Self-Organizing Pathway Expansion for Non-Exemplar Class-Incremental Learning

Kai Zhu[1,2,*] Kecheng Zheng[3,4,*] Ruili Feng[1,2]
Deli Zhao[1] Yang Cao[2,5,†] Zheng-Jun Zha[2]
[1] Alibaba Group [2] University of Science and Technology of China [3] Zhejiang University
[4] Ant Group [5] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center
{kaizhuustc, zkechengzk, ruilifengustc, zhaodeli}@gmail.com {forrest, zhazj}@ustc.edu.cn

Abstract

Non-exemplar class-incremental learning aims to recognize both the old and new classes without access to old class samples. The conflict between old and new class optimization is exacerbated since the shared neural pathways can only be differentiated by the incremental samples. To address this problem, we propose a novel self-organizing pathway expansion scheme. Our scheme consists of a class-specific pathway organization strategy that reduces the coupling of optimization pathway among different classes to enhance the independence of the feature representation, and a pathway-guided feature optimization mechanism to mitigate the update interference between the old and new classes. Extensive experiments on four datasets demonstrate significant performance gains, outperforming the state-of-the-art methods by a margin of 1%, 3%, 2% *and* 2%*, respectively.*

1. Introduction

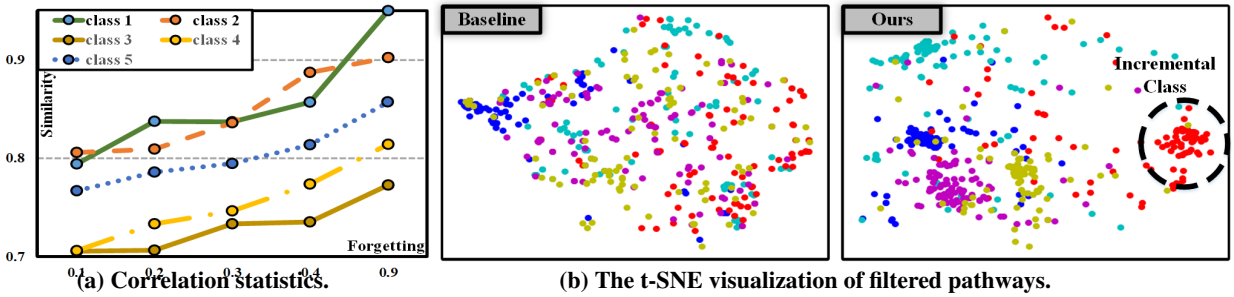
Since deep neural networks have achieved good performance in fully supervised scenarios, how to extend this learning capability to open environment has attracted great attention. Particularly, it is essential to ensure that the network can continuously learn new knowledge while maintaining the abilities to identify old tasks (*i.e.*, incremental learning [8, 18]). Fine-tuning the network directly with new data can lead to a serious bias of the representation and classifier, which is often referred to as catastrophic forgetting. Due to privacy and hardware limits, old samples are usually unavailable for joint training, making it more difficult to maintain the old class performance in the subsequent optimization process. In this paper, we focus on this ability to continuously learn new tasks without any old samples or exemplars, which is called non-exemplar class-incremental learning (NECIL) [25, 27, 30, 31, 33].

Most methods maintain the feature representation of old classes by means of various distillation loss functions [8, 10]. Although catastrophic forgetting is somewhat mitigated, incremental performance still suffers from the confusion between the old and new class in the feature space. Furthermore, in the absence of old class samples, the degree of forgetting is only related to the initial model and incremental samples [31]. Existing NECIL works [25, 30] mainly focus on enhancing the overall performance by improving the discrimination and generalization of the initial model, which brings a significant improvement on the incremental performance.

Instead, we focus on the impact of incremental samples on the optimization process. Intuitively, since different incremental classes cause disparate feature confusion, the interference on the old class performance is also different even if initialized from the same model [31, 33]. To further explore the association, we estimate the inter-class confusion by measuring the status of feature activation [29] in existing incremental model. As shown in Fig. 1 (b), we filter out the positions of strongly activated modules as the class-specific pathways [19], and find that the pathway of incremental class is commonly confused with the previous ones in the baseline. Furthermore, it can be seen in Fig. 1 (a) that the degree of pathway overlap (*i.e.,* similarity) between the old class and incremental class is positively correlated with the forgetting degree, which motivates us to address the interference problem from the perspective of pathway optimization.

Based on the above observation, we propose a self-organizing pathway expansion scheme to learn a pathway-aware representation, mitigating the feature interference during the subsequent incremental process. The scheme is mainly manifested in two aspect. Firstly, during the initial phase, we adopt the class-specific pathway organization

∗Co-first Author. †Corresponding Author.



(a) Correlation statistics. (b) The t-SNE visualization of filtered pathways.

Figure 1. Motivation of our method. (a) The accuracy degradation of old classes (*i.e.* forgetting rates in the horizontal coordinate) is positively correlated to the corresponding pathway similarity with incremental classes. The concept of pathway [19] is formed from the aggregation of important modules, which are filtered out by the contribution to the final recognition performance. (b) In standard classification method (*i.e.*, baseline in Sec. 3.2), the direct adoption of an activation-like rule [33] makes it hard to measure inter-class overlap as the distribution implicitly optimized for classes is haphazard. In contrast, the discriminative pathway in our method brings out lower inter-class overlap, which benefits the mitigation of feature confusion. All above experiments are conduct on ImageNet-Full dataset.

strategy to enhance the independence of feature representation by forcing the optimization pathways specific to different classes. A global pathway planner is utilized to explicitly select the most relevant modules, facilitating the pathway identification. It is noted that we do not modify the network structure, but only divide the output channels of each convolution module to match the output of the pathway planner. Secondly, during the incremental phases, we introduce a pathway-guided feature update mechanism to promote the effectiveness of new classes involved in incremental optimization by adjusting the classification weight with the pathway similarity. Since the pathway value is either 0 or 1, we calculate the intersection of union (*i.e.*, IoU) value to better measure the class relevance, reducing the interference of vector normalization. Furthermore, an incremental pathway update mechanism is proposed to ensure the long-term effect by alternating the optimization of the pathway planner and feature representation. To summarize, our main contributions are as follows:

1) A self-organizing pathway expansion scheme is proposed for non-exemplar incremental learning, in which a progressive decoupling optimization is accomplished by a class-specific pathway organization strategy, resulting in a pathway-aware representation.

2) A pathway-guided feature update mechanism is proposed, which utilizes the similarity of pathways to guide the optimization of incremental samples.

3) Extensive experiments are performed on benchmark including CIFAR-100, TinyImageNet, ImageNet-Subset and ImageNet-Full datasets, and the results demonstrate the superiority of our method over the state-of-the-art.

2. Related Work

2.1. Incremental Learning

As deep learning research advances, there is a growing demand for continual learning [1, 5, 11, 28, 32], which requires the network to learn new tasks without forgetting the old knowledge to achieve the stability-plasticity trade-off. Class-incremental learning (CIL [8, 9, 18, 23, 24]), a difficult type in continual learning, has attracted much attention due to the agnosticism to task identity [21, 22].

Recently, some works [25,27,30,31] focus on a challenging but practical NECIL problem, where no past data can be stored due to equipment limits or privacy security. [27] estimates the semantic drift of the initial model inherited from the base phase, and compensates the prototypes in each test phase. [25] inverts the old samples from the initial model for the joint distillation. [30, 31] consider to enhance the generalization of the representation to learn more transferable features for future tasks. We follow their NECIL settings. However, different from their work focusing on the utilization and enhancement of the initial model, we mainly consider the rectification of the incremental samples on joint classification and distillation process.

2.2. Neural Pathways

To enhance the adaptation of the network to new tasks, several continual learning methods [4, 17] have been proposed to decouple the learning process from the perspective of pathway. However, the targeted models are continuously expanded with the update of pathway, which is difficult to adapt to the standard classification network. The expansion direction of pathway tends to be selected randomly, making it hard to search for an explanation. In this paper, we target on the pathway learning on the standard network without changing the structure, and guiding the incremental optimization based on the pathway relationship.

3. Method

3.1. Problem Description

The NECIL problem is defined as follows. Here we denote D_t as the training set at the current phase t, which

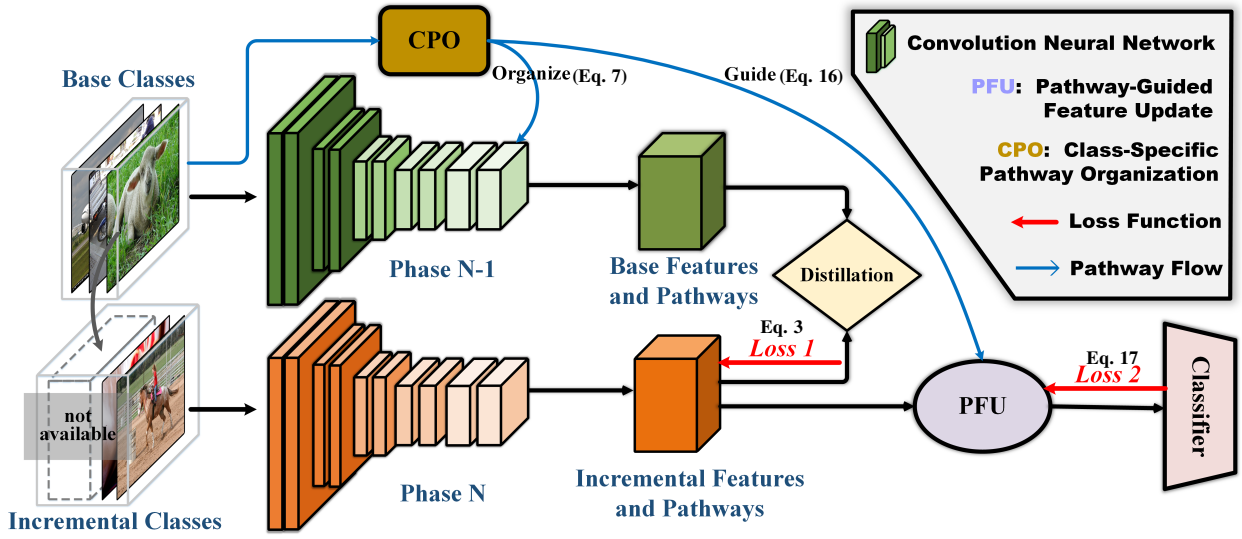


Figure 2. Overall pipeline of our proposed self-organized pathway expansion scheme for NECIL. Our scheme consists of a class-specific pathway organization strategy that reduces the coupling of optimization pathway among different classes, and a pathway-guided feature update mechanism to mitigate the update interference between the old and new classes.

consists of the sample set X_t and label set Y_t. Our task is to train the model from a continuous data stream, *i.e.*, training sets $D_0, D_1, \cdots D_T$, where labels of a set X_i ($0 \leq i \leq T$) are from the set Y_i, and T represents the number of incremental phases. It should be mentioned that all the incremental classes are disjoint, that is, $Y_i \cap Y_j = \varnothing (i \neq j)$. At the current phase t, there are no old training sets (*i.e.*, $D_{0:t-1}$) in memory, but incremental samples (*i.e.*, D_t) for the current phase. To measure the performance of models at current phase t, we calculate the classification accuracy on the test set Z_t, in which the classes are from all the seen label sets $Y_0 \cup Y_1 \cdots \cup Y_t$.

3.2. Baseline for NECIL

Following the paradigm of existing NECIL work [25, 30, 31, 33], we adapt distillation-based CIL methods [18] to the NECIL setting as the baseline. Specifically, at the incremental phase (*i.e.*, $t > 0$), a standard classification model that consists of the feature extractor $f_{\boldsymbol{\theta}_t}$ and classifier $g_{\boldsymbol{\phi}_t}$ should be optimized under full supervision (*i.e.*, $D_{0:t}$),

$$\min_{\boldsymbol{\theta}_t, \boldsymbol{\phi}_t} \mathcal{L}_t = \mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t; D_{0:t-1}) + \mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t; D_t), \quad (1)$$

$$\mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t; D_t) = \sum_{x \in X_t} \sum_{y \in Y_t} y \cdot log(g_{\boldsymbol{\phi}_t}(f_{\boldsymbol{\theta}_t}(x))), \quad (2)$$

where $\mathcal{L}_t$ represents the overall loss function for feature optimization. However in the NECIL setting, since the previous training sets are unavailable, the corresponding loss $\mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t; D_{0:t-1})$ for both the feature extractor and classifier is missing, leading to a serious bias to current classes. To solve the problem, existing methods [8, 9] replace the old classification supervision with the feature distillation and classifier correction. Specifically, the parameters $\boldsymbol{\theta}_{t-1}$ of the old feature extractor from previous phase $t-1$ is frozen and saved during each incremental phase t. To maintain the old informative feature, the knowledge distillation $\mathcal{L}_{kd}$ is used to ensure the similarity between the current representation $f_{\boldsymbol{\theta}_t}(x)$ and the previous one $f_{\boldsymbol{\theta}_{t-1}}(x)$:

$$\min_{\boldsymbol{\theta}_t} \mathcal{L}_{kd}(\boldsymbol{\theta}_t; \boldsymbol{\theta}_{t-1}, D_t) = \sum_{x \in X_t} \left\| f_{\boldsymbol{\theta}_t}(x) - f_{\boldsymbol{\theta}_{t-1}}(x) \right\|_2, \quad (3)$$

where $\|\cdot\|_2$ denotes Euclidean Norm. As there are no exemplars for balanced classifier optimization in NECIL, we turn to consider the class-representative prototypes $\boldsymbol{P}_{0:t-1}$ [31] in the deep feature space. Specifically, we compute and memorize one prototype $\boldsymbol{p}^c \in \boldsymbol{P}_{0:t-1}$ for each class c as:

$$\boldsymbol{p}^c = \mathbb{E}_{(x,y) \sim D_{0:t-1}} \left[f_{\boldsymbol{\theta}_t}(x) \mid y = c \right]. \quad (4)$$

In each training iteration, we choose to oversample [3] memorized prototypes $\boldsymbol{P}_{0:t-1}$ as training prototypes $\tilde{\boldsymbol{P}}_{0:t-1}$ by the ratio of batch size. Training prototypes are directly involved in the standard classification optimization, achieving the augmentation of the classifier, which is consistent with the baseline in PASS [31] and IL2A [30]:

$$\min_{\boldsymbol{\phi}_t} \mathcal{L}_{aug}(\boldsymbol{\phi}_t; \tilde{\boldsymbol{P}}_{0:t-1}) = \sum_{\boldsymbol{p}^c \in \tilde{\boldsymbol{P}}_{0:t-1}} \sum_{y \in Y_{0:t-1}} y \cdot log(g_{\boldsymbol{\phi}_t}(\boldsymbol{p}^c)). \quad (5)$$

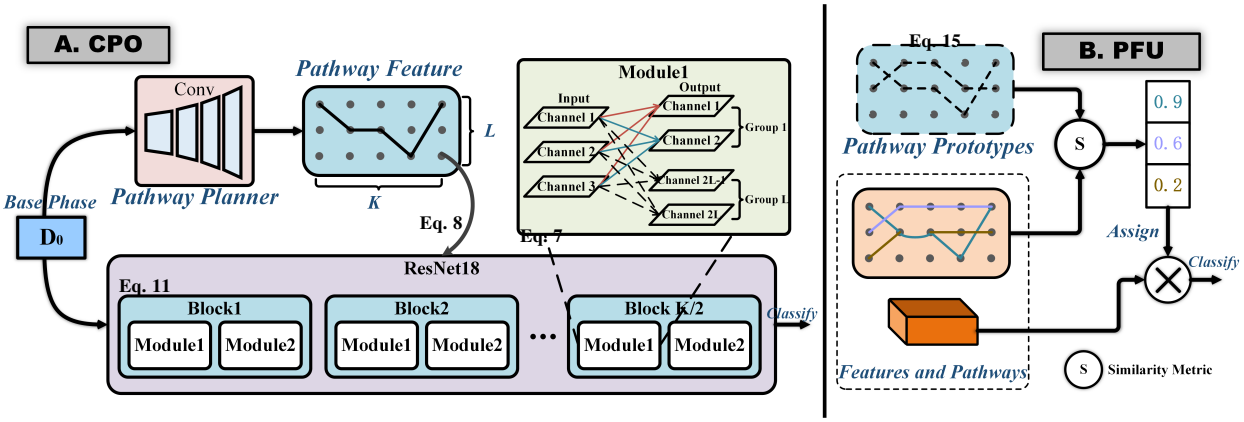


Figure 3. (A) During the base phase, a CPO strategy is proposed to mitigate the incremental interference, in which the pathway feature extracted by a pathway planner is utilized to organize the class-specific learning in feature extractor (*e.g.*, ResNet18). (B) During the incremental phase, the similarity scores between the pathway feature and saved pathway prototypes are assigned to the optimization process as loss weights, facilitating the pathway-guided feature update (PFU).

In conclusion, the overall feature optimization problem for the baseline method can be written as follows,

$$\min_{\boldsymbol{\theta}_t,\boldsymbol{\phi}_t} \mathcal{L}_t = \mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t; D_t) + \mathcal{L}_{kd}(\boldsymbol{\theta}_t; \boldsymbol{\theta}_{t-1}, D_t) + \mathcal{L}_{aug}(\boldsymbol{\phi}_t; \tilde{\boldsymbol{P}}_{0:t-1}). \tag{6}$$

3.3. Self-Organizing Pathway Expansion

Our proposed self-organizing pathway expansion scheme consists of a class-specific pathway organization strategy that reduces the pathway overlap during the base phase to mitigate the overall feature confusion, and a pathway-guided feature optimization mechanism to refine the incremental optimization guided by the inter-class pathway correlation. The main procedures are summarized in Algorithms 1 and 2 respectively, and the specific implementation is described below.

Class-Specific Pathway Organization. To mitigate the interference during the feature optimization process, we perform a structural decomposition on the feature extractor and organize the class-specific pathway adaptively. As shown in Fig. 2, each standard convolution module consists of a 3×3 convolution layer and a BatchNorm layer. We firstly reorganize K convolution modules, each of which is equally divided into L groups along the output channels. We define $\boldsymbol{\theta}_t^k \in \mathbb{R}^{C_{in}\times C_{out}}$ as the parameters of k_{th} convolution module $\mathrm{Conv}_{\boldsymbol{\theta}_t^k}$ of the feature extractor $f_{\boldsymbol{\theta}_t}$, in which $\boldsymbol{\theta}_t^{k,l} \in \mathbb{R}^{C_{in}\times C_{out}/L}$ denotes the parameters of l_{th} group $\mathrm{Conv}_{\boldsymbol{\theta}_t^{k,l}}$. C_{in} and C_{out} represent the number of input and output channels. Let $\boldsymbol{z}_t^{k-1}$ be the input feature of $\mathrm{Conv}_{\boldsymbol{\theta}_t^k}$, the convolution operation is organized as follows,

$$\boldsymbol{z}_t^k = \mathrm{Conv}_{\boldsymbol{\theta}_t^k}(\boldsymbol{z}_t^{k-1}) = \mathrm{Concat}[\mathrm{Conv}_{\boldsymbol{\theta}_t^{k,1}}(\boldsymbol{z}_t^{k-1}), \dots \mathrm{Conv}_{\boldsymbol{\theta}_t^{k,L}}(\boldsymbol{z}_t^{k-1})], (1 < k \leq K), \tag{7}$$

where Concat denotes the concatenation along the output channels. The output feature $\boldsymbol{z}_t^k$ is the same as the that of standard convolution module before reorganization.

Then, we introduce a pathway planner $f_{\boldsymbol{\alpha}_t}$, which consists of several standard convolution blocks. It receives the image x as input, and output a probability score $\boldsymbol{S} \in \mathbb{R}^{K\times L} = f_{\boldsymbol{\alpha}_t}$, representing the pathway importance of K modules and L groups in the feature extractor. According to the obtained score, a gradually decreasing sparse rate is adopted to filter the most adequate components of the global pathway to guide the feature optimization. Specifically, given a target sparse rate ζ, we solve the minimum pathway threshold ε from the equation as follows (See A.7 in supplementary materials for examples),

$$1-\zeta = \frac{\left|\{s^{k,l} \mid s^{k,l} > \varepsilon, s^{k,l} \in \boldsymbol{S}\}\right|}{|\{s^{k,l}, s^{k,l} \in \boldsymbol{S}\}|}, \tag{8}$$

where $|\cdot|$ means the element number. The pathway score can be filtered by the calculated threshold:

$$\hat{\boldsymbol{S}} = \mathrm{Filter}(\boldsymbol{S}, \zeta) = \boldsymbol{S} * \mathrm{Bool}(\boldsymbol{S} - \varepsilon > 0), \tag{9}$$

where $*$ represents the element-wise multiplication, and Bool denotes the element-wise boolean operation. As the threshold ε is not a given hard value [6] but a filtered soft one in Eq. 8, no special gradient correction is required. To stabilize the optimization process with the threshold, we use a three-step strategy to jointly optimize features and pathways in which different values of sparse rate are adopted at different epoch e:

$$\zeta = \begin{cases} 0, & e < e_1 \\ \frac{e-e_1}{e_2-e_1}\zeta_{max}, & e_1 \leq e < e_2 \\ \zeta_{max}, & e \geq e_2, \end{cases} \tag{10}$$

where e_1 and e_2 are two hyper-parameters. ζ_{max} is another hyper-parameter that defines the maximum value of sparse rate. According to the filtered scores $\hat{\boldsymbol{S}}$, we reorganize the pathway of the network, and Eq. 7 can be rewritten as follows,

$$\begin{aligned} \boldsymbol{z}_t^k = \mathrm{Conv}_{\boldsymbol{\theta}_t^k}(\boldsymbol{z}_t^{k-1}, \hat{\boldsymbol{S}}) = \mathrm{Concat}[\hat{s}^{k,1} * \\ \mathrm{Conv}_{\boldsymbol{\theta}_t^{k,0}}(\boldsymbol{z}_t^{k-1}), \dots \hat{s}^{k,L} * \mathrm{Conv}_{\boldsymbol{\theta}_t^{k,L}}(\boldsymbol{z}_t^{k-1})], \end{aligned} \tag{11}$$

$$\begin{aligned} \boldsymbol{z}_t^K = f_{\boldsymbol{\theta}_t}(\boldsymbol{z}_t^0, \hat{\boldsymbol{S}}) = f_{\boldsymbol{\theta}_t}(x; \hat{\boldsymbol{S}}) \\ = f_{\boldsymbol{\theta}_t}(x; f_{\boldsymbol{\alpha}_t}(x)) = f_{\boldsymbol{\theta}_t, \boldsymbol{\alpha}_t}(x), \quad x \in X_t, \end{aligned} \tag{12}$$

where $\hat{s}^{k,l}$ denotes the element in $\hat{\boldsymbol{S}}$ at the (k, l) position. Eq. 2 can be be rewritten as follows,

$$\mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t, \boldsymbol{\alpha}_t; D_t) = \sum_{x \in X_t} \sum_{y \in Y_t} y \cdot log(g_{\boldsymbol{\phi}_t}(f_{\boldsymbol{\theta}_t, \boldsymbol{\alpha}_t}(x))). \tag{13}$$

Finally, we binarize the filtered pathway and improve inter-class discriminability with a learnable pathway classifier $g_{\boldsymbol{\beta}_t}$:

$$\min_{\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t} \mathcal{L}_{cls}^{path}(\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t; D_t) = \sum_{x \in X_t} \sum_{y \in Y_t} y \cdot log(g_{\boldsymbol{\beta}_t}(\delta(f_{\boldsymbol{\alpha}_t}(x))), \tag{14}$$

where δ, $\boldsymbol{\alpha}_t$ and $\mathcal{L}_{cls}^{path}$ denotes the gate function [20], the learnable parameters in the pathway planner $f_{\boldsymbol{\alpha}_t}$ and the overall pathway classification loss, respectively.

Pathway-Guided Feature Update. To promote the efficiency of incremental learning, we adopt a pathway-guided feature update mechanism in the incremental phase. Specifically, as shown in Fig. 3, we involve new samples into the classification process according to the pathway overlap with old ones. We preserve the class-specific pathway prototype $\boldsymbol{a}^c \in \boldsymbol{A}_{0:t-1}$ for class c at the phase end,

$$\boldsymbol{a}^c = \mathrm{Filter}(\mathbb{E}_{(x,y) \sim D_{0:t-1}} [f_{\boldsymbol{\alpha}_t}(x) \mid y = c], \zeta), \tag{15}$$

where Filter is the same as that in Eq. 9. The binarized pathway score $\delta(f_{\boldsymbol{\alpha}_t}(x))$ is compared to the saved pathway prototype with intersection over union (IoU [14]), thus measuring the relevance λ of the corresponding samples to the previous parameter space:

$$\lambda(x) = \frac{1}{C} \sum_{c=1}^{C} (\mathrm{IoU}(\delta(f_{\boldsymbol{\alpha}_t}(x)), \boldsymbol{a}^c)), \tag{16}$$

where C represents the number of pathway prototypes. To ensure the stability of the incremental representation optimization, we freeze the parameters of the pathway planner (*i.e.,* $\boldsymbol{\alpha}_t$). More relevant samples are assigned smaller weights to reduce the optimization confusion of novel classes, thus the classification loss in Eq. 6 (*i.e.,* $\mathcal{L}_t$) can be rewritten as follows,

$$\begin{aligned} \min_{\boldsymbol{\theta}_t, \boldsymbol{\phi}_t} \mathcal{L}_{cls}(\boldsymbol{\theta}_t, \boldsymbol{\phi}_t; \boldsymbol{\alpha}_t, D_t, \boldsymbol{A}_{0:t-1}) = \\ \sum_{x \in X_t} (1 - \lambda(x)) \sum_{y \in Y_t} y \cdot log(g_{\boldsymbol{\phi}_t}(f_{\boldsymbol{\theta}_t, \boldsymbol{\alpha}_t}(x))). \end{aligned} \tag{17}$$

Incremental Pathway Update. To enhance the effectiveness of the pathway planner, we then freeze the parameters of feature extractor $\boldsymbol{\theta}_t$, and adopt the incremental pathway update mechanism, which is similar to the optimization process of incremental feature in Eq. 6,

$$\begin{aligned} \min_{\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t} L_t^{path} = \mathcal{L}_{cls}^{path}(\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t; D_t) \\ + \mathcal{L}_{kd}^{path}(\boldsymbol{\alpha}_t; \boldsymbol{\alpha}_{t-1}, D_t) + \mathcal{L}_{aug}^{path}(\boldsymbol{\beta}_t, \boldsymbol{A}_{0:t-1}). \end{aligned} \tag{18}$$

The old pathway planner with frozen parameters $\boldsymbol{\alpha}_{t-1}$ is utilized to distill with the current planner, and the pathway

Algorithm 1 Class-Specific Pathway Organization

Input: Feature extractor $f_{\boldsymbol{\theta}_t}$, pathway planner $f_{\boldsymbol{\alpha}_t}$, base set D_0 and maximum sparse rate ζ_{max},
Initialize: Reorganize the structure $f_{\boldsymbol{\theta}_t}$ by Eq. 7;
for all $(x, y) \in D_0$ **do**
  Extract the pathway score $\boldsymbol{S} = f_{\boldsymbol{\alpha}_t}(x)$;
  Compute the epoch-specific sparse rate ζ ($\leq \zeta_{max}$)by Eq. 10;
  Confirm the position (l, k) of filtered pathway with the soft threshold ε by Eq. 8;
    $\hat{\boldsymbol{S}} \leftarrow \left\{ s^{l,k} \mid s^{l,k} > \varepsilon, s^{l,k} \in \boldsymbol{S} \right\}$
  Guide the feature optimization with filtered pathway by Eq. 11;
  Update $\boldsymbol{\theta}_t$ and $\boldsymbol{\alpha}_t$ by taking a SGD step on the image and pathway loss (Eq. 13 14);
end for
Output: Calculated feature prototypes $\boldsymbol{P}_0$ and pathway prototypes $\boldsymbol{A}_0$ by Eq. 4 15.

Algorithm 2 Pathway-Guided Feature Update

Input: Old $f_{\boldsymbol{\theta}_{t-1}}$ and new feature extractor $f_{\boldsymbol{\theta}_t}$, old $f_{\boldsymbol{\alpha}_{t-1}}$ and new pathway planner $f_{\boldsymbol{\alpha}_t}$, incremental set $D_t (t > 0)$, feature prototypes $\boldsymbol{P}_{0:t-1}$ and pathway prototypes $\boldsymbol{A}_{0:t-1}$.
Initialize: Freeze the parameters of $f_{\boldsymbol{\alpha}_t}$;
for all $(x, y) \in D_t$ **do**
  Filter the pathway with ζ_{max} by Eq. 9;
  Compute feature classification and distillation loss weighted with pathway similarity by Eq. 17;
  Compute the augmentation loss by Eq. 5;
  Update $\boldsymbol{\theta}_t$ and $\boldsymbol{\alpha}_t$ based on above losses;
end for
Freeze the parameters of $f_{\boldsymbol{\theta}_t}$, and unfreeze $f_{\boldsymbol{\alpha}_t}$;
for all $(x, y) \in D_t$ **do**
  Update incremental pathway planner with pathway update loss $\mathcal{L}_t^{path}$ by Eq. 18.
end for

+CPO	+PFU	+IPU	5	10	20
			48.51	46.66	40.29
√			51.55	49.87	48.60
√	√		52.61	51.97	51.17
√	√	√	53.69	52.88	51.94

Table 1. Ablation study of our method on TinyImageNet. CPO, PFU and IPU represent the proposed components in Sec. 3. 5, 10 and 20 represents the number of incremental phases (*i.e.,* P).

Method	5	10	20
Rps [17]	63.74	62.71	59.06
Hat [20]	59.44	57.69	55.68
Iap [4]	56.00	55.11	52.79
Piggy [16]	55.79	54.36	38.78
Ours	66.64	65.84	61.83

Table 2. The impact of the pathway structure on CIFAR-100. Rps, Hat, Iap and Piggy are detailed in Sec. 4.3.

prototypes are oversampled to correct the pathway classifier bias to the old class. Overall, the loss functions L_t and L_t^{path} are utilized sequentially in the incremental phase t.

4. Experiments

4.1. Datasets and Settings

Datasets. Following the setting in [31], we conduct comprehensive experiments on four datasets CIFAR-100 [12], TinyImageNet [13], ImageNet-Subset and ImageNet-Full. CIFAR-100 contains 60,000 images of 32 × 32 size from 100 classes, and each class includes 500 training images and 100 test images. TinyImageNet contains 200 classes, and each class contains 500 training images, 50 validation images and 50 test images. It provides more incremental phases and classes for the sensitivity analysis on different methods. ImageNet-Subset is a 100-class subset of ImageNet-Full [7], which provides a large-scale evaluation scenery. Except for 40 base classes in 20 incremental phases setting of CIFAR-100, we train the model on half of classes for the base phase, and equal classes in the rest incremental phases. We conduct different incremental settings (5, 10 and 20 phases) for both CIFAR-100 and TinyImageNet, and 10 incremental phases setting for the rest datasets, which is consistent with [31].

Settings and Metric. For a fair comparison with [31], we adopt the same backbone network (*i.e.*, ResNet-18), and maintain the same accuracy at the first phase for all datasets. We report average incremental accuracy and average forgetting [31]. Average incremental accuracy A_T is computed as the average accuracy of all incremental phases a_t (including the first phase), which compares the

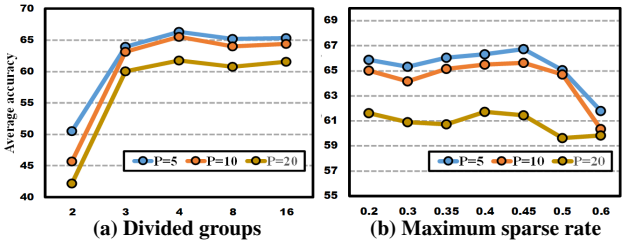


Figure 4. The impact of values of divided groups (*i.e.*, L) and maximum sparse rate (*i.e.*, ζ_{max}) on the incremental performance.

overall performance of different methods fairly,

$$A_T = \frac{1}{T}\sum_{t=0}^{T} a_t. \tag{19}$$

Average forgetting is computed as the average forgetting throughout the incremental process, which directly measures the ability of different methods to resist catastrophic forgetting. The forgetting at phase t ($t > 0$) is calculated as $F_t = \frac{1}{t}\sum_{j=1}^{t-1} f_j^t$, where f_j^t denotes the performance drop of classes:

$$f_j^t = \max_{i\in\{j,\cdots t-1\}} a_{i,j} - a_{t,j}, \tag{20}$$

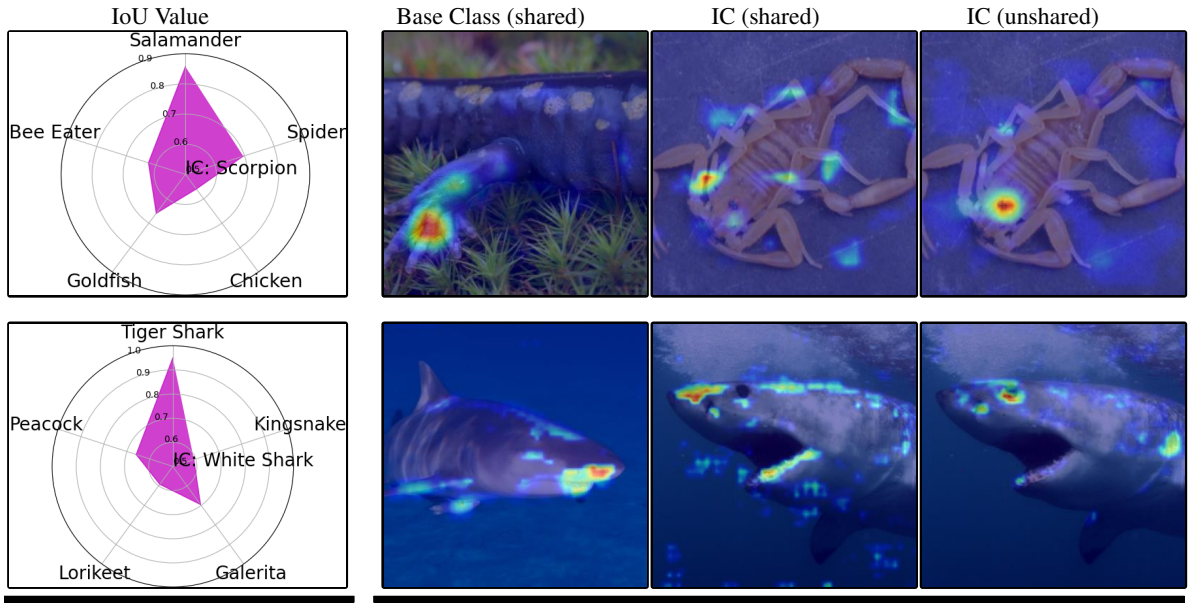
where $a_{i,j}$ represents the accuracy of classes first encountered in phase j after the model has been incrementally trained up to phase i ($i > j$). Other implementation details on the settings are available in the supplementary material.

4.2. Ablation Study

To prove the effectiveness of our proposed method, we conduct several ablation experiments on TinyImageNet. The performance of our scheme is mainly attributed to three prominent components: the class-specific pathway organization strategy (CPO), the pathway-guided feature update (PFU) and the incremental pathway update (IPU) mechanism. Since the three components are sequential, we add them gradually for comparison. As can be seen in Tab. 1, CPO bring a 3.04%, 3.21% and 8.31% improvement in overall performance. It demonstrates that the initial pathway decoupling plays an important role in mitigating the interference during the incremental process, especially in the case of longer phases. IPU and PFU also achieves average improvement of 1 and 2 points, facilitating the rectification of features and pathways during the subsequent incremental processes.

4.3. Analysis

The impact of the pathway optimization strategy. To explore the impact of pathway optimization strategy on the incremental representation learning, we compare our methods to some classical pathway-related ones. Since most of methods are not designed for class-incremental



(a) Pathway Similarity **(b) T-SNE results of the shared and unshared pathways**

Figure 5. Effect of our scheme on the pathway learning. (a) CPO realizes the organization of distinguishable pathways, thus mitigating the overlap between the incremental classes (*i.e.*, IC) and old ones. (b) PFU promotes the pathway expansion of similar classes. The first two columns represent the activated features of shared pathways, and the last represents the unshared ones.

learning, we adapt their core strategies in our settings. As shown in Tab. 2, our method is obviously superior to other ones in three settings. Piggy [16] simply optimizes the mask of parameters on the basis of the initial model, which is not sufficient to handle the complex incremental process. The hard threshold adopted in Hat [20] and Iap [4] brings great optimization difficulty. Although the RPS [17] achieves good results, its complex network structure and random path search strategy are not efficient.

The impact of the numbers of divided groups (*i.e.,* L). To explore the sensitivity of divided groups on the incremental performance, we design the following experiments. We divide the output channels into different channels equally. If the channels are not divisible, we round down it. It can be seen in Fig. 4 (a) that the performance fluctuates little except for exceptionally few divisions, demonstrating the stability of our pathway learning. When the number of division is equal to 2, the overall decoupling space for pathways is too small to promote sparse learning.

The impact of the maximum sparse rate (*i.e.,* ζ_{max}). To explore the effect of sparse rate on the incremental performance, we conduct multiple experiments with different sparse rates on CIFAR-100. As shown in Fig. 4 (b), the performance with high sparse rate is obviously worse than that with other values. In this case, due to the increase of difficulty of pathway independence, the initial classification accuracy is greatly disturbed. When the sparse rate is too low (*e.g.*, 0.2), the initial accuracy is obviously higher, bring the overall improvement of the incremental performance. When the sparsity value is between 0.3 and 0.45, the initial accuracy is consistent and the incremental performance gets better with heavier sparsity, demonstrating the effectiveness of the pathway decoupling.

4.4. Visualization

To better demonstrate the role of CPO and PFU during optimization, we show the corresponding visualization results. In Fig. 5 (a), the center of the circle represents the incremental class, and the surrounding represents the five different base classes. The middle values represent the intersection of union (IoU) of pathways between the new and old classes. It can be seen the pathways are class-specific, and the similarity is also positively related to the class relationship. For example, the pathway of white sharp is closer to the one of tiger sharp. As shown in Fig. 5 (b), for the incremental class, the features of shared and unshared pathways are visualized by t-SNE [15]. For example, the white sharp and tiger sharp are discriminatory to other classes due to the features of teeth. To further distinguish between these two ones, the white shark expands new pathways to learn the texture features on their bodies. Owing to our PFU, the incremental pathways are promoted to differentiate from the old ones, thus improving the separation of novel clusters.

4.5. Comparison with SOTA

To better assess the overall performance, we compare it to the SOTA of NECIL (LwF_MC [18], MUC, SDC, PASS,

	Methods	Average Accuracy (↑)			Average Forgetting (↓)		
		P=5	*P=10*	*P=20*	*P=5*	*P=10*	*P=20*
(1) E=20	iCaRL-CNN*	51.07	48.66	44.43	42.13	45.69	43.54
	iCaRL-NCM*	58.56	54.19	50.51	24.90	28.32	35.53
	EEIL* [2]	60.37	56.05	52.34	23.36	26.65	32.40
	UCIR* [9]	63.78	62.39	59.07	21.00	25.12	28.65
	PODNet‡ [8]	64.88	63.05	61.62	19.12	22.55	25.64
(2) E=0	LwF_MC	45.93	27.43	20.07	44.23	50.47	55.46
	MUC [26]	49.42	30.19	21.27	40.28	47.56	52.65
	SDC‡ [27]	56.77	57.00	58.90	6.96	7.50	10.77
	PASS [31]	63.47	61.84	58.09	25.20	30.25	30.61
	IL2A‡ [30]	65.72	62.69	59.90	27.25	37.35	39.27
	ABD‡ [25]	63.85	62.46	57.40	23.12	27.34	33.42
	SSRE [33]	65.88	65.04	61.70	18.37	19.48	19.00
	Ours	**66.64**+0.76	**65.84**+0.80	**61.83**+0.13	**6.50**+0.46	**3.30**+4.20	**9.14**+1.63

Table 3. Comparisons with other methods on CIFAR-100. P represents the phase number and E represents the exemplar number. Models with an asterisk * represent the reproduced results in [31]. Models with a marker ‡ represent the reproduced results by this paper. The red footnotes in the last row represent the relative improvement compared with the results of SOTA.

	Methods	TinyImageNet			ImageNet-Subset	ImageNet-Full
		P=5	*P=10*	*P=20*	*P=10*	*P=10*
(1) E=20	iCaRL-CNN*	34.64	31.15	27.90	50.53	38.43
	iCaRL-NCM* [18]	45.86	43.29	38.04	60.79	46.72
	EEIL* [2]	47.12	45.01	40.50	63.34	-
	UCIR* [9]	49.15	48.52	42.83	66.16	61.28
(2) E=0	LwF_MC [18]	29.12	23.10	17.43	31.18	-
	MUC [26]	32.58	26.61	21.95	35.07	-
	MAS [1]	18.97	11.82	7.17	19.11	-
	EWC [11]	19.64	16.18	17.09	27.32	-
	PASS [31]	49.55	47.29	42.07	61.80	55.90‡
	SSRE [33]	50.39	48.93	48.17	67.69	58.12‡
	Ours	**53.69**+3.30	**52.88**+3.95	**51.94**+3.77	**69.22**+1.53	**60.20**+2.08

Table 4. Comparisons of the average incremental accuracy (%) with other methods on TinyImageNet, ImageNet-Subset and ImageNet-Full. P represents the number of phases and E represents the number of exemplars.

IL2A, ABD and SSRE) and some classical exemplar-based CIL methods (iCARL [18], EEIL, UCIR and PODNet [8]).

As shown in Tab. 3, compared to the SOTA of non-exemplar methods (*i.e.,* E=0), our method achieves average improvement of about 1 point and 2 points on the average accuracy and average forgetting of CIFAR-100 dataset, respectively. The performance of our method is comparable to the classical exemplar-based methods (*i.e.,* E=20), which shows that our method further mitigate the gap between the two settings. To provide further insight into the behaviors of different methods on larger benchmarks, we compare their average accuracy on TinyImageNet, ImageNet-Subset and ImageNet-Full. As shown in Tab. 4, our method achieves average improvement of 3 points. Due to the larger-scale images in these datasets, the pathway independence during the feature optimization is clearer, bringing greater performance improvement.

5. Conclusion

In this paper, a novel self-organized pathway expansion scheme is presented for the NECIL task. A class-specific pathway organization strategy is first proposed to mitigate the feature interference during the optimization of pathway-aware representation. Based on the learnable pathway planner, a pathway-guided feature update mechanism is introduced to adjust the involvement in joint training of classification and distillation. Experimental results show that our method is superior in both performance and adaptability to the state-of-the-art methods.

Acknowledgments

This work was supported by National Key R&D Program of China under Grant 2020AAA0105700, National Natural Science Foundation of China (NSFC) under Grants 62225207, U19B2038 and 62121002, Alibaba Group through Alibaba Innovative Research Program.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[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)*, pages 233–248, 2018.

[3] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[4] Hung-Jen Chen, An-Chieh Cheng, Da-Cheng Juan, Wei Wei, and Min Sun. Mitigating forgetting in online continual learning via instance-aware parameterization. *Advances in Neural Information Processing Systems*, 33:17466–17477, 2020.

[5] Zhen Cheng, Zhiwei Xiong, Chang Chen, Dong Liu, and Zheng-Jun Zha. Light field super-resolution with zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10010–10019, 2021.

[6] Róbert Csordás, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Are neural nets modular? inspecting functional modularity through differentiable weight masks. In *International Conference on Learning Representations*, 2020.

[7] 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*, pages 248–255. Ieee, 2009.

[8] 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*, pages 86–102. Springer, 2020.

[9] 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*, pages 831–839, 2019.

[10] Xinting Hu, Kaihua Tang, Chunyan Miao, Xiansheng Hua, and Hanwang Zhang. Distilling causal effect of data in class-incremental learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3956–3965, 2021.

[11] 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.

[12] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

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

[14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[15] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[16] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 67–82, 2018.

[17] Jathushan Rajasegaran, Munawar Hayat, Salman Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for incremental learning. *Advances in Neural Information Processing Systems*, 2019.

[18] 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*, pages 2001–2010, 2017.

[19] Sudip Roy, Jeff Dean, Sanjay Ghemawat, Ryan Sepassi, Hyeontaek Lim, Michael Isard, Paul Barham, Yonghui Wu, Laurent Shafey, Aakanksha Chowdhery, et al. Pathways: Asynchronous distributed dataflow for ml. *Proceedings of Machine Learning and Systems*, 4, 2022.

[20] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR, 2018.

[21] Gido M. van de Ven and A. Tolias. Three scenarios for continual learning. *ArXiv*, abs/1904.07734, 2019.

[22] Jiamin Wu, Tianzhu Zhang, Zheng-Jun Zha, Jiebo Luo, Yongdong Zhang, and Feng Wu. Self-supervised domain-aware generative network for generalized zero-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12767–12776, 2020.

[23] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 374–382, 2019.

[24] 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*, pages 3014–3023, 2021.

[25] 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*, pages 8715–8724, 2020.

[26] L. Yu, S. Parisot, G. Slabaugh, J. Xu, and T. Tuytelaars. More classifiers, less forgetting: A generic multi-classifier

paradigm for incremental learning. *European Conference on Computer Vision*, 2020.

[27] 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*, pages 6982–6991, 2020.

[28] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[29] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2921–2929, 2016.

[30] Fei Zhu, Zhen Cheng, Xu-yao Zhang, and Cheng-lin Liu. Class-incremental learning via dual augmentation. *Advances in Neural Information Processing Systems*, 34, 2021.

[31] 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*, pages 5871–5880, 2021.

[32] Kai Zhu, Yang Cao, Wei Zhai, Jie Cheng, and Zheng-Jun Zha. Self-promoted prototype refinement for few-shot class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6801–6810, 2021.

[33] 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*, pages 9296–9305, 2022.