DS-AL: A Dual-Stream Analytic Learning for Exemplar-Free Class-Incremental Learning

Huiping Zhuang[1], **Run He**[1], **Kai Tong**[1], **Ziqian Zeng**[1], **Cen Chen**[2,3*], **Zhiping Lin**[4]

[1]Shien-Ming Wu School of Intelligent Engineering, South China University of Technology, China
[2] School of Future Technology, South China University of Technology, China
[3] Pazhou Laboratory, China
[4] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
{hpzhuang, zqzeng, chencen}@scut.edu.cn, {wirh, wikaitong}@mail.scut.edu.cn, ezplin@ntu.edu.sg

Abstract

Class-incremental learning (CIL) under an exemplar-free constraint has presented a significant challenge. Existing methods adhering to this constraint are prone to catastrophic forgetting, far more so than replay-based techniques that retain access to past samples. In this paper, to solve the exemplar-free CIL problem, we propose a Dual-Stream Analytic Learning (DS-AL) approach. The DS-AL contains a main stream offering an analytical (i.e., closed-form) linear solution, and a compensation stream improving the inherent under-fitting limitation due to adopting linear mapping. The main stream redefines the CIL problem into a Concatenated Recursive Least Squares (C-RLS) task, allowing an equivalence between the CIL and its joint-learning counterpart. The compensation stream is governed by a Dual-Activation Compensation (DAC) module. This module re-activates the embedding with a different activation function from the main stream one, and seeks fitting compensation by projecting the embedding to the null space of the main stream's linear mapping. Empirical results demonstrate that the DS-AL, despite being an exemplar-free technique, delivers performance comparable with or better than that of replay-based methods across various datasets, including CIFAR-100, ImageNet-100 and ImageNet-Full. Additionally, the C-RLS' equivalent property allows the DS-AL to execute CIL in a phase-invariant manner. This is evidenced by a never-before-seen 500-phase CIL ImageNet task, which performs on a level identical to a 5-phase one. Our codes are available at https://github.com/ZHUANGHP/Analytic-continual-learning.

Introduction

Class-incremental learning (CIL) (Li and Hoiem 2018; Rebuffi et al. 2017) updates a network's parameters in a phase-by-phase manner, with data arriving separately in each training phase. CIL has gained popularity for its ability to adapt trained models to unseen data categories, reducing energy consumption. Its development is emphasized by the fact that data and target categories are often available at specific locations or time slots. Furthermore, CIL is intuitively inspired by the human learning process, where individuals build upon their existing knowledge by assimilating new information continuously.

*Corresponding author

CIL offers several benefits but can also lead to *catastrophic forgetting* (Belouadah, Popescu, and Kanellos 2021), where models quickly lose previously learned knowledge when acquiring new tasks. This issue is exacerbated under an exemplar-free constraint, where previous experience cannot be stored or revisited. Researchers have proposed various solutions, including replay-based CIL and exemplar-free CIL (EFCIL). Replay-based methods perform competitively but violate the exemplar-free constraint. EFCIL techniques honor data privacy but often yield inferior results.

The EFCIL family consists of a few sub-branches, with regularization-based methods being the most prevalent branch. These methods reduce catastrophic forgetting by imposing constraints to prevent the change of important weights or activations. However, they are usually inadequate as their performances cannot match those of replay-based counterparts (Liu, Schiele, and Sun 2021b). Recently, a new branch of CIL, the analytic learning (AL) based CIL (Zhuang et al. 2022), has shown promising results even while adhering to the exemplar-free constraint.

The AL-based methods identify the iterative mechanism as the primary cause of catastrophic forgetting and replace it with linear recursive tools. For the first time, they provide results on par with those of replay-based techniques. Additionally, the recursive operation allows the AL-based CIL to give strong performance especially under large-phase scenarios (e.g., 50-phase). However, existing AL-based methods may suffer from an under-fitting dilemma because they rely solely on one linear projection. This has motivated the exploration of potential approaches to compensate for the limited fitting power of vanilla linearity inherited from the AL-based CIL.

In this paper, we introduce a Dual-Stream Analytic Learning (DS-AL). The DS-AL contains a main stream offering an analytical (i.e., closed-form) linear solution, and a compensation stream that improves the inherent under-fitting limitation due to adopting linear mapping. The DS-AL develops the AL-based CIL family by compensating the lack of fitting power, without losing the basic benefits inherited from this category. The key contributions are summarized as follows.

- We present the DS-AL, an exemplar-free technique that offers an analytical solution to the CIL problem.
- The DS-AL's main stream redefines the CIL problem

into a Concatenated Recursive Least Squares (C-RLS) task, allowing an equivalence between the CIL and its joint-learning. Therefore, models trained in a CIL manner yield identical results to those employing data from both current and historical phases concurrently.

• The DS-AL compensation stream introduces a Dual-Activation Compensation (DAC) module. This enables our method to overcome the under-fitting limitation inherited from the AL-based CIL.

• Our experiments on benchmark datasets show that the DS-AL, despite being an EFCIL technique, delivers performance better than that of existing AL-based techniques and even surpasses most replay-based methods. Additionally, we conduct a never-before-seen 500-phase task on ImageNet. It achieves a near-identical result to its 5-phase counterpart, further highlighting DS-AL's equivalent property.

Related Works

In this section, we review CIL techniques related to our proposed method, which fall into two categories: replay-based and EFCIL techniques.

Replay-based CIL

Replay-based CIL reinforces models' memory of past knowledge by replaying historical experience. The replay mechanism was first introduced by iCaRL (Rebuffi et al. 2017) and has gained popularity with various attempts follow due to its competitive performance. For instance, end-to-end incremental learning (Castro et al. 2018) introduces balanced training via replaying. Bias correction (Wu et al. 2019) includes an extra trainable layer. LUCIR (Hou et al. 2019) replaces the softmax layer with a cosine one. PODNet (Douillard et al. 2020) implements spatial-based distillation loss. FOSTER (Wang et al. 2022) adopts a two-stage learning by expanding and reducing the network.

There are also plug-in techniques that can be attached to existing replay-based methods, achieving state-of-the-art (SOTA) performance. For instance, AANets (Liu, Schiele, and Sun 2021a) incorporates stable and plastic blocks to balance stability and plasticity, improving performance when plugged into techniques such as PODNet. Similarly, the Mnemonics technique (Liu et al. 2020) explores exemplar storing mechanism. Reinforced memory management (RMM) (Liu, Schiele, and Sun 2021b) leverages reinforcement learning and constructs dynamic memory management for exemplars, achieving outstanding results when attached to PODNet and AANets. Online hyperparameter optimization (Liu et al. 2023) adaptively optimizes the stability-plasticity balance without prior knowledge.

In general, replay-based CIL achieves satisfactory results with the need to store previously visited samples.

Exemplar-free CIL

EFCIL methods branch roughly into regularization-based CIL, prototype-based CIL, and AL-based CIL.

Regularization-based CIL Regularization-based methods introduce additional constraints to mitigate forgetting by constructing new loss functions. These constraints can be applied to network activations and penalize changes of important ones. Examples include less-forgetting learning (Jung et al. 2016), which penalizes activation difference, and learning without forgetting (LwF) (Li and Hoiem 2018), which prevents activation changes between old and new networks. Alternatively, regularization can be imposed on network parameters, such as elastic weight consolidation (EWC) (Kirkpatrick et al. 2017), which captures prior importance by a diagonally approximated Fisher information matrix. Building upon EWC, (Liu et al. 2018) seeks a more appropriate replacement of the Fisher information matrix.

Prototype-based CIL Prototype-based CIL mitigates forgetting by storing prototypes for each class to avoid overlapping representations of new and old classes. For instance, PASS (Zhu et al. 2021b) augments the prototypes of features to distinguish previous classes. SSRE (Zhu et al. 2022) introduces a prototype selection mechanism that selectively incorporates new samples into the distillation to enhance the distinctiveness between old and new classes. Fetril (Petit et al. 2023) reduces forgetting by generating pseudo-features of old classes from new representations.

Analytic Learning based CIL AL-based methods are newly developed exemplar-free tools specifically designed to protect data privacy during incremental learning. They are inspired by AL (Guo, Lyu, and Mastorakis 2001; Zhuang, Lin, and Toh 2021), where the training of neural networks yields a closed-form solution using least squares (LS). Analytic CIL (ACIL) (Zhuang et al. 2022) was first developed, reformulating the CIL procedure into a recursive analytic learning process. The need for storing exemplars is evicted by preserving a correlation matrix. ACIL is subsequently followed by Gaussian Kernel Embedded Analytic Learning (GKEAL) (Zhuang et al. 2023). GKEAL specializes in few-shot CIL settings by adopting a Gaussian kernel process that excels in data-scarce scenarios.

AL-based CIL is an emerging CIL branch, exhibiting strong performance even at its first proposal as ACIL. However, relying solely on one linear projection would likely lead to under-fitting. In this paper, we overcome this limitation by introducing DS-AL, where the DAC module compensates for the lack of fitting ability.

The Proposed Method

In this section, we present our DS-AL training algorithm (see Figure 1). This starts with a backpropagation (BP) based backbone training on the base dataset (see Figure 1(a)). The AL-based training steps follow, including an AL-based re-training (i.e., Figure 1(b)) and follow-up CIL procedures (i.e., Figure 1(c)-(d)). We focus on the AL-based training steps, each of which is delivered in a dual-stream manner, with the main stream contributed by the C-RLS and the compensation stream governed by the DAC module.

BP-based Training

The network is first trained with an iterative BP on the base dataset (see Figure 1(a)) for multiple epochs with an appropriate learning scheduling. For simplicity, we only discuss

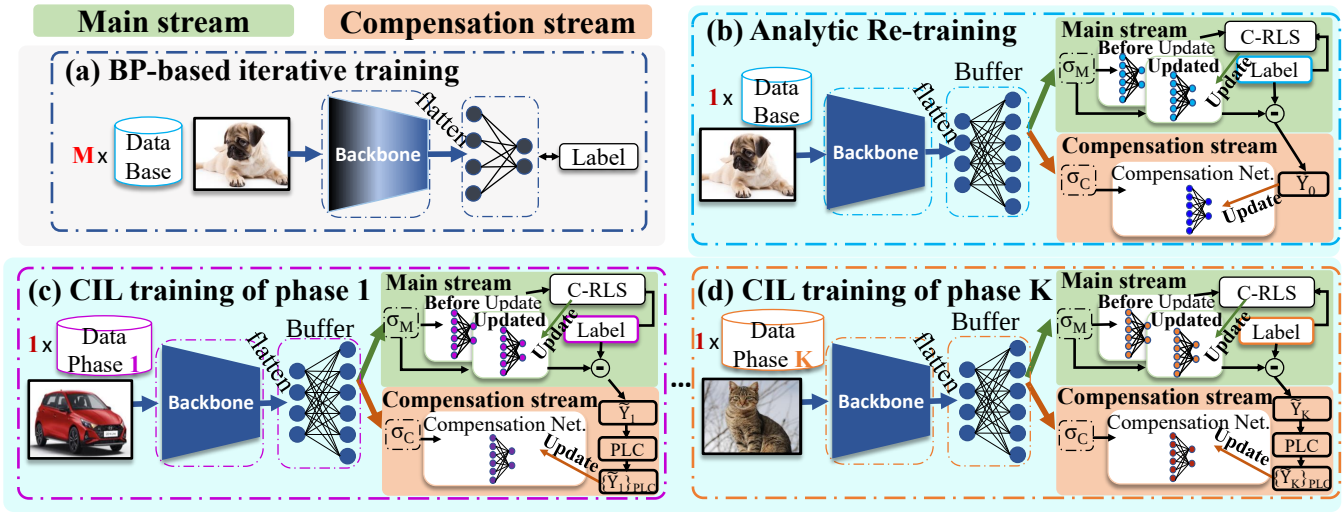


Figure 1: The DS-AL consists of (a) BP-based training on the base dataset, followed by (b)-(d) AL-based training steps. Each step includes a main stream (green block) with a C-RLS module and a compensation stream (orange block) using the mapping residue from main stream as the label. (b) DS-AL initializes CIL by replacing and re-training the classification head with an AL-based one that includes a buffer layer and a linear classifier. (c)-(d) The CIL is then recursively conducted, incorporating a PLC module (defined in (15)) to ensure the incremental constraint in the compensation stream.

convolutional neural networks (CNN) examples. Let $\mathbf{W}_{\text{CNN}}$ and $\mathbf{W}_{\text{FCN}}$ be the parameters representing the CNN backbone and the fully-connected classifier. After the BP training, given an input $\mathbf{X}$, the output of the network is

$$\mathbf{Y} = f_{\text{softmax}}(f_{\text{flat}}(f_{\text{CNN}}(\mathbf{X}, \mathbf{W}_{\text{CNN}}))\mathbf{W}_{\text{FCN}}) \tag{1}$$

where $f_{\text{CNN}}(\mathbf{X}, \boldsymbol{W}_{\text{CNN}})$ indicates the CNN output; f_{softmax} and f_{flat} are softmax function and flattening operator (i.e., reshaping a tensor into a 1-D vector).

After the BP training, the backbone's weights are obtained. We then *freeze the backbone and replace the classifier with a 2-layer AL network for re-training and the forthcoming CIL steps*. This is delivered in a dual-stream manner as follows.

The Main Stream of DS-AL

Prior to further processing, a few definitions are made. Let the network be incrementally trained for K phases where training data of each phase comes with different classes. Let $\mathcal{D}_k^{\text{train}} \sim \{\boldsymbol{X}_k^{\text{train}}, \boldsymbol{Y}_k^{\text{train}}\}$ and $\mathcal{D}_k^{\text{test}} \sim \{\boldsymbol{X}_k^{\text{test}}, \boldsymbol{Y}_k^{\text{test}}\}$ be the training and testing datasets at phase k ($k = 0, 1, \ldots, K$). $\boldsymbol{X}_k \in \mathbb{R}^{N_k \times c \times w \times h}$ (e.g., N_k images with a shape of $c \times w \times h$) and $\boldsymbol{Y}_k^{\text{train}} \in \mathbb{R}^{N_k \times d_{y_k}}$ (with phase k including d_{y_k} classes) are stacked input and label (one-hot) tensors.

AL-based Re-training. We first conduct the AL-based re-training (see Figure 1(b)) on the base training set (represented by $\mathcal{D}_0^{\text{train}} \sim \{\boldsymbol{X}_0^{\text{train}}, \boldsymbol{Y}_0^{\text{train}}\}$). The AL-based classifier includes a *Buffer* layer and a linear mapping. The buffer layer is a common trick to improve AL-based methods' performance (Zhuang, Lin, and Toh 2021). To show this, we feed the input $\boldsymbol{X}_0^{\text{train}}$ through the trained CNN backbone, followed by a flattening operation, to extract features, i.e.,

$$\boldsymbol{X}_0^{\text{cnn}} = f_{\text{flat}}(f_{\text{CNN}}(\boldsymbol{X}_0^{\text{train}}, \boldsymbol{W}_{\text{CNN}})) \tag{2}$$

where $\boldsymbol{X}_0^{\text{cnn}} \in \mathbb{R}^{N_0 \times d_{\text{cnn}}}$. Then the buffer layer (denoted by B) is inserted to map the feature to a different space. That is, the feature $\boldsymbol{X}_0^{\text{cnn}}$ is mapped to the *main stream activation* $\boldsymbol{X}_{\text{M},0}$ as follows

$$\boldsymbol{X}_{\text{M},0} = \sigma_{\text{M}}(B(f_{\text{CNN}}(\boldsymbol{X}_0^{\text{train}}, \boldsymbol{W}_{\text{CNN}})) = \sigma_{\text{M}}(B(\boldsymbol{X}_0^{\text{cnn}})) \tag{3}$$

where σ_{M} is an activation function. Here, we adopt *ReLU* following many AL-based techniques (Zhuang et al. 2022).

There are a few choices to construct the buffer layer. The most common one is the random projection (Zhuang et al. 2022). This is simply a linear projection to map the feature to a higher-dimension space, i.e., $B(\boldsymbol{X}_0^{\text{cnn}}) = \boldsymbol{X}_0^{\text{cnn}}\boldsymbol{X}_{\text{B}}$. Another known buffer structure is the Gaussian kernel mapping (Zhuang et al. 2023), which projects the feature to a space spanned by computation of the Gaussian kernel function. The buffer layer selection is not the focus of this paper, so we simply follow the ACIL to adopt the random projection represented by $\boldsymbol{X}_{\text{B}} \in \mathbb{R}^{d_{\text{cnn}} \times d_B}$.

The second layer of the AL-based network is a linear layer (denoted by $\boldsymbol{W}_{\text{M}}^{(0)}$). This is constructed by linearly mapping the $\boldsymbol{X}_{\text{M},0}$ to the label matrix $\boldsymbol{Y}_0^{\text{train}}$ via solving the following regularized linear problem

$$\underset{\boldsymbol{W}_{\text{M}}^{(0)}}{\text{argmin}} \quad \left\|\boldsymbol{Y}_0^{\text{train}} - \boldsymbol{X}_{\text{M},0}\boldsymbol{W}_{\text{M}}^{(0)}\right\|_F^2 + \gamma \left\|\boldsymbol{W}_{\text{M}}^{(0)}\right\|_F^2 \tag{4}$$

where $\|\cdot\|_F$ indicates the Frobenius norm, and γ is a regularization parameter. The optimal estimation (Zhuang, Lin, and Toh 2021) of $\boldsymbol{W}_{\text{M}}^{(0)}$ can be found in

$$\hat{\boldsymbol{W}}_{\text{M}}^{(0)} = (\boldsymbol{X}_{\text{M},0}^T\boldsymbol{X}_{\text{M},0} + \gamma\boldsymbol{I})^{-1}\boldsymbol{X}_{\text{M},0}^T\boldsymbol{Y}_0^{\text{train}} \tag{5}$$

in which $\cdot^T$ is the matrix transpose operator. Upon obtaining $\hat{\boldsymbol{W}}_{\mathrm{M}}^{(0)}$, the AL-based re-training is completed.

The AL-based CIL. Upon completing the base training, we proceed with the CIL steps using C-RLS in a recursive and analytical manner. To illustrate this, without loss of generality, assume that we are given $\mathcal{D}_0^{\text{train}}, \ldots, \mathcal{D}_{k-1}^{\text{train}}$, and let $\boldsymbol{X}_{\mathrm{M},0:k-1} \in \mathbb{R}^{N_{0:k-1} \times d_B}$ and $\boldsymbol{Y}_{0:k-1} \in \mathbb{R}^{N_{0:k-1} \times \sum_{i=1}^{k-1} d_{y_i}}$ be the concatenated activation and label tensors respectively from phase 0 to $k-1$, i.e.,

$$\boldsymbol{X}_{\mathrm{M},0:k-1} = \begin{bmatrix} \boldsymbol{X}_{\mathrm{M},0} \\ \vdots \\ \boldsymbol{X}_{\mathrm{M},k-1} \end{bmatrix} \quad \boldsymbol{Y}_{0:k-1} = \begin{bmatrix} \boldsymbol{Y}_0^{\text{train}} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Y}_1^{\text{train}} & \boldsymbol{0} & \ldots & \boldsymbol{0} \\ \vdots & \vdots & & \ddots & \\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \ldots & \boldsymbol{Y}_{k-1}^{\text{train}} \end{bmatrix} \tag{6}$$

where

$$\boldsymbol{X}_{\mathrm{M},k-1} = \sigma_{\mathrm{M}}(B(f_{\text{flat}}(f_{\text{CNN}}(\boldsymbol{X}_{k-1}^{\text{train}}, \boldsymbol{W}_{\text{CNN}})))) \tag{7}$$

and $\boldsymbol{N}_{0:k-1}$ indicates the total number of data samples from phase 0 to $k-1$. $\boldsymbol{Y}_{0:k-1}$'s sparse structure is because data classes among different phases are mutually exclusive. The learning problem can then be extended to

$$\underset{\boldsymbol{W}_{\mathrm{M}}^{(k-1)}}{\operatorname{argmin}} \quad \left\| \boldsymbol{Y}_{0:k-1} - \boldsymbol{X}_{\mathrm{M},0:k-1} \boldsymbol{W}_{\mathrm{M}}^{(k-1)} \right\|_F^2 + \gamma \left\| \boldsymbol{W}_{\mathrm{M}}^{(k-1)} \right\|_F^2. \tag{8}$$

According to (5), at phase $k-1$, we have

$$\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k-1)} = \left(\boldsymbol{X}_{\mathrm{M},0:k-1}^T \boldsymbol{X}_{\mathrm{M},0:k-1} + \gamma \boldsymbol{I} \right)^{-1} \boldsymbol{X}_{\mathrm{M},0:k-1}^T \boldsymbol{Y}_{0:k-1} \tag{9}$$

where $\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k-1)} \in \mathbb{R}^{d_{\mathrm{B}} \times \sum_{i=1}^{k} d_{y_i}}$ whose column size is expanded as k increases. Let

$$\boldsymbol{R}_{\mathrm{M},k-1} = (\boldsymbol{X}_{\mathrm{M},0:k-1}^T \boldsymbol{X}_{\mathrm{M},0:k-1} + \gamma \boldsymbol{I})^{-1} \tag{10}$$

be an *inverted auto-correlation matrix* (iACM), which captures the correlation information of both present and past samples. Building upon the above derivations, *the goal of the main stream CIL is to calculate $\hat{\boldsymbol{W}}_M^{(k)}$ using only $\hat{\boldsymbol{W}}_M^{(k-1)}$, $\boldsymbol{R}_{M,k-1}$ and data $\boldsymbol{X}_k^{train}$ of phase k, without involving historical samples such as $\boldsymbol{X}_{0:k-1}$*. To this end, we formulate the CIL process as a C-RLS indicated in the following theorem.

Theorem 1. *Let $\hat{\boldsymbol{W}}_M^{(k)}$ be the optimal estimation of $\boldsymbol{W}_M^{(k)}$ using* (9) *with all the training data from phase* 0 *to* k*. Let $\hat{\boldsymbol{W}}_M^{(k-1)\prime} = [\hat{\boldsymbol{W}}_M^{(k-1)} \quad \boldsymbol{0}]$, and $\hat{\boldsymbol{W}}_M^{(k)}$ can be equivalent calculated via*

$$\hat{\boldsymbol{W}}_M^{(k)} = \hat{\boldsymbol{W}}_M^{(k-1)\prime} + \boldsymbol{R}_{M,k} \boldsymbol{X}_{M,k}^T (\boldsymbol{Y}_k^{train} - \boldsymbol{X}_{M,k} \hat{\boldsymbol{W}}_M^{(k-1)\prime}) \tag{11}$$

where

$$\boldsymbol{R}_{M,k} = \boldsymbol{R}_{M,k-1} - \boldsymbol{R}_{M,k-1} \boldsymbol{X}_{M,k}^T (\boldsymbol{I} + \boldsymbol{X}_{M,k} \boldsymbol{R}_{M,k-1} \boldsymbol{X}_{M,k}^T)^{-1} \boldsymbol{X}_{M,k} \boldsymbol{R}_{M,k-1}. \tag{12}$$

Proof. See Supplementary material A[1]. □

As shown in Theorem 1, the C-RLS concatenates the weight matrix (i.e., building $\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k-1)\prime}$) and updates it recursively. Notably, the recursive formulas in (11) and (12) are the widely recognized RLS (Hayes 1996). Hence, with C-RLS, the main stream has constructed a non-forgetting CIL mechanism. Upon freezing the backbone, the CIL is equivalent to its joint-learning counterpart as shown in Theorem 1. That is, the model trained incrementally yields identical weights to that trained employing the entire data.

[1]The supplementary materials can be found in https://github.com/ZHUANGHP/Analytic-continual-learning

The Compensation Stream of DS-AL

The process of C-RLS is built entirely on one linear projection. When training samples are complex, under-fitting may occur. To overcome this, we introduce the compensation stream governed by the DAC module to help overcome the under-fitting limitation of linear mapping.

The compensation stream operates in a similar manner to the main stream, but it differs in that the label matrix is generated using the residue from the main stream. Without loss of generality, assume that we have executed the main stream at phase k (i.e., obtaining $\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)}$ and $\boldsymbol{R}_{\mathrm{M},k}$) and the compensation stream at phase $k-1$ (i.e., obtaining compensation weight matrix $\hat{\boldsymbol{W}}_{\mathrm{C}}^{(k-1)}$ and corresponding correlation matrix $\boldsymbol{R}_{\mathrm{C},k-1}$). Let $\tilde{\boldsymbol{Y}}_k$ be the residue after conducting the main stream, i.e.,

$$\tilde{\boldsymbol{Y}}_k = [\boldsymbol{0}_{N_{0:k-1} \times d_{y_{k-1}}} \quad \boldsymbol{Y}_k^{\text{train}}] - \boldsymbol{X}_{\mathrm{M},k} \hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)} \tag{13}$$

where the zero matrix is due to the mutual-exclusive CIL setting. Let

$$\boldsymbol{X}_{\mathrm{C},k} = \sigma_{\mathrm{C}}(B(f_{\text{flat}}(f_{\text{CNN}}(\boldsymbol{X}_k^{\text{train}}, \boldsymbol{W}_{\text{CNN}})))) \tag{14}$$

be the *compensation stream activation*, where σ_{C} is an activation function different from σ_{M}. Here σ_{C} is a hyperparameter to be determined (e.g., can be $Tanh$, $Mish$ or others), which will be explored during the experiments.

The $\tilde{\boldsymbol{Y}}_k$ can be treated as the null space of $\boldsymbol{X}_{\mathrm{M},k}$, where the embedding cannot reach. The key idea of the DAC module is to *map to the null space using an alternative embedding $\boldsymbol{X}_{C,k}$*, attempting to further improve the fitting ability. To achieve this, we construct a dual recursive CIL process. Similar to the C-RLS indicated in Theorem 1, the DAC module would lead to a resembling recursive structure.

Before proceeding to the later step, a Previous Label Cleansing (PLC) process is adopted, i.e.,

$$\{\tilde{\boldsymbol{Y}}_k\}_{\text{PLC}} = [\boldsymbol{0}_{N_{0:k-1} \times d_{y_{k-1}}} \quad (\tilde{\boldsymbol{Y}}_k)_{\text{new}}] \tag{15}$$

where $(\tilde{\boldsymbol{Y}}_k)_{\text{new}}$ indicates a submatrix by keeping the last d_{y_k} columns in $\tilde{\boldsymbol{Y}}_k$. Note that PLC does not apply during the initial phase, i.e., $\{\tilde{\boldsymbol{Y}}_0\}_{\text{PLC}} = \tilde{\boldsymbol{Y}}_0$.

To illustrate the need for PLC, let $(\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{old}}$ and $(\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{new}}$ represent the submatrices by keeping the first $\sum_{i=0}^{k-1} d_{y_i}$ and the last d_{y_k} columns respectively, namely, the weight components corresponding to the previous phases and the current phase k. We then rewrite (13) into

$$\begin{aligned} \tilde{\boldsymbol{Y}}_k &= [\boldsymbol{0}_{N_{0:k-1} \times d_{y_{k-1}}} \quad \boldsymbol{Y}_k^{\text{train}}] - \boldsymbol{X}_k [(\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{old}} \quad (\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{new}}] \\ &= [-\boldsymbol{X}_k (\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{old}} \quad \boldsymbol{Y}_k^{\text{train}} - \boldsymbol{X}_k (\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{new}}]. \end{aligned} \tag{16}$$

By comparing (15) with (16), we should find that the label using PLC in (15) is more reasonable. That is because, during the CIL, the data classes are mutually exclusive. This setting should also be applied in the proposed DAC module, otherwise (16) could provide non-zero false supervision for previous phases (i.e., $-\boldsymbol{X}_k (\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)})_{\text{old}}$). This will be empirically evidenced in the experiment section.

	Method	EFCIL?	*CIFAR-100*			*ImageNet-100*			*ImageNet-Full*		
			K=5	25	50	K=5	25	50	K=5	25	50
$\bar{\mathcal{A}}$	LUCIR	×	63.17	57.54	-	70.84	61.44	-	64.45	56.56	-
	PODNet	×	64.83	60.72	57.98	75.54	68.31	62.48	66.95	59.17	-
	AANets	×	66.31	62.31	-	76.96	71.78	-	67.73	61.78	-
	RMM	×	68.36	64.12	-	79.50	75.01	-	69.21	63.93	-
	FOSTER	×	-	67.95	-	-	69.34	-	-	-	-
	LwF	✓	49.59	45.51	-	53.62	44.32	-	51.50	43.14	-
	ACIL	✓	66.30	65.95	66.01	74.81	74.59	74.13	65.34	64.63	64.35
	PASS	✓	(63.88*)	(56.86*)	(41.11*)	72.24*	52.02*	30.59*	-	-	-
	IL2A	✓	(65.53*)	(53.15*)	(21.49*)	-	-	-	-	-	-
	FeTrIL	✓	(66.30)	-	-	72.20	-	-	66.10	-	-
	iVoro-ND	✓	(67.55)	-	-	-	-	-	-	-	-
	DS-AL	✓	**66.39** ± 0.09	**66.20** ± 0.05	**66.33** ± 0.11	-	-	-	-	-	-
	DS-AL	✓	**(68.39)** ± 0.16	**(68.40)** ± 0.20	**(68.26)** ± 0.08	**75.19** ± 0.10	**75.03** ± 0.07	**74.77** ± 0.11	**67.18** ± 0.03	**66.81** ± 0.03	**66.79** ± 0.02
$\mathcal{A}_K$	LUCIR	×	54.30	48.35	-	60.00	49.26	-	56.60	46.23	-
	PODNet	×	54.60	51.40	-	67.60	55.34	-	58.90	50.51	-
	AANets	×	59.39	53.55	-	69.40	63.69	-	60.84	53.21	-
	RMM	×	59.00	56.50	-	73.80	68.84	-	62.50	55.50	-
	LwF	✓	43.36	41.66	-	55.32	55.12	-	48.70	49.84	-
	ACIL	✓	57.78	57.65	57.83	66.98	67.16	67.22	56.11	55.43	56.09
	PASS	✓	(55.75*)	(44.76*)	(28.02*)	61.76*	37.46*	18.22*	-	-	-
	IL2A	✓	(53.36*)	(35.27*)	(11.03*)	-	-	-	-	-	-
	iVoro-ND	✓	(57.25)	-	-	-	-	-	-	-	-
	DS-AL	✓	**58.26** ± 0.09	**58.32** ± 0.12	**58.37** ± 0.15	-	-	-	-	-	-
	DS-AL	✓	**(61.44)** ± 0.08	**(61.41)** ± 0.26	**(61.35)** ± 0.26	**68.00** ± 0.17	**67.72** ± 0.13	**67.80** ± 0.15	**58.17** ± 0.04	**58.10** ± 0.02	**58.15** ± 0.03

Table 1: Comparison of average accuracy $\bar{A}$ and last-phase accuracy $\mathcal{A}_K$ among EFCIL and replay-based methods. Results from replay-based methods are cited from their papers. On CIFAR-100, data in bracket is for ResNet-18. Data in Bold are the best within EFCIL methods and data underlined are the best considering both categories. Data with "*" are those we reproduce via official codes. "-" means the results are not available. Lower lines in results of DS-AL are standard deviations.

Upon providing the input $\boldsymbol{X}_{\mathrm{C},k}$ and label $\{\tilde{\boldsymbol{Y}}_k\}_{\mathrm{PLC}}$, we can proceed to recursively update the compensation weight $\boldsymbol{W}_{\mathrm{C}}^{(k)}$ following the same C-RLS structure indicated in Theorem 1. Let $\hat{\boldsymbol{W}}_{\mathrm{C}}^{(k-1)\prime} = [\hat{\boldsymbol{W}}_{\mathrm{C}}^{(k-1)} \quad \mathbf{0}]$, and we have

$$\hat{\boldsymbol{W}}_{\mathrm{C}}^{(k)} = \hat{\boldsymbol{W}}_{\mathrm{C}}^{(k-1)\prime} + \boldsymbol{R}_{\mathrm{C},k}\boldsymbol{X}_{\mathrm{C},k}^{T}(\{\tilde{\boldsymbol{Y}}_k\}_{\mathrm{PLC}} - \boldsymbol{X}_{\mathrm{C},k}\hat{\boldsymbol{W}}_{\mathrm{C}}^{(k-1)\prime}) \quad (17)$$

where

$$\boldsymbol{R}_{\mathrm{C},k} = \boldsymbol{R}_{\mathrm{C},k-1} - \boldsymbol{R}_{\mathrm{C},k-1}\boldsymbol{X}_{\mathrm{C},k}^{T}(\boldsymbol{I} + \boldsymbol{X}_{\mathrm{C},k}\boldsymbol{R}_{\mathrm{C},k-1}\boldsymbol{X}_{\mathrm{C},k}^{T})^{-1}\boldsymbol{X}_{\mathrm{C},k}\boldsymbol{R}_{\mathrm{C},k-1} \quad (18)$$

which concludes the compensation stream.

Finally, during inference, the prediction is produced by combining both streams as follows.

$$\hat{\boldsymbol{Y}}_k^{(\mathrm{all})} = \boldsymbol{X}_{\mathrm{M},k}\hat{\boldsymbol{W}}_{\mathrm{M}}^{(k)} + \mathcal{C}\boldsymbol{X}_{\mathrm{C},k}\hat{\boldsymbol{W}}_{\mathrm{C}}^{(k)} \quad (19)$$

where $\mathcal{C}$ is a compensation ratio indicating to what extent the network relies on compensation to enhance its fitting ability. The DS-AL is summarized in algorithm framework (we place the algorithm in Supplementary material B.

Experiments

In the section, we conduct experiments on various benchmark datasets, and compare the DS-AL with SOTA EFCIL methods, including LwF (Li and Hoiem 2018), PASS (Zhu et al. 2021b), IL2A (Zhu et al. 2021a), ACIL (Zhuang et al. 2022), FeTrIL (Petit et al. 2023) and iVoro-ND (Ma et al. 2023). We also include replay-based methods, i.e., LUCIR (Hou et al. 2019), PODNet (Douillard et al. 2020), AANets (Liu, Schiele, and Sun 2021a), RMM (Liu, Schiele, and Sun 2021b) and FOSTER (Wang et al. 2022).

Experimental Setup

Basic Setup. We follow the setting from the ACIL, including datasets, architectures, training strategies, and CIL evaluation protocols. Datasets include CIFAR-100, ImageNet-100 and ImageNet-Full. Architectures are ResNet-32 on CIFAR-100 and ResNet-18 on both ImageNet-100 and ImageNet-Full, respectively. Additionally, we train a ResNet-18 on CIFAR-100 as many EFCIL methods (e.g. (Zhu et al. 2021b) and (Ma et al. 2023)) adopt this structure. We adopt the same training strategies as that of the ACIL (see details in Supplementary material C.

For CIL evaluation, the network is first trained (i.e., phase 0) on the base dataset containing half of the full data classes. Subsequently, the network gradually learns the remaining classes evenly for K phases. Most existing methods only report small-phase results, e.g., those of $K = 5, 25$. We include $K = 50, 100, 250, 500$ as well to validate DS-AL's large-phase performance.

Hyperparameters. Two unique hyperparameters (i.e., σ_{C} and $\mathcal{C}$) have been introduced in this paper. We utilize grid search to determine their values. For other hyperparameters shared among AL-based methods, we copy them from the

ACIL for simplicity. The details can be found in Supplementary material C.

Evaluation Metric

Two metrics are adopted for evaluation. The overall performance is evaluated by the *average incremental accuracy* (or average accuracy) $\bar{\mathcal{A}}$ (%): $\bar{\mathcal{A}} = \frac{1}{K+1}\sum_{k=0}^{K}\mathcal{A}_k$, where $\mathcal{A}_k$ indicates the average test accuracy at phase k by testing the network on $\mathcal{D}_{0:k}^{\text{test}}$. A higher $\bar{\mathcal{A}}$ score is preferred when evaluating CIL algorithms. The other evaluation metric is the *last-phase accuracy* $\mathcal{A}_K$ (%) measuring the network's last-phase performance upon completing all CIL tasks. $\mathcal{A}_K$ (%) is an important metric as reveals the gap between the CIL and the joint training, a gap all CIL methods stride to close.

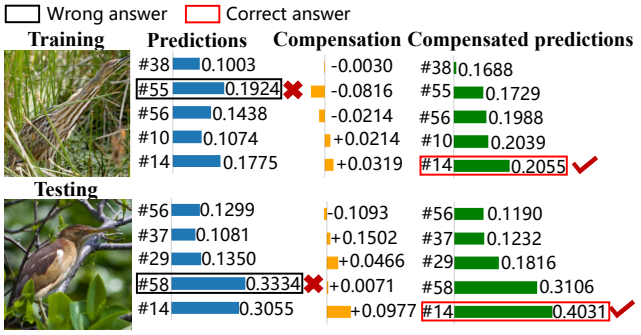


Figure 2: The change of top-5 predictions of images w/ and w/o compensation ($\mathcal{C} = 1.0$). The compensation improves the fitting as well as generalization abilities.

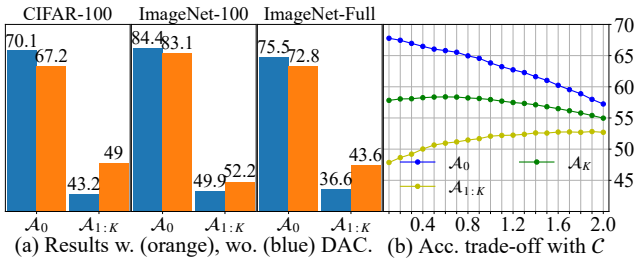
Comparison with State-of-the-arts

We compared with both EFCIL and reply-based methods as demonstrated in Table 1. The upper panel shows the average accuracy $\bar{\mathcal{A}}$ and the lower panel shows the last-phase accuracy $\mathcal{A}_K$. The activation function chosen in DS-AL is *Tanh*. The compensation ratios $\mathcal{C} = 0.6, 0.8, 1.4$ on CIFAR-100, ImageNet-100 and ImageNet-Full respectively.

Compare with EFCIL Methods. As shown in the upper panel in Table 1, the DS-AL delivers the most competitive results among EFCIL methods. On CIFAR-100, it slightly outperforms the previous best technique (i.e., ACIL) by 0.48%, 0.65%, 0.67% and 0.50% of $\mathcal{A}_K$ for different phase settings. In particular, the performance of AL-based CIL (e.g., DS-AL and ACIL) remains roughly the same as K changes, while other techniques receive degrading performance as K increases. This allows our DS-AL to excel more significantly for a growing K. The DS-AL gives a similar performance on ImageNet-100.

On ImageNet-Full, the $\bar{\mathcal{A}}$ achieves 67.18%, 66.91%, 66.81% and 66.79%, overtaking the previous best EFCIL by **1.08**%, **1.91**%, **2.18**% and **2.44**% respectively. The DS-AL's last-phase accuracy yields 58.17%, 58.13%, 58.10% and 58.15%, leads the ACIL by **2.1**%-**2.7**% among various K scenarios. The DS-AL has been shown to produce a more significant improvement on ImageNet-Full ($\approx 2\%$ v.s. $\approx 0.5\%$). This is because CIL tasks on ImageNet-Full are more challenging, requesting extra ability of fitting. Although the DS-AL and the ACIL belong to the same AL-based CIL family, the ACIL suffers from under-fitting on large-scale datasets such as ImageNet-Full.

Compare with Replay-based Methods. Replay-based methods have access to historical samples during the CIL procedure, allowing an easier way to address catastrophic forgetting. As shown in Table 1, the DS-AL runs behind replay-based counterparts for small-phase scenarios (e.g., $K = 5$). For instance, the DS-AL gives 67.18%, worse than the 69.21% from the RMM on ImageNet-Full. However, advantages of replaying samples are consumed as K increases. For instance, for $K = 25$, the DS-AL leads the best replaying method by **2.88**% (66.81% v.s. 63.93%) averagely, and by **2.60**% (58.10% v.s. 55.50%) at the last phase. This pattern on ImageNet-Full is consistent with those on CIFAR-100 and ImageNet-100. In general, the DS-AL begins to outperform replay-based methods from $K \geq 25$, except on ImageNet-100 where the DS-AL falls behind the RMM at $K = 25$, but the $\bar{\mathcal{A}}$ gap is very small (74.93% v.s. 75.01%).



(a) Results w. (orange), wo. (blue) DAC. (b) Acc. trade-off with $\mathcal{C}$

Figure 3: (a) Stability-plasticity changes via the DAC module. (b) Analysis of compensation ratio $\mathcal{C}$ on CIFAR-100.

Analysis on Compensation Stream

The compensation stream utilizes the DAC module to improve the main stream's fitting ability. To help understand this, we give specific examples (i.e., 5-phase experiments on ImageNet-100) during the CIL training. We plot the top-5 predictions with before and after the DAC module's contribution. As shown in the upper part of Figure 2, the prediction for the sample class is inaccurate during training in the main stream. After applying the DAC module, the compensation stream provides an extra gain, thereby correcting the predicted results. In this example, we can observe a significant prediction change, suggesting a non-trivial enhancement on DS-AL's fitting ability, mitigating its limitation as an AL-based CIL technique. Such a fitting improvement also benefits the generalization power. As indicated in the lower panel of Figure 2, during inference, the compensation stream can in fact correct the wrong prediction into an accurate one, especially when the top-2 predictions are comparable.

Compensation Stream Enhances the Plasticity. Balancing the stability (old knowledge) and plasticity (new knowledge) is crucial in CIL. Conventional AL-based CIL such as ACIL prioritizes stability as the backbone is trained completely on the base dataset. Through the compensation stream, our DS-AL can achieve a more reasonable stability-plasticity balance. To show this, we report the $\mathcal{A}_K$ on the base dataset (i.e., the first half) and the CIL dataset (i.e., the other half) respectively as shown in Figure 3(a). By

introducing the DAC module, novel classes learned incrementally receive a significant improvement (plasticity) that overtakes the performance decline on the base dataset (stability). For instance, the DAC module improves the novel class accuracy by **6.85**% (36.61%→43.56%) while only costing the base class accuracy to be reduced by 2.39% (75.48%→72.79%). This allows an improved overall accuracy, demonstrating that the DAC module is a beneficial adjustment by tuning the stability-plasticity balance.

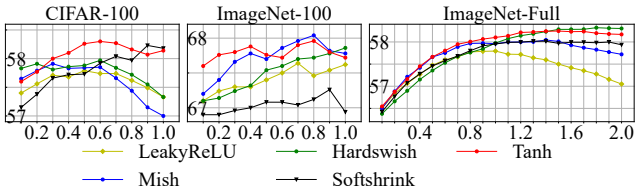


Figure 4: Last-phase accuracy of DS-AL with different activation functions and compensation ratio.

Hyperparameter Analysis

Activation function types and compensation ratio $\mathcal{C}$ influence the DS-AL's performance. Here we analyze these parameters in detail (with 5-phase setting). We plot the $\mathcal{A}_K$ with various activation functions under the scenarios of $\mathcal{C} = \{0.1, 0.2, \dots 1.0\}$ in Figure 4 and specifically extend the $\mathcal{C}$ up to 2.0 on ImageNet-Full.

Activation Function. As shown in Figure 4, the performance with all activation functions changes with $\mathcal{C}$. Among all candidates, we discover that the *Tanh* function delivers constantly good performance across all three datasets while other functions fluctuate. Although it fails behind *Mish* on ImageNet-100 and *Hardwish* on ImageNet-Full, the gap is relatively negligible. This observation can be explained by the fact that the *Tanh* function produces embedding values with a distribution vastly different from the main stream one (i.e., produced by *ReLU*), which helps improve the fitting task to the null space of the main stream. Most commonly-seen activation functions resemble the *ReLU* function curve and therefore lead to quite similar data distribution.

Compensation Ratio. As shown in Figure 4, all results of different activation functions evolve when $\mathcal{C}$ changes. On CIFAR-100, results for most activation functions peak at the middle (e.g., $0.2 < \mathcal{C} < 0.8$). For example, the *Tanh* reaches the top at around $\mathcal{C} = 0.6$ and falls after. This is because the model needs appropriate compensation to enhance the fitting, while overcompensation may mislead the model. On ImageNet-100 and ImageNet-Full, similar patterns can be found but peak $\mathcal{C}$ values become higher in general (e.g. from $\mathcal{C} = 0.6$ on CIFAR-100 to $\mathcal{C} = 0.8$ and $\mathcal{C} = 1.4$ of *Tanh* on ImageNet-100 and ImageNet-Full). That may be caused by more challenging tasks, rendering the under-fitting issue more obvious, leading to an increase of compensation.

In addition, we explore $\mathcal{C}$'s effect on networks' stability and plasticity (see Figure 3(b)). The $\mathcal{A}_0$ is found to decrease while $\mathcal{A}_{1:K}$ acts oppositely as $\mathcal{C}$ evolves, showing that more compensation will enhance the plasticity and suppress the stability. There is a leverage point of $\mathcal{C}$ after which the suppression contributes more and lead to degradation on $\mathcal{A}_K$.

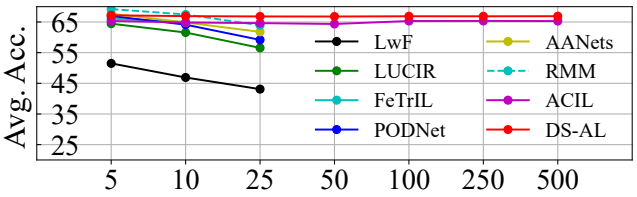


Figure 5: The evolution of $\bar{\mathcal{A}}$ with the growing K.

Large-phase Performance

We have theoretically demonstrated that the DS-AL achieves a phase-invariant property. To empirically support our claim, we include large-phase examples with up to $K = 100, 250, 500$ on ImageNet-Full (see Figure 5). We observe that the DS-AL receives an unchanged $\bar{\mathcal{A}}$ across various large-phase scenarios even under the extreme case of $K = 500$. As the ACIL and DS-AL both belong to the AL-based CIL, they share the same invariant property, with our method delivering a better performance. Other methods (including replay-based ones), however, lead to declining patterns as K increases from 5 to 25. If K continues to increase, a further performance reduction is expected.

Ablation Study

Here we conduct an ablation study to justify the contributions of DAC and PLC. Experiments are done on ImageNet-Full with activation function *Tanh* under the 5-phase setting. As shown in Table 2, performance with only the C-RLS is already competitive. However, with the DAC, the performance can be further improved. This improvement comes from the fitting and generalization enhancement from the compensation. With extra PLC that reduce the unnecessary compensation from previous classes, DS-AL can have even better performance.

Modules	$\mathcal{A}_K$ (%)	$\bar{\mathcal{A}}$ (%)
Concatenated-Recursive Least Squares (C-RLS)	56.11	65.34
+ Dual-Activation Compensation (DAC)	57.43	66.71
+ Previous Label Cleansing (PLC)	**58.17**	**67.18**

Table 2: Ablation study of the DAC ($\mathcal{C} = 1.4$) and PLC.

Conclusion

In this paper, we propose a Dual-Stream Analytic Learning (DS-AL) to tackle the challenging EFCIL problem. The DS-AL comprises a main stream that formulates the CIL problem as a C-RLS solution, and a compensation stream with a DAC module to mitigate the under-fitting nature of linearity in the main stream via projecting an alternatively activated embedding onto the main stream's null space. The DS-AL establishes equivalence between CIL and its joint-learning counterpart while improving fitting power as an AL-based method. Experimental results demonstrate comparable performance to CIL across different phase counts. Introducing the compensation stream consistently enhances both fitting and generalization abilities.

Acknowledgements

This research was supported by the National Natural Science Foundation of China (Grant No. 6230070401), 2023 South China University of Technology-TCL Technology Innovation Fund, and Guangzhou Basic and Applied Basic Research Foundation (2023A04J1687).

References

Belouadah, E.; Popescu, A.; and Kanellos, I. 2021. A comprehensive study of class incremental learning algorithms for visual tasks. *Neural Networks*, 135: 38–54.

Castro, F. M.; Marin-Jimenez, M. J.; Guil, N.; Schmid, C.; and Alahari, K. 2018. End-to-End Incremental Learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Douillard, A.; Cord, M.; Ollion, C.; Robert, T.; and Valle, E. 2020. 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*, 86–102. Springer.

Guo, P.; Lyu, M. R.; and Mastorakis, N. 2001. Pseudoinverse learning algorithm for feedforward neural networks. *Advances in Neural Networks and Applications*, 321–326.

Hayes, M. H. 1996. *Statistical digital signal processing and modeling*. John Wiley & Sons.

Hou, S.; Pan, X.; Loy, C. C.; Wang, Z.; and Lin, D. 2019. Learning a Unified Classifier Incrementally via Rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Jung, H.; Ju, J.; Jung, M.; and Kim, J. 2016. Less-forgetting learning in deep neural networks. *arXiv preprint arXiv:1607.00122*.

Kirkpatrick, J.; Pascanu, R.; Rabinowitz, N.; Veness, J.; Desjardins, G.; Rusu, A. A.; Milan, K.; Quan, J.; Ramalho, T.; Grabska-Barwinska, A.; et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526.

Li, Z.; and Hoiem, D. 2018. Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12): 2935–2947.

Liu, X.; Masana, M.; Herranz, L.; Van de Weijer, J.; López, A. M.; and Bagdanov, A. D. 2018. Rotate your Networks: Better Weight Consolidation and Less Catastrophic Forgetting. In *2018 24th International Conference on Pattern Recognition (ICPR)*, 2262–2268.

Liu, Y.; Li, Y.; Schiele, B.; and Sun, Q. 2023. Online Hyperparameter Optimization for Class-Incremental Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(7): 8906–8913.

Liu, Y.; Schiele, B.; and Sun, Q. 2021a. Adaptive Aggregation Networks for Class-Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2544–2553.

Liu, Y.; Schiele, B.; and Sun, Q. 2021b. RMM: Reinforced Memory Management for Class-Incremental Learning. *Advances in Neural Information Processing Systems*, 34.

Liu, Y.; Su, Y.; Liu, A.-A.; Schiele, B.; and Sun, Q. 2020. Mnemonics Training: Multi-Class Incremental Learning Without Forgetting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ma, C.; Ji, Z.; Huang, Z.; Shen, Y.; Gao, M.; and Xu, J. 2023. Progressive Voronoi Diagram Subdivision Enables Accurate Data-free Class-Incremental Learning. In *The Eleventh International Conference on Learning Representations*.

Petit, G.; Popescu, A.; Schindler, H.; Picard, D.; and Delezoide, B. 2023. FeTrIL: Feature Translation for Exemplar-Free Class-Incremental Learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 3911–3920.

Rebuffi, S.-A.; Kolesnikov, A.; Sperl, G.; and Lampert, C. H. 2017. iCaRL: Incremental Classifier and Representation Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wang, F.-Y.; Zhou, D.-W.; Ye, H.-J.; and Zhan, D.-C. 2022. FOSTER: Feature Boosting and Compression for Class-Incremental Learning. In Avidan, S.; Brostow, G.; Cissé, M.; Farinella, G. M.; and Hassner, T., eds., *Computer Vision – ECCV 2022*, 398–414. Cham: Springer Nature Switzerland. ISBN 978-3-031-19806-9.

Wu, Y.; Chen, Y.; Wang, L.; Ye, Y.; Liu, Z.; Guo, Y.; and Fu, Y. 2019. Large Scale Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhu, F.; Cheng, Z.; Zhang, X.-y.; and Liu, C.-l. 2021a. Class-Incremental Learning via Dual Augmentation. In Ranzato, M.; Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*, volume 34, 14306–14318. Curran Associates, Inc.

Zhu, F.; Zhang, X.-Y.; Wang, C.; Yin, F.; and Liu, C.-L. 2021b. Prototype Augmentation and Self-Supervision for Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5871–5880.

Zhu, K.; Zhai, W.; Cao, Y.; Luo, J.; and Zha, Z.-J. 2022. Self-Sustaining Representation Expansion for Non-Exemplar Class-Incremental Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9296–9305.

Zhuang, H.; Lin, Z.; and Toh, K.-A. 2021. Blockwise Recursive Moore-Penrose Inverse for Network Learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 1–14.

Zhuang, H.; Weng, Z.; He, R.; Lin, Z.; and Zeng, Z. 2023. GKEAL: Gaussian Kernel Embedded Analytic Learning for Few-Shot Class Incremental Task. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7746–7755.

Zhuang, H.; Weng, Z.; Wei, H.; XIE, R.; Toh, K.-A.; and Lin, Z. 2022. ACIL: Analytic Class-Incremental Learning with Absolute Memorization and Privacy Protection. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho,

K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 11602–11614. Curran Associates, Inc.