ECAI 2023
K. Gal et al. (Eds.)
© 2023 The Authors.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/FAIA230599

Data-Free Class-Incremental Learning with Implicit Representation of Prototypes

Tianwen Yang[a], Leixiong Huang[a] and Ronghua Luo[a;*]

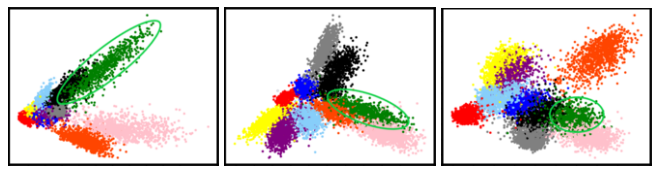
[a]South China University of Technology, Guangzhou, China
g920002673@gmail.com, {202221044358@mail, rhluo}.scut.edu.cn

Abstract. Class-incremental learning (CIL) has attracted much attention in deep learning due to the challenge problem of catastrophic forgetting. Various methods have been proposed for CIL, including exemplar-based class-incremental learning (EBCIL), non-exemplar class-incremental learning (NECIL) and data-free class-incremental learning (DFCIL). Without storing any information (such as examples and prototypes) about the old classes, DFCIL is obviously the most challenging one. To address the problem of lacking information in DFCIL and with the assumption that the learned representations are not linearly separable, we propose a method called IRP. We use the L2-similarity classifier instead of the FC classifier, where each weight vector represents a prototype that implicitly records information about the classes. We use representation-prototype distance minimization (RPDM) to solve the problem of loose representation caused by overfitting. To alleviate the excessive deviation of old prototypes under long-term CIL, we add prototype changing limitation (PCL) and prototype momentum updating (PMU) in incremental stages. In addition, we design a method for resampling around old prototypes (RAOP) to maintain the decision boundary of the old classes. Numerous experiments on three benchmarks have shown that IRP is significantly superior to other DFCIL methods and performs comparably to NECIL and partial EBCIL methods.

1 Introduction

Deep learning has made significant progress in visual recognition, with accuracy comparable to or surpassing that of humans in daily tasks such as facial recognition and autonomous driving. However, a large amount of labeled data is needed in the training process to achieve these achievements. Furthermore, data is often non-independent identically distributed (Non-IID), making it difficult to obtain a large amount of training data at once in many cases. Usually, over time, more and more new data will be available, while old data will gradually become unavailable due to storage limitations or privacy protection. If we train a deep learning model with this kind of streamed data, especially when the data has unlearned classes, the model would be more inclined to adapt to new knowledge and forget the old knowledge, which is the catastrophic forgetting problem faced by deep learning [24, 14]. To alleviate the catastrophic forgetting problem, incremental learning has been proposed. Class-incremental learning (CIL) [10, 27] is relatively tricky among numerous incremental learning scenarios.

* Corresponding Author.



(a) FC (b) L2 (c) L2 + RPDM

Figure 1: Illustrattion of the representations learned by different classifiers on the MNIST [19]. (a) The assumption of linear separability influences the FC classifier, and the intra-class distance is greater than the inter-class distance. (b) When using only the L2-similarity classifier, the learned representation has the same problem of loose representation as the FC classifier. (c) RPDM effectively alleviates the problem of loose representation in L2-similarity classifiers.

Many works have focused on storing and replaying a small portion of samples from the old classes, namely the EBCIL method [27, 2, 4, 11], which alleviates catastrophic forgetting through joint training samples of the new and old classes. However, due to the need to retain old samples, these EBCIL methods have to face the storage pressure of old samples and cannot protect the privacy and security of data, which is unsuitable for many practical scenarios. To address these shortcomings, recent works on CIL have proposed several methods that only retain some processed information of old classes (such as prototypes), namely the NECIL method [38, 39, 37, 35, 16, 21, 23], which tries to alleviate the forgetting problem by retaining the processed representation of old classes and maintaining the decision boundary of the old classes. However, the information of old classes is still extracted from the old samples. It may still involve privacy and security issues, and as CIL progresses, the capacity of old information increases linearly, which still brings a storage burden. To completely overcome privacy threats and storage pressures, a more difficult DFCIL [29, 6, 30] scenario is proposed where old samples and information cannot be retained. In other words, except for the model (encoder and classifier), the capacity does not increase with the CIL stage.

DFCIL often has the worst performance among these three types of CIL, and it is easy to recognize that this is because DFCIL does not retain any samples and information from the old classes, which makes the model more inclined to learn new classes and forget old ones. In the incremental training process, the new and old classes overlap in the embedding space, and it is difficult to maintain the decision boundary of the old classes. Recently, some DFCIL works have been inspired by generating old pseudo-samples based on the

generative model [32, 13]. They introduce inversion [29, 6] into the model, use the old model to generate pseudo-samples and alleviate catastrophic forgetting through the joint training of pseudo-samples and new samples. However, due to the poor quality of pseudo-sample generation, it cannot effectively represent the old classes, resulting in unsatisfactory model performance.

In addition, many CIL works pay too much attention to improving the stability of the encoder while neglecting the vital role of classifier in mitigating catastrophic forgetting. The general CIL uses a linear layer as the classifier, known as the FC classifier. From the perspective of representation learning, using the FC classifier implies the assumption that the representation learned by the model is linearly separable, as shown in Fig. 1(a). This kind of representation can easily lead to the intra-class distance being more significant than the inter-class distance and reducing the robustness of the model. When learning for new tasks, the model tends to learn new classes, while the old knowledge is quickly forgotten. As [11, 4] has noted, using Cosine-similarity classifier instead of FC classifier can alleviate the negative impact caused by class imbalance.

To address these issues, we propose the DFCIL method with implicit representation of prototypes, called IRP[1], which replaces the FC classifier used in general CIL models with the L2-similarity classifier. Each weight vector of the L2-similarity classifier represents the corresponding prototype, implicitly providing the model with old information while avoiding the assumption of linear separability of representations in FC classifier. Then, the representation-prototype distance minimization (RPDM) is used to shorten the L2 distance between the representations of new class and the new prototypes to solve the problem of loose representation caused by the overfitting of the L2-similarity classifier. In addition, we use prototype changing limitation (PCL) and prototype momentum updating (PMU) to alleviate the problem of excessive deviation of the old prototypes during the training process. Finally, a method of resampling around old prototypes (RAOP) is proposed. The new classifier is jointly trained by the resampled old prototypes and the new representations to maintain the decision boundary of old classes. In addition, [38, 5, 12, 30] points out that self-supervised learning (SSL) can help alleviate the problem of model overfitting in CIL and improve performance. Therefore, our model uses the same SSL as [38] to alleviate overfitting. In summary, our main contributions are as follows:

1. We propose a new DFCIL method which implicitly represents the prototypes with an L2-similarity classifier while avoiding linearly separable assumptions of representation. Moreover, propose a representation-prototype distance minimization (RPDM) mechanism to solve the problem of loose representation.
2. We propose a prototype changing limitation (PCL) and a prototype momentum updating (PMU) method to alleviate the excessive deviation of the old prototypes.
3. We also propose a method for resampling around old prototypes (RAOP) to maintain the decision boundary of the old classes.
4. Many experiments on three benchmarks have shown that our method is significantly superior to the DFCIL methods and has comparable performance to NECIL and partial EBCIL methods.

2 Related works

With the development of deep learning, the demand for incremental learning of new knowledge in models is increasing. The focus of incremental learning is to enable the model to have the ability to learn new knowledge without forgetting old knowledge and to achieve a good balance between stability and plasticity. The class-incremental learning (CIL) has always been one of the most challenging scenarios in incremental learning.

CIL can be divided into three categories from the perspective of utilizing old data: exemplar-base class-incremental learning (EBCIL), non-exemplar class-incremental learning (NECIL) and data-free class-incremental learning (DFCIL). The EBCIL has always been a hot topic in solving catastrophic forgetting, using joint training of new and old samples to achieve a balance between stability and plasticity [27, 2, 4, 11]. Some works go further based on sample replay, expanding the model during incremental learning and utilizing additional modules to adapt to new tasks [33, 26, 1]. In addition, many works [4, 11, 27] have focused on using knowledge distillation [9] to alleviate catastrophic forgetting in the encoder. Recently, [11, 31] have focused on addressing the imbalance between old and new classes, alleviating catastrophic forgetting in models from the perspective of suppressing models that lean more towards new class. Due to the need for the EBCIL method to retain some old samples for each old class, the model capacity increases linearly with the CIL process, which can cause significant storage pressure in the long-term CIL process. Furthermore, the EBCIL method involves privacy issues, and retaining old samples is not allowed in many practical scenarios. These two problems prevent the EBCIL method from being well applied to practical tasks.

In order to be closer to the actual situation, many work focus on a more complicated scenario called NECIL, which preserves some old information to alleviate catastrophic forgetting without saving old samples. Some works attempt to add a generator and use its pseudo old samples for joint training [32, 13]. Recently, [38, 39] have found that only retaining the old prototypes can provide enough old information for the model. While using knowledge distillation to alleviate the encoder's catastrophic forgetting, the decision boundary of the old classes can be maintained through the joint training of old prototypes and new representations. The NECIL method performs incremental learning using information extracted from the old training data, and does not require old samples. But the extracted information may be still sensitive and the requirement of additional structures to store the information of the old classes will increase the model capacity as the CIL stage progresses.

Some works explore the balance between stability and plasticity in the DFCIL scenario. [29, 6] are inspired by generating old pseudo samples based on the generative model. They introduce inversion[34] and use the old model to generate pseudo samples. [30] attempt to use multiple classifiers to deepen the understanding of old samples by the encoder, to alleviate catastrophic forgetting. Unlike previous works, we use the implicit representation of prototypes to implement DFCIL.

3 Problem statement

CIL is a process in which a model continuously learns new knowledge while preventing forgetting old knowledge. Given a continuous data stream D, $D^t = \{X^t, Y^t\}$ is the dataset in task t, where X is the sample set and Y is the label set. The class set in D^t is C^t, and the number of classes is $|C^t|$. The datasets of different tasks are not intersecting, that is, $D^i \cap D^j = \emptyset$, $i \neq j$. When $t = 0$, it is the initial stage and $t > 0$ is the incremental stage. The model can only be trained using the current stage dataset D^t and tested using all the trained datasets $D^{0:t}$ so far in the CIL process. On this basis, the EBCIL allows for retaining a small number of old samples and train-

[1] IRP code is available at https://github.com/YTWWW/IRP

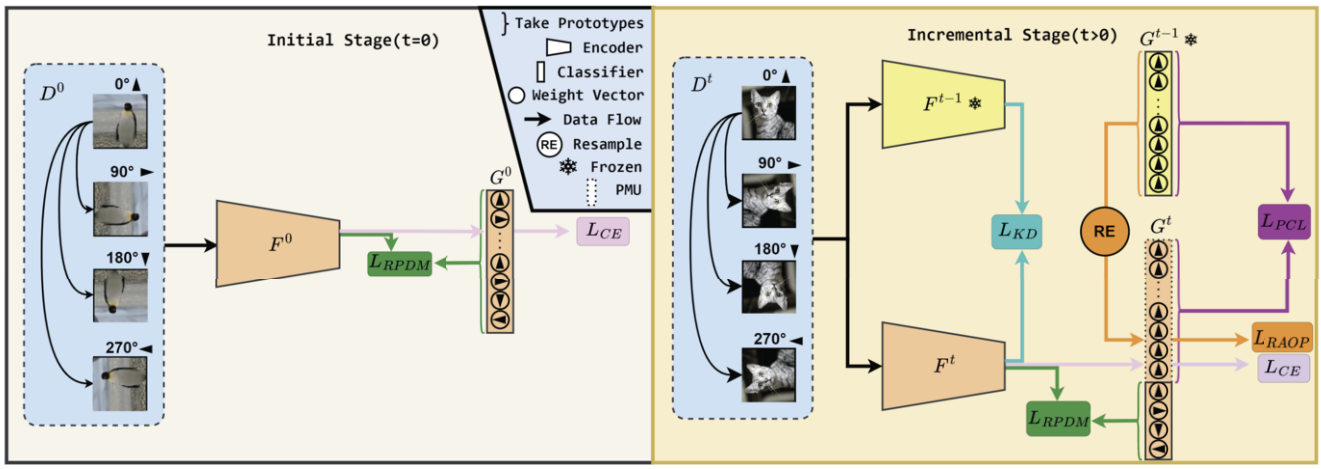


Figure 2: Illustration of IRP. (i) IRP is a DFCIL method where the L2-similarity classifier implicitly provides prototypes for the model. (ii) RPDM narrows the L2 distance between the representation and the corresponding prototype, solving the problem of loose representation in L2-similarity classifier. (iii) PCL and PMU alleviate prototype offset. (iv) RAOP maintains the decision boundary of the old classes. (v) The SSL transforms the k-class classification problem into the 4k-class classification problem.

ing the model together with D^t during training. The NECIL does not allow the preservation of old samples but allows the preservation of information about old classes beyond the storage model (encoder and classifier) itself, such as prototypes. The DFCIL does not allow for retaining old samples and information; only the model can be retained.

4 Methodology

The framework of IRP is shown in Fig. 2. IRP is a DFCIL method that does not explicitly store any old samples and information. Firstly, the L2-similarity classifier replaces the FC classifier commonly used in CIL. Each weight vector of the L2-similarity classifier represents the prototype of corresponding class. In the process of new classes training, the new representations and the new prototypes are used to calculate the representation-prototype distance minimization (RPDM) to shorten the L2 distance between them and solve the loose representation problem caused by the overfitting of the L2-similarity classifier. Then, the old prototypes of the new and old classifiers are constrained by prototype changing limitation (PCL) and update the old prototypes by prototype momentum updating (PMU) method, to alleviate the excessive deviation of the old prototypes. In order to maintain the decision boundary of the old classes, we use the resampling around old prototypes (RAOP) method to jointly train the classifier with the resampled old prototypes and the new representations.

Similar to the previous CIL methods, we use the cross entropy loss function to adapt the model to image classification tasks; use knowledge distillation to alleviate catastrophic forgetting in the encoder; use the same SSL as [38] to alleviate the problem of overfitting while making the current embedding space more compact and leaving room for new classes.

In the following sections, we will provide the detailed description of our main contributions.

4.1 *Implicit representation of prototypes*

In DFCIL, due to not retaining any old samples and information, the representations of new classes will inevitably overlap and cover the representations of old classes, resulting in the forgetting of the old knowledge learned by the model. To address this issue, we use the L2-similarity classifier instead of the FC classifier, implicitly providing the model with information of old prototypes while avoiding the assumption of linear separability of representation in the FC classifier.

In the t-th incremental stage, only D^t can be used for training, where the old class set that the model has already seen is $C^{0:t-1}$. The new class set that has not been seen is C^t. Unlike FC classifiers, L2-similarity classifiers use L2 distance as a metric. The process of calculating classification probability is as follows:

$$p_c(x^t) = \frac{\exp(-||w_c - F(x^t)||_2)}{\sum_{c \in C^{0:t}} \exp(-||w_c - F(x^t)||_2)} \tag{1}$$

Where F is the encoder, $w = [w_1, w_2, w_3, \cdots, w_{|C^{0:t}|}]$ is the weight vector of the L2-similarity classifier and $x^t \in X^t$ is the new sample. Image classification loss uses the standard cross entropy loss function, defined as:

$$L_{CE} = -\frac{1}{|X^t|} \sum_{x^t \in X^t} y_{(x^t)} \cdot \log p(x^t) \tag{2}$$

Where $|X^t|$ represents the number of new samples, $y_{(x^t)}$ is the label of x^t, $y_{(x^t)} \in Y^t$. The weight w_c of the L2-similarity classifier represents the prototype of class c in the embedding space. Without explicitly storing the prototypes, the L2-similarity classifier still implicitly provides the prototypes for the model, thus overcoming the difficulty of DFCIL lacking old information. The essence of the L2-similarity classifier is to shorten the L2 distance between the representations and the corresponding prototypes, that is, to minimize $||w_{(x)} - F(x)||_2$. As shown in Fig. 1(b), the representation is loose when only the L2-similarity classifier is used instead of the FC classifier, which is caused by the overfitting. To address the problem of loose representation in L2-similarity classifiers, we propose a representation-prototype distance minimization as a regularization, described in section 4.2. For the convenience of understanding, we will refer to the weight vector w_c of class c in the L2-similarity classifier as the prototype of class c.

4.2 Representation-prototype distance minimization (RPDM)

During the experiment, it was found that simply using the L2-similarity classifier to replace the FC classifier does not achieve ideal results. Representations of some samples are far from their corresponding prototypes, which will lead to a loose representation of classes. So, within incremental learning, the decision boundary of the old classes will gather around the old prototypes, making it impossible to effectively classify the old representations far away from the old prototypes. This is the loose representation problem faced by the L2-similarity classifier. To address this issue, we use the representation-prototype distance minimization to shorten the L2 distance between the representations and the corresponding prototypes, which is defined as:

$$L_{RPDM} = \frac{1}{|X^t|} \sum_{x^t \in X^t} ||w_{(x^t)} - F(x^t)||_2 \tag{3}$$

Where $|X^t|$ represents the number of new samples, $w_{(x)}$ represents the weight vector of classifier corresponding to the class to which sample x belongs. As shown in Fig. 1(c), the RPDM effectively shortens the L2 distance between the representations and the corresponding prototypes so that the representations can be clustered around the corresponding prototypes to solve the problem of loose representation. Thus, when the new classes arrives and the decision boundary of old classes is compressed, the old representations are not easy to classify wrongly as it is close to the corresponding prototypes.

4.3 Prototype changing limitation (PCL)

Inspired by the idea of limiting encoder changes through knowledge distillation [9, 27, 4], we improve the stability of the model by limiting the changes of old prototypes in the L2-similarity classifier. Specifically, we use the L2 distance between the old prototypes in the new classifier and the prototypes in the old classifier as a constraint to alleviate the severe deviation phenomenon of the old prototypes when only new data is available. The PCL is defined as:

$$L_{PCL} = \frac{1}{|C^{0:t-1}|} \sum_{c \in C^{0:t-1}} ||w_c^t - w_c^{t-1}||_2 \tag{4}$$

Where $|C^{0:t-1}|$ represents the number of old classes, and w_c^t is the weight vector corresponding to class c in the classifier of stage t. By minimizing this loss function, we limit the excessive deviation of the old prototypes in the incremental learning process. On the one hand, the old prototypes play a role in occupying the new embedding space for the old classes and improving the stability of the model. On the other hand, the appropriate deviation of the old prototypes can leave room for the new classes, improve the plasticity of the model, and finally achieve a better balance between the plasticity and stability of the model.

4.4 Prototype momentum updating (PMU)

In the CIL process, there are often multiple incremental stages. The old prototypes are prone to excessive deviation during the training process due to the lack of constraints from old samples, making them no longer representative. In the experiment, we found that even if the PCL is used to constrain the old prototypes, catastrophic forgetting problems still inevitably occur in the long-term CIL process. Inspired by [7], we applied the momentum update to the L2-similarity classifier to alleviate the excessive deviation of the old prototypes through this slow update process. Specifically, after each batch of training, we use the PMU method to update the old prototypes:

$$W_{C_{0:t-1}}^t = \alpha \cdot W^{t-1} + (1-\alpha) \cdot W_{C_{0:t-1}}^t \tag{5}$$

We set $\alpha = 0.5$ in the experiment.

4.5 Resampling around old prototypes (RAOP)

IRP is a DFCIL method, which cannot preserve the old prototypes and use it to maintain the decision boundary of the old classes like NECIL methods [38, 39]. Inspired by [38], we use the resampled prototypes of the old classifier and the new representations to jointly feedback to the L2-similarity classifier to maintain the decision boundary of old classes. Specifically, when the new incremental stage comes, we feed the resampled prototypes of the old classifier to the new classifier and use cross entropy as the loss function:

$$L_{RAOP} = -\frac{1}{|K^t|} \sum_{k^t \in K^t} y_{(k^t)} \cdot \log p(k^t) \tag{6}$$

Among them, p represents the calculation of classification probability by the L2-similarity classifier. The definition of the resampled prototypes K is as follows:

$$K^t = W^{t-1} + e \cdot r \tag{7}$$

Among them, $e \sim N(0,1)$ is a Gaussian noise with the same dimension as the representation. r is used to control the magnitude of uncertainty in resampling. Specifically, the calculation method for scale r is as follows:

$$r = \frac{1}{|X^0| \cdot ||e||_2} \sum_{x^0 \in X^0} ||w_{(x^0)} - F(x^0)||_2 \tag{8}$$

Specifically, we only calculate r in the initial stage and use the same r in subsequent incremental stages.

4.6 Self-supervised learning (SSL)

To alleviate the overfitting in the model, we introduce the same SSL as [38] for IRP. Specifically, for each class, rotate the samples by 90, 180, and 270 degrees and expand them into three newly expanded classes. Then, the expanded samples and the samples themselves are combined as training data to train the model. For the t-th incremental stage, we do not care about the classes extended in the $0:t-1$ stage but only focus on the old classes in the $0:t-1$ stage and the new and extended classes in the t stage. In short, this SSL method extends the original k-class classification problem to a 4k-class classification problem.

4.7 Final objective

As shown in Fig. 2, our final optimization objective consists of 5 items:

$$L_{TOTAL}^t = L_{CE}^t + L_{RPDM}^t + L_{PCL}^t + \lambda \cdot L_{RAOP}^t + L_{KD}^t \tag{9}$$

λ is the loss weight, which we set $\lambda = 10$ in the experiment. Among them, L_{KD}^t is a knowledge distillation function used to alleviate catastrophic forgetting in the encoder, defined as:

$$L_{KD}^t = ||F^t(x^t) - F^{t-1}(x^t)||_2 \tag{10}$$

Table 1: The average top-1 accuracy of different incremental protocols in the CIFAR100, TinyImageNet and ImageNet-Subset. Where T represents the number of incremental stages, and E represents the number of samples retained for each old class. The * indicates that the data result was replicated in its initial paper or [39, 6]. **Bold** represents the optimal result and underline represents the suboptimal result.The subscript 32 represents that ResNet-32 [8] is used as the backbone of this result.

	Methods	CIFAR100			TinyImageNet			ImageNet-Subset		
		$T=5$	$T=10$	$T=20$	$T=5$	$T=10$	$T=20$	$T=5$	$T=10$	$T=20$
NECIL ($E=0$)	EWC*(PNAS'17)	24.5	21.2	15.9	18.8	15.8	12.4	-	20.4	-
	LwF-MC*(CVPR'17)	45.9	27.4	20.1	29.1	23.1	17.4	-	31.2	-
	MUC*(ECCV'20)	49.4	30.2	21.3	32.6	26.6	22.0	-	35.1	-
	SDC*(CVPR'20)	56.8	57.0	58.9	-	-	-	-	61.1	-
	PASS (CVPR'21)	65.3	63.3	59.4	49.7	46.7	41.7	67.5	64.9	56.4
	IL2A*(NeurIPS'21)	66.0	60.3	57.9	47.3	44.7	40.0	-	-	-
	SSRE (CVPR'22)	66.1	65.5	62.0	50.3	49.1	**48.3**	-	**67.5**	-
DFCIL	ABD^*_{32}(ICCV'21)	62.4	59.0	-	44.6	41.6	-	-	-	-
	$R\text{-}DFCIL^*_{32}$(ECCV'22)	64.8	61.7	-	48.9	47.6	-	-	-	-
	R-DFCIL (ECCV'22)	62.6	59.5	50.4	51.6	49.7	43.4	63.3	60.6	51.7
	IRP (ours)	**68.5**	**68.3**	**63.1**	**51.9**	**51.3**	46.5	**68.3**	67.1	**59.4**

Table 2: Comparison of average top-1 accuracy with EBCIL methods under different incremental protocols on the CIFAR100, TinyImageNet and ImageNet-Subset. The * indicates that the data result was replicated in its initial paper or [39, 20, 33].

	Backbone	Methods	CIFAR100			TinyImageNet			ImageNet-Subset	
			$T=5$	$T=10$	$T=20$	$T=5$	$T=10$	$T=20$	$T=5$	$T=10$
EBCIL ($E=20$)	ResNet-18	iCaRL-CNN*(CVPR'17)	51.1	48.7	44.4	34.6	31.2	27.9	-	50.5
		iCaRL-NCM*(CVPR'17)	58.6	54.2	50.5	45.9	43.3	38.0	-	60.8
		EEIL*(ECCV'18)	60.4	56.1	52.3	47.0	45.0	40.5	-	63.3
		UCIR*(CVPR'19)	63.8	62.4	59.1	49.2	48.5	42.8	-	66.2
		BiC*(CVPR'19)	66.6	60.3	-	-	-	-	-	-
		WA*(CVPR'20)	64.0	57.9	-	-	-	-	-	-
		PODNet*(ECCV'20)	67.3	64.0	-	-	-	-	-	74.3
	ResNet-32	UCIR*(CVPR'19)	63.2	60.1	-	-	-	-	70.8	68.3
		w/ AANets*(CVPR'21)	66.8	65.3	-	-	-	-	72.6	69.2
		Mnemonics*(CVPR'20)	63.3	62.3	-	-	-	-	-	71.4
		w/ AANets*(CVPR'21)	67.6	65.7	-	-	-	-	72.9	71.9
		PODNet*(ECCV'20)	64.8	63.2	-	-	-	-	-	74.3
		w/ AANets*(CVPR'21)	66.3	64.3	-	-	-	-	**77.0**	**75.6**
DFCIL	ResNet-18	IRP (ours)	**68.5**	**68.3**	**63.1**	**51.9**	**51.3**	**46.5**	68.3	67.1

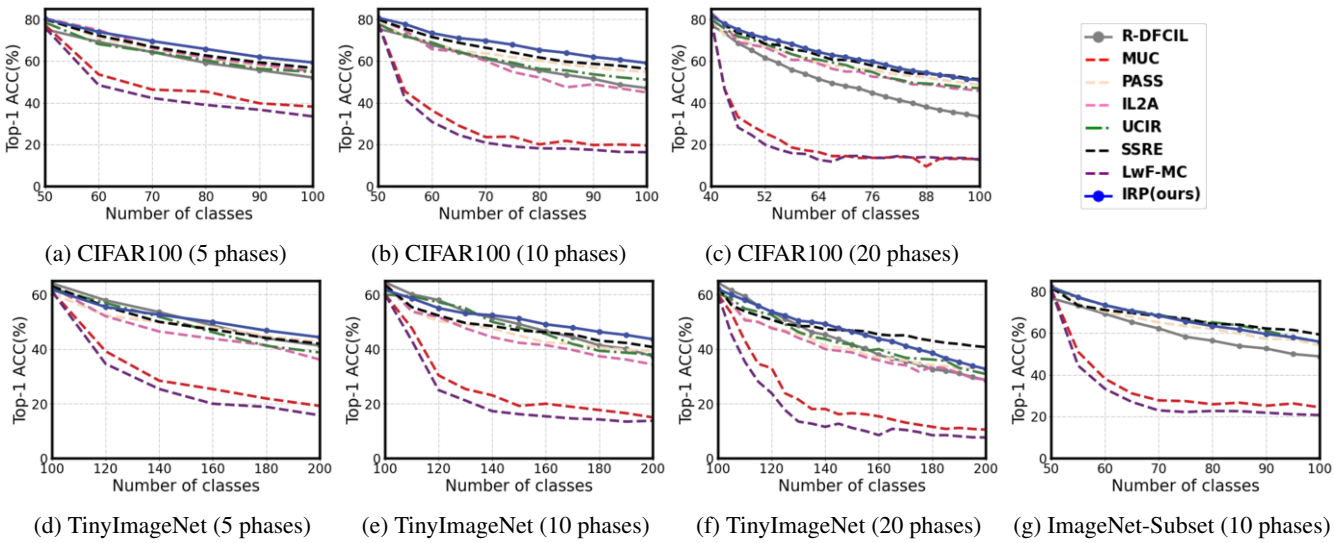
5 Experiments

5.1 Datasets and Setting

Datasets. To evaluate the performance of our method, we conducted comprehensive experiments on three datasets: CIFAR100 [17], TinyImageNet [18] and ImageNet-Subset [3]. CIFAR100 contains 100 classes of images, with a total of 60000 images of 32×32 size. Each class contains 500 training images and 100 test images. TinyImageNet contains images from 200 classes, with pixel sizes of 64×64, and each class contains 500 training images and 50 test images. ImageNet-Subset contains 100 classes. Each class contains about 1300 training images and 50 test images. We follow the settings of [38] and use 1993 as a random seed to select ImageNet-Subset from Imagenet1K [3].

Incremental protocols. We use the classic CIL protocol to train the initial stage with a portion of the data and divide the remaining data equally to train the incremental stages. For CIFAR100 and ImageNet-Subset, three different task partitioning configurations are used: (1) 5 phases: 50 classes are used in the initial stage and 10 classes are used in each incremental stage; (2) 10 phases: 50 classes are used in the initial stage and 5 classes are used in each incremental stage; (3) 20 phases: 40 classes are used in the initial stage and 3 classes are used in each incremental stage. For TinyImageNet, three different task partitioning configurations are used: (1) 5 phases: 100 classes are used in the initial stage and 20 classes are used in each incremental stage; (2) 10 phases: 100 classes are used in the initial stage and 10 classes are used in each incremental stage; (3) 20 phases: 100 classes are used in the initial stage and 5 classes are used in each incremental stage.

Compared methods. We chose some of the most advanced DFCIL and NECIL methods for comparison. To better demonstrate the performance of IRP, we referred to [38, 39] to select some classic EBCIL methods for comparison. The methods for DFCIL include ABD [29] and R-DFCIL [6]. Methods for NECIL include EWC [16], LwF-MC [21], MUC [23], SDC [35], PASS [38], IL2A [37] and SSRE [39]. Methods for EBCIL include iCaRL [27], EEIL [2],



(a) CIFAR100 (5 phases) (b) CIFAR100 (10 phases) (c) CIFAR100 (20 phases)

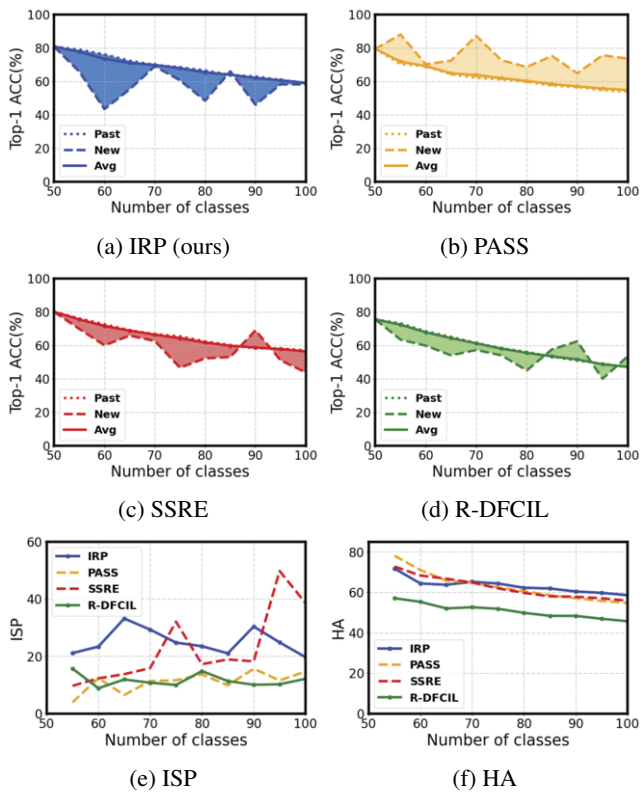
(d) TinyImageNet (5 phases) (e) TinyImageNet (10 phases) (f) TinyImageNet (20 phases) (g) ImageNet-Subset (10 phases)

Figure 3: Trend of top-1 accuracy in the CIFAR100, TinyImageNet and ImageNet-Subset.

UCIR [11], Mnemonics [22], AANets [20], WA [36] and PODNet [4].

Evaluation metrics. We report three CIL metrics for measuring the quality of the methods: average top-1 accuracy, indicator of stability and plasticity (ISP) and harmonic accuracy (HA) [25]. The calculation of average top-1 accuracy includes all stages (including the initial stage), reflecting the overall incremental performance. The indicator of stability and plasticity reflects the balance between stability and plasticity at each incremental stage. The ISP of the t-th incremental stage is defined as $ISP^t = \frac{Avg^t}{2\cdot(Avg^{t-1}-Past^t)}$, in which Avg^t and $Last^t$ are the average top-1 accuracy and accuracy of the old classes in the t-th incremental stage, respectively. There is a positive correlation between ISP and the balance between stability and plasticity. The HA at t-th incremental stage is $HA^t = \frac{2\cdot A_b^t \cdot A_i^t}{A_b^t + A_i^t}$, where A_b^t is the top-1 accuracy of initial classes and A_i^t is the top-1 accuracy of incremental classes. Both ISP and HA represent a balance between stability and plasticity, but the focus is different. With the highest average accuracy possible, ISP focus on maintaining the performance of old classes, while HA expects small differences in performance between incremental and initial classes.

Implementation details. For a fair comparison, we follow the experimental settings of [39]. All experiments without specific instructions default to using ResNet-18 [8] as the backbone. At the same time, the batch size is set to 128. For CIFAR100 and TinyImageNet, use Adam [15] as the optimizer, set the initial learning rate to 0.0015, and set the weight decay to 5e-5. When training CIFAR100, 40 epochs are trained in each stage, and the learning rate is divided by 10 at the 35 epoch. When training TinyImageNet, 20 epochs are trained in each stage, and the learning rate is divided by 10 at the 15 epoch. For ImageNet-Subset, SGD [28] is used as the optimizer in the initial stage, with a learning rate of 0.1 and weight decay of 5e-5. 160 epochs are trained, and the learning rate is divided by 10 at 80, 120, and 150 epochs, respectively. In the incremental stages, use Adam as the optimizer, set the initial learning rate to 0.0015, and set the weight decay to 2e-5. 70 epochs are trained in each stage, and the learning rate is divided by 10 at the 35 epoch. All the experiments are repeated three times, and the average results are reported.



(a) IRP (ours) (b) PASS

(c) SSRE (d) R-DFCIL

(e) ISP (f) HA

Figure 4: The trend of past, new and average top-1 accuracy of three models on CIFAR100 ($T = 10$) in subfigures (a), (b), (c) and (d). In the t-th stage, the past and new accuracy account for $\frac{|C^{0:t-1}|}{|C^{0:t}|}$ and $\frac{|C^t|}{|C^{0:t}|}$ of the average accuracy, respectively. Since $|C^{0:t-1}|$ is much greater than $|C^t|$, maintaining the past accuracy is more important than improving the new accuracy. The best scenario is to prioritize alleviating the decrease in past accuracy while improving new accuracy as much as possible. Subfigure (e) illustrates the quantification of stability and plasticity at each incremental stage, where the y-axis represents ISP. Subfigure (f) illustrates the harmonic accuracy of each incremental stage.

5.2 Comparative results

As shown in Table 1, IRP is significantly superior to other DFCIL methods and has comparable performance with the most advanced NECIL method SSRE [39]. It proves that IRP effectively alleviates the catastrophic forgetting problem without explicitly storing any old samples and prototypes, allowing the model to achieve a good balance between stability and plasticity. Specifically, IRP outperforms the state-of-the-art DFCIL method in the three task partitioning of the CIFAR100. It even improves by a margin of 2.4%, 2.8%, and 1.1% compared to the state-of-the-art NECIL method SSRE [39], respectively. In addition, on the TinyImageNet and ImageNet-Subset, IRP exhibits better performance than DFCIL methods and comparable to NECIL methods. Since EBCIL allows partial sample retention, the performance is often the best among the three types of CIL. We choose some classic and high-performance EBCIL methods as a comparison to better demonstrate the excellent performance of IRP, as shown in Table 2. Most EBCIL methods use ResNet-32 as the backbone for training the CIFAR100. In order to compare more comprehensively with EBCIL methods, we included the results of ResNet-18 and ResNet-32 when referencing these EBCIL results. It can be seen that on the CIFAR100 and TinyImageNet, DFCIL method IRP performs better than the classic EBCIL methods.

Trend of accuracy. In order to more intuitively see the performance of different methods, we demonstrated the trend of accuracy on three datasets. As shown in Fig. 3, IRP has the slowest decline in accuracy and has good class-incremental learning ability on the CIFAR100 and TinyImageNet. Slightly inferior to the state-of-the-art NECIL method SSRE [39] on the ImageNet-Subset.

Stability-Plasticity balance. To better demonstrate the balance between the stability and plasticity of the model, we record the accuracy of new and old tasks at each stage. As shown in Fig. 4(b-d), SSRE [39], PASS [38] and R-DFCIL [6] maintain high accuracy of new classes during the CIL process, while the accuracy of old classes decreases rapidly, resulting in the model having high plasticity and low stability. On the contrary, due to the large proportion of old classes in the total class, IRP tends to maintain the stability of old classes, with relatively low accuracy of new classes and low forgetting of old classes. At the end of incremental learning, IRP achieved higher average top-1 accuracy. Furthermore, we quantitatively compared these four methods using indicator of stability and plasticity (ISP) and harmonic accuracy (HA). As shown in Fig. 4(e-f), the ISP and HA of IRP is significantly superior to the other three methods in most incremental stages, showing a better balance between stability and plasticity.

5.3 Ablation study

The model proposed in this article consists of a total of six parts: L2-similarity classifier, representation-prototype distance minimization (RPDM), prototype changing limitation (PCL), prototype momentum updating (PMU), resampling around old prototypes (RAOP) and self-supervised learning (SSL). To demonstrate the effectiveness of these methods, we conducted ablation experiments on the CIFAR100. The detailed results of the ablation experiment are shown in Table 3. We can see that: (1) If the L2-similarity classifier is not used to replace the FC classifier, the results will ultimately collapse. (2)

Table 3: Ablation study of IRP on CIFAR100.

Classifier	♦ RPDM ■ PCL ▲ PMU ▼ ROAP ✓ SSL	CIFAR100					
		$T=5$		$T=10$		$T=20$	
		Last	Avg	Last	Avg	Last	Avg
FC	◊ ■ ▲ ▼ ✓	13.6	28.0	11.8	18.9	11.3	16.0
L2	◊ ■ ▲ ▼ ✓	28.7	47.2	28.8	41.6	29.0	35.5
L2	♦ □ ▲ ▼ ✓	**59.6**	**68.6**	58.4	67.4	50.0	63.0
L2	♦ ■ △ ▼ ✓	59.3	68.3	57.6	67.2	41.8	58.2
L2	♦ ■ ▲ ▽ ✓	57.7	67.8	56.7	67.1	48.6	62.0
L2	♦ ■ ▲ ▼	54.1	64.1	51.4	62.5	36.7	53.9
L2	♦ ■ ▲ ▼ ✓	59.4	68.5	**59.1**	**68.3**	**50.9**	**63.1**

Table 4: Top-1 accuracy of updating old prototypes with different methods on CIFAR100. "Normal" represents normal backpropagation updates.

Update Methods	CIFAR100					
	$T=5$		$T=10$		$T=20$	
	Last	Avg	Last	Avg	Last	Avg
Normal	59.0	68.0	57.9	67.6	39.0	57.5
Frozen	59.1	68.0	57.7	67.6	49.7	63.1
PMU($\alpha=0.1$)	59.2	**68.6**	58.0	67.4	49.4	62.8
PMU($\alpha=0.5$)	**59.4**	68.5	**59.1**	**68.3**	**50.9**	63.1
PMU($\alpha=0.9$)	58.8	68.0	58.2	67.6	50.1	**63.6**

The RPDM brings a 25.2% performance improvement to the model. It can be seen from Fig. 1(b-c) that using RPDM can solve the problem of loose representation, resulting in improved performance. (3) Both PCL and PMU contribute to alleviating excessive deviation of prototypes. After using PCL and PMU separately, the accuracy of long-term class-incremental learning such as 10 and 20 phases improved by a margin of 0.5% and 3%, respectively. (4) After using RAOP to maintain the decision boundary of old classes, the overall performance is improved by a margin of 1%. (5) SSL brings a 6.5% performance improvement to the model. To better illustrate the impact of updating the old prototypes on model performance, we attempted multiple different updating methods and compared them, as shown in Table 4. It can be seen that using PMU to update the old prototypes significantly improves the performance of the model.

6 Conclusion

This paper proposes a new DFCIL method, called IRP, to solve the catastrophic forgetting problem from three aspects: classifier, alleviating the deviation of old prototypes, and maintaining the decision boundary of old classes. Firstly, the L2-similarity classifier is used to replace the FC classifier commonly used in CIL, implicitly providing the old prototypes for the model, and the representation-prototype distance minimization is used to solve the problem of loose representation. Then, update the old prototype using prototype changing limitation and prototype momentum updating to alleviate its excessive deviation. In particular, we feed the resampled old prototypes back to L2-similarity classifier to maintain the decision boundary of the old classes. The experimental results show that IRP is significantly superior to other DFCIL methods and achieves comparable performance compared to the most advanced NECIL and some classic EBCIL methods.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3931–3940, (2020).

[2] Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari, 'End-to-end incremental learning', in *Proceedings of the European conference on computer vision (ECCV)*, pp. 233–248, (2018).

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

[4] Arthur Douillard, Matthieu Cord, Charles Ollion, Thomas Robert, and Eduardo Valle, 'Podnet: Pooled outputs distillation for small-tasks incremental learning', in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pp. 86–102. Springer, (2020).

[5] Enrico Fini, Victor G Turrisi Da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal, 'Self-supervised models are continual learners', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9621–9630, (2022).

[6] Qiankun Gao, Chen Zhao, Bernard Ghanem, and Jian Zhang, 'R-dfcil: Relation-guided representation learning for data-free class incremental learning', in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIII*, pp. 423–439. Springer, (2022).

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, 'Momentum contrast for unsupervised visual representation learning', in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, (2020).

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

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, 'Distilling the knowledge in a neural network', *arXiv preprint arXiv:1503.02531*, (2015).

[10] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin, 'Learning a unified classifier incrementally via rebalancing', in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 831–839, (2019).

[11] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin, 'Learning a unified classifier incrementally via rebalancing', in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 831–839, (2019).

[12] Jayateja Kalla and Soma Biswas, 'S3c: Self-supervised stochastic classifiers for few-shot class-incremental learning', in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pp. 432–448. Springer, (2022).

[13] Ronald Kemker and Christopher Kanan, 'Fearnet: Brain-inspired model for incremental learning', *arXiv preprint arXiv:1711.10563*, (2017).

[14] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan, 'Measuring catastrophic forgetting in neural networks', in *Proceedings of the AAAI conference on artificial intelligence*, volume 32, (2018).

[15] Diederik P Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*, (2014).

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

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

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

[19] Yann LeCun, 'The mnist database of handwritten digits', *http://yann. lecun. com/exdb/mnist/*, (1998).

[20] Yaoyao Liu, Bernt Schiele, and Qianru Sun, 'Adaptive aggregation networks for class-incremental learning', in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 2544–2553, (2021).

[21] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun, 'Mnemonics training: Multi-class incremental learning without forgetting', in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 12245–12254, (2020).

[22] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun, 'Mnemonics training: Multi-class incremental learning without forgetting', in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pp. 12245–12254, (2020).

[23] 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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pp. 699–716. Springer, (2020).

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

[25] Can Peng, Kun Zhao, Tianren Wang, Meng Li, and Brian C Lovell, 'Few-shot class-incremental learning from an open-set perspective', in *European Conference on Computer Vision*, pp. 382–397. Springer, (2022).

[26] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao, 'Random path selection for continual learning', *Advances in Neural Information Processing Systems*, **32**, (2019).

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

[28] Herbert Robbins and Sutton Monro, 'A stochastic approximation method', *The annals of mathematical statistics*, 400–407, (1951).

[29] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9374–9384, (2021).

[30] Guile Wu, Shaogang Gong, and Pan Li, 'Striking a balance between stability and plasticity for class-incremental learning', in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1124–1133, (2021).

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

[32] Ye Xiang, Ying Fu, Pan Ji, and Hua Huang, 'Incremental learning using conditional adversarial networks', in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6619–6628, (2019).

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

[34] Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz, 'Dreaming to distill: Data-free knowledge transfer via deepinversion', in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8715–8724, (2020).

[35] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 6982–6991, (2020).

[36] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia, 'Maintaining discrimination and fairness in class incremental learning', in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13208–13217, (2020).

[37] Fei Zhu, Zhen Cheng, Xu-Yao Zhang, and Cheng-lin Liu, 'Class-incremental learning via dual augmentation', *Advances in Neural Information Processing Systems*, **34**, 14306–14318, (2021).

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

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