,

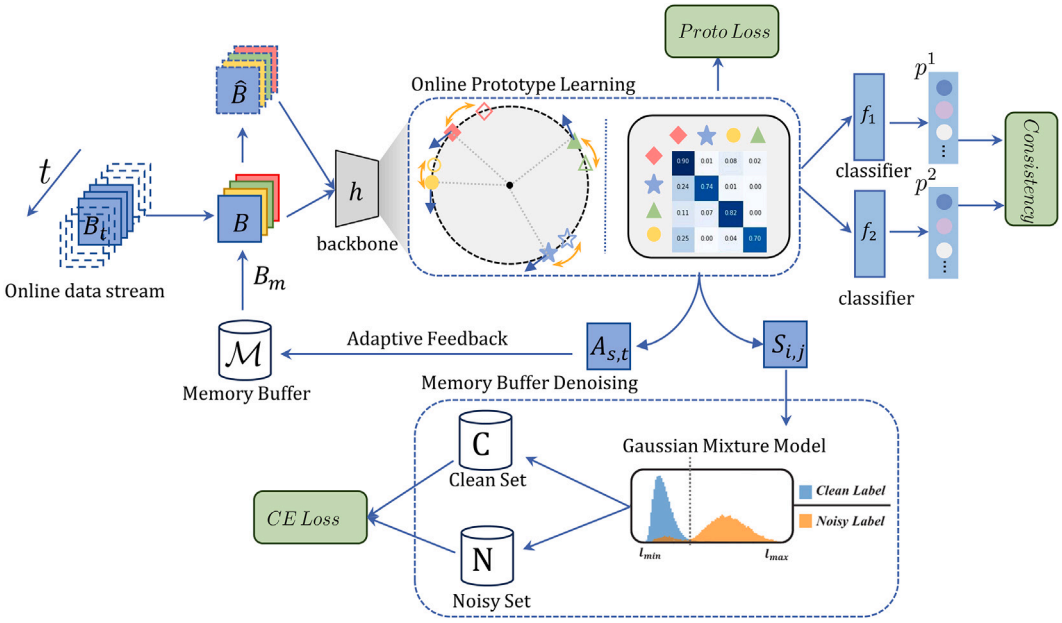

Fig. 2. The learning process of PAA. A memory buffer $\mathcal{M}$ and a delay buffer B_t are kept to store selected data from previous and current tasks. The network is composed of a feature extractor $\boldsymbol{v} = h(\boldsymbol{x};\theta_h)$ followed by a linear projecting head $g(\boldsymbol{v};\theta_g)$ and a dual-classifier $f(\boldsymbol{v};\theta_{f_1},\theta_{f_2})$. PAA adopts an online prototype learning framework to learn representative and discriminative prototypes for new and old classes. These prototypes serve as anchors in the denoising process to distinguish between clean and erroneous samples based on representation similarities. Additionally, a KNN score-based strategy is used to correct high-confidence samples within erroneous ones and discard the *unseen* class samples with low confidence.

2022), OnPro (Wei, Ye, Huang, Zhang, & Shan, 2023), and PCR (Lin, Zhang, Feng, Li, & Ye, 2023) utilize instance contrastive learning or prototype contrastive learning to enhance the model's fitting capability on new data streams.

2.2. Noisy label learning

Learning with noisy labels is an important topic in weakly-supervised learning, attracting much recent interest in several directions. (1) Methods based on sample selection (Bo et al., 2018; Hongxin, Lei, Xiangyu, & Bo, 2020; Li, Socher, & Hoi, 2020). These methods are based on a common starting point, the small-loss trick. in noise learning due to memory utility, resulting in clean samples often being fitted first and noisy samples second. Therefore small-loss can be utilized as a basis for filtering samples. (2) Methods for modeling noise (Lu, Zhengyuan, Thomas, Li-Jia, & Li, 2018; Tongliang & Dacheng, 2015; Xia et al., 2019; Yao et al., 2020). This type of approach models the noise in the dataset based on some assumptions, such as the anchor point assumption, to obtain a noise transfer matrix. Using the noise transfer matrix it is then possible to derive the clean a posteriori probability from the noise a posteriori probability. This method has the guarantee of statistical consistency. (3) Robust loss functions (Liu & Guo, 2020; Yilun, Peng, Yuqing, & Yizhou, 2019; Zhilu & Mert, 2018). The robustness of the model can be improved by using robust losses. However, this approach is often difficult to apply in real-world noisy scenarios.

To improve the robustness of the model to open-set noise samples, Oza and Patel (2019), Sun, Yang, Zhang, Ling, and Peng (2020) and Yoshihashi et al. (2019) learn conditional Gaussian distributions for known classes and then detect open-set noise. Li, Xiong, and Hoi (2021), Lu, Xu, Li, Cheng, and Niu (2022) and Yang, Zhang, Yin, Yang, and Liu (2020) leverage prototype-based contrastive learning for acquiring robust representations. Sun et al. (2022) and Yazhou et al. (2021) exploit the Jensen–Shannon (JS) divergence and prediction disagreement to globally select different types of noisy data.

3. Problem definition

In online CIL, a series of tasks come in a sequential data stream $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_T\}$, with T denoting the total number of tasks. Each task t corresponds to a dataset $\mathcal{D}_t = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{|\mathcal{D}_t|}$ concerning a set of C_t classes. The classes across tasks are disjoint. In practical scenarios, label noise occurs commonly, i.e., for some sample $\boldsymbol{x}_i \in \mathcal{D}_t$, its observed label $\tilde{\boldsymbol{y}}_i$ is potentially erroneous. We formally define the following *closed-set* and *open-set* noise in online CIL:

Definition 1. Closed-set/Open-set noise in online CIL: For some sample $\boldsymbol{x}_i$ of task t wrongly tagged with some label $\tilde{\boldsymbol{y}}_i \in C_t$, we call it as a closed-set noisy sample if its true class $\boldsymbol{y}_i$ falls within the union class range of all tasks, i.e.,

$$(\boldsymbol{x}_i, \tilde{\boldsymbol{y}}_i) \in \mathcal{D}_t, \quad with \quad \boldsymbol{y}_i \in \{\cup_{t=1}^{T} C_t\}. \tag{1}$$

Otherwise, it is called an open-set noisy sample.

Significantly different from single-task scenarios where *closed-set*/*open-set* refer to whether the sample falls within the class range of the current task, in online CIL, the *closed-set* noisy sample $\boldsymbol{x}_i$ of task t can be either *seen closed-set* noise with its true class in (previously) seen tasks, i.e.,

$$(\boldsymbol{x}_i, \tilde{\boldsymbol{y}}_i) \in \mathcal{D}_t, \quad with \quad \boldsymbol{y}_i \in \{\cup_{ti=1}^{t} C_{ti}\}, \tag{2}$$

or *unseen closed-set* noise from future unseen classes:

$$(\boldsymbol{x}_i, \tilde{\boldsymbol{y}}_i) \in \mathcal{D}_t, \quad with \quad \boldsymbol{y}_i \in \{\cup_{ti=t+1}^{T} C_{ti}\}. \tag{3}$$

Without being aware of the essential difference of label noise in online CIL with that in traditional single-task learning, a few works have designed techniques and conducted validations for online CIL with noisy labels, by treating the contaminated samples as traditional *seen closed-set* noise through discarding the potentially noisy samples (Kim et al., 2021), predicting pseudo-labels for them (Karim et al., 2022), or utilizing them in a semi-supervised manner (Bang et al., 2022). When encountering the *unseen closed-set* and *open-set* noisy samples, they

would incorporate them as previously seen classes, which is harmful and misleading.

To our knowledge, we are the first to formally define the types of label noise in online CIL and highlight their difference from single-task learning. To effectively tackle the general label noise from either *seen* or *unseen* classes, in the following we propose our replay-based approach PAA.

4. Methodology: Prototypes as anchors (PAA)

Fig. 2 presents the overall procedure of PAA. The network is composed of a feature extractor $\boldsymbol{v} = h(\boldsymbol{x}; \theta_h)$ followed by a linear projecting head $g(\boldsymbol{v}; \theta_g)$ and a dual-classifier $f(\boldsymbol{v}; \theta_{f_1}, \theta_{f_2})$. A memory buffer $\mathcal{M}$ and a delay buffer B_t are kept to respectively store selected data from previous tasks and streaming data from the current task $\mathcal{D}_t$. In alignment with existing replay-based methods, at each time step, the memory buffer $\mathcal{M}$ is updated by uniformly and randomly incorporating samples from current B_t. Should $\mathcal{M}$ reach its capacity, an equivalent number of samples are selectively removed from it to maintain balance. During training, PAA alternately conducts denoising on the memory buffer $\mathcal{M}$ and efficient online prototype-based representation learning using a mini-batch $B = B_t \cup B_m$ with B_m from the denoised $\mathcal{M}$ through a reservoir sampling strategy (Riemer et al., 2018). Note that the originally observed labels for the delay buffer B_t are used during learning since its samples appear only once. Thus the model on the one hand does not quickly fit the potential noise affected by the memory effect, and on the other hand, having learned only once, it also struggles to reliably detect and refine the noise. In the following, we give implementation details for the online prototype learning and denoising procedure.

Note that we adopt the uniform and random sampling strategy for updating the memory buffer in this paper. The reason behind this choice is that sampling strategies are not the primary focus of this paper. Moreover, complicate sampling algorithms often introduce additional computational complexity. Referencing recent studies, such as OnPro (Wei et al., 2023), which suggest that enhancing the model's ability to fit new tasks is more beneficial for online continual learning than focusing on sophisticated sampling strategies. Therefore, in this paper, we opt for a simpler random sampling approach, which allows us to concentrate on improving the model's adaptability to new tasks without the added computational burden of more complex sampling methods. This approach aligns with our goal of maintaining a balance between computational efficiency and model performance in online continual learning scenarios.

4.1. Online prototype learning with dual-classifier consistency

To foster noise detection and filtering in online CIL, efficiency and computational simplicity are paramount. Therefore in this research, we adopt the online prototype learning framework described by Wei et al. (2023). This approach learns representative and discriminative representation over classes and instances on each batch of the data stream, to prevent class confusion and effectively maintain balance among all seen classes when learning new ones.

Specifically, we conduct representation learning for new and old classes respectively on the delay buffer B_t and the mini-batch B_m sampled from memory buffer $\mathcal{M}$ through prototype-based contrastive learning:

$$\mathcal{L}_{\text{NEW}} = \frac{-1}{|\boldsymbol{\mathcal{P}}|} \sum_{j=1}^{|\boldsymbol{\mathcal{P}}|} \log \frac{\exp\left(\frac{\boldsymbol{p}_j^{\mathrm{T}} \widehat{\boldsymbol{p}}_j}{\tau_1}\right)}{\sum_k \exp\left(\frac{\boldsymbol{p}_j^{\mathrm{T}} \widehat{\boldsymbol{p}}_k}{\tau_1}\right) + \sum_{k \neq j} \exp\left(\frac{\boldsymbol{p}_j^{\mathrm{T}} \boldsymbol{p}_k}{\tau_1}\right)}. \tag{4}$$

$\mathcal{L}_{\text{NEW}}$ aligns the online prototypes $\{\boldsymbol{p}_j\}$ with their augmented view $\{\widehat{\boldsymbol{p}}_j\}$ for each class j in B_t to enhances class-specific feature representation and achieve a balance between feature alignment and class differentiation. Here $\boldsymbol{p}_j$ and $\widehat{\boldsymbol{p}}_j$ respectively denote the prototype and *augmented* prototype of class j in B_t, computed as the normalized mean embedding of samples $\boldsymbol{x}$ and *augmented* samples $aug(\boldsymbol{x})$:

$$\begin{aligned} \boldsymbol{p}_j &= mean\{g(h(\boldsymbol{x}))|(\boldsymbol{x}, \tilde{\boldsymbol{y}}) \in B_t, \tilde{\boldsymbol{y}} = j\}, \quad \boldsymbol{p}_j = \frac{\boldsymbol{p}_j}{\left\|\boldsymbol{p}_j\right\|_2} \\ \widehat{\boldsymbol{p}}_j &= mean\{g(h(aug(\boldsymbol{x})))|(\boldsymbol{x}, \tilde{\boldsymbol{y}}) \in B_t, \tilde{\boldsymbol{y}} = j\}, \quad \widehat{\boldsymbol{p}}_j = \frac{\widehat{\boldsymbol{p}}_j}{\left\|\widehat{\boldsymbol{p}}_j\right\|_2} \end{aligned} \tag{5}$$

This prototype computation on each batch, rather than a more complex, holistic approach, significantly reduces computational complexity while maintaining accuracy. Eq. (4) is computing prototype comparison learning, the prototype is the center of each class, and the number of prototypes is much smaller than the number of samples, so the computational efficiency is very high compared to the comparison learning (from $O(N^2)$ down to $O(C^2)$, with N denoting the number of samples in the buffer and C denoting the number of classes in the buffer). Compared to the method of fitting with only the cross-entropy function, prototype comparison learning allows the model to have a better fit on new data streams without bringing a lot of extra overhead. So it is all about balancing efficiency and accuracy.

Similarly, $\mathcal{L}_{\text{OLD}}$ is adopted on the replay data B_m to maintain knowledge of previously learned classes, ensuring a continuous and balanced understanding of all classes. With $\boldsymbol{p}_c^m$ denoting the prototype for old class c in replay data, $\mathcal{L}_{\text{OLD}}$ contrasts between the samples in each class with the corresponding prototype to achieve compact representation:

$$\mathcal{L}_{\text{OLD}} = \frac{-1}{|B_m|} \sum_{i=1}^{|B_m|} \log \frac{\exp\left(\hat{\boldsymbol{z}}_i \cdot \boldsymbol{p}_{\hat{\boldsymbol{y}}_i}^m / \tau_2\right)}{\sum_{c=1}^{C_{old}} \exp\left(\hat{\boldsymbol{z}}_i \cdot \boldsymbol{p}_c^m / \tau_2\right)}. \tag{6}$$

Here $\hat{\boldsymbol{z}}_i = g(h(aug(\boldsymbol{x}_i)))$ denotes embedding of the *augmented* view of sample $\boldsymbol{x}_i \in B_m$, with its cleansed label being $\hat{\boldsymbol{y}}_i$. In the learning beginning, $\hat{\boldsymbol{y}}_i$ is initialized as the training label $\tilde{\boldsymbol{y}}_i$.

To equally treat old classes and new classes, the replay data B_m are sampled from the memory bank $\mathcal{M}$ adaptively based on the confusion level between different classes:

$$A_{s,t} \propto \exp(-\|\boldsymbol{p}_{cs}^m - \boldsymbol{p}_{ct}^m\|_2^2). \tag{7}$$

$A_{s,t}$ calculates distances between prototypes and transforming them into a probability distribution using a Gaussian kernel. Larger $A_{s,t}$ indicates classes prone to misclassification and guides sample selection in $\mathcal{M}$. A two-stage sampling strategy is employed, with n_{PR} percent samples firstly chosen based on the probability $A_{s,t}$, and the remaining $1 - n_{\text{PR}}$ percent are selected uniformly from $\mathcal{M}$.

Consistency Regularization In an ideal scenario, a model trained to high proficiency should render consistent predictions across permutations of in-distribution (ID) samples, while exhibiting inconsistency with out-of-distribution (OOD) samples. Notably, challenging ID samples, typically proximal to the class boundaries in the feature space, can be discerned through the consistency analysis between two independently trained classifiers, f_1 and f_2, each possessing distinct decision boundaries. By incorporating f_1 and f_2 into our network architecture, we obtain two separate predictions for the same sample x. Subsequently, the consistency between these classifiers on the training sample $\boldsymbol{x}$ in mini-batch $B_t \cup B_m$ is quantified by the following L1 norm measure:

$$\begin{aligned} \mathcal{L}_{\text{CON}}(\boldsymbol{x}) &= \left|f_1(h(\boldsymbol{x})) - f_2(h(\boldsymbol{x}))\right|, \\ \mathcal{L}_{\text{CON}} &= \sum_{\boldsymbol{x} \in B_t \cup B_m} \mathcal{L}_{\text{CON}}(\boldsymbol{x}). \end{aligned} \tag{8}$$

This consistency regularization method not only incrementally enhances the representation learning of the model but also significantly improves its ability to distinguish between ID and OOD noise. Such a methodology is crucial for ensuring robust and discerning performance of the model amidst diverse data types.

Classifier Refinement After each online prototype learning session concludes, we further refine the classifiers f_1 and f_2 through classification loss minimization using the cleansed samples $(\boldsymbol{x}, \hat{\boldsymbol{y}})$ in the memory buffer $\mathcal{M}$:

$$\mathcal{L}_{\mathrm{CE}} = \sum_{(\boldsymbol{x},\hat{\boldsymbol{y}})\in\mathcal{M}} \mathrm{CE}(\hat{\boldsymbol{y}}, 0.5 * [f_1(h(\boldsymbol{x})) + f_2(h(\boldsymbol{x}))]). \tag{9}$$

Note that $\hat{\boldsymbol{y}}$ is the cleansed labeled of $\boldsymbol{x}$ after the denoising process conducted on the memory buffer $\mathcal{M}$. In the beginning, $\hat{\boldsymbol{y}}$ is initialized as observed the training label $\tilde{\boldsymbol{y}}$. Thus, the total loss of our framework is given as:

$$\mathcal{L} = \mathcal{L}_{\mathrm{NEW}} + \mathcal{L}_{\mathrm{OLD}} + \mathcal{L}_{\mathrm{CON}} + \mathcal{L}_{\mathrm{CE}}. \tag{10}$$

4.2. Memory buffer denoising

After the above prototype-based representation learning, we expect the network to maintain the decision boundaries and reduce feature overlaps for distinct classes to a certain extent. To further address the challenge of label-corrupted samples, we propose an additional representation-based denoising process. This enhances the model's ability to distinguish between clean and noisy samples, thereby improving accuracy and robustness.

Specifically, with the updated feature extractor $h(\boldsymbol{x}; \theta_h)$, we compute the similarity $S_{i,j}$ between the feature representation of sample $\boldsymbol{x}_i \in \mathcal{M}$ and the prototypes of each class j of $\mathcal{M}$:

$$S_{i,j} \propto \exp(-\|\boldsymbol{p}_j - g(h(\boldsymbol{x}_i))\|_2^2), \tag{11}$$

where $\boldsymbol{p}_j$ is the prototype of class j, i.e., the mean normalized embedding of samples belonging to class j in $\mathcal{M}$, computed in the same way as Eq. (5).

$S_{i,j}$ serves as a centrality score metric to ascertain the most influential or cleanest samples among those sharing identical class labels. Nevertheless, $\mathcal{M}$ encompasses both clean and noisy samples. The latter can misleadingly affect the centrality score, resulting in an ambiguous separation between the centrality scores of clean and noisy samples. To address this, we propose calculating the cleanliness probability of each sample by applying a Gaussian Mixture Model (GMM) (A & Others, 2009) over $S_{i,j}$. This method allows for a more precise differentiation between the clean and noisy samples based on their respective influence:

$$p_G\left(g \mid S_{i,j}\right) = \frac{\pi_g \cdot \mathcal{N}\left(S_{i,j} \mid \mu_g, \sigma_g^2\right)}{\sum_{l=1}^{C_M} \pi_l \cdot \mathcal{N}\left(S_{i,j} \mid \mu_l, \sigma_l^2\right)}, \tag{12}$$

The posterior probability $p_G\left(g \mid S_{i,j}\right)$ denotes the probability of sample $\boldsymbol{x}_i$ having true class label g, where g is the Gaussian component with a larger mean (larger centrality score). C_M denotes the total number of classes in the memory buffer $\mathcal{M}$. The parameters of the GMM model $\{\mu_g, \sigma_g, \pi_g\}$ are inferred through the Expectation–Maximization algorithm. We finally obtain the clean subset $\mathbf{C}$ and noisy subset $\mathbf{N}$ of $\mathcal{M}$ by thresholding the posterior probability with a parameter λ_1:

$$\mathbf{C} := \left\{(\boldsymbol{x}, \tilde{\boldsymbol{y}}) \in \mathcal{M} : p_G\left(g = \tilde{\boldsymbol{y}} \mid S_{i,j}\right) \geq \lambda_1\right\}, \tag{13}$$

$$\mathbf{N} := \left\{(\boldsymbol{x}, \tilde{\boldsymbol{y}}) \in \mathcal{M} : p_G\left(g = \tilde{\boldsymbol{y}} \mid S_{i,j}\right) < \lambda_1\right\}, \tag{14}$$

Influenced by the findings in Sun, Ming, Zhu and Li (2022), and considering that the partitioned noisy subset $\mathbf{N}$ contains *unseen* class samples, we employ the KNN Score $s_k(\boldsymbol{x})$ to determine whether a sample belongs to the *seen* classes:

$$\mathbf{N}_{\mathbf{c}} := \left\{(\boldsymbol{x}, \tilde{\boldsymbol{y}}) \in \mathbf{N} : \mathbf{1}\left\{-s_k(\boldsymbol{x}) < \lambda_2\right\}\right\}. \tag{15}$$

Here $s_k(\boldsymbol{x}) = \left\|g(h(\boldsymbol{x})) - g(h(\boldsymbol{x}_{(k)}))\right\|_2$ is the distance of sample $\boldsymbol{x}$ to its kth nearest neighbor $\boldsymbol{x}_{(k)}$ in the representation space, $\mathbf{1}\{\cdot\}$ is the indicator function. Subsequently, we reassign pseudo-label j to each sample $\boldsymbol{x}_i \in \mathbf{N}_{\mathbf{c}}$ with maximum value of $S_{i,j}$. The other *unseen* class samples in $\mathbf{N}$ are discarded to avoid their negative effect.

Algorithm 1 Training Process of PAA at Time Step t

Input: Delay buffer B_t, memory buffer $\mathcal{M}$, hyperparameters λ_1, λ_2, k
Output: Network parameters $\{\theta_h, \theta_g, \theta_{f_1}, \theta_{f_2}\}$ for encoder $h(\boldsymbol{x}; \theta_h)$, projector $g(\boldsymbol{x}; \theta_g)$, classifier $f(\boldsymbol{x}; \theta_{f_1}, \theta_{f_2})$)

```
/* Online Prototype Learning */
Construct mini-bach B_m ← M based on Eq. (7)
// online prototype loss computation
Compute L_NEW on B_t via Eq.(4)
Compute L_OLD on B_m via Eq.(6)
// dual classifier consistency
Compute L_CON on B_t ∪ B_m via Eq.(8)
// online loss minimization
Minimize L_online = L_NEW + L_OLD + L_CON in an online manner
Compute L_CE on M via Eq.(9)
for epoch = 1 to MaxEpoch do
    /* Classifier Refinement */
    Minimize L_CE
end for
/* Memory Buffer Denoising */
Obtain C, N_c ← M. via Eq.(13),(15)
Update M as the combination of C and corrected N_c
```

After the denoising process, the memory buffer $\mathcal{M}$ is updated as the combination of clean samples and corrected *seen* class noisy samples $\mathcal{M} = \mathbf{C} \cup \mathbf{N}_{\mathbf{c}}$. Algorithm 1 shows the pipeline of PAA at each time step t.

4.3. PAA+

In the original implementation, prototype representations were updated batch-wise, leading to potential instability and information loss as data distributions changed. To address these issues, we introduced an Exponential Moving Average (EMA) mechanism to update the prototype representations continuously. This approach not only enriches the class feature information by accumulating more comprehensive data over time but also ensures more stable representations, which are crucial for robustness in dynamic environments.

We propose an enhancement to the original PAA method's online prototype learning loss L_{NEW} by integrating an Exponential Moving Average (EMA) approach to update class prototypes p_j in a continuous data stream. Initially, prototypes are initialized for each class. Upon arrival of each new mini-batch B_t, the prototypes are updated using the EMA formula:

$$p_j = \alpha \cdot p_j + (1 - \alpha) \cdot \hat{p}_j. \tag{16}$$

where α is the smoothing factor balancing the contribution of new and existing prototypes. This method ensures cross-batch consistency and gradual integration of new data, mitigating fluctuations due to mini-batch randomness. Post-EMA update, the updated prototypes are employed in the L_{NEW} loss function for continued learning. It is crucial to adjust the learning rate and monitor model performance to maintain the model's generalization capabilities and robustness. This refinement of p_j aims to enhance the stability and performance of the model when learning from a persistently incoming stream of data.

4.4. Efficiency and memory analysis

On each mini-batch, the time complexity of PAA primarily comes from its prototype learning process. Prototype updates (Eq. (5)) and contrastive losses ($\mathcal{L}_{\mathrm{NEW}}, \mathcal{L}_{\mathrm{OLD}}$) incur $O(Bd)$ and $O(B^2d)$ operations per mini-batch, respectively, where B is the batch size and d is the feature dimension. Thus, for a training dataset with N samples, the total computation complexity is $O(Nd + N^2d)$.

As for space complexity, memory usage hinges on buffer storage and prototype storage. Buffer storage require $O(Kd)$ for a memory buffer of size K storing d-dimensional feature embeddings. Prototype storage ($O(Cd)$, C = total classes have seen) grows incrementally with tasks. Thus the total space complexity is $O(Kd + Cd)$.

Buffer management is critical: larger K enhances robustness but strains memory, while smaller K risks underfitting. Efficiency hinges on balancing K against task complexity and hardware constraints.

5. Experiments

5.1. Settings

Datasets.

Synthesized Datasets. We generate synthetic datasets on the widely used CIFAR-100 dataset (Alex, 2009) and ImageNet dataset (Deng et al., 2009). CIFAR-100 dateset contains 50,000 training and 10,000 testing 32×32 color images. Following JoCoR (Yazhou et al., 2021), we crafted the closed-set synthetic noise dataset on CIFAR100N and ImageNet100N, featuring a noise ratio n_c ranging from 0 to 1. ImageNet dataset is a large-scale dataset consisting of 1000 classes with more than 1000 images per class, which is a more challenging benchmark for incremental learning. In total, there are roughly 1.2 million training images and 50k validation images. We conduct experiments on ImageNet100, which is a subset of ImageNet with randomly sampled 100 classes. We follow Hou, Pan, Loy, Wang, and Lin (2019) to use the fixed random seed (1993) for dataset generation.

The dataset incorporates two noise types, classified as either "Symmetry" or "Asymmetry". Symmetric noise occurs when a portion of the training data's labels are randomly switched out for any other labels available. In contrast, asymmetric noise is crafted to reflect the actual patterns of label noise found in the real world, where labels are only substituted with those of closely related categories, such as mistaking a deer for a horse or swapping a dog for a cat. To create the open-set synthetic dataset CIFAR80N, we initially designated the final 20 categories of CIFAR100 as out-of-distribution. Subsequently, we generated noisy in-distribution samples by randomly altering the labels of a n_c proportion of the remaining samples, by the specified noise type. This approach resulted in an overall noise ratio of $n_{all} = 0.2 + 0.8n_c$.

Real-World Datasets. We conduct experiments on two common real-world crowdsourced datasets: **WebVision** (Wen, Limin, Wei, Eirikur, & Luc, 2017) and **Food101N** (Kuang-Huei, Xiaodong, Lei, & Linjun, 2018). WebVision is a large-scale web image dataset designed to facilitate research on learning visual representations from noisy web data. It contains more than 2.4 million images crawled from the Flickr website and Google Images search. Food-101N dataset comprises approximately 310,009 food recipe images, categorized into 101 classes. However, the class labels are noisy, meaning they may be inaccurate. Each image in the dataset is assigned a class label, and the estimated noise rate is approximately 20%. More information about online CIL settings, on both synthetic and real-world datasets can be found in Section 5.2.

Baseline. In our research focusing on continual learning from streams of data with open-set noisy labels, we strategically combined state-of-the-art methods from both continual learning and open-set noisy label learning domains. To address the challenges of learning in an online task-free setting, we explored replay-based approaches. Specifically, we incorporated four key methods: (i) Maximally Interfered Retrieval (MIR) (Aljundi & Lucas, 2019), (ii) GDumb (Prabhu et al., 2020), (iii) Online Prototype(OnPro) (Wei et al., 2023), and (iv) Proxy-based Contrastive Replay(PCR) (Lin et al., 2023).

Simultaneously, to effectively handle the aspect of open-set noisy label learning, we selected two models representing a broad spectrum of strategies for classifying open-set noisy labeled data. These models are (i) semi-supervised (JoCoR) (Yazhou et al., 2021), (ii) probabilistic prediction (PNP) (Sun, Shen et al., 2022).

Furthermore, we conducted comparisons with state-of-the-art methods under the existing noisy labeled online CL settings, even though these methods did not address the issue of open-set noise. These methods include SPR (Kim et al., 2021), CNLL (Karim et al., 2022), and PuriDivER (Bang et al., 2022). Additional information regarding the configuration of baselines can be found in the supplementary materials.

Evaluation Metrics. We use the last test (or validation) accuracy as the primary performance metric, a standard widely adopted in CIL. Here, "last" denotes the point at which all tasks have arrived.

5.2. Detailed configuration for online CIL

Synthesized Datasets. To accommodate the dynamics of an online CL setup, we split CIFAR100N into 5 disjoint tasks, where each task has 20 disjoint classes, and split CIFAR80N into 20 disjoint tasks where each task has 16 disjoint classes. For ImageNet100N, we set the number of tasks to 10 and the memory buffer size to 2000. For a fair comparison, all methods use the same data augmentations, including resized-crop, horizontal-flip, and gray-scale.

Real-World Datasets. In our work utilizing the WebVision dataset, as guided by Kim et al. (2021), we selected the 14 largest classes based on their data volume. We constructed seven tasks from this selection, each comprising randomly paired classes to ensure a diverse and representative sampling. Regarding the Food-101N dataset, we developed five distinct tasks. Considering that Food-101N encompasses 101 classes, the composition of the final task is larger, incorporating 21 major classes to accommodate the extensive class range.

5.3. Baseline and the proposed approach configurations/settings

In our experiments with CIFAR100N, CIFAR80N, ImageNet100N, and WebVision datasets, we uniformly employ ResNet18, as delineated in He, Zhang, Ren, and Sun (2016), as the fundamental architecture for all algorithms under comparison. Concurrently, a linear layer is utilized as the projection head g, with the hidden dimension (dim) in g fixed at 128. For the Food101N dataset, ResNet34 serves as the backbone for all compared algorithms. A linear layer is also adopted as the classifier f_1 and f_2. Each model is trained from scratch using the Adam optimizer, with an initial learning rate set at 1×10^{-3} for all datasets. The weight decay parameter is maintained at 1.0×10^{-4}. In accordance with the methodologies described in Bang et al. (2022) and Kim et al. (2021), the batch size N is established at 10, and following Bang et al. (2022), the replay batch size m is set at 64. The temperature parameters are defined as $\tau_1 = 0.5$ and $\tau_2 = 0.1$. For baseline comparisons, we employ ResNet18 as the backbone architecture, ensuring parity in batch sizes and replay batch sizes for equitable evaluation. All baseline models are reproduced in an identical computational environment using their original source code and default configurations. The results reported herein represent the average outcomes of 15 independent runs for each experimental setup.

5.4. Results

Synthesized Datasets.

The comparative analysis of the last test accuracy on synthesized CIFAR100N, CIFAR80N, and ImageNet100N datasets under various noise conditions is presented in Tables 1 and 2. The results indicate a superior performance of our method over others in most scenarios. Specifically, our proposed method enhances accuracy by 2% on CIFAR100N and 10% on CIFAR80N compared to competing methods. Notably, PAA demonstrates a significant performance increase, particularly under high noise rates, irrespective of the presence of open-set noisy samples in the online data stream. The PAA algorithm effectively balances plasticity and stability, thereby enhancing the model's robustness. Traditional replay-based methods such as MIR and GDumb exhibit suboptimal performance, showing limited efficacy in enhancing the

Table 1
Last test accuracy on CIFAR100N and CIFAR80N under various noise scenarios. The size of the memory buffer both are 2000. Bold value represents the best method.

Methods	CIFAR100N					CIFAR80N				
	Sym.			Asym.		Sym.			Asym.	
	20	40	60	20	40	20	40	60	20	40
MIR	15.1 ± 1.0	7.7 ± 0.3	5.1 ± 0.4	11.3 ± 1.5	7.2 ± 0.6	11.6 ± 0.5	4.2 ± 1.6	1.4 ± 0.3	7.7 ± 0.6	4.4 ± 1.1
+ JoCoR	15.7 ± 1.6	7.9 ± 0.4	5.5 ± 2.1	13.7 ± 0.8	7.7 ± 0.9	12.9 ± 0.4	5.8 ± 1.2	2.6 ± 0.3	10.6 ± 1.4	3.7 ± 0.6
+ PNP	15.5 ± 0.7	8.2 ± 1.3	5.4 ± 1.7	12.9 ± 0.3	7.9 ± 1.2	13.1 ± 0.8	4.7 ± 0.9	2.3 ± 0.5	10.9 ± 0.5	4.8 ± 0.7
GDumb	15.8 ± 1.7	8.1 ± 0.5	5.6 ± 1.1	13.9 ± 1.3	9.0 ± 0.9	13.0 ± 0.2	5.0 ± 1.7	2.5 ± 0.8	10.7 ± 0.9	5.8 ± 0.5
+ JoCoR	16.1 ± 0.3	8.9 ± 0.4	6.1 ± 2.6	15.0 ± 0.2	9.5 ± 1.3	12.8 ± 0.7	5.9 ± 1.3	3.6 ± 2.6	11.3 ± 0.2	6.9 ± 1.3
+ PNP	16.5 ± 0.2	9.3 ± 0.7	6.4 ± 0.4	14.7 ± 0.8	10.1 ± 2.4	12.7 ± 0.8	7.0 ± 1.1	2.6 ± 0.4	11.9 ± 0.7	7.4 ± 1.1
OnPro	24.7 ± 0.1	22.8 ± 1.4	16.5 ± 0.3	22.4 ± 0.7	19.2 ± 0.2	22.5 ± 1.3	20.1 ± 0.9	14.4 ± 0.4	19.7 ± 1.0	16.2 ± 1.2
+ JoCoR	23.6 ± 1.4	23.1 ± 0.9	17.8 ± 1.2	23.1 ± 1.5	20.8 ± 1.4	21.1 ± 0.8	19.6 ± 1.0	15.4 ± 0.7	20.8 ± 1.3	17.8 ± 0.9
+ PNP	24.1 ± 0.9	20.6 ± 1.3	16.4 ± 0.2	22.7 ± 1.0	19.4 ± 0.6	21.6 ± 0.9	17.3 ± 1.1	12.8 ± 0.2	19.1 ± 0.7	17.1 ± 1.1
PCR	23.8 ± 0.1	21.7 ± 0.4	17.1 ± 0.7	22.3 ± 1.9	18.2 ± 0.8	21.6 ± 1.3	18.9 ± 0.8	14.6 ± 0.7	18.9 ± 1.1	14.5 ± 0.8
+ JoCoR	24.3 ± 0.5	22.6 ± 0.8	18.9 ± 1.5	22.7 ± 0.4	19.6 ± 0.5	22.1 ± 0.6	19.6 ± 1.0	16.3 ± 0.8	19.2 ± 1.2	16.3 ± 0.5
+ PNP	23.1 ± 0.4	21.6 ± 0.6	18.7 ± 1.1	22.1 ± 0.1	18.9 ± 0.2	19.6 ± 1.0	18.5 ± 0.7	15.6 ± 0.7	19.0 ± 0.6	15.8 ± 1.0
SPR	21.5 ± 0.1	21.1 ± 0.3	18.1 ± 1.0	20.5 ± 0.6	19.8 ± 2.2	18.3 ± 0.2	18.7 ± 0.3	16.1 ± 1.1	18.4 ± 0.8	17.7 ± 2.0
CNLL	38.7 ± 0.6	32.1 ± 0.4	26.2 ± 1.2	39.0 ± 0.9	32.6 ± 1.1	30.1 ± 0.5	28.8 ± 0.4	22.5 ± 0.7	34.1 ± 0.9	26.5 ± 1.1
PuriDivER	32.1 ± 0.4	30.6 ± 0.3	24.7 ± 0.9	31.4 ± 0.3	22.5 ± 0.2	28.9 ± 0.4	24.0 ± 0.3	21.5 ± 0.8	28.1 ± 0.5	20.4 ± 0.6
PAA(ResNet)	40.2 ± 1.6	35.4 ± 0.5	28.3 ± 1.4	37.8 ± 1.9	34.7 ± 1.7	36.6 ± 0.6	32.8 ± 0.4	29.7 ± 0.1	35.4 ± 0.7	31.9 ± 0.2
PAA+	**41.4 ± 1.1**	**37.8 ± 0.9**	**30.2 ± 2.1**	**39.6 ± 2.4**	**36.1 ± 1.2**	**38.2 ± 1.0**	**34.3 ± 0.7**	**31.5 ± 0.3**	**37.9 ± 1.5**	**34.1 ± 0.5**

Table 2
Last test accuracy on ImageNet100N under various noise scenarios. The size of the memory buffer is 2000. Bold value represents the best method.

Methods	ImageNet100N				
	Sym.			Asym.	
	20	40	60	20	40
MIR	7.8 ± 2.3	4.1 ± 0.8	2.8 ± 1.2	6.0 ± 1.0	3.9 ± 1.5
+ JoCoR	8.2 ± 0.9	4.4 ± 0.6	3.1 ± 1.0	7.3 ± 0.7	4.2 ± 1.1
+ PNP	8.0 ± 1.1	4.5 ± 0.9	3.0 ± 1.3	6.8 ± 0.5	4.4 ± 0.8
GDumb	8.3 ± 1.3	4.6 ± 0.7	3.1 ± 1.4	7.4 ± 1.2	4.9 ± 0.6
+ JoCoR	8.5 ± 0.5	5.1 ± 0.8	3.5 ± 1.6	8.2 ± 0.4	5.4 ± 0.9
+ PNP	8.8 ± 0.4	5.4 ± 0.9	3.7 ± 0.6	7.9 ± 0.9	5.6 ± 1.3
OnPro	13.1 ± 0.3	12.4 ± 0.9	9.4 ± 0.5	12.0 ± 0.5	10.3 ± 0.4
+ JoCoR	12.4 ± 0.8	12.5 ± 0.6	10.2 ± 0.8	12.4 ± 0.9	11.2 ± 0.9
+ PNP	12.7 ± 0.6	11.2 ± 0.8	9.5 ± 0.4	12.0 ± 0.7	10.6 ± 0.5
PCR	12.5 ± 0.2	11.6 ± 0.6	10.0 ± 0.8	11.8 ± 1.2	10.2 ± 0.7
+ JoCoR	12.9 ± 0.7	12.0 ± 0.9	10.8 ± 1.3	12.0 ± 0.6	10.4 ± 0.6
+ PNP	12.3 ± 0.5	11.5 ± 0.7	10.7 ± 0.9	11.8 ± 0.3	10.8 ± 0.5
SPR	11.3 ± 0.3	11.2 ± 0.5	9.6 ± 0.9	10.9 ± 0.8	10.7 ± 1.7
CNLL	21.2 ± 0.8	17.3 ± 0.6	13.8 ± 1.4	**21.3 ± 1.1**	17.5 ± 1.3
PuriDivER	17.0 ± 0.6	16.2 ± 0.5	13.1 ± 1.0	16.8 ± 0.6	12.1 ± 0.5
PAA	22.1 ± 1.4	19.0 ± 0.7	15.0 ± 1.3	20.8 ± 1.6	18.7 ± 1.5
PAA+	**22.7 ± 1.0**	**20.4 ± 0.9**	**16.0 ± 1.7**	20.3 ± 2.0	**19.4 ± 1.1**

model's adaptability to new samples. State-of-the-art methods in online continuous learning, such as OnPro and PCR, demonstrate commendable performance at lower noise rates. However, when combined with robust algorithms, there is a discernible improvement in performance across various settings. The PAA+ method, an enhanced version of PAA, improves accuracy by 1% to 2% compared to PAA under all noise rates, indicating that updating the prototypes on the data stream using the EMA method achieves a more stable representation.

On the ImageNet100N dataset, PAA and PAA+ demonstrate superior performance, achieving 22.1% and 22.7% accuracy under 20% symmetric noise, respectively. These results outperform all baselines by more than 1% to 5%. The performance gap widens under higher noise levels, such as 60% symmetric noise, where PAA+ achieves 16.0% accuracy compared to CNLL's 13.8%. Under asymmetric noise, PAA+ achieves 19.4% accuracy at 40% noise, surpassing the best baseline (CNLL: 17.5%). This aligns with results on CIFAR datasets, confirming the robustness of our framework to diverse noise types. While PuriDivER and CNLL show moderate performance, their reliance on closed-set noise assumptions limits their adaptability to streaming tasks. In contrast, PAA's dynamic prototype alignment ensures robustness across sequential noisy updates.

In summary, our proposed PAA and PAA+ methods demonstrate significant improvements in handling noisy data streams across multiple datasets, showcasing their robustness and adaptability in various scenarios. The consistent performance gains across CIFAR100N, CIFAR80N, and ImageNet100N highlight the effectiveness of our approach in addressing the challenges of noisy-label learning and continual learning.

Real-World Datasets. The test accuracy results for various methods applied to the WebVision and Food-101N datasets are delineated in Table 3. An analysis of these results, as depicted in Table 2, reveals that our PAA method consistently surpasses competing methods by a margin exceeding 6% in test accuracy. The enhanced version of PAA, denoted as PAA+, further improves upon this performance, achieving a validation accuracy of 58.6%, thus solidifying its position as the top-performing method. The robustness and adaptability of the PAA+ method are thus affirmed in complex, real-world noisy label learning scenarios.

Focusing on the Food-101N dataset, it is important to note that the exact noise ratio has not been quantified, presenting a challenge for methods that are tailored to specific types of noise. Despite this lack of specific noise ratio data, the PAA method maintains a lead, outperforming existing methods by over 3% in test accuracy. The PAA+ method also shows an improvement in this dataset, with a validation accuracy of 17.7%.

5.5. Ablation study

Revision of effects of each component. Our ablation study, which is a part of the PAA+ approach, thoroughly examined how different loss components—specifically, $\mathcal{L}_{\text{NEW}}$, $\mathcal{L}_{\text{OLD}}$, $\mathcal{L}_{\text{CON}}$, and $\mathcal{L}_{\text{CE}}$—contribute to the model's effectiveness under various noisy conditions. This analysis was conducted on the CIFAR100N and CIFAR80N datasets to understand their impact on the model's performance. Notably, the absence of $\mathcal{L}_{\text{NEW}}$, which is imperative for adapting to newly emergent patterns in the data streams, markedly diminished the model accuracy. This reduction was evident as the performance declined to 30.1% and 27.3% in Sym.20% and Asym.20% noise scenarios respectively on CIFAR100N, underscoring the pivotal role of $\mathcal{L}_{\text{NEW}}$ in bolstering model robustness. Furthermore, this trend was mirrored in CIFAR80N with corresponding declines, reinforcing the crucial necessity of $\mathcal{L}_{\text{NEW}}$ in

Table 3
Last validation accuracy on WebVision (K = 1000) and Food-101N (K = 2000), where K is memory buffer size.

Methods	WebVision	Food-101N
MIR	17.2 ± 0.8	8.8 ± 1.8
+ JoCoR	19.0 ± 0.4	8.9 ± 1.2
+ PNP	16.5 ± 0.3	8.8 ± 0.4
GDumb	30.4 ± 2.1	9.7 ± 0.7
+ JoCoR	24.2 ± 2.1	10.1 ± 0.5
+ PNP	26.5 ± 1.8	10.2 ± 0.3
OnPro	23.1 ± 0.1	12.5 ± 0.2
+ JoCoR	21.4 ± 1.1	12.6 ± 0.6
+ PNP	21.8 ± 0.8	9.5 ± 1.3
PCR	23.1 ± 0.1	11.7 ± 0.2
+ JoCoR	21.4 ± 1.1	13.4 ± 0.8
+ PNP	21.8 ± 0.8	13.2 ± 1.1
SPR	40.0 ± 0.5	12.1 ± 0.2
CNLL	47.7 ± 0.7	14.2 ± 0.9
PuriDivER	51.8 ± 0.4	13.4 ± 0.3
PAA	57.4 ± 0.7	16.8 ± 0.4
PAA+	**58.6 ± 0.3**	**17.7 ± 0.9**

Table 4
Ablation Studies on CIFAR100N and CIFAR80N under Sym.20% and Asym.20% Noise Scenarios. The size of the memory buffer both are 2000. "baseline" means $\mathcal{L}_{CE}$."DP" means Denoising Process.

Method	CIFAR100N		CIFAR80N	
	Sym.20%	Asym.20%	Sym.20%	Asym.20%
baseline	10.2 ± 1.5	7.8 ± 0.7	6.6 ± 0.5	5.4 ± 1.2
w/o DP	29.0 ± 0.2	28.1 ± 2.1	25.4 ± 0.9	25.7 ± 0.4
w/o $\mathcal{L}_{NEW}$	30.1 ± 1.7	27.3 ± 1.3	26.8 ± 0.9	29.7 ± 0.2
w/o $\mathcal{L}_{OLD}$	33.6 ± 1.2	23.5 ± 0.4	32.4 ± 0.5	28.3 ± 0.8
w/o $\mathcal{L}_{CON}$	36.6 ± 1.3	33.1 ± 0.6	33.0 ± 0.7	30.7 ± 1.9
w/o $\mathcal{L}_{CE}$	21.3 ± 0.8	19.5 ± 2.1	17.7 ± 1.2	17.1 ± 2.0
PAA+	**40.2 ± 1.6**	**37.8 ± 1.9**	**36.6 ± 0.6**	**35.4 ± 0.7**

the learning process. Conversely, elimination of $\mathcal{L}_{OLD}$, while impactful, did not lead to as significant a performance degradation as $\mathcal{L}_{NEW}$. This distinction highlights the differential contributions of these components towards handling noise and retaining model accuracy. Similarly, removal of $\mathcal{L}_{CON}$ and $\mathcal{L}_{CE}$ also noticeably impacted the accuracy, with $\mathcal{L}_{CE}$'s absence showing considerable effects across both CIFAR100N and CIFAR80N, thereby emphasizing the integral role of cross-entropy loss in incorporating re-labeled samples and enhancing the learning efficacy.

Effects of denoising process. Removing the Denoising Process resulted in a reduction in accuracy, though the impact was less than the loss functions. On CIFAR100N, the accuracy in Sym.20% decreased from 40.2% to 29.0%, and in Asym.20% from 37.8% to 28.1%. This pattern was also evident in CIFAR80N, indicating that DP plays a significant role in the model's performance, particularly in managing noisy data (see Table 4).

5.6. Buffer size analysis

Examining the experimental results presented in the Table 5, it is apparent that the method designated as "PAA" (ours) demonstrates remarkable superiority over its counterparts, maintaining the highest accuracy across both symmetric and asymmetric noise conditions, regardless of whether the buffer size is set to 500 or 1000. This uniform dominance strongly suggests that its effectiveness is inherent to its approach rather than just a consequence of larger memory capacity. Impressively, while all methods benefit from an increased buffer size—showing enhanced accuracy—"PAA" (ours) showcases a potent robustness to buffer size variations, hinting that its performance is likely underpinned by an efficient learning mechanism and optimal buffer management. The improvements witnessed when doubling the buffer size elucidate the value of memory in managing noisy environments, particularly in the more demanding scenarios evidenced by the higher noise levels. Furthermore, the SPR method's consistent accuracy exemplifies its resilience, though its unique memory usage warns against direct comparison with other baselines. Collectively, these findings affirm that strategic buffer utilization and replay techniques are critical in addressing label noise within continual learning frameworks.

5.7. Sensitivity analysis

In this section, the outcomes of the sensitivity analysis experiment about the proposed method are delineated. The crux of the method's sensitivity hinges on the threshold for noise and open-set decision bound. For λ_1, the test accuracy across both CIFAR80N and CIFAR100N datasets shows a decreasing trend with increasing λ_1 values. This indicates that excessively high thresholds can significantly reduce the proportion of credible clean samples, affecting the model's fitting speed. In contrast, the sensitivity of λ_2 is less clear-cut. While there are fluctuations in test accuracy with varying λ_2 values, the trend is not as pronounced or consistent as with λ_1. This suggests that relatively conservative thresholds λ_2 can minimize the negative impact of open set samples on the model and reduce the cumulative error of the model (see Fig. 3).

5.8. Time cost

In the realm of online continual learning, the time cost required for a model to adapt to the changing data distribution is a crucial factor. However, there exists a trade-off between time efficiency and accuracy. As depicted in Fig. 4, methods like OnPro and PCR achieve relatively short training times but at the expense of lower accuracy. Conversely, SPR incurs excessive time costs, rendering it impractical for real-time online applications. Although PAA involves additional time for the denoising process compared to CNLL and PuriDivER, it demonstrates a favorable balance between accuracy and time cost, outperforming other approaches in this regard.

6. Conclusion

In conclusion, our study addresses the critical challenges of label noise and unknown classes in the realm of online class-incremental continual learning (CIL), which are vital for model success in dynamic and evolving domains. We have explored and defined the concepts of *closed-set* and *open-set* noise, illuminating how both can introduce *unseen classes* into the current training classifier. Our proposed solution, the Prototypes as Anchors (PAA) method, innovatively utilizes replay-based techniques to learn representative and discriminative prototypes for each class. PAA's strength lies in its similarity-based denoising schema within the representation space, effectively distinguishing and mitigating the adverse effects of unseen classes. The dual-classifier architecture further enhances the robustness of PAA by implementing consistency checks. Our extensive experiments across various datasets have demonstrated that PAA significantly outperforms existing approaches in terms of model performance and robustness.

6.1. Overhead

The overhead of our proposed algorithm primarily lies in the accuracy of the computed prototypes, as the fitting of new samples and the denoising process both rely on the accuracy of the computed prototype representations. Although we use Exponential Moving Average (EMA) to obtain relatively stable and accurate representations, the presence of semantic drift in the continual learning scenario inevitably leads to dynamic changes and shifts in the relative positions of the prototypes.

Table 5
Last test accuracy on CIFAR80N under various buffer size. Bold value represents the best method.

Methods	CIFAR80N (Buffer size = 500)					CIFAR80N (Buffer size = 1000)				
	Sym.			Asym.		Sym.			Asym.	
	20	40	60	20	40	20	40	60	20	40
SPR	15.1 ± 0.2	15.0 ± 0.3	12.5 ± 1.1	15.7 ± 0.8	14.0 ± 2.0	17.5 ± 0.2	**18.0 ± 0.3**	15.4 ± 1.1	17.6 ± 0.8	16.9 ± 2.0
CNLL	26.8 ± 0.5	25.2 ± 0.4	19.0 ± 0.7	31.0 ± 0.9	23.3 ± 1.1	29.4 ± 0.5	27.9 ± 0.4	21.8 ± 0.7	**33.3 ± 0.9**	25.8 ± 1.1
PuriDivER	25.6 ± 0.4	22.0 ± 0.3	18.9 ± 0.8	26.0 ± 0.5	17.8 ± 0.6	28.2 ± 0.4	23.5 ± 0.3	20.9 ± 0.8	27.3 ± 0.5	20.1 ± 0.6
PAA (ours)	**32.9 ± 0.6**	**29.3 ± 0.4**	**26.2 ± 0.1**	**32.7 ± 0.7**	**28.4 ± 0.2**	**35.3 ± 0.6**	31.5 ± 0.4	**27.0 ± 0.1**	34.6 ± 0.7	**30.6 ± 0.2**

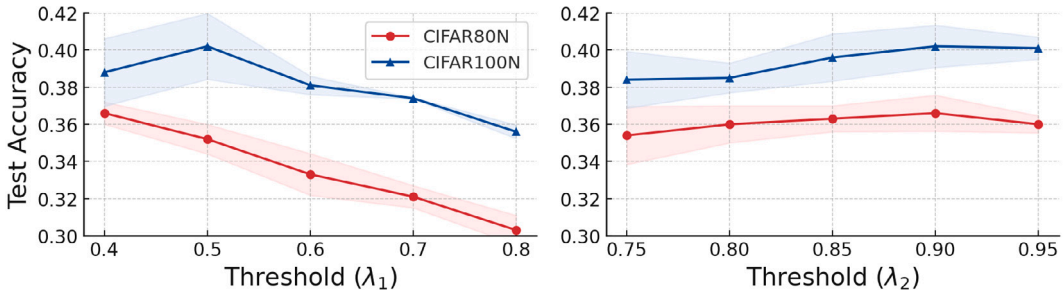

Fig. 3. The sensitivity of hyper-parameter λ_1 and λ_2. Experiments are conducted on CIFAR100N and CIFAR80N with Sym.20%.

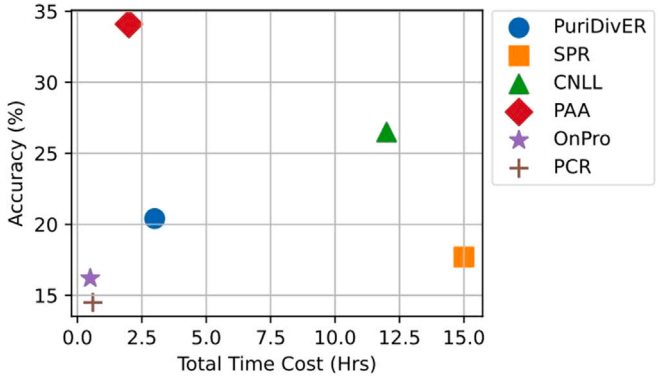

Fig. 4. Total training time and accuracy on CIFAR80N with Asym.40% for various methods.

6.2. Limitation

Although PAA outperforms prior arts by large margins, developing methods in open-set CIL setups are in its nascent steps; e.g., no new domain images is added as the task progresses. Another limitation is related to the scalability of the method. As the number of tasks and classes in online continual learning grows, the computational cost of maintaining and updating the memory buffer, as well as computing prototypes, may increase substantially. The current approach may struggle to handle extremely large- scale problems efficiently. This could limit its application in resource - constrained environments, such as edge devices, where computational power and memory are scarce.

6.3. Directions for future research

Since continual learning (CL) methods aim to apply AI in a wide range of scenarios, data can be sourced from various places. This situation may give rise to data privacy problems. Even though the proposed method does not deliberately cause such issues, it might still have negative impacts on data privacy unknowingly. In the field of secure machine learning research, efforts to address and prevent these privacy - related concerns will be a major focus.

CRediT authorship contribution statement

Shao-Yuan Li: Writing – review & editing, Supervision, Conceptualization. **Yu-Xiang Zheng:** Writing – original draft, Visualization, Software, Methodology, Investigation. **Sheng-Jun Huang:** Writing – review & editing. **Songcan Chen:** Writing – review & editing. **Kangkan Wang:** Writing – review & editing.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work, some of the authors used ChatGPT (GPT-4) in order to improve the readability and language of certain sections. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

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 is supported by National Science and Technology Major Project (2022ZD0114801), National Natural Science Foundation of China (62472224), Open Project Funds for the Joint Laboratory of Spatial Intelligent Perception and Large Model Application (SIPLMA-2024-YB-05), Fundamental Research Funds for the Central Universities, China (NS2024059).

References

A, R. D., & Others (2009). Gaussian mixture models. *Encyclopedia of Biometrics*, 741.

Alex, K. (2009). *Learning multiple layers of features from tiny images* (Master's Thesis), Department of Computer Science, University of Toronto.

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., & Tuytelaars, T. (2018). Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision* (pp. 139–154).

Aljundi, R., & Lucas, C. (2019). Online continual learning with maximally interfered retrieval. In *Advances in neural information processing systems*.

Bang, J., Koh, H., Park, S., Song, H., Ha, J. W., & Choi, J. (2022). Online continual learning on a contaminated data stream with blurry task boundaries. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 9275–9284).

Bo, H., Quanming, Y., Xingrui, Y., Gang, N., Miao, X., Weihua, H., et al. (2018). Co-teaching: Robust training of deep neural networks with extremely noisy labels. *vol. 31*, In *Advances in neural information processing systems*.

Buzzega, P., Boschini, M., Porrello, A., Abati, D., & Calderara, S. (2020). Dark experience for general continual learning: A strong, simple baseline. a. *Dvances in Neural Information Processing Systems*, *33*, 15920–15930.

Chaudhry, A., Dokania, P. K., Ajanthan, T., & Torr, P. H. (2018). Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European conference on computer vision* (pp. 532–547).

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H., et al. (2019). On tiny episodic memories in continual learning. arXiv preprint arXiv:1902.10486.

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).

Guo, Y., Liu, B., & Zhao, D. (2022). Online continual learning through mutual information maximization. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, & S. Sabato (Eds.), *Proceedings of the international conference on machine learning* (pp. 8109–8126). PMLR..

He, X., & Jaeger, H. (2018). Overcoming catastrophic interference using conceptor-aided backpropagation. In *Proceedings of the international conference on learning representations*.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778).

Hongxin, W., Lei, F., Xiangyu, C., & Bo, A. (2020). Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13726–13735).

Hou, S., Pan, X., Loy, C. C., Wang, Z., & Lin, D. (2019). Learning a unified classifier incrementally via rebalancing. In *The IEEE conference on computer vision and pattern recognition*. pp.

Karim, N., Khalid, U., Esmaeili, A., & Rahnavard, N. (2022). Cnll: A semi-supervised approach for continual noisy label learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 3878–3888).

Kim, D. C., Jeong, J., Moon, S., & Kim, G. (2021). Continual learning on noisy data streams via self-purified replay. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 537–547).

Kuang-Huei, L., Xiaodong, H., Lei, Z., & Linjun, Y. (2018). Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5447–5456).

Lange, M. D., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., et al. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*, 3366–3385.

Li, J., Socher, R., & Hoi, S. C. (2020). Dividemix: Learning with noisy labels as semi-supervised learning. In *Proceedings of the international conference on learning representations*.

Li, J., Xiong, C., & Hoi, S. C. (2021). Learning from noisy data with robust representation learning. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9485–9494).

Lin, H., Zhang, B., Feng, S., Li, X., & Ye, Y. (2023). Pcr: Proxy-based contrastive replay for online class-incremental continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 24246–24255).

Liu, Y., & Guo, H. (2020). Peer loss functions: Learning from noisy labels without knowing noise rates. In *Proceedings of the international conference on machine learning* (pp. 6226–6236). PMLR..

Lu, J., Xu, Y., Li, H., Cheng, Z., & Niu, Y. (2022). Pmal: Open set recognition via robust prototype mining. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 1872–1880).

Lu, J., Zhengyuan, Z., Thomas, L., Li-Jia, L., & Li, F. F. (2018). Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the international conference on machine learning* (pp. 2304–2313). PMLR..

Oza, P., & Patel, V. M. (2019). C2ae: Class conditioned auto-encoder for open-set recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2307–2316).

Prabhu, A., Torr, P. H., & Dokania, P. K. (2020). Gdumb: A simple approach that questions our progress in continual learning. In *Computer vision–ECCV 2020: 16th European conference, glasgow, UK, August (2020) 23–28, proceedings, part II* (pp. 524–540). Springer.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., et al. (2018). Learning to learn without forgetting by maximizing transfer and minimizing interference. arXiv preprint arXiv:1810.11910.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., et al. (2016). Progressive neural networks. arXiv preprint arXiv:1606.04671.

Shim, D., Mai, Z., Jeong, J., Sanner, S., Kim, H., & Jang, J. (2021). Online class-incremental continual learning with adversarial shapley value. In *Proceedings of the AAAI conference on artificial intelligence* (pp. 9630–9638).

Sun, Y., Ming, Y., Zhu, X., & Li, Y. (2022). Out-of-distribution detection with deep nearest neighbors. In *Proceedings of the international conference on machine learning* (pp. 20827–20840). PMLR.

Sun, Z., Shen, F., Huang, D., Wang, Q., Shu, X., Yao, Y., et al. (2022). Pnp: Robust learning from noisy labels by probabilistic noise prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5311–5320).

Sun, X., Yang, Z., Zhang, C., Ling, K. V., & Peng, G. (2020). Conditional gaussian distribution learning for open set recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13480–13489).

Tongliang, L., & Dacheng, T. (2015). Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*, 447–461.

Wei, Y., Ye, J., Huang, Z., Zhang, J., & Shan, H. (2023). Online prototype learning for online continual learning. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 18764–18774).

Wen, L., Limin, W., Wei, L., Eirikur, A., & Luc, V. G. (2017). Webvision database: Visual learning and understanding from web data. arXiv preprint arXiv:1708.02862.

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., et al. (2019). Large scale incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 374–382).

Xia, X., Liu, T., Wang, N., Han, B., Gong, C., Niu, G., et al. (2019). Are anchor points really indispensable in label-noise learning? *Advances in Neural Information Processing Systems*, *32*.

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., Yang, Q., & Liu, C. L. (2020). Convolutional prototype network for open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44*, 2358–2370.

Yao, Y., Liu, T., Han, B., Gong, M., Deng, J., Niu, G., et al. (2020). Dual t: Reducing estimation error for transition matrix in label-noise learning. *Advances in Neural Information Processing Systems*, *33*, 7260–7271.

Yazhou, Y., Zeren, S., Chuanyi, Z., Fumin, S., Qi, W., Jian, Z., et al. (2021). Jo-src: A contrastive approach for combating noisy labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 5192–5201).

Yilun, X., Peng, C., Yuqing, K., & Yizhou, W. (2019). L_*dmi*: A novel information-theoretic loss function for training deep nets robust to label noise. *Advances in Neural Information Processing Systems*, *32*.

Yoshihashi, R., Shao, W., Kawakami, R., You, S., Iida, M., & Naemura, T. (2019). Classification-reconstruction learning for open-set recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 4016–4025).

Zhilu, Z., & Mert, S. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in Neural Information Processing Systems*, *31*.

Zhou, D. W., Wang, Q. W., Qi, Z. H., Ye, H. J., Zhan, D. C., & Liu, Z. (2023). Deep class-incremental learning: A survey. arXiv preprint arXiv:2302.03648.

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., Cheng, Z., yao Zhang, X., & lin Liu, C. (2021). Class-incremental learning via dual augmentation. *Advances in Neural Information Processing Systems*, *34*, 14306–14318.