Knowledge Memorization and Rumination for Pre-trained Model-based Class-Incremental Learning

Zijian Gao[1,2]†, Wangwang Jia[1,2]†, Xingxing Zhang[3], Dulan Zhou[1,2],
Kele Xu[1,2]*, Feng Dawei[1,2], Yong Dou[1], Xinjun Mao[1,2], Huaimin Wang[1,2]
[1]College of Computer Science and Technology, National University of Defense Technology
[2]State Key Laboratory of Complex & Critical Software Environment
[3]School of Computer Science, Tsinghua University
{gaozijian19, wangwangjia, dulan_zhou, xukelele, yongdou, xjmao, hmwang}@nudt.edu.cn,
xxzhang1993@gmail.com, davyfeng.c@qq.com

Abstract

*Class-Incremental Learning (CIL) enables models to continuously learn new classes while mitigating catastrophic forgetting. Recently, Pre-Trained Models (PTMs) have greatly enhanced CIL performance, even when fine-tuning is limited to the first task. This advantage is particularly beneficial for CIL methods that freeze the feature extractor after first-task fine-tuning, such as analytic learning-based approaches using a least squares solution-based classification head to acquire knowledge recursively. In this work, we revisit the analytical learning approach combined with PTMs and identify its limitations in adapting to new classes, leading to sub-optimal performance. To address this, we propose the **Mo**mentum-based **A**nalytical **L**earning (**MoAL**) approach. MoAL achieves robust knowledge memorization via an analytical classification head and improves adaptivity to new classes through momentum-based adapter weight interpolation, leading to forgetting outdated knowledge. Importantly, we introduce a knowledge rumination mechanism that leverages refined adaptivity, allowing the model to revisit and reinforce old knowledge, thereby improving performance on old classes. MoAL facilitates the acquisition of new knowledge and consolidates old knowledge, achieving a win-win outcome between plasticity and stability. Extensive experiments on various incremental settings show MoAL's state-of-the-art performance*[1].

1. Introduction

In conventional deep learning, models are trained on a fixed dataset where all data is available upfront. In contrast, real-world scenarios often involve data arriving sequentially with new classes, necessitating a continual learning process—referred to as Class-Incremental Learning (CIL) [20]. The primary objective of CIL is to address catastrophic forgetting [9, 21, 22]—a sharp decline in performance on earlier tasks following sequential learning—thus preserving stability. CIL demands that models balance stability and plasticity to retain prior knowledge while continuously integrating new information [40].

Recently, Pre-Trained Models (PTMs) have shown state-of-the-art results across a broad range of applications [24]. Their strong generalization capability also has led to significant advancements in CIL, outperforming methods that do not leverage PTMs [6, 19, 34, 42, 43, 52, 53]. Empirical evidence from existing works [52, 54] suggests that PTMs' inherent representations can remain fixed throughout incremental learning, with prototype-based classification heads often surpassing the performance of prompt-based approaches [32, 34, 42, 43]. As a result, many methods either limit PTM fine-tuning to the first task [52] or design careful continual fine-tuning strategies [6, 53], balancing the pre-trained representations' generalization with the adaptivity required for learning new classes.

Naturally, traditional CIL approaches, especially those leveraging frozen feature extractors [26, 59, 60, 62], can benefit significantly from the strong generalizability of PTMs. Among these, analytical learning-based methods stand out, utilizing a fixed feature extractor after first-task fine-tuning and relying on the classification head to recursively acquire knowledge of each sample [59]. While they effectively mitigate representation drift and accommodate new classes through updates of the classification head alone, we find that they often show inadequate adaptivity to novel classes (see Section 3.3.2), resulting in sub-optimal performance. Achieving a balance between stability and plasticity

*Corresponding author. † Equal Contribution.
[1]Link to code-https://github.com/Zi-Jian-Gao/Moal.

remains a fundamental challenge in CIL, driving renewed interest in developing mechanisms to improve PTM adaptivity while preserving stability.

To address this challenge, we propose a **Mo**mentum-based **A**nalytic **L**earning (**MoAL**) approach that continually enhances adaptivity across incremental tasks while balancing stability and plasticity. MoAL achieves robust knowledge memorization through an analytical classification head, effectively capturing task-specific knowledge. To enhance adaptivity for new tasks, MoAL employs momentum-based adapter weight interpolation, which facilitates efficient adaptation and helps discard outdated information caused by the previous model's limited adaptivity. Additionally, MoAL incorporates a knowledge rumination mechanism to revisit and reinforce old task knowledge in a more fine-grained manner.

In nature, ruminant animals rapidly consume forage with minimal chewing, followed by a period of soaking and softening in the rumen [44]. The forage is then regurgitated for thorough re-chewing, allowing efficient nutrient extraction. Inspired by this process, we conceptualize model adaptivity similarly: continuous adaptive learning refines representations incrementally, akin to the rumen's soaking phase. Building on this analogy, our knowledge rumination mechanism leverages prototype-based fine-grained feature generation to systematically revisit and reinforce old knowledge, aligning retained information more precisely with the model's evolving representations.

Building on continually adaptive training, MoAL incorporates two key mechanisms—Knowledge Memorization and Knowledge Rumination—to achieve a win-win outcome between plasticity and stability. Key contributions include: (1) Identifying the often-overlooked sub-optimality (i.e., limited adaptivity) in PTM-based analytical learning methods and addressing this limitation with MoAL; (2) Developing an adapter-based training paradigm paired with a knowledge memorization mechanism to effectively balance adaptivity and generalization while recursively capturing new task knowledge; (3) Introducing a knowledge rumination mechanism that revisits and reinforces old knowledge without relying on instances from previous tasks; (4) Demonstrating through extensive experiments that MoAL significantly outperforms existing methods across various settings and even surpasses replay-based approaches.

2. Related Work

2.1. Class-Incremental Learning

Class-Incremental Learning (CIL) aims to continuously learn new classes while retaining knowledge of previously learned ones [54]. Traditional CIL methods are generally classified into replay-based [27, 36, 47, 50, 51], replay-free [31, 55–57], regularization-based [1, 8, 15, 18, 29, 41, 48], and analytical learning-based approaches [59, 60, 62]. Replay-based methods selectively store or generate past samples for incorporation in current training, while replay-free methods avoid storing instances, instead retaining prototypes to overcome forgetting. Regularization-based approaches use techniques like knowledge distillation [13, 18] to maintain key model parameters. ACIL [59] is the first analytic continual learning method with a closed-form solution. It treats CIL as a recursive process to achieve absolute memory thereby avoiding forgetting caused by backpropagation. It inspires a new CIL branch, with new intakes such as GKEAL [60], GACL [61], RAIL [46].

2.2. Pre-Trained Model-Based CIL

Unlike the aforementioned CIL methods that typically train models from scratch, pre-trained model (PTM)-based CIL methods have gained considerable attention in recent years due to their strong generalizability. To minimize training costs, these methods often employ parameter-efficient tuning techniques, mainly including prompts [17] and adapters [4]. Prompt-based approaches [34, 38, 42, 43] focus on designing architectures that adaptively generate task-specific prompts. In contrast, LAE [6] introduces a unified framework that integrates predictions from online and offline modules based on parameter-efficient tuning. Recently, EASE [53] proposed training task-specific adapters and ensembling them without conflicts, achieving state-of-the-art performance. Additionally, Aper [52] demonstrates that a prototype-based classification head performs strongly even without model fine-tuning, motivating us to revisit analytical learning-based methods [59] using PTMs.

3. Preliminary Analysis

3.1. Problem Definition

We first introduce the background of CIL with pre-trained models. Consider a classification model $M_\theta(\cdot) = h(f(\cdot))$ with trainable parameters $\theta = \{\theta_{\text{fe}}, \theta_{\text{cls}}\}$, where $f(\cdot)$ represents the feature extractor and $h(\cdot)$ denotes the classification head. The feature extractor f_{fe} is initialized with parameters θ_0 from a PTM and subsequently trained sequentially on a series of incremental tasks. We denote the training dataset at task t ($t = 1, 2, \ldots, T$) by $\mathcal{D}_t = \{\boldsymbol{X}_t, \boldsymbol{Y}_t\}$. Here, $\boldsymbol{X}_t \in \mathbb{R}^{N_t \times c \times w \times h}$ represents N_t training samples, each with dimensions $c \times w \times h$, and $\boldsymbol{Y}_t \in \mathbb{R}^{N_t \times d_{y_t}}$ corresponds to N_t one-hot labels, where each label reflects one of the d_{y_t} classes in task t. The tasks satisfy the non-overlapping class condition, such that $\boldsymbol{Y}_t \cap \boldsymbol{Y}_{t'} = \varnothing$ for $t \neq t'$. In this paper, we adopt the replay-free setting [31, 55–57], where no instances from previous tasks are retained. The goal of M_θ is to achieve performance comparable to joint learning after sequential training on a total of T tasks.

3.2. Adapter-Based Parameter-Efficient Tuning

In the era of PTMs, to preserve pre-trained knowledge and reduce training costs, existing studies [6, 34, 42, 43, 52, 53] commonly freeze the pre-trained weights and introduce an additional trainable module. The adapter [4, 14] is a bottleneck structure that can be incorporated into a pre-trained model to improve performance on downstream tasks. The adapter consists of a down-projection layer $W_{\text{down}} \in \mathbb{R}^{d\times r}$ that reduces the feature dimensionality, followed by a non-linear activation function σ, and an up-projection layer $W_{\text{up}} \in \mathbb{R}^{r\times d}$ to restore the original dimensionality. Given an input x, the output of the adapter is formalized as:

$$x + \sigma(x * W_{\text{down}}) * W_{\text{up}}, \tag{1}$$

where σ represents the activation function (e.g., ReLU) and $*$ denotes matrix multiplication. In cases where the pre-trained weights are frozen, only the parameters of the adapter ($\theta_{W_{\text{down}}} \cup \theta_{W_{\text{up}}} \cup \theta_{\text{cls}}$) and the classification head are optimized, allowing for efficient adaptation with minimal parameter updates. Compared to prompt tuning, adapter tuning has shown greater effectiveness and superior anti-forgetting capacity in CIL [6, 52, 53]. Thus, we adopt the adapter-based tuning used in Aper [52] and EASE [53].

3.3. Analytical Learning-Based Sub-Optimality

3.3.1. Analytical Learning-Based CIL

Here, we introduce the foundational principles of analytical learning, excluding considerations specific to CIL. Analytic learning [10], also called pseudoinverse learning, was developed to overcome limitations of backpropagation [45], such as gradient vanishing/exploding and the need for long training epochs. It first extracts sample features via a feature extractor and then applies a feature transformation in the initial buffer layer. Using the least squares solution, it subsequently learns the classifier parameters in the second layer with shallow networks [2] in only a single epoch [58].

Analytical learning operates through a two-layer analytic classification head, consisting of a random linear buffer layer f_{B} that expands the feature space to d_B dimensions, followed by a linear layer f_{L} that aligns features with labels. Using randomization techniques, analytical learning effectively captures valuable information [10, 58]. During learning, inputs $\boldsymbol{X}$ are passed through the feature extractor f_{fe} to obtain features $\boldsymbol{X}^{\text{fe}}$, which are then projected by f_{B} to produce the expanded features $\boldsymbol{X}^{\text{B}}$, as formulated below:

$$\boldsymbol{X}^{\text{B}} = \sigma(f_{\text{B}}(f_{\text{flat}}(\boldsymbol{X}^{\text{fe}}))), \tag{2}$$

where f_{flat} denotes the flattening operator and σ represents the activation function, with ReLU applied as recommended in [59, 62]. The second layer, f_{L}, with parameters $\boldsymbol{W}^{\text{L}}$, aligns the expanded features $\boldsymbol{X}^{\text{B}}$ with labels $\boldsymbol{Y}$ by solving

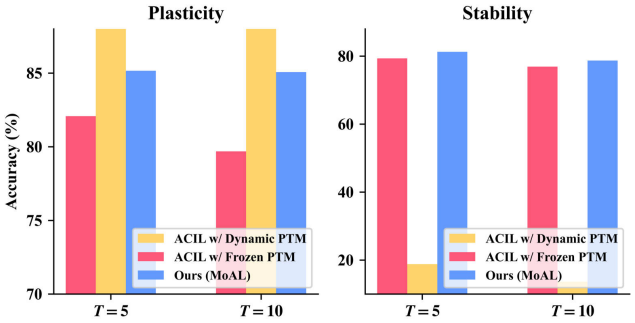

Figure 1. Plasticity and Stability Comparison.

the following minimization problem:

$$\underset{\boldsymbol{W}^{\text{L}}}{\operatorname{argmin}} \left\|\boldsymbol{Y} - \boldsymbol{X}^{\text{B}}\boldsymbol{W}^{\text{L}}\right\|_F^2 + \gamma \left\|\boldsymbol{W}^{\text{L}}\right\|_F^2, \tag{3}$$

where $\|\cdot\|_F$ is the Frobenius norm and γ is a regularization hyperparameter. The optimal solution, obtained through least squares estimation, is given by:

$$\hat{\boldsymbol{W}}^{\text{L}} = \left(\boldsymbol{X}^{\text{B}\,\text{T}}\boldsymbol{X}^{\text{B}} + \gamma \boldsymbol{I}\right)^{-1} \boldsymbol{X}^{\text{B}\,\text{T}}\boldsymbol{Y}, \tag{4}$$

in which $\cdot^{\text{T}}$ denotes the matrix transpose. With the optimal parameters $\hat{\boldsymbol{W}}^{\text{L}}$ determined, the analytical learning framework is established, demonstrating that its upper performance bound depends on the quality of the feature extractor: the better the feature extractor, the higher the achievable performance [10, 58].

Building on this foundation, analytical learning-based CIL methods [59, 60, 62] use a frozen feature extractor f_{fe} that undergoes supervised adaptation via backpropagation on the first task and then remains frozen for subsequent tasks, allowing the analytic classification head to be recursively updated and to consistently achieve robust performance across various settings. Due to their strong generalization capabilities, PTMs serve as ideal feature extractors for analytical learning-based methods. Thus, we revisit the analytical learning-based CIL method (ACIL [59]) utilizing PTMs to explore its potential.

3.3.2. Overlooked Sub-Optimality

To investigate the latent sub-optimality caused by a frozen feature extractor, we conducted comprehensive experiments using the challenging ImageNet-R [11] and ImageNet-A [12] datasets, with the ViT-B/16-IN21K [5] model pre-trained on ImageNet-21K [30]. Adapter tuning was applied, and the model was incrementally trained on an equal number of new classes across a total of 5 tasks. Our evaluation compared the plasticity (average accuracy on new classes) and stability (average accuracy on old classes) of three approaches: ACIL with a frozen PTM after initial task fine-tuning, ACIL with a dynamic PTM (consisting of a frozen PTM and a trainable adapter) incrementally fine-tuned on new tasks, and our proposed MoAL.

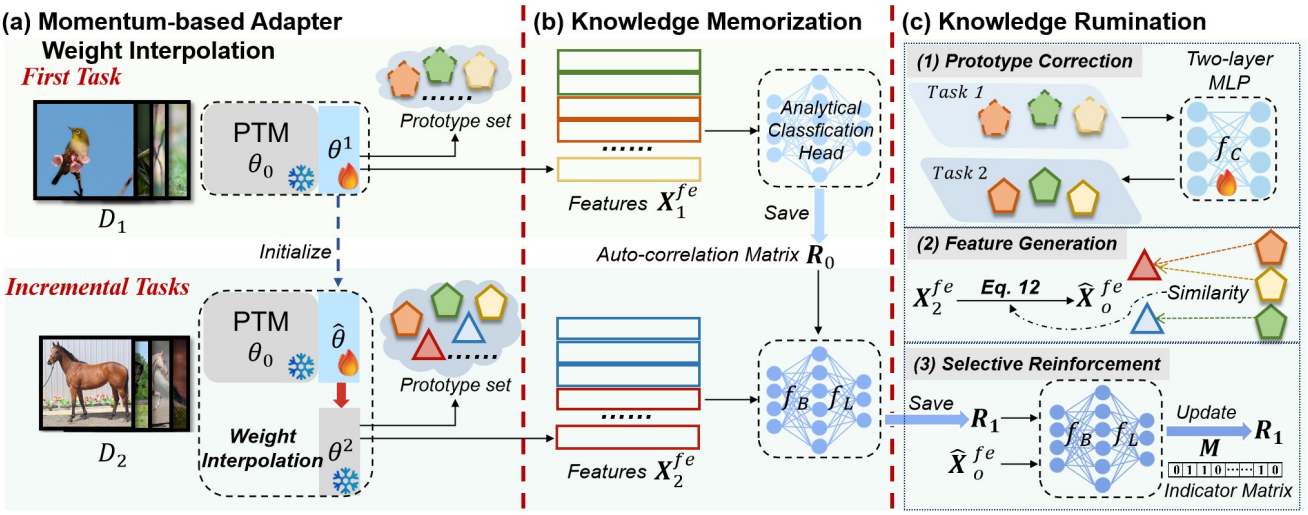

Figure 2. Illustration of MoAL. **(a)** In the first task, we train an adapter to memorize new knowledge using the analytical classification head and store prototypes post-training. For incremental tasks, momentum-based adapter weight interpolation is employed to enhance adaptability while preserving generalizability. **(b)** The knowledge memorization mechanism is executed by storing the auto-correlation matrix and recursively updating the weights. **(c)** Following knowledge memorization, the rumination mechanism corrects old prototypes and generates fine-grained features for old classes, selectively reinforcing old knowledge that has not been fully absorbed by the model.

As shown in Figure 1, ACIL with a frozen feature extractor demonstrates limited adaptability to new tasks, while both our method and the unrestricted fine-tuning approach significantly improve plasticity, underscoring the benefits of continuously enhancing PTM adaptivity. Notably, our method not only increase plasticity but also improve stability, thanks to our knowledge rumination mechanism. To conclude, freezing the feature extractor in ACIL trades adaptability for stability, leading to marked performance declines on datasets that diverge from the pre-training distribution. Conversely, unrestricted PTM fine-tuning increases plasticity but undermines the recursive manner and stability. Our approach, however, achieves a win-win outcome in learning and anti-forgetting.

4. Methodology

The success of MoAL can be attributed to three key components: a lightweight, momentum-based adapter weight interpolation mechanism that enhances model adaptability while preserving generalizability; a knowledge memorization mechanism that recursively memorizes new task knowledge; and a knowledge rumination mechanism that revisits and reinforces old task knowledge by leveraging improved model adaptability and refined representations. Figure 2 illustrates the overall framework of our method.

4.1. Momentum-based Adapter Weight Interpolation

In our work, we use the adapter [4] to fine-tune the PTM due to its parameter efficiency and performance superiority [52, 53]. Formally, we define the parameters of the PTM f_θ as $\theta = \theta_0 + \theta_{\text{adapter}}$, where θ_0 denotes the initialized parameters from a PTM, and θ_{adapter} represents the learnable parameters of the adapter, comprising down-projection low-rank parameters $\theta_{W_{\text{down}}}$ and up-projection low-rank parameters $\theta_{W_{\text{up}}}$. Previous studies have shown that maintaining generalization while enhancing adaptivity is essential for PTM-based CIL [52] to learn new tasks effectively while retaining knowledge from previous tasks.

To facilitate continual adaptation to new tasks while preserving the PTM's inherent generalizability, we introduce a momentum-based adapter weight interpolation mechanism (see Figure 2 (a)) that efficiently retains prior knowledge while integrating new information. Specifically, we assume the adapter parameters θ_{adapter} are initially updated with the fully-connected classification head $h(\cdot)$ using cross-entropy loss L_{CE} during the first task, where continual adaptation is not yet involved. During subsequent tasks, at the end of each epoch, we iteratively update the adapter's weights through weight-space interpolation:

$$\theta_{\text{adapter}} = \alpha\theta_{\text{adapter}} + (1-\alpha)\hat{\theta}_{\text{adapter}}, \tag{5}$$

where the momentum hyperparameter $\alpha \in [0, 1]$ controls the adapter updates, with sensitivity illustrated in Table 3. Here, $\hat{\theta}_{\text{adapter}}$ is initialized with θ_{adapter} at the beginning of each task and optimized using the classification head $h(\cdot)$ and cross-entropy loss via backpropagation.

Our method offers several advantages over existing approaches: (1) Parameter and Resource Efficiency: Training involves only a single adapter, greatly reducing the number of trainable parameters and the computational load than those use multiple adapters or models. (2) Inference Efficiency: In the inference stage, our method retains only

a single adapter, minimizing storage requirements and enabling faster inference with a single forward pass. (3) Enhanced Performance: Unlike knowledge distillation-based methods [13, 18], which depend on regularization losses, our approach eliminates the need for additional regularization. This leads to superior learning capabilities, effectively consolidating old knowledge through our knowledge rumination mechanism (see Table 2 for comparisons).

4.2. Knowledge Memorization

After enhancing the adaptivity of PTM for each task, an effective mechanism is required to learn and memorize knowledge incrementally. To achieve this, our knowledge memorization mechanism (Figure 2 (b)) employs an efficient, one-pass learning approach that computes the analytical classification head weights and auto-correlation matrix through a straightforward recursive least squares solution, enabling task memorization during the incremental task without the resource demands of conventional backpropagation-based learning.

Specifically, following the classical analytical learning process and the notations in Section 3.3.1, we define the PTM f_θ as the feature extractor f_{fe} and the analytical classification head, which consists of a buffer layer f_{B} and a second linear layer f_{L}. After training on the first task without considering continual learning, we obtain the optimal parameters $\hat{\boldsymbol{W}}_1^{\text{L}} = \left(\boldsymbol{X}_1^{\text{B}\,\text{T}}\boldsymbol{X}_1^{\text{B}} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{X}_1^{\text{B}\,\text{T}}\boldsymbol{Y}_1$ for the linear layer f_{L} based on Eq. 4. These parameters map the expanded features $\boldsymbol{X}_1^{\text{B}}$ to the labels $\boldsymbol{Y}_1$ and are calculated using the training samples $\boldsymbol{X}_1$ from the first task.

In incremental tasks, at task t, the goal of the knowledge memorization mechanism is to incorporate the existing mapping relationships for old task samples $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{t-1}$ along with the new mapping relationship for the current task samples $\boldsymbol{X}_t$. Assuming that all task samples are available simultaneously, we can get an equivalent target to our recursive solution:

$$\underset{\boldsymbol{W}_t^{\text{L}}}{\operatorname{argmin}} \left\|\boldsymbol{Y}_{1:t} - \boldsymbol{X}_{1:t}^{\text{B}}\boldsymbol{W}_t^{\text{L}}\right\|_F^2 + \gamma\left\|\boldsymbol{W}_t^{\text{L}}\right\|_F^2, \tag{6}$$

where

$$\boldsymbol{X}_{1:t}^{\text{B}} = \begin{bmatrix} \boldsymbol{X}_1^{\text{B}} \\ \vdots \\ \boldsymbol{X}_t^{\text{B}} \end{bmatrix} \boldsymbol{Y}_{1:t} = \begin{bmatrix} \boldsymbol{Y}_1 & 0 & 0 & \ldots & 0 \\ 0 & \boldsymbol{Y}_2 & 0 & \ldots & 0 \\ \vdots & \vdots & & \ddots & \\ 0 & 0 & 0 & \ldots & \boldsymbol{Y}_t \end{bmatrix}. \tag{7}$$

The sparse structure of $\boldsymbol{Y}_{1:t}$ results from the mutually exclusive classes across tasks. Thus, according to Eq. 4, we can get the least squares solution:

$$\boldsymbol{W}_t^{\text{L}} = \left(\boldsymbol{X}_{1:t}^{\text{B}\,\text{T}}\boldsymbol{X}_{1:t}^{\text{B}} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{X}_{1:t}^{\text{B}\,\text{T}}\boldsymbol{Y}_{1:t}. \tag{8}$$

At the task t, we aim to get the identical recursive solution $\boldsymbol{W}_t^{\text{L}}$ based on the last task solution $\boldsymbol{W}_{t-1}^{\text{L}}$, current task samples $\boldsymbol{X}_t$ and labels $\boldsymbol{Y}_t$. The recursive solution can be summarized in the following theorem:

Theorem 1. *Let* $\boldsymbol{R}_t = \left(\boldsymbol{X}_{1:t}^{B\,\text{T}}\boldsymbol{X}_{1:t}^{B} + \gamma\boldsymbol{I}\right)^{-1}$ *be the auto-correlation matrix, the weights* $\boldsymbol{W}_t^L$ *are recursively obtained by*

$$\boldsymbol{W}_t^L = \boldsymbol{W}_{t-1}^L + \boldsymbol{R}_t\boldsymbol{X}_t^{B\,\text{T}}\left(\boldsymbol{Y}_t - \boldsymbol{X}_t^B\boldsymbol{W}_{t-1}^L\right) \tag{9}$$

where

$$\boldsymbol{R}_t = \boldsymbol{R}_{t-1} - \boldsymbol{R}_{t-1}\boldsymbol{X}_t^{B\,\text{T}}\left(\boldsymbol{I} + \boldsymbol{X}_t^B\boldsymbol{R}_{t-1}\boldsymbol{X}_t^{B\,\text{T}}\right)^{-1}\boldsymbol{X}_t^B\boldsymbol{R}_{t-1}. \tag{10}$$

Proof. See the supplementary materials. □

Based on Theorem 1, we derive a recursive solution that does not require samples $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{t-1}$ from old tasks. However, as discussed in Section 3.3.2, this recursive solution depends on a frozen PTM, which limits adaptability and results in sub-optimal performance. To overcome this limitation, we enhance the adaptivity of the PTM f_θ through momentum-based adapter weight interpolation, simultaneously enabling forgetting to discard out-dated feature-to-label mappings (i.e., knowledge).

Forgetting has been shown to be essential in both biological systems [7, 28, 33] and continual learning [37, 39], as it enables selective removal of outdated information, aiding adaptation to new knowledge [3]. While this disrupts the recursive solution, it allows the model to discard obsolete knowledge, creating a solid foundation for our knowledge rumination mechanism to revisit and reinforce previous task knowledge with enhanced adaptivity, ultimately achieving better performance on old tasks.

4.3. Knowledge Rumination

The knowledge rumination mechanism (Figure 2 (c)) reproduces fine-grained knowledge of previous tasks with enhanced adaptivity without relying on old task data. To achieve this, we store prototypes (typically the class means in feature space) and implement a straightforward prototype correction network to adjust these prototypes over time. Additionally, we exploit the relationships between old and new class prototypes to generate refined features for old tasks. This selective process enables the model to revisit and reinforce knowledge that was inadequately learned, thus improving performance on previous tasks. This approach is inspired by biological rumination: after food is initially ingested and softened in the rumen, it is regurgitated for fine-grained re-chewing, allowing thorough nutrient breakdown and enhanced absorption.

Prototype correction: Prototypes are widely used in CIL to address forgetting, especially when previous task

samples [55, 56] are unavailable. For each task t, we compute the prototype p_k for samples $\boldsymbol{X}^k$ of a new class k using the PTM f_{θ^t}: $p_k^t = \text{mean}(f_{\theta^t}(\boldsymbol{X}^k))$. However, prototypes obtained by the PTM f_{θ^t} for task t will differ from those produced by its predecessor $f_{\theta^{t-1}}$ in task $t-1$, leading to inferior reliability when generating features for old classes. To mitigate this, we transform the old prototype p_k^{t-1} from the previous feature space to the current one, obtaining the corrected prototype p_k^t. Specifically, we save the adapter $\theta^{t-1}_{\text{adapter}}$ before training on task t, representing the PTM parameters for task $t-1$ as $\theta^{t-1} = \theta_0 + \theta^{t-1}_{\text{adapter}}$. Similarly, we denote the PTM parameters for task t as $\theta^t = \theta_0 + \theta^t_{\text{adapter}}$. To align old and new feature spaces, we introduce a prototype correction network f_{C}, implemented as a two-layer MLP, which utilizes $f_{\theta^{t-1}}$, f_{θ^t}, and the current task samples $\boldsymbol{X}_t$ to minimize the Euclidean distance between features extracted by $f_{\theta^{t-1}}$ and f_{θ^t}:

$$\mathcal{L}_{\text{PC}} = \left\| f_{\theta^t}(\boldsymbol{X}_t) - f_{\text{C}}(f_{\theta^{t-1}}(\boldsymbol{X}_t)) \right\|_2 . \tag{11}$$

In this way, after each task, we use the correction network f_{C} to obtain the corrected prototype $p_k^t = f_{\text{C}}(p_k^{(t-1)})$. All previously learned prototypes are updated with each new task, resulting in an updated prototype set $\{p_1^t, p_2^t, \ldots, p_k^t\}$ across all classes.

Fine-grained feature generation for old classes: Using the prototype set, we calculate the cosine similarity between the prototypes of new and old classes to generate fine-grained features for old classes. In task t, let i be an old class most similar to a new class j; the features of old class i can then be generated as follows:

$$\hat{\boldsymbol{X}}_i^{\text{fe}} = \boldsymbol{X}_j^{\text{fe}} + p_j^t - p_i^t. \tag{12}$$

where $\boldsymbol{X}_j^{\text{fe}} = f_{\theta^t}(\boldsymbol{X}^j)$. This equation translates the refined features of new classes into more precise representations of old classes, leveraging prototype distances. It provides a computationally efficient solution by relying solely on addition and subtraction operations, thus enabling the model to revisit knowledge from previous tasks.

Selective reinforcement of old task knowledge: For simplicity, we denote the fine-grained features of all old classes as $\hat{\boldsymbol{X}}_o^{\text{fe}}$. These features are then passed through the buffer layer f_{B} to produce expanded features $\hat{\boldsymbol{X}}_o^{\text{B}}$ (see Equation 2). To selectively reinforce the old task knowledge that the model has not fully learned, we introduce an indicator matrix $\boldsymbol{M}$, defined as follows:

$$\boldsymbol{M} = \mathbb{I}\left(f_{\text{L}}(\hat{\boldsymbol{X}}_o^{\text{B}}) \neq \boldsymbol{Y}_o \right), \tag{13}$$

where $\boldsymbol{Y}_o$ represents the corresponding labels and $\mathbb{I}(\cdot)$ is the indicator function, which takes the value 1 if the condition is true and 0 otherwise. The reinforcement process is formulated in the following theorem:

Theorem 2. *Given the weights $\boldsymbol{W}_t^L$ of the linear layer f_L, the auto-correlation matrix $\boldsymbol{R}_t$, the expanded features $\hat{\boldsymbol{X}}_o^B$, the corresponding labels $\boldsymbol{Y}_o$, and the indicator matrix $\boldsymbol{M}$, the weights $\boldsymbol{W}_t^L$ are updated as follows:*

$$\boldsymbol{W}_t^L = \boldsymbol{W}_t^L + \boldsymbol{R}_t \left(\boldsymbol{M} \odot \hat{\boldsymbol{X}}_o^{B^{\text{T}}} \right) \left(\boldsymbol{M} \odot \boldsymbol{Y}_o - \boldsymbol{M} \odot \hat{\boldsymbol{X}}_o^B \boldsymbol{W}_t^L \right) \tag{14}$$

where

$$\boldsymbol{R}_t = \boldsymbol{R}_t - \boldsymbol{R}_t \left(\boldsymbol{M} \odot \hat{\boldsymbol{X}}_o^{B^{\text{T}}} \right) \left(\boldsymbol{I} + \boldsymbol{M} \odot \hat{\boldsymbol{X}}_o^B \boldsymbol{R}_t \left(\boldsymbol{M} \odot \hat{\boldsymbol{X}}_o^{B^{\text{T}}} \right) \right)^{-1} \left(\boldsymbol{M} \odot \hat{\boldsymbol{X}}_o^B \right) \boldsymbol{R}_t. \tag{15}$$

By integrating enhanced adaptivity with the relational structure between old and new class feature spaces, our knowledge rumination mechanism effectively revisits and reinforces old knowledge. Although momentum-based adapter weight interpolation disrupts the strict recursion of analytical learning in CIL, the rumination mechanism mitigates forgetting and further enhances performance on old tasks by refining fine-grained knowledge. The pseudo-algorithm is included in the supplementary material.

5. Experiments

5.1. Experimental Setting

Datasets. To evaluate the performance of our method, we conduct experiments on four commonly used benchmarks: CIFAR-100 [16], ImageNet-A [12], ImageNet-R [11], and OmniBenchmark [49]. For CIFAR-100, ImageNet-A, and ImageNet-R, we use three incremental settings (5, 10, and 20 tasks), while for OmniBenchmark, we use settings with 5, 15, and 30 tasks. All classes are arranged in a fixed order and the model is trained on an equal number of classes in both the first task and each incremental task.

Implementation details. Our methods are implemented using PyTorch [25] and PILOT [35] on an NVIDIA 4090. For fair comparison, we utilize the widely adopted Vision Transformer (ViT) architecture, specifically ViT-B/16-IN21K [5], following the setups in [43, 52]. This architecture features a transformer block with an input dimension of 224 and an output dimension of 768. The momentum hyperparameter α is set to 0.999 and The dimension d_B of the buffer layer f_{B} is set to $5k$ for CIFAR-100, $15k$ for ImageNet-A, $20k$ for ImageNet-R, and $10k$ for OmniBenchmark. We use the SGD optimizer with a momentum of 0.9, and the learning rate starts at 0.01, decaying with cosine annealing. The batch size is set to 48, and the model is trained for 80 epochs (20 epochs for OmniBenchmark). Meanwhile, we employ ridge regression [23] to compute regularization hyperparameter γ in a self-adaptive manner. Following [27], we shuffle the class order using a random seed of 1993 for all methods.

Comparison methods. To establish the effectiveness of our approach, we select state-of-the-art PTM-

Table 1. Comparison of average incremental accuracy $\bar{A}(\%)$, last-task accuracy $A_T(\%)$ and the average performance among EFCIL methods with different T tasks on different datasets. The best results of existing methods are highlighted in blue font.

Metric	Methods	CIFAR-100				ImageNet-A				ImageNet-R				OmniBenchmark			
		$T=20$	10	5	*avg.*	$T=20$	10	5	*avg.*	$T=20$	10	5	*avg.*	$T=30$	15	10	*avg.*
$\bar{A}$	Finetune	84.40	87.52	90.51	87.48	46.06	50.45	57.23	51.25	71.14	74.94	78.15	74.74	67.70	72.05	73.30	71.02
	LwF (TPAMI 2018) [18]	76.19	82.88	88.10	82.39	46.03	50.36	56.75	51.05	70.74	74.31	77.00	74.02	67.42	72.31	73.91	71.21
	L2P (CVPR 2022) [43]	86.99	89.48	91.02	89.16	50.87	54.03	56.35	53.75	75.59	77.92	77.86	77.12	70.14	71.93	73.22	71.76
	DualPrompt (ECCV 2022) [42]	86.88	88.86	89.78	88.51	54.59	58.60	59.59	57.59	73.61	75.06	75.12	74.60	72.81	73.66	75.13	73.87
	ACIL (NeurIPS 2022) [59]	90.22	91.96	94.00	92.06	64.42	70.93	72.04	69.97	79.43	81.21	83.36	81.34	76.96	76.54	76.09	76.53
	CODA-Prompt (CVPR 2023) [34]	86.06	91.19	92.20	89.82	57.19	61.86	65.97	61.67	71.63	76.69	80.17	76.16	68.64	70.78	72.19	70.54
	LAE (ICCV 2023) [6]	80.96	86.97	88.50	85.47	50.30	58.56	59.09	55.98	72.85	75.42	75.48	74.58	71.00	73.82	73.65	72.82
	DS-AL (AAAI 2024) [62]	86.11	83.50	88.82	86.14	63.38	63.47	61.82	63.22	75.90	78.37	80.39	78.22	79.73	80.20	76.91	78.95
	SimpleCIL (IJCV 2024) [52]	82.79	82.31	81.12	82.07	60.05	59.33	58.09	59.16	67.60	67.09	65.89	66.86	79.46	79.23	78.51	79.07
	Aper (IJCV 2024) [52]	88.48	90.91	91.56	90.32	61.36	65.74	68.90	65.33	76.28	79.01	80.48	78.59	79.72	79.79	79.82	79.78
	EASE (CVPR 2024) [53]	90.62	92.01	92.81	91.81	60.62	62.93	67.93	63.83	78.15	81.33	82.25	80.58	73.09	75.32	81.11	76.51
	MoAL (Ours)	**93.27**	**94.22**	**94.03**	**93.84**	**67.26**	**74.29**	**75.22**	**72.26**	**82.94**	**84.45**	**85.39**	**84.26**	**84.04**	**85.68**	**84.45**	**84.72**
		+3.05	+2.26	+0.03	+1.78	+2.84	+3.36	+3.18	+2.29	+3.51	+3.24	+2.03	+2.92	+4.31	+5.48	+3.34	+5.77
A_T	Finetune	78.58	82.06	86.31	82.32	33.77	40.62	47.14	40.51	63.72	68.63	73.97	68.77	56.41	60.82	62.42	59.88
	LwF (TPAMI 2018) [18]	67.36	77.57	84.28	76.40	33.77	40.22	45.89	39.96	64.45	69.55	73.27	69.09	56.52	61.40	64.59	60.84
	L2P (CVPR 2022) [43]	81.22	84.47	86.27	83.99	42.40	45.49	48.52	45.47	68.73	72.25	73.73	71.57	59.85	62.32	63.84	62.00
	DualPrompt (ECCV 2022) [42]	79.90	84.23	84.76	82.96	43.38	47.93	49.18	46.83	67.12	69.10	70.37	68.86	62.84	62.91	65.60	63.78
	ACIL (NeurIPS 2022) [59]	88.79	90.33	90.73	89.95	51.02	60.90	62.54	59.65	75.55	77.38	78.90	77.28	73.28	66.52	77.06	72.29
	CODA-Prompt (CVPR 2023) [34]	79.55	87.24	88.67	85.15	46.15	51.02	56.35	51.17	67.93	73.10	76.40	72.48	64.61	67.64	68.84	67.03
	LAE (ICCV 2023) [6]	74.26	81.13	82.76	79.38	39.43	47.73	50.03	45.73	65.57	69.83	71.05	68.82	62.44	63.88	64.76	63.70
	DS-AL (AAAI 2024) [62]	85.90	86.05	85.91	85.95	51.74	52.67	51.02	51.81	74.05	77.48	76.55	76.03	73.40	73.52	72.95	73.29
	SimpleCIL (IJCV 2024) [52]	76.21	76.21	76.21	76.21	49.24	49.24	49.24	49.24	61.35	61.35	61.35	61.35	72.18	72.18	72.18	72.18
	Aper (IJCV 2024) [52]	82.75	85.81	87.58	85.38	50.49	55.69	59.91	55.36	69.25	72.05	74.95	72.08	72.51	72.95	73.25	72.90
	EASE (CVPR 2024) [53]	84.21	87.25	89.22	86.89	49.70	51.74	57.93	53.13	71.10	76.00	78.05	75.05	64.44	67.74	74.37	67.75
	MoAL (Ours)	**88.36**	**90.49**	**90.85**	**89.90**	**52.01**	**64.06**	**67.22**	**61.10**	**76.85**	**79.33**	**81.38**	**79.19**	**74.02**	**77.23**	**78.61**	**76.62**
		-0.43	+0.16	+0.12	-0.05	+0.27	+3.16	+4.68	+1.45	+1.30	+1.85	+2.48	+1.91	+0.62	+3.71	+1.55	+3.33

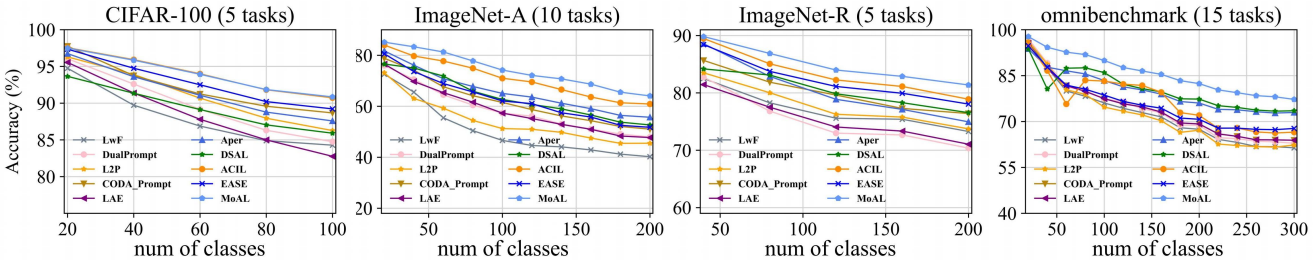

Figure 3. The average incremental accuracy $\bar{A}(\%)$ curves of various methods.

based CIL methods for comparative analysis, which include L2P [43], DualPrompt [42], ACIL [59], CODA-Prompt [34], LAE [6], DS-AL [62], SimpleCIL [52], Aper [52], and EASE [53]. Furthermore, we also benchmark our approach against typical CIL methods by equipping them with the same PTM, such as LwF [18], iCaRL [27], DER [47], FOSTER [36], and MEMO [51]. For Finetune, LwF [18] and Aper [52], we construct a fair comparison that includes a frozen PTM and a trainable adapter module, both serving as the backbone for training.

Evaluation Metrics. Following the widely used benchmarks [35, 54], two metrics are adopted for evaluation. The overall performance is evaluated by the average incremental accuracy $\bar{A}(\%) : \bar{A} = \frac{1}{T}\sum_{t=1}^{T} A_t$ where T and A_t respectively denote the number of incremental tasks and the test accuracy on all seen classes at task t. Another metric is the last-task accuracy $A_T(\%)$, which denotes the final average accuracy on all classes.

5.2. State-of-the-art Comparison

In Table 1, we compare MoAL with several state-of-the-art (SOTA) methods. Notably, the non-PTM-based ACIL [59], benefiting from its recursive structure, achieves competitive results relative to the PTM-optimized EASE [53], highlighting the strong compatibility between analytical learning-based CIL methods and PTMs. However, MoAL outperforms all baselines in both $\hat{A}$ and A_T across nearly all datasets and settings. In particular, on datasets that differ from the pre-training data, such as ImageNet-A and ImageNet-R, MoAL surpasses the best baseline by an average of 2.29% and 2.92% in $\hat{A}$, respectively. Furthermore, our method demonstrates minimal performance degradation as the number of tasks increases.

It's worth note that ACIL's final accuracy on CIFAR-100 is close to MoAL's, due to the inherent generalization capabilities of PTMs on this dataset, even without continual adaptation. However, on more challenging datasets like ImageNet-A and ImageNet-R, MoAL significantly outperforms ACIL, thanks to its enhanced PTM adaptivity, which ensures superior stability and plasticity. These trends are also reflected in the accuracy curves shown in Figure 3. These comparisons highlight the value of our method, particularly when downstream datasets differ significantly from pre-trained data, showcasing its strong generalization across diverse domains. Please see the supplementary material for more results on different PTM backbones and

Table 2. Ablation study comparing the impact of using different components in MoAL on the average incremental accuracy $\bar{A}(\%)$.

Knowledge Distillation	Weight Interpolation	Knowledge Rumination	Imagenet-A		Imagenet-R		OmniBenchmark	
			$T=10$	5	$T=10$	5	$T=15$	10
			70.93	72.04	81.21	83.36	76.54	76.09
✓			66.93	69.48	64.39	63.12	76.99	82.89
	✓		72.66	74.01	84.27	84.08	81.77	82.21
✓		✓	70.56	71.41	82.60	82.65	80.57	83.84
	✓	✓	**74.29**	**75.22**	**84.45**	**85.39**	**85.68**	**84.45**

Table 3. Average incremental accuracy $\bar{A}(\%)$ comparison of different values of α across various benchmarks and settings.

α	CIFAR-100		Imagenet-A		OmniBenchmark	
	$T=10$	5	$T=10$	5	$T=15$	10
0.99	93.16	93.73	72.96	74.50	77.63	79.96
0.995	93.64	93.84	74.20	75.01	84.67	82.10
0.999	94.11	94.03	**74.29**	75.22	85.68	84.45
0.9995	94.21	**94.00**	74.19	**75.24**	85.00	84.62
0.9999	**94.22**	93.91	73.98	75.15	**85.57**	**84.68**

datasets, and the comparison to replay-based methods.

5.3. Ablation Study

In Table 2, we present an ablation study to assess each component's contribution in MoAL. Removing both weight interpolation and knowledge rumination reduces MoAL to the classical ACIL method [59]. For further insights, we replace our momentum-based weight interpolation with the widely-used feature-based knowledge distillation regularization. The results indicate that knowledge distillation alone results in a noticeable performance drop, while our weight interpolation improves performance even without the recursive principle, underscoring the importance of continual adaptation. Incorporating knowledge rumination alongside these techniques further boosts performance, particularly with an increasing number of tasks, demonstrating its effectiveness in reinforcing old knowledge. The complementary roles of weight interpolation and knowledge rumination are clear: interpolation enhances representation adaptability, allowing finer-grained feature generation for old classes. Together, these findings validate the effectiveness and mutual interplay of the two core components.

5.4. Sensitivity Analysis

There are two key hyperparameters in MoAL: the momentum parameter α in Eq. 5 and the expanded dimension d_B of buffer layer f_B. Table 3 shows results from experiments conducted over a wide range of α values across different datasets and incremental settings. As observed, when $\alpha > 0.99$, MoAL exhibits strong robustness and close performance. However, setting $\alpha = 0.99$ leads to performance degradation across benchmarks. Higher values of α better preserve the PTM's inherent generalization. In comparison, lower values can disrupt the balance between adaptivity and generalizability, especially in challenging datasets that differ significantly from the pre-training data. For d_B, from

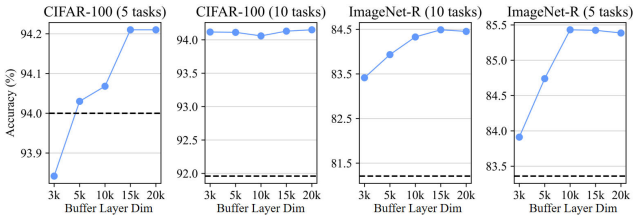

Figure 4. The impact of buffer layer dim d_B on metric $\bar{A}(\%)$. The dotted black line denotes the best performance of other methods.

Table 4. Average incremental accuracy $\bar{A}(\%)$ comparison among different architectures of prototype correction network f_C.

Correction Network f_C	CIFAR-100		Imagenet-A		OmniBenchmark	
	$T=10$	$T=5$	$T=10$	$T=5$	$T=15$	$T=10$
[768, 768]	**94.22**	**94.03**	**74.29**	75.22	**85.68**	84.45
[768, 768, 768]	94.01	93.97	73.56	75.08	85.67	**84.63**
[768, 1024, 768]	94.02	93.91	73.59	**75.26**	84.67	84.39

Figure 4, we find that values exceeding $3k$ consistently yield performance surpassing the best results from other methods. These findings confirm the robustness of MoAL with respect to these parameters. Based on these results, we recommend setting α to 0.999 and $d_B > 10k$ to achieve near-optimal, reliable performance. As shown in Table 4, we implement the prototype correction network f_C with three different architectures. The results demonstrate consistent performance across these variations, with the two-layer configuration offering both simplicity and strong performance. More analyses are provided in the supplementary file.

6. Conclusion

Incremental learning represents an essential capability for intelligent systems in dynamic environments. While effective, PTM-based analytical CIL methods rely on first-task adaptivity, prioritizing stability over learning flexibility, which limits adaptability to new data distributions and restricts real-world applicability. In this paper, we propose MoAL to address these key challenges. By enhancing PTM adaptivity through momentum-based adapter weight interpolation, MoAL enables continual adaption without sacrificing generalizability. It integrates knowledge memorization and rumination mechanisms to effectively acquire new knowledge and selectively reinforce old knowledge, achieving a better balance between plasticity and stability. Extensive experiments validate MoAL's effectiveness, marking a significant progress in analytical learning-based CIL by removing the constraint of a frozen feature extractor.

Acknowledgments. This work was supported by the National Science and Technology Major Project (2023ZD0121101), the National University of Defense Technology (ZZCX-ZZGC-01-04), and National Natural Science Foundation of China (No.62172426, 62106123).

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

[2] Simon A. Barton. A Matrix Method for Optimizing a Neural Network. *Neural Computation*, 3(3):450–459, 1991. 3

[3] Isaac Cervantes-Sandoval, Molee Chakraborty, Courtney MacMullen, and Ronald L Davis. Scribble scaffolds a signalosome for active forgetting. *Neuron*, 90(6):1230–1242, 2016. 5

[4] Shoufa Chen, Chongjian GE, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 16664–16678. Curran Associates, Inc., 2022. 2, 3, 4

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3, 6

[6] Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11483–11493, 2023. 1, 2, 3, 7

[7] Yang Gao, Yichun Shuai, Xuchen Zhang, Yuwei Peng, Lianzhang Wang, Jing He, Yi Zhong, and Qian Li. Genetic dissection of active forgetting in labile and consolidated memories in drosophila. *Proceedings of the National Academy of Sciences*, 116(42):21191–21197, 2019. 5

[8] Zijian Gao, Kele Xu, Huiping Zhuang, Li Liu, Xinjun Mao, Bo Ding, Dawei Feng, and Huaimin Wang. Less confidence, less forgetting: Learning with a humbler teacher in exemplar-free class-incremental learning. *Neural Networks*, 179:106513, 2024. 2

[9] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *Computer Science*, 84 (12):1387–91, 2013. 1

[10] Ping Guo and Michael R Lyu. A pseudoinverse learning algorithm for feedforward neural networks with stacked generalization applications to software reliability growth data. *Neurocomputing*, 56:101–121, 2004. 3

[11] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8340–8349, 2021. 3, 6

[12] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15262–15271, 2021. 3, 6

[13] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. 2, 5

[14] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning (ICML)*, pages 2790–2799. PMLR, 2019. 3

[15] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. 2

[16] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases*, 1(4), 2009. 6

[17] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3045–3059. Association for Computational Linguistics, 2021. 2

[18] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(12):2935–2947, 2017. 2, 5, 7

[19] Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. Weighted ensemble models are strong continual learners. *Proceedings of the European conference on computer vision (ECCV)*, 2024. 1

[20] Marc Masana, Xialei Liu, Bartłomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: Survey and performance evaluation on image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5513–5533, 2023. 1

[21] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995. 1

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

[23] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012. 6

[24] OpenAI. Gpt-4 technical report. *arxiv 2303.08774*, 2023. 1

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[26] Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. Fetril: Feature translation for exemplar-free class-incremental learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3911–3920, 2023. 1

[27] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2001–2010, 2017. 2, 6, 7

[28] Blake A Richards and Paul W Frankland. The persistence and transience of memory. *Neuron*, 94(6):1071–1084, 2017. 5

[29] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. *arXiv preprint arXiv:1810.11910*, 2018. 2

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115:211–252, 2015. 3

[31] Wuxuan Shi and Mang Ye. Prototype reminiscence and augmented asymmetric knowledge aggregation for non-exemplar class-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1772–1781, 2023. 2

[32] Yanyan Shi, Dianxi Shi, Ziteng Qiao, Zhen Wang, Yi Zhang, Shaowu Yang, and Chunping Qiu. Multi-granularity knowledge distillation and prototype consistency regularization for class-incremental learning. *Neural Networks*, 164:617–630, 2023. 1

[33] Yichun Shuai, Binyan Lu, Ying Hu, Lianzhang Wang, Kan Sun, and Yi Zhong. Forgetting is regulated through rac activity in drosophila. *Cell*, 140(4):579–589, 2010. 5

[34] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 11909–11919, 2023. 1, 2, 3, 7

[35] Hai-Long Sun, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Pilot: A pre-trained model-based continual learning toolbox. *arXiv preprint arXiv:2309.07117*, 2023. 6, 7

[36] Fu-Yun Wang, Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Foster: Feature boosting and compression for class-incremental learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 398–414, 2022. 2, 7

[37] Liyuan Wang, Mingtian Zhang, Zhongfan Jia, Qian Li, Chenglong Bao, Kaisheng Ma, Jun Zhu, and Yi Zhong. Afec: Active forgetting of negative transfer in continual learning. In *Advances in Neural Information Processing Systems*, pages 22379–22391. Curran Associates, Inc., 2021. 5

[38] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 69054–69076. Curran Associates, Inc., 2023. 2

[39] Liyuan Wang, Xingxing Zhang, Qian Li, Mingtian Zhang, Hang Su, Jun Zhu, and Yi Zhong. Incorporating neuro-inspired adaptability for continual learning in artificial intelligence. *Nature Machine Intelligence*, 5(12):1356–1368, 2023. 5

[40] Liyuan Wang, Xingxing Zhang, Hang Su, and Jun Zhu. A comprehensive survey of continual learning: Theory, method and application. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 46(8):5362–5383, 2024. 1

[41] Yueqing Wang, Yong Dou, Xinwang Liu, and Yuanwu Lei. Pr-elm: Parallel regularized extreme learning machine based on cluster. *Neurocomputing*, 173:1073–1081, 2016. 2

[42] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648. Springer, 2022. 1, 2, 3, 7

[43] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 139–149, 2022. 1, 2, 3, 6, 7

[44] JG Welch. Rumination, particle size and passage from the rumen. *Journal of animal science*, 54(4):885–894, 1982. 2

[45] Paul Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. *PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA*, 1974. 3

[46] Yicheng Xu, Yuxin Chen, Jiahao Nie, Yusong Wang, Huiping Zhuang, and Manabu Okumura. Advancing cross-domain discriminability in continual learning of vision-language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[47] 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 (CVPR)*, pages 3013–3022, 2021. 2, 7

[48] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning (ICML)*, pages 3987–3995. PMLR, 2017. 2

[49] Yuanhan Zhang, Zhenfei Yin, Jing Shao, and Ziwei Liu. Benchmarking omni-vision representation through the lens

of visual realms. In *European Conference on Computer Vision (ECCV)*, pages 594–611. Springer, 2022. 6

[50] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13208–13217, 2020. 2

[51] Da-Wei Zhou, Qi-Wei Wang, Han-Jia Ye, and De-Chuan Zhan. A model or 603 exemplars: Towards memory-efficient class-incremental learning. In *International Conference on Learning Representations (ICLR)*, 2023. 2, 7

[52] Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision (IJCV)*, 2024. 1, 2, 3, 4, 6, 7

[53] Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 23554–23564, 2024. 1, 2, 3, 4, 7

[54] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Class-incremental learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 46(12):9851–9873, 2024. 1, 2, 7

[55] Fei Zhu, Zhen Cheng, Xu-yao Zhang, and Cheng-lin Liu. Class-incremental learning via dual augmentation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14306–14318, 2021. 2, 6

[56] 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 (CVPR)*, pages 5871–5880, 2021. 6

[57] 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 (CVPR)*, pages 9296–9305, 2022. 2

[58] Huiping Zhuang, Zhiping Lin, and Kar-Ann Toh. Correlation projection for analytic learning of a classification network. *Neural Processing Letters*, 53:3893–3914, 2021. 3

[59] Huiping Zhuang, Zhenyu Weng, Hongxin Wei, Renchunzi Xie, Kar-Ann Toh, and Zhiping Lin. Acil: Analytic class-incremental learning with absolute memorization and privacy protection. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 11602–11614, 2022. 1, 2, 3, 7, 8

[60] Huiping Zhuang, Zhenyu Weng, Run He, Zhiping Lin, and Ziqian Zeng. 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 (CVPR)*, pages 7746–7755, 2023. 1, 2, 3

[61] Huiping Zhuang, Yizhu Chen, Di Fang, Run He, Kai Tong, Hongxin Wei, Ziqian Zeng, and Cen Chen. GACL: Exemplar-free generalized analytic continual learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[62] Huiping Zhuang, Run He, Kai Tong, Ziqian Zeng, Cen Chen, and Zhiping Lin. DS-AL: A dual-stream analytic learning for exemplar-free class-incremental learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(15):17237–17244, 2024. 1, 2, 3, 7