Prototype Reminiscence and Augmented Asymmetric Knowledge Aggregation for Non-Exemplar Class-Incremental Learning

Wuxuan Shi[1], Mang Ye[1,2]*
[1]School of Computer Science, Wuhan University, Wuhan, China
[2] Hubei Luojia Laboratory, Wuhan, China
https://shiwuxuan.github.io/PRAKA-project

Abstract

Non-exemplar class-incremental learning (NECIL) requires deep models to maintain existing knowledge while continuously learning new classes without saving old class samples. In NECIL methods, prototypical representations are usually stored, which inject information from former classes to resist catastrophic forgetting in subsequent incremental learning. However, since the model continuously learns new knowledge, the stored prototypical representations cannot correctly model the properties of old classes in the existence of knowledge updates. To address this problem, we propose a novel prototype reminiscence mechanism that incorporates the previous class prototypes with arriving new class features to dynamically reshape old class feature distributions thus preserving the decision boundaries of previous tasks. In addition, to improve the model generalization on both newly arriving classes and old classes, we contribute an augmented asymmetric knowledge aggregation approach, which aggregates the overall knowledge of the current task and extracts the valuable knowledge of the past tasks, on top of self-supervised label augmentation. Experimental results on three benchmarks suggest the superior performance of our approach over the SOTA methods.

1. Introduction

In recent years, deep neural networks have achieved great success on various tasks. In dynamic and open environments, deep models also require the ability to continuously learn new tasks as the input stream is updated. Hence, class-incremental learning (CIL), which aims to learn a unified classifier that can classify all seen classes under progressive changes in the classes to be learned, has attracted extensive attention [30, 41, 16, 47, 31, 28].

As new data becomes available, it is computationally expensive to jointly retrain the model with new and old class samples. Worse still, the old class samples could not be fully accessible. In this case, an alternative is to fine-tune the model on new data, yet catastrophic forgetting [38, 15] will be a serious challenge. The decision boundary of the unified classifier would be significantly changed and biased towards the new classes. Conversely, another direction is to fix the feature embedding space of a trained model, which suffers from frustrating generalization ability and thus performs poorly on new tasks, *i.e.*, the plasticity of the model is greatly degraded.

To overcome the catastrophic forgetting issue, many CIL methods [42, 8, 6, 56, 2, 53] store a fraction of the old data in memory and replay them in subsequent incremental phases to maintain the existing knowledge. Unfortunately, storing data poses privacy concerns and comes at a sharp cost to memory and computation. In this paper, we follow a paradigm holding for more extensive applications,

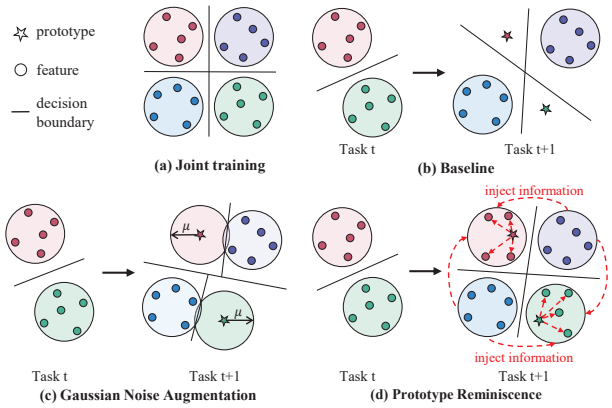

Figure 1. Idea illustration. (a) Joint training with abundant samples. (b) Baseline: Updating the model with the memorized prototype and new data, which narrows the decision boundary of the old classes. (c) Gaussian Noise Augmentation [66]: The decision boundary of the old class is retained, but it introduces an overlap between the old and new class distributions due to the change in representation space. (d) With prototype reminiscence, newly learned information can be injected while reshaping the feature distribution of past data to reduce overlap and resist forgetting.

*Corresponding Author: Mang Ye (yemang@whu.edu.cn)

termed non-exemplar class-incremental learning (NECIL) [66, 67], which solves the catastrophic forgetting problem in CIL scenario without preserving old class samples.

For NECIL, a natural substitute for storing data is to generate pseudo-samples of previous classes by deep generative models [62, 54, 46, 49, 57] such as GAN [4, 17]. However, it is unstable and ineffective to train generative models for non-stationary data streams [65]. Catastrophic forgetting can also have a negative impact on the generative model resulting in a simultaneous decrease in the effectiveness of both models. Instead of focusing on old data, some works turn to estimating model parameters that are important for previous tasks and constraining their changes [23, 61, 3]. Nevertheless, the constraints on the model parameters lead to poor generalization ability to long-sequence tasks. Besides, several studies propose to dynamically expand the network structure during the process of incremental learning [44, 58, 59, 37, 64]. Although this strategy can efficiently handle long sequences of tasks and ultimately maintain the performance of the old classes, the computational resource requirements associated with creating and storing additional network components and reasoning about multiple forward propagations are frustrating.

Recently, some prototype-based NECIL methods have achieved impressive performance [66, 67, 60, 50]. They use prototypical representations (typically the class mean in the deep feature space) memorized for each old class to model the feature distribution of past data and inject information from the previous classes in subsequent incremental learning. Rather than storing samples, this strategy is more memory efficient and privacy secure. Nevertheless, as shown in Fig. 1 (b), direct training with saved prototypes and current data struggles to prevent the collapse of decision boundaries, due to the lack of old class features. Some works augment the prototypes by adding Gaussian noise [66] or over-sampling [67] to enrich old class features. However, updates of the model on continuous data streams could lead to inevitable changes in the representation of old classes, making the saved prototypes increasingly outdated.

The feature distribution simulated by the above strategies cannot accommodate such changes due to the missing consideration of knowledge updates. It could result in overlap between the distributions of different classes, especially between new and old classes, as shown in Fig. 1 (c). Consequently, *combining newly acquired information with stored prototypes to dynamically model past data distributions is crucial to resist catastrophic forgetting in NECIL.*

To address this challenge, we propose a prototype reminiscence mechanism to track the evolution of the old class representations by injecting new knowledge from the updating network while reshaping the feature distribution. Specifically, we perform a random bidirectional interpolation operation between the extracted new class features and

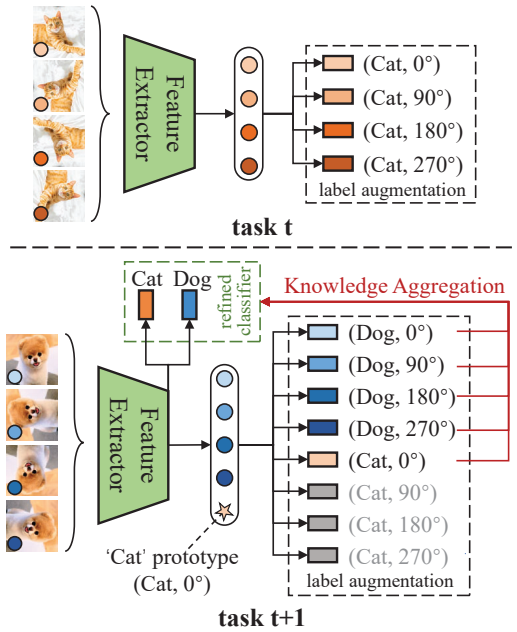

Figure 2. We introduce self-supervised label augmentation to learn generalizable and transferable representations. The knowledge of the self-supervised classifier is aggregated and transferred to another classifier to take full advantage of it.

the saved old class prototypes to enrich old class features. As shown in Fig. 1 (d), it expands the prototype to protect the decision boundaries of old classes and to counteract catastrophic forgetting. Since the feature distribution of past tasks is dynamically adjusted to the current representation space, the overlap between the old and new classes is reduced. Thus, the discrimination and balance between the old and new classes are maintained. Cooperating with the well-known knowledge distillation (KD) [20, 19], the mismatch between the preserved prototypes and the continuously updated network is alleviated.

In addition to dealing with catastrophic forgetting, when new data arrives, performance on the current task is also of great concern, necessitating the plasticity of the incremental learner. This mainly involves two aspects: learning generalizable and transferable representations, fully utilizing the information from new data. Previous works [66, 55] have achieved good progress on the first aspect by self-supervised label augmentation [27], however they ordinarily disregard the second. The new task contains abundant information that the network has never encountered and will have a stronger influence on model updates. *Improving the plasticity of the model from continuous data streams requires a simultaneous approach from both aspects.*

To solve this bottleneck, we contribute an augmented asymmetric knowledge aggregation approach to enhance the plasticity of the model noninvasively. Taking inspiration from [24, 27], we first augment the new classes with

rotation as self-supervision, which requires the model to acquire task-agnostic representations to improve its generalization ability. Furthermore, as illustrated in Fig. 2, we selectively aggregate the valuable knowledge in the augmented classifier—valid weights of past tasks are extracted, and the information captured on the current task is sufficiently exploited. This asymmetric knowledge aggregation scheme can condense the knowledge learned via self-supervised label augmentation (SLA) to make the classifier more purified. It further improves the incremental learner's performance on the new tasks without discrimination scarification on old classes.

To summarize, our main contributions are as follows:

- We propose a simple yet effective method of prototype reminiscence for NECIL, which models feature distribution for the past data in a continuously updated representation space to resist catastrophic forgetting.
- We contribute augmented asymmetric knowledge aggregation, which learns task-agnostic representations and fully captures the newly acquired knowledge to improve the plasticity of incremental learners.
- Extensive experiments on three benchmarks demonstrate that our method achieves state-of-the-art performance. We also provide a detailed ablation study to analyze the influence of each component.

2. Related Work

2.1. Class-Incremental Learning

Current CIL methods can be roughly divided into three categories: rehearsal-based methods, regularization-based methods and structure-based methods.

Rehearsal-based methods maintain the distribution of old classes by saving exemplars of fixed memory size to jointly train the model with current data. Based on the saved samples, some works investigate knowledge distillation to prevent forgetting [56, 42, 13], while others try to combine regularization of gradients and rehearsal to make more efficient use of the preserved data [43, 34, 11]. In addition, several papers have studied the impact of memory management schemes [5, 22, 33]. *Regularization-based methods* identify the key model parameters for the previous tasks, penalizing their changes when learning the current task [23, 61, 3, 40]. The difference between these methods is the association of each network parameter with importance weights stored in incremental learning. *Structure-based methods* dynamically modify the network architecture to maintain existing knowledge and adapt to new tasks. They usually expand the network structure in depth or width when facing a new task [59, 21, 29, 58], or mask parts of the network for different tasks [45, 36, 1].

Recently, some works have researched the use of class prototypes to implement non-exemplar class-incremental learning [66, 60, 67, 50], which somewhat reduces the issues of data privacy security and memory constraints. Yu *et al.* [60] approximate and compensate for the semantic drift of previous tasks during the training of the new task. Based on this, Toldo *et al.* [50] propose to jointly exploit semantic drift and feature drift to update the representations of past classes. Apart from investigating the evolution of prototypes as incremental learning occurs, there are some studies that focus on the ways to model past data distributions by prototypes [66, 67]. Note that this strategy is orthogonal to the above methods for drift estimation, and our method falls into this category. Zhu *et al.* [66] introduce Gaussian noise to augment the prototype to restrain the decision boundaries of old classes. Zhu *et al.* [67] propose a prototype selection mechanism that uses samples of the new data similar to the old classes for distillation to reduce confusion between the newly added classes and the original classes.

2.2. Self-supervised Learning in CIL

The objective of self-supervised learning (SSL) is to acquire transferable representations that are applicable for other tasks. This coincides with the need for IL to inhibit task-level overfitting phenomenon and rapidly adapt to new tasks. Hence, some works explore improving the quality of the learned representations by SSL in the CIL setting [66, 55, 7, 14]. Wu *et al.* [55] learn class-independent knowledge and multi-perspective knowledge by SSL to make a trade-off between gaining new information and maintaining old knowledge. Fini *et al.* [14] attempt to seamlessly convert the existing self-supervised loss function into distillation mechanisms plugged into the CIL framework. It also points out that simply introducing SSL can approach or outperform supervised learning in the CIL setting, while this is not always the case for other settings (data-incremental and domain-incremental). Zhu *et al.* [66] provide better initialization for learning the next task by using rotation as self-supervised information for label augmentation to reduce forgetting from model updates.Our SLA is implemented in the same way as the SSL in [66]. On top of [66], we further improve the upper bound of the incremental learner's understanding of new tasks.

3. Methodology

Problem Statement. In this paper, we consider non-exemplar class-incremental learning (NECIL), where no samples from old classes are stored. The training data for the incremental task $t \in \{1, \ldots, T\}$ is denoted as $D_t = \{X_t, Y_t\} = \{x_t^i, y_t^i\}_{i=1}^{n_t}$, where n_t, x_t^i and $y_t^i \in C_t$ represent the number of training samples, an input sample and its corresponding label for task t, respectively. C_t is the class set of task t and all the incremental classes are disjoint, *i.e.*,

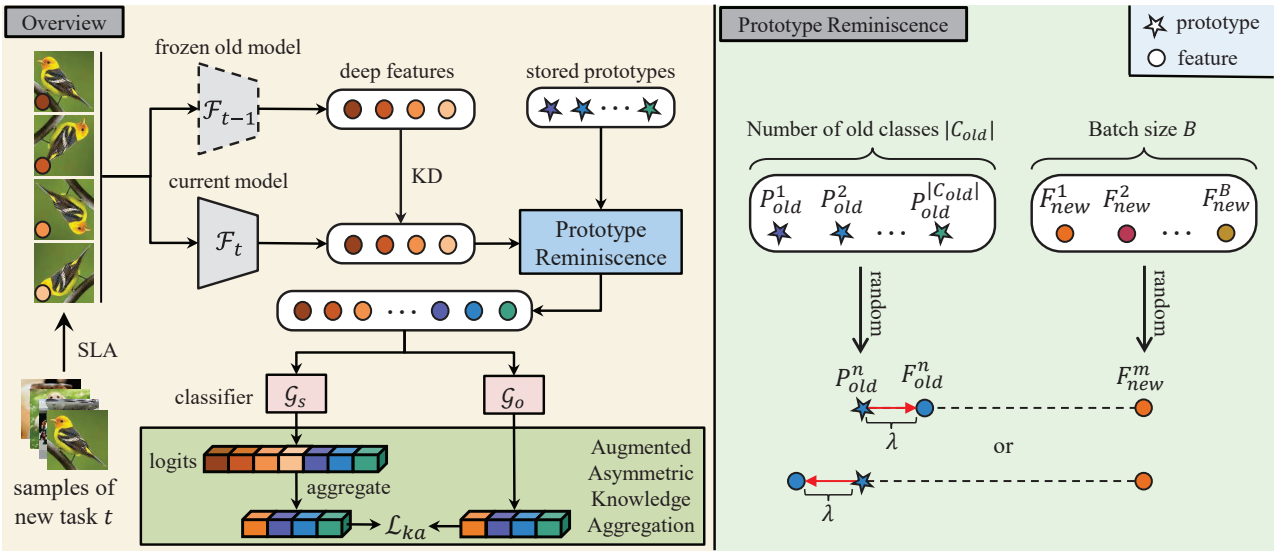

Figure 3. Illustration of our method for NECIL. The classes of current task are augmented by rotation transformation [27]. We expand the stored prototype by prototype reminiscence in the deep feature space to generate the same number of old class features as the batch size each time. During the evaluation, only the current feature extractor $\mathcal{F}_t$ and the refined classifier $\mathcal{G}_o$ are used.

$C_1 \cap \ldots \cap C_i = \varnothing$. We represent the model with two components: a feature extractor $\mathcal{F}$ with parameters θ and a unified classifier $\mathcal{G}$ with parameters φ. The goal is to find a classification model minimized for some loss function $\mathcal{L}(\cdot,\cdot)$:

$$\underset{\theta,\varphi}{\operatorname{argmin}} \sum_{t=1}^{C_t} \mathbb{E}_{(x_t^i, y_t^i)\sim D_t} \left[\mathcal{L}\left(y_t^i, \mathcal{G}(\mathcal{F}(x_t^i;\theta);\varphi)\right)\right]. \quad (1)$$

As other class-incremental methods, the algorithm has no access to the task label t and is expected to classify all observed classes $C_{total} = \{C_1, \ldots, C_t\}$ at any given point.

Overview of Framework. Fig. 3 shows an overview of our method. The classes of the current task are augmented by rotation transformation [27] and the augmented data is fed to the feature extractors. For each old class, we memorize a class-representative prototype. Then the prototype reminiscence uses the extracted new class features and the stored old class prototypes to generate old class features for joint training. We aggregate the knowledge from the classifier $\mathcal{G}_S$ learned with SLA and transfer it to another classifier $\mathcal{G}_O$ that only recognizes classes without augmentation. At test time, the current feature extractor $\mathcal{F}_t$ and the refined classifier $\mathcal{G}_O$ are used for evaluation.

3.1. Prototype Reminiscence

At each incremental phase of task t, only D_t is available in the NECIL setting. Following [66, 60], when learning a new task, we compute and memorize one prototype in the deep feature space for each class:

$$P_{t,k_{new}} = \frac{1}{N_{t,c_{new}}} \sum_{n=1}^{N_{t,c_{new}}} \mathcal{F}\left(\mathbf{X}_{t,c_{new}};\theta_t\right), \quad (2)$$

where $c_{new} \in C_{new} = C_t$ is one class of the current task. To alleviate the catastrophic forgetting, the model is trained jointly on the memorized prototypes $\{P_{old}^1, \ldots, P_{old}^{|C_{old}|}\}$ and current data D_t, where $C_{old} = C_1 \cup \ldots \cup C_{t-1}$ denotes the set of past classes.

Whereas prototypes can inject information of past classes in training, the decision boundaries are narrowed due to the lack of rich old class features. Moreover, as the model continuously learns new data, the deep representation space is changing, which poses a challenge to reproduce the properties of old classes. To this end, we propose a prototype reminiscence strategy to dynamically model the feature distribution of past data in an updating representation space. Our strategy is based on a simple yet effective interpolation operation. In general, as shown in Fig. 3, we randomly select a prototype of the old classes $P_{old}^n \in \{P_{old}^1, \ldots, P_{old}^{|C_{old}|}\}$ and a feature of the new classes $F_{new}^m \in \{F_{new}^1, \ldots, F_{new}^B\}$, where B is the batch size. Then we perform a random bidirectional interpolation operation between P_{old}^n and F_{new}^m:

$$F_{old}^n = \begin{cases} (1-\lambda)\left(P_{old}^n\right) + \lambda\left(F_{new}^m\right), p_e < 0.5 \\ (1+\lambda)\left(P_{old}^n\right) - \lambda\left(F_{new}^m\right), otherwise \end{cases} \quad (3)$$

where F_{old}^n is the feature generated in the current representation space that has the same label as P_{old}^n and p_e is randomly sampled from [0, 1). Considering the design principle of mixup [63, 52], we ensure that the coefficient summary of the two terms for the prototype reminiscence is equal to 1. Different from them, our prototype reminiscence introduces an extrapolation operation, which avoids the generated features being concentrated on one side of the decision boundary. Meanwhile, we choose

$\lambda \sim \text{Beta}(0.5, 0.5) \in [0, \eta]$, where η is a threshold value to control the maximum distance between the generated old class features and the corresponding prototype for avoiding outliers. To make a fair comparison, as in [66], we generate the same number of old class features as the batch size for each batch of new data.

3.2. Augmented Asymmetric Knowledge Aggregation

While counteracting forgetting, incremental learners also need to adapt to new tasks. We design an augmented asymmetric knowledge aggregation scheme that endows the model with stronger plasticity. First, we augment the current classes as in [27]. For the current task, the N-way classification problem is extended into a 4N-way classification problem. Specifically, the input data for each class is rotated by 90°, 180°, and 270° to produce three new classes:

$$\tilde{x_t^i} = \{x_t^{i,j}\}_{j=0}^{3} = \left\{rotate(x_t^i, j \times 90°)\right\}_{j=0}^{3}, \quad (4)$$

and we allocate new labels $\tilde{y_t^i}$ to the augmented data:

$$\tilde{y_t^i} = \left\{y_t^i \times 4 + j\right\}_{j=0}^{3}. \quad (5)$$

Compared to the widely used 4-way self-supervised task, the above approach removes unnecessary invariant properties when learning both the original task and the self-supervised task simultaneously, thereby reducing the loss of discriminative information while obtaining generalizable and transferable representations [27].

However, for NECIL, the self-supervised label augmentation can only be applied to new classes (past data is not available). Borrowing from some works [39, 48], the classification weight $w_{c,j}$ is denoted as a *proxy*, where $c \in C_{total}$, and j represents different rotation transformations. In the subsequent incremental learning, the proxies for the augmented classes (90°, 180°, 270°) of the previous tasks gradually fail, leading to redundancy within the classifier. When using the cross-entropy loss for optimization, all augmented classes are viewed independently, and the relationship between them is not explored. Moreover, only the predictions of the non-augmented classes (0°) are considered valid during testing [66], which leads to a large amount of information valuable for the current task being discarded.

To precisely utilize the valuable knowledge obtained from the above self-supervised label augmentation, we propose an asymmetric knowledge aggregation approach for NECIL. Concretely, for the past tasks, we take in the knowledge of the proxies associated with the original classes:

$$\mathcal{P}_{agg}(c|F_{old}) = \frac{\exp\left(w_{c,0}^T F_{old}\right)}{\sum_{k=1}^{K} \exp\left(w_{k,0}^T F_{old}\right)}, \quad (6)$$

where $K = |C_{total}|$, w is the proxy of $\mathcal{G}_S$ and F_{old} is the feature generated by prototype reminiscence. For the current task, the conditional probabilities corresponding to all transformations of each class are aggregated:

$$\mathcal{P}_{agg}(c|\tilde{F_{new}}) = \frac{\exp\left(\frac{1}{4}\sum_{j=0}^{3} w_{c,j}^T F_{new}^j\right)}{\sum_{k=1}^{K} \exp\left(\frac{1}{4}\sum_{j=0}^{3} w_{k,j}^T F_{new}^j\right)}, \quad (7)$$

where $\tilde{F_{new}} = \{F_{new}^j\}_{j=0}^{3}$ represents the set of features extracted from the augmented exemplars $\tilde{x_t^i}$ by current model $\mathcal{F}_t$. As presented in Fig. 3, we transfer the aggregated knowledge to another classifier $\mathcal{G}_O$, which only needs to distinguish between non-augmented classes. The loss function $\mathcal{L}_{ka}$ can be expressed as follows:

$$\begin{aligned} \mathcal{L}_{ka} &= KLD(\mathcal{P}_{agg}(\cdot|F) \| \mathcal{G}_O(F;\phi)), \\ &= \mathcal{P}_{agg}(\cdot|F) \log \frac{\mathcal{P}_{agg}(\cdot|F)}{\mathcal{G}_O(F;\phi)} \end{aligned} \quad (8)$$

where ϕ denotes the parameters of $\mathcal{G}_O$ and $KLD(\cdot|\cdot)$ is the Kullback-Leibler divergence. The knowledge learned from the new data is utilized as much as possible and the invalid weights are discarded to obtain a more refined classifier. During testing inference, only the classifier $\mathcal{G}_O$ is used.

3.3. Integrated Optimization Objective

Inspiring ourselves from [66], we employ the well-known knowledge distillation (KD) [68, 19] to regularize the feature extractor. Specifically, we minimize the Euclidean distance between the features of new data extracted by current model $\mathcal{F}_t$ and that of previous model $\mathcal{F}_{t-1}$ to constrain the feature extractor, which can be formulated as:

$$\mathcal{L}_{kd} = \|\mathcal{F}_t(\tilde{x_t^i};\theta_t) - \mathcal{F}_{t-1}(\tilde{x_t^i};\theta_{t-1})\|_2. \quad (9)$$

With the assistance of prototype reminiscence, the disparity between the stored prototype and the updated model can be significantly mitigated. We also calculate the cross-entropy loss for $\mathcal{G}_S$ and $\mathcal{G}_O$, respectively. With these considerations above, the total loss function $\mathcal{L}_{total}$ can be expressed as:

$$\begin{aligned} \mathcal{L}_{new} =& \mathcal{L}_{ce}(\mathcal{G}_S(\mathcal{F}_t(\tilde{x_t^i};\theta_t);\varphi_t)) \\ &+ \mathcal{L}_{ce}(\mathcal{G}_O(\mathcal{F}_t(\tilde{x_t^i};\theta_t);\phi_t)) + \mathcal{L}_{ka}, \end{aligned} \quad (10)$$

$$\begin{aligned} \mathcal{L}_{old} =& \mathcal{L}_{ce}(\mathcal{G}_S(F_{old}^i;\varphi_t)) \\ &+ \mathcal{L}_{ce}(\mathcal{G}_O(F_{old}^i);\phi_t)) + \mathcal{L}_{ka}, \end{aligned} \quad (11)$$

$$\mathcal{L}_{total} = \mathcal{L}_{new} + \alpha\mathcal{L}_{old} + \gamma\mathcal{L}_{kd}, \quad (12)$$

where $\mathcal{L}_{new}$ is the loss of new data $\tilde{x_t^i}$, $\mathcal{L}_{old}$ is the loss of the feature F_{old}^i generated by prototype reminiscence. The loss weights α and γ are both set to 15 in our experiments.

4. Experiments

4.1. Experimental Setting

Dataset. To evaluate the performance of our method, we conduct sufficient experiments on three benchmarks includ-

Table 1. Quantitative comparisons of the average incremental accuracy (%) with other methods at different task number settings on CIFAR-100, TinyImageNet, and ImageNet-Subset. *E=20* represents exemplar-based methods and storing 20 exemplars for each old class. *E=0* represents non-exemplar methods. The relative improvement compared to the SOTA NECIL method is shown by the red footnotes.

	Methods	CIFAR-100			TinyImageNet			ImageNet-Subset
		5 phases	10 phases	20 phases	5 phases	10 phases	20 phases	10 phases
E = 20	iCaRL-CNN	51.07	48.66	44.43	34.64	31.15	27.90	50.53
	iCaRL-NCM [42]	58.56	54.19	50.51	45.86	43.29	38.04	60.79
	EEIL [9]	60.37	56.05	52.34	47.12	45.01	40.50	63.34
	UCIR [20]	63.78	62.39	59.07	49.15	48.52	42.83	66.16
	DER [58]	73.21	72.81	69.97	—	—-	—	75.36
E = 0	EWC [23]	24.48	21.20	15.89	18.80	15.77	12.39	20.40
	LwF_MC [42]	45.93	27.43	20.07	29.12	23.10	17.43	31.18
	MUC [32]	49.42	30.19	21.27	32.58	26.61	21.95	35.07
	SDC [60]	56.77	57.00	58.90	—	—	—	61.12
	PASS [66]	63.47	61.84	58.09	49.55	47.29	42.07	61.80
	SSRE [67]	65.88	65.04	61.70	50.39	48.93	48.17	67.69
	Ours	**70.02**+4.14	**68.86**+3.82	**65.86**+4.16	**53.32**+2.93	**52.61**+3.68	**49.83**+1.66	**68.98**+1.29

ing CIFAR-100 [25], TinyImageNet [26] and ImageNet-Subset (random seed 1993) [12]. For CIFAR-100 and TinyImageNet there are three incremental settings (5, 10, and 20 phases) and for ImageNet-Subset there is one (10 phases). For the sequencing and division of all dataset classes, we strictly follow the specification in [66].

Implementation details. Following [66], we use ResNet-18 [18] as the backbone network. The model is trained for 100 epochs on each new task. For comparisons on all datasets, the threshold of our prototype reminiscence is set to 0.6. During training, the batch size is set to 128 and the model is optimized by the Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e^{-8}$ (weight decay 2e-4). The initial learning rate is 0.001 and is adjusted with the cosine annealing algorithm [35] with a period of 32 epochs.

Evaluation metrics. As with [66], we employ *average incremental accuracy* [42] and *average forgetting* [10] as the evaluation metric. The average incremental accuracy is defined as the average of the accuracies over seen classes across all incremental phases (including the first phase), which reflects the overall incremental performance of the method. The average forgetting is the average difference between the peak task accuracy and the final task accuracy after incremental learning is completed, and the lower value represents better performance.

4.2. Comparison with SOTA

We compare our method with the state-of-the-art (SOTA) methods of NECIL (EWC [23], LwF_MC [42], MUC [32], SDC [60], PASS [66] and SSRE [67]) and several classical exemplar-based CIL approaches (iCaRL [42], EEIL [9] and UCIR [20]). The results reported for PASS are obtained with self-supervised learning. We reproduce the SSRE with label augmentation in the supplementary materials. As adopted in [67], we memorize 20 samples for the exemplar-based methods.

Accuracy and forgetting. The quantitative comparisons of the average incremental accuracy are reported in Tab. 1. Compared to the suboptimal NECIL method (SSRE), we improve the accuracy on CIFAR-100, TinyImageNet and ImageNetSubset by 6.30%, 5.59%, and 1.91%, respectively. Moreover, our method shows competitive or even better performance than the classical exemplar-based methods. Certainly, it still slightly inferior to the SOTA exemplar-based method DER [58]. In Fig. 4 we show the accuracy change curves on three benchmarks. Our method is superior at almost all phases, achieving a better balance between stability and plasticity. To further evaluate the above methods, we provide the results of average forgetting in Tab. 2, where our method also performs superiorly. In addition, since the proposed asymmetric knowledge aggregation (AKA) improves the classification accuracy on the current task, the forgetting metric is increased, however this is harmless to overall performance. We will report more analysis in the following and supplementary materials. Experimental results show that our approach can mitigate catastrophic forgetting despite the lack of past task data, which is a more privacy-safe and memory-friendly manner to achieve CIL.

Table 2. Results of average forgetting on 5, 10 and 20 phases.

Methods	CIFAR-100			TinyImageNet		
	5	10	20	5	10	20
iCaRL-CNN	42.13	45.69	43.54	36.89	36.70	45.12
iCaRL-NCM	24.90	28.32	35.53	27.15	28.89	37.40
EEIL	23.36	26.65	32.40	25.56	25.51	35.04
UCIR	21.00	25.12	28.65	20.61	22.25	33.74
LwF_MC	44.23	50.47	55.46	54.26	54.37	63.54
MUC	40.28	47.56	52.65	51.46	50.21	58.00
PASS	25.20	30.25	30.61	18.04	23.11	30.55
SSRE	18.37	19.48	19.00	9.17	14.06	14.20
Ours	12.59	14.65	17.39	11.84	13.95	18.51
Ours w/o AKA	**7.18**	**6.42**	**10.30**	**4.57**	**5.10**	**9.05**

4.3. Ablation Study

In this section, we conduct several ablation studies on the proposed method. To better analyze the impact of the core designs, our approach is divided into three compo-

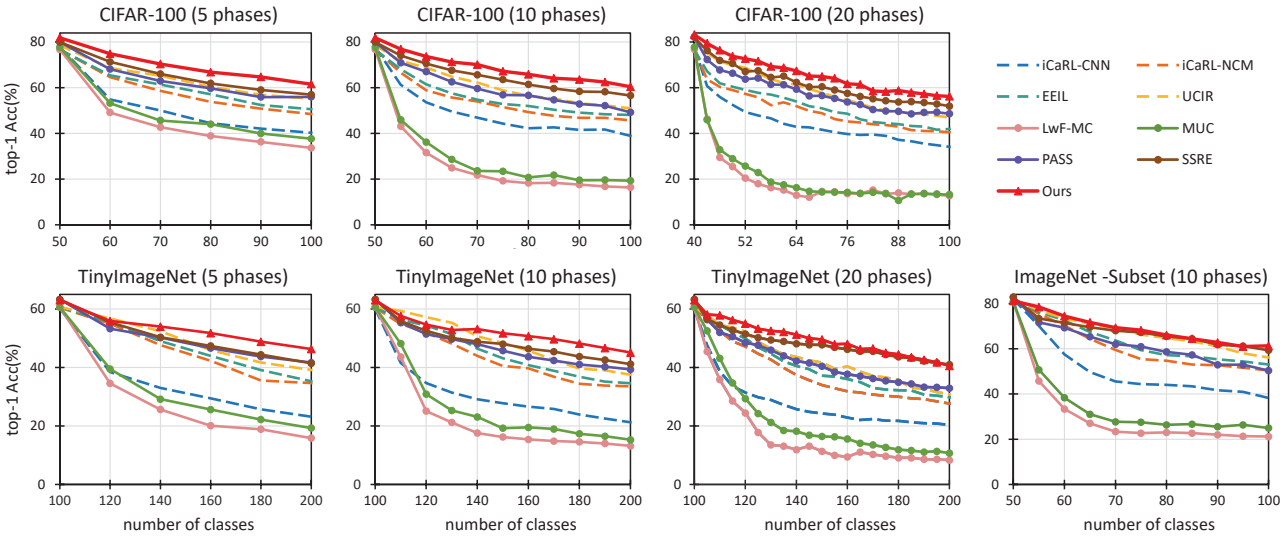

Figure 4. Illustration of the classification accuracy changes as tasks are being learned on CIFAR-100, TinyImageNet and ImageNet-Subset, which contains the complete curves. Precise data of our method is presented in the supplementary materials.

Table 3. Ablation study (in average incremental accuracy) of our method on CIFAR-100 and TinyImageNet datasets.

Components			CIFAR-100			TinyImageNet		
PR	SLA	AKA	5 phases	10 phases	20 phases	5 phases	10 phases	20 phases
			56.27	51.02	43.98	37.93	32.44	23.98
✓			66.21	63.80	57.31	45.85	44.04	35.93
	✓		61.27	59.59	55.14	46.65	43.88	38.13
✓	✓		68.48	67.56	65.03	52.15	51.67	49.30
✓	✓	✓	**70.02**	**68.86**	**65.86**	**53.32**	**52.61**	**49.83**

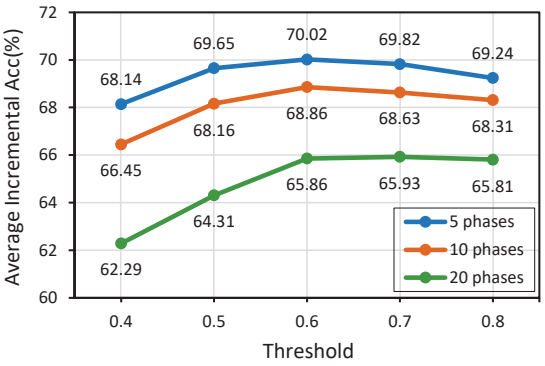

Figure 5. Influence of the threshold in prototype reminiscence.

nents: prototype reminiscence (PR), self-supervised label augmentation (SLA) and asymmetric knowledge aggregation (AKA). As a general trick, KD was used in all experimental settings. To explicitly illustrate the effectiveness of PR, in the baseline model we save the prototypes and directly use them for training. As shown in the 1^{st} and 2^{nd} rows of Tab. 3, PR successfully maintains the discrimination and balance between the old and new classes and makes an impressive progress on the baseline model. When SLA acts on the baseline model alone, it is effective, but there is still a bias in the classifier, while using it with PR achieves promising results. AKA and SLA are adopted as a whole. In addition to the overall performance improvement, AKA is more about enhancing the model's understanding of the new task to satisfy the latest business requirements.

To investigate the sensitivity of the threshold η in our prototype reminiscence, we plot its fluctuation curve on CIFAR-100 dataset. As presented in Fig. 5, when η is low, the generated old class features are restricted to concentrate around the preserved prototypes thus lacking the ability to adapt to the changes of the representation space, which results in poor performance. As the constraint is relaxed (η increases), the effect of PR gradually comes into its own and peaks in performance at about $\eta = 0.6$. When the threshold continues to increase, the performance gain from PR starts to diminish because as the constraint fades away, the number of outlier features generated by PR gradually increases, which disturbs the learning of the model.

4.4. Analysis

Visualization. To demonstrate the advantages of our prototype reminiscence, we visualise the 2D embeddings of feature vectors with t-SNE [51]. In Fig. 6 (c) and (d), the old class features are generated by the stored prototypes. Compared with fine-tuning, the confusion of old class features is alleviated in the results of Gaussian noise augmentation. However, the overlap between the distributions of different

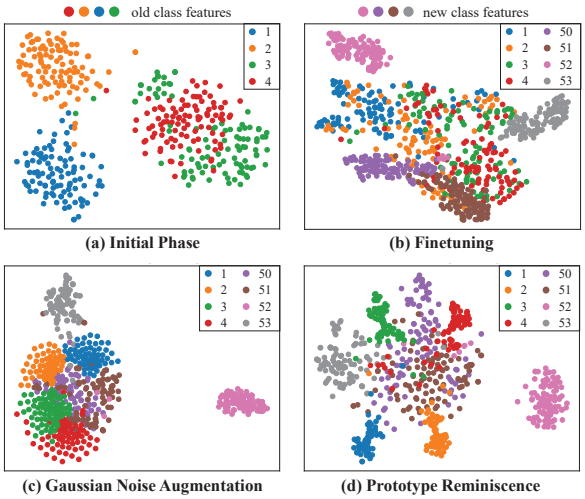

Figure 6. Visualization of different augmentation schemes.

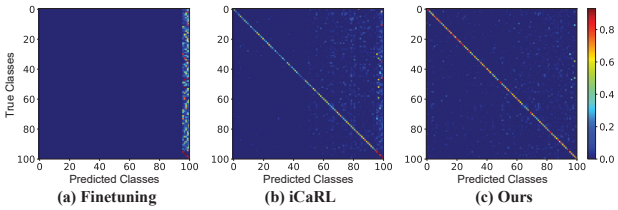

Figure 7. The comparison of confusion matrix of finetuning, iCaRL and our method on CIFAR-100 (10 phases).

classes is still significant, and in particular the confusion between the generated features. Instead, the feature distributions of different old classes reshaped by our method are relatively dispersed among each other.

Classification bias of the model. To evaluate the performance differences between the old and new classes, we compared the confusion matrix of finetuning, iCaRL and our method on CIFAR-100. As shown in Fig. 7, since samples of old classes are not available, finetuning tends to classify the samples into new classes. The confusion is alleviated by iCaRL, but still shows a preference for new classes. The confusion matrix of our method exhibits better overall performance without favouring old or new classes. Thanks to PR and AKA, the classification bias between old and new classes is well handled.

The role of asymmetric knowledge aggregation. To better illustrate the role of the asymmetric knowledge aggregation, we compare the changes in classification accuracy for the initial task during subsequent incremental learning, and the performance on each new task in the presence or absence of AKA. The results for the three settings on CIFAR-100 are shown in Fig. 8. On the one hand, the performance degradation in the initial task caused by the introduction of AKA is negligible; on the other hand, the discrimination of new classes is considerably improved. As analyzed in the introduction, AKA can improve the model's understanding of new tasks as non-destructively as possible.

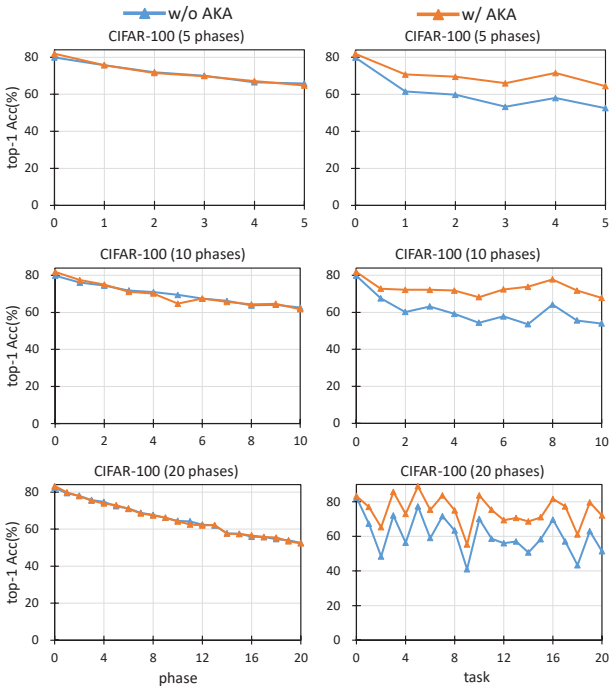

Figure 8. The first column is the classification accuracy on the initial task, which changes with the incremental phase. The second column shows the change in classification accuracy on the new task. The proposed knowledge aggregation allows better learning of new tasks with little impact on existing knowledge.

5. Conclusion

In this work, we address the NECIL issue from two perspectives of resisting catastrophic forgetting and boosting plasticity of the model, proposing prototype reminiscence and augmented asymmetric knowledge aggregation, respectively. In particular, we pay attention to the performance of the incremental learner on new tasks which often represent the latest business requirements, and design an asymmetric knowledge aggregation strategy for better adaptation to new tasks. Extensive evaluations demonstrate that our approach outperforms the SOTA NECIL methods and shows strong competitiveness with classical exemplar-based methods in the absence of stored samples.

Limitations. For each prototype reminiscence, the prototype and the feature are selected randomly. When the selected prototype and feature are less similar (*i.e.*, farther apart in the feature space), there may be outliers in the generated features of old classes. Exploring the impact of outliers on training may be of interest.

Acknowledgement. This work is partially supported by the Key Research and Development Program of Hubei Province (2021BAA187), National Natural Science Foundation of China under Grant (62176188), Zhejiang lab (NO.2022NF0AB01), the Special Fund of Hubei Luojia Laboratory (220100015) and CAAI-Huawei MindSpore Open Fund.

References

[1] Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *CVPR*, pages 3931–3940, 2020. 3

[2] Hongjoon Ahn, Jihwan Kwak, Subin Lim, Hyeonsu Bang, Hyojun Kim, and Taesup Moon. Ss-il: Separated softmax for incremental learning. In *ICCV*, pages 844–853, 2021. 1

[3] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *ECCV*, pages 139–154, 2018. 2, 3

[4] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *ICML*, pages 214–223, 2017. 2

[5] Jihwan Bang, Heesu Kim, YoungJoon Yoo, Jung-Woo Ha, and Jonghyun Choi. Rainbow memory: Continual learning with a memory of diverse samples. In *CVPR*, pages 8218–8227, 2021. 3

[6] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *CVPR*, pages 583–592, 2019. 1

[7] Prashant Bhat, Bahram Zonooz, and Elahe Arani. Task agnostic representation consolidation: a self-supervised based continual learning approach. *arXiv preprint arXiv:2207.06267*, 2022. 3

[8] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *NeurIPS*, pages 15920–15930, 2020. 1

[9] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. End-to-end incremental learning. In *ECCV*, pages 233–248, 2018. 6

[10] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, pages 532–547, 2018. 6

[11] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a-gem. In *ICLR*, 2018. 3

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 6

[13] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle. Podnet: Pooled outputs distillation for small-tasks incremental learning. In *ECCV*, pages 86–102, 2020. 3

[14] Enrico Fini, Victor G Turrisi da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. Self-supervised models are continual learners. In *CVPR*, pages 9621–9630, 2022. 3

[15] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, pages 128–135, 1999. 1

[16] Qiankun Gao, Chen Zhao, Bernard Ghanem, and Jian Zhang. R-dfcil: Relation-guided representation learning for data-free class incremental learning. In *European Conference on Computer Vision*, pages 423–439. Springer, 2022. 1

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 2

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6

[19] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 5

[20] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, pages 831–839, 2019. 2, 6

[21] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. In *NeurIPS*, 2019. 3

[22] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In *AAAI*, 2018. 3

[23] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, pages 3521–3526, 2017. 2, 3, 6

[24] Nikos Komodakis and Spyros Gidaris. Unsupervised representation learning by predicting image rotations. In *ICLR*, 2018. 2

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

[26] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 6

[27] Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Self-supervised label augmentation via input transformations. In *ICML*, pages 5714–5724, 2020. 2, 4, 5

[28] Guopeng Li, Yue Xu, Jian Ding, and Gui-Song Xia. Towards generic and controllable attacks against object detection. *arXiv preprint arXiv:2307.12342*, 2023. 1

[29] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *ICML*, pages 3925–3934, 2019. 3

[30] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, pages 2935–2947, 2017. 1

[31] Yaoyao Liu, Yingying Li, Bernt Schiele, and Qianru Sun. Online hyperparameter optimization for class-incremental learning. In *AAAI*, 2023. 1

[32] Yu Liu, Sarah Parisot, Gregory Slabaugh, Xu Jia, Ales Leonardis, and Tinne Tuytelaars. More classifiers, less forgetting: A generic multi-classifier paradigm for incremental learning. In *ECCV*, pages 699–716, 2020. 6

[33] Yaoyao Liu, Bernt Schiele, and Qianru Sun. Rmm: Reinforced memory management for class-incremental learning. In *NeurIPS*, pages 3478–3490, 2021. 3

[34] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *NeurIPS*, 2017. 3

[35] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6

[36] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *ECCV*, pages 67–82, 2018. 3

[37] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *CVPR*, pages 7765–7773, 2018. 2

[38] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, pages 109–165. 1989. 1

[39] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *CVPR*, pages 360–368, 2017. 5

[40] Inyoung Paik, Sangjun Oh, Taeyeong Kwak, and Injung Kim. Overcoming catastrophic forgetting by neuron-level plasticity control. In *AAAI*, pages 5339–5346, 2020. 3

[41] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *ECCV*, pages 524–540, 2020. 1

[42] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, pages 2001–2010, 2017. 1, 3, 6

[43] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *ICLR*, 2018. 3

[44] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 2

[45] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, pages 4548–4557, 2018. 3

[46] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *NeurIPS*, 2017. 2

[47] James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *ICCV*, pages 9374–9384, 2021. 1

[48] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *NeurIPS*, 30, 2017. 5

[49] Yu-Ming Tang, Yi-Xing Peng, and Wei-Shi Zheng. Learning to imagine: Diversify memory for incremental learning using unlabeled data. In *CVPR*, pages 9549–9558, 2022. 2

[50] Marco Toldo and Mete Ozay. Bring evanescent representations to life in lifelong class incremental learning. In *CVPR*, pages 16732–16741, 2022. 2, 3

[51] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008. 7

[52] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, pages 6438–6447, 2019. 4

[53] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *European conference on computer vision*, pages 398–414. Springer, 2022. 1

[54] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. In *NeurIPS*, 2018. 2

[55] Guile Wu, Shaogang Gong, and Pan Li. Striking a balance between stability and plasticity for class-incremental learning. In *ICCV*, pages 1124–1133, 2021. 2, 3

[56] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, pages 374–382, 2019. 1, 3

[57] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang. Incremental learning using conditional adversarial networks. In *ICCV*, pages 6619–6628, 2019. 2

[58] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *CVPR*, pages 3014–3023, 2021. 2, 3, 6

[59] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. In *ICLR*, 2018. 2, 3

[60] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *CVPR*, pages 6982–6991, 2020. 2, 3, 4, 6

[61] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, pages 3987–3995, 2017. 2, 3

[62] Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong gan: Continual learning for conditional image generation. In *ICCV*, pages 2759–2768, 2019. 2

[63] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 4

[64] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *WACV*, pages 1131–1140, 2020. 2

[65] Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-lin Liu. Class-incremental learning via dual augmentation. *NeurIPS*, pages 14306–14318, 2021. 2

[66] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *CVPR*, pages 5871–5880, 2021. 1, 2, 3, 4, 5, 6

[67] Kai Zhu, Wei Zhai, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Self-sustaining representation expansion for non-exemplar class-incremental learning. In *CVPR*, pages 9296–9305, 2022. 2, 3, 6

[68] Xiatian Zhu, Shaogang Gong, et al. Knowledge distillation by on-the-fly native ensemble. *NeurIPS*, 31, 2018. 5