Sparse Retrieval Memory with Dynamic Expansion for Class-Incremental Learning

Xiangyu Zhou
Northwestern Polytechnical University
Xi'an, China
2021300119@mail.nwpu.edu.cn

Yue Lu
Northwestern Polytechnical University
Xi'an, China
zgxd@mail.nwpu.edu.cn

Shizhou Zhang*
Northwestern Polytechnical University
Xi'an, China
szzhang@nwpu.edu.cn

Yinghui Xing*
Northwestern Polytechnical University
Xi'an, China
xyh_7491@nwpu.edu.cn

Lingyan Ran
Northwestern Polytechnical University
Xi'an, China
lran@nwpu.edu.cn

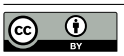
Di Xu
Huawei Technologies Co., Ltd.
Xi'an, China
xudi21@huawei.com

Peng Wang
Northwestern Polytechnical University
Xi'an, China
peng.wang@nwpu.edu.cn

Abstract

Recent work has demonstrated that sparse memory fine-tuning offers a principled solution to catastrophic forgetting in continual learning by decoupling task-specific knowledge through selective slot-level updates. However, a critical pre-training bottleneck limits practical adoption. The memory layer replaces an internal feed-forward network of the Transformer, and the resulting architecture fundamentally departs from any standard pre-trained model, necessitating expensive re-pre-training from scratch. To overcome this bottleneck, we propose Sparse Retrieval Memory, an external sparse memory architecture that can be directly deployed on frozen pre-trained Vision Transformers without architectural surgery or re-pre-training. Concretely, we attach learnable key-value memory modules as parallel residual branches alongside the multi-head self-attention at specific intermediate layers. Through Top-K sparse routing, each input token dynamically retrieves a small subset of the most relevant memory slots, significantly mitigating gradient interference via sparse parameter activation. Furthermore, the memory pool grows via coverage-based dynamic expansion, automatically adapting its capacity to the complexity of the incoming data stream. Extensive experiments on ImageNet-R, CIFAR-100, and DomainNet demonstrate state-of-the-art performance, achieving 80.50% on 10-task ImageNet-R and 84.68% on 10-task DomainNet, while training fewer than 3% of backbone parameters and storing zero historical exemplars. The source code is available at https://github.com/xyz-npu/SRM-CIL.

*Corresponding authors.



This work is licensed under a Creative Commons Attribution 4.0 International License.
MM '26, Rio de Janeiro, Brazil
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2213-4/2026/11
https://doi.org/10.1145/3767308.3835874

CCS Concepts

• **Computing methodologies** → **Lifelong machine learning**; *Computer vision*; Neural networks.

Keywords

Continual Learning, Class-Incremental Learning, Vision Transformer, Sparse Retrieval Memory, Dynamic Expansion

ACM Reference Format:
Xiangyu Zhou, Yue Lu, Shizhou Zhang, Yinghui Xing, Lingyan Ran, Di Xu, and Peng Wang. 2026. Sparse Retrieval Memory with Dynamic Expansion for Class-Incremental Learning. In *Proceedings of the 34th ACM International Conference on Multimedia (MM '26), November 10–14, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3767308.3835874

1 Introduction

Deep neural networks suffer from catastrophic forgetting [16, 20] when novel categories emerge sequentially, a challenge formalized as class-incremental learning (CIL) [49]. A prevailing exemplar-free paradigm freezes a pre-trained backbone and attaches lightweight parameter-efficient fine-tuning (PEFT) modules, such as learnable prompts or dense adapters, to absorb incremental knowledge. However, these densely shared PEFT parameters are vulnerable to **representation drift**: sequentially updating shared weights across tasks inevitably alters the foundational feature flow, causing cross-task feature interference and degrading previously acquired representations.

Recent PEFT methods mitigate this interference through structured subspace constraints. InfLoRA [21] confines each task's gradient updates to an orthogonal low-rank subspace, and SD-LoRA [42] freezes old adapters and stacks new ones; yet both rely on dense,

globally shared adapter parameters, limiting plasticity and accumulating feature drift over long sequences. Sparse memory fine-tuning [22] instead replaces an internal feed-forward network (FFN) with a massive key-value lookup layer, selectively updating only the highest-scored slots via a TF-IDF sparsity criterion to decouple task-specific knowledge. However, this structural replacement departs from any standard pre-trained model, necessitating expensive re-pre-training from scratch and rendering the method impractical for practitioners with off-the-shelf frozen foundation models.

To overcome this bottleneck, we propose **Sparse Retrieval Memory**, an *external* sparse memory architecture for exemplar-free CIL. We deploy learnable key-value memory modules as parallel residual branches alongside the multi-head self-attention at intermediate layers of a *completely frozen* pre-trained ViT, leaving the backbone unmodified and requiring **zero re-pre-training**. Through Top-K sparse routing, each token retrieves only a small subset of the most relevant memory slots, significantly mitigating gradient interference with previously stored representations via sparse parameter activation. Retrieved representations are injected into the pre-trained feature flow through a channel-wise gated residual branch, a logarithmic usage penalty prevents a few popular slots from monopolizing retrieval, and the memory pool grows organically via coverage-based dynamic expansion.

Extensive experiments on ImageNet-R, CIFAR-100, and DomainNet under protocols ranging from 10 to 100 sequential tasks show that our method consistently sets new state-of-the-art results, achieving 80.50% CIL accuracy on 10-task ImageNet-R and 84.68% on 10-task DomainNet, while training fewer than 3% of the backbone parameters and storing zero historical exemplars.

Our main contributions are as follows:

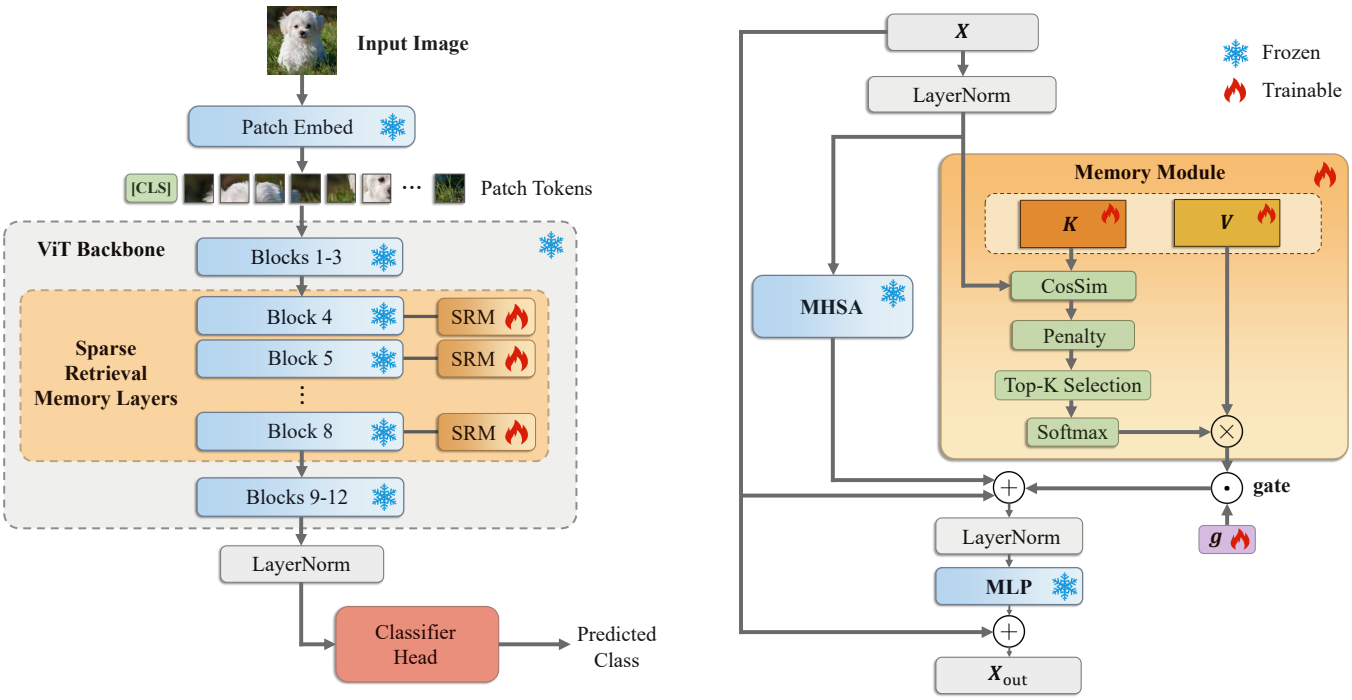
- We propose an external Sparse Retrieval Memory architecture that augments a frozen pre-trained ViT with learnable key-value memory modules at intermediate layers, achieving strong plasticity for new tasks via Top-K sparse routing while keeping the backbone intact and requiring no re-pre-training.
- We design a channel-wise gated residual integration, together with a logarithmic usage penalty for load balancing, coverage-based dynamic expansion for flexible capacity management, and key orthogonality regularization for slot diversity.
- Extensive experiments across ImageNet-R, CIFAR-100, and DomainNet (10–100 tasks) establish a new state-of-the-art for exemplar-free CIL, e.g., 80.50% on 10-task ImageNet-R with less than 3% trainable parameters.

2 Related Work

Continual Learning. Continual learning typically grapples with catastrophic forgetting when non-stationary data distributions are encountered sequentially. Regularization-based methods [16, 20] constrain weight updates or distill knowledge, but their effectiveness degrades over long sequences, while rehearsal-based methods [1, 30] retain historical knowledge via replay buffers yet raise privacy, memory, and copyright concerns. Consequently, exemplar-free continual learning has become a crucial setting, where recent state-of-the-art approaches leverage prompt learning [33, 40, 41]: L2P [41] and DualPrompt [40] maintain pools of learnable prompts on a frozen ViT, CODA-Prompt [33] learns attention-weighted prompt combinations, and variants [14, 24, 25, 37, 38] specialize prompts for domain-incremental, disentangled, or discrete-selection settings. Orthogonally, analytical classifier approaches [12, 27, 47, 52] achieve strong results via prototypes or Ridge Regression with random projections, avoiding iterative classifier optimization. While effective, prompt-based methods inherently alter the input sequence length; in contrast, our method embeds memory natively within the feature transformation layers, operating entirely in the latent space.

Parameter-Efficient Fine-Tuning. Transferring massive pre-trained models has popularized PEFT: Visual Prompt Tuning [13] and LoRA [10] show that optimizing a small fraction of parameters can match full fine-tuning, and lightweight adapters [3, 9] are inserted alongside attention or feed-forward layers. However, standard PEFT in continual learning exposes a critical flaw: densely shared parameters are overwritten by new-task gradients, causing cross-task interference. Recent methods address this via structured subspace constraints: InfLoRA [21] confines each task's adaptation to orthogonal low-rank subspaces, SD-LoRA [42] freezes old adapters and stacks new ones, and BiLoRA [51] and LoRA-DRS [23] enforce near-orthogonal or drift-resistant parameter spaces [7]. VPT-NSP [26, 45] instead project prompt gradients into the null space of previously learned features, providing theoretical guarantees against backward interference in visual prompt tuning. Meanwhile, SLCA [44] and SSITuning [34] improve stability via slow learning rates and semantically-shifted adapters. Despite these advances, all such methods rely on dense globally connected weights that limit plasticity and accumulate feature drift over long sequences. Our approach instead employs sparsely routed memory blocks, where only K out of M slots are activated per token, significantly mitigating gradient interference across tasks.

Sparse Memory and Routing Mechanisms. Sparsity is a powerful inductive bias for scaling models and decoupling knowledge. Mixture of Experts [32] and sparse attention route tokens to specific subnetworks via gating networks, physically isolating parameters and preventing gradient interference among distinct patterns. In the continual learning context, MoE-based adapters [43] decompose knowledge across expert modules for vision-language models, and KMR [6] introduces a knowledge memorization and rumination framework via external memory modules. More recently, Lin et al. [22] showed that sparse memory fine-tuning, which replaces an internal FFN with a massive key-value lookup and selectively updates only the highest-scored slots via a TF-IDF criterion, decouples task-specific knowledge and mitigates continual forgetting, albeit with a prohibitive re-pre-training bottleneck. Inspired by these advances, we introduce an *external* sparse memory architecture that augments frozen pre-trained Vision Transformers without modifying any internal components, completely eliminating the re-pre-training requirement while preserving the benefits of sparse knowledge isolation.



(a) Overall Architecture

(b) SRM-Augmented Transformer Block

Figure 1: Overview of Sparse Retrieval Memory. Learnable key-value memory modules are attached as parallel residual branches at intermediate layers of a frozen pre-trained ViT. Each input token is routed to the Top-K most relevant memory slots via cosine similarity, and the retrieved values are injected into the feature flow through a channel-wise gated residual connection. The memory pool grows organically via coverage-based dynamic expansion.

3 Methodology

3.1 Problem Definition and Overview

We consider class-incremental learning over a sequential curriculum $\mathcal{T} = \{1, 2, \ldots, T\}$. At each phase t, the model observes dataset $\mathcal{D}_t = \{(\boldsymbol{x}_{i,t}, y_{i,t})\}$ with labels $y_{i,t} \in C_t$, where label spaces are mutually disjoint ($C_i \cap C_j = \emptyset$ for $i \neq j$). No historical data is retained: samples from $\mathcal{D}_1, \ldots, \mathcal{D}_{t-1}$ are completely inaccessible when learning task t. At evaluation, the model must classify test samples over the cumulative label space $\bigcup_{i=1}^{T} C_i$ without task identifiers. We employ a pre-trained Vision Transformer (ViT) [4] whose backbone parameters remain frozen throughout training, and introduce a minimal learnable parameter set, realized as sparse external memory modules, to absorb incremental task knowledge without interfering with previously acquired representations.

As illustrated in Figure 1, our method augments a frozen pre-trained ViT with Sparse Retrieval Memory modules attached as parallel residual branches at intermediate layers (Section 3.2). Each module maintains a pool of key-value slots; input tokens are routed to their Top-K most relevant slots via cosine-similarity sparse routing, and the retrieved values are injected into the feature flow through a channel-wise gated residual connection. To prevent a few popular slots from monopolizing retrieval, we introduce a logarithmic-usage-penalty load balancing mechanism (Section 3.3), and the memory pool grows organically via coverage-based dynamic expansion, activating zero-initialized dormant slots to accommodate novel distributions (Section 3.4).

3.2 Sparse Retrieval Memory

Existing PEFT methods densely share their adapter parameters across all tokens, causing destructive gradient interference when sequentially optimized for different tasks. We instead draw inspiration from sparse Mixture-of-Experts, where only a small subset of parameters is activated per input, reducing gradient interference and enabling selective knowledge reuse.

To endow the locked backbone with task-agnostic plasticity, we augment specific intermediate layers of the ViT with learnable Sparse Retrieval Memory modules. Let M denote the total capacity (number of available memory slots) and C denote the embedding dimension. The memory pool is mathematically parameterized by a Key matrix $\boldsymbol{K} \in \mathbb{R}^{M \times C}$, a Value matrix $\boldsymbol{V} \in \mathbb{R}^{M \times C}$, and an active mask $\boldsymbol{A} \in \{0, 1\}^M$: entries $A_m = 1$ mark active slots, whereas dormant slots ($A_m = 0$) are excluded from retrieval until activated by coverage-based expansion (Section 3.4).

Given a batch of input token sequences $\boldsymbol{X} \in \mathbb{R}^{B \times N \times C}$, where B represents the batch size and N represents the sequence length (comprising [CLS] token and patch tokens), we initially evaluate

the semantic affinity between the incoming tokens and the persistent memory slots. We employ L_2-normalized cosine similarity to ensure that the retrieval gradients remain mathematically stable. The attention score $S_{b,n,m}$ between the n-th token of the b-th sample ($X_{b,n} \in \mathbb{R}^C$) and the m-th memory key ($K_m \in \mathbb{R}^C$) is computed as:

$$S_{b,n,m} = \gamma \left(\frac{X_{b,n}}{\|X_{b,n}\|_2} \right) \left(\frac{K_m}{\|K_m\|_2} \right)^\top \tag{1}$$

where $S \in \mathbb{R}^{B \times N \times M}$ constitutes the holistic attention score matrix, and γ serves as a constant scaling factor (empirically set to 10.0).

To realize effective knowledge separation and reduce gradient coupling across divergent visual tasks, we enforce extreme sparsity by performing a Top-K masking operation over the slot dimension M. Let $\mathcal{I}_{\text{top-}K}(b,n)$ denote the index set of the K largest values within the score vector $S_{b,n,m}$. The masked attention scores $\widetilde{S}$ are formulated as:

$$\widetilde{S}_{b,n,m} = \begin{cases} S_{b,n,m} & \text{if } A_m = 1 \wedge m \in \mathcal{I}_{\text{top-}K}(b,n) \\ -\infty & \text{otherwise} \end{cases} \tag{2}$$

The retrieved memory features $Z \in \mathbb{R}^{B \times N \times C}$ are subsequently generated via a softmax-weighted aggregation of the corresponding Values:

$$Z = \text{Softmax}(\widetilde{S})V \tag{3}$$

To seamlessly integrate the retrieved representations Z into the pre-trained feature flow, we design a gated residual integration strategy.

Gated Residual Integration. The Sparse Retrieval Memory operates as a parallel auxiliary branch strictly localized to the transformer block. It ingests the identically normalized input as the frozen Multi-Head Self-Attention (MHSA) layer. To prevent the freshly initialized memory parameters from disrupting the well-established pre-trained latent space, we introduce a channel-wise gating mechanism $g \in \mathbb{R}^{1\times1\times C}$. The block output is computed as:

$$X_{\text{out}} = X + \text{MHSA}(\text{LN}_1(X)) + \tanh(g) \odot Z \tag{4}$$

where Z is derived using $\text{LN}_1(X)$ as the query. We initialize g such that $\tanh(g) \approx 0.1$, so the memory branch injects only ~10% of its activation at the onset of training, guaranteeing stable forward propagation while leaving the gate capacity to scale specific channels as training advances. The overall architecture is illustrated in Figure 1, and the internal structure of a single memory layer is detailed in Figure 2.

3.3 Dynamic Load Balancing via Logarithmic Penalty

A persistent vulnerability of sparse Top-K routing is representation collapse: a few slots monopolize the gradient flow while others starve. In continual learning this is particularly severe, as early-task overuse dominates routing for all subsequent tasks. To enforce uniform exploitation of the memory pool, we design a load balancing scheme guided by a logarithmic usage penalty.

We track an unnormalized cumulative usage count vector $c \in \mathbb{R}^M$ for all memory slots across the training iterations. During the forward pass, we first map the raw cosine similarities to a non-negative range to ensure a well-behaved multiplicative penalty:

$$S^{\text{nonneg}}_{b,n,m} = \frac{1 + S_{b,n,m}}{2} \in [0,1] \tag{5}$$

Immediately preceding the Top-K masking operation, we penalize the score of the m-th slot based upon its historical activation frequency c_m:

$$P_m = \frac{1}{\log(1 + c_m) + 1} \tag{6}$$

$$S^{\text{penalized}}_{b,n,m} = S^{\text{nonneg}}_{b,n,m} \times P_m \tag{7}$$

The Top-K selection is subsequently executed exclusively on $S^{\text{penalized}}$. Unlike aggressive linear penalties, the logarithmic decay smoothly suppresses excessively used slots (e.g., $P_m \approx 0.10$ when $c_m = 10000$) while mildly regularizing occasionally used ones, averting gradient starvation without the oscillatory instabilities of abrupt linear suppression. The count vector c is updated under a stop-gradient context.

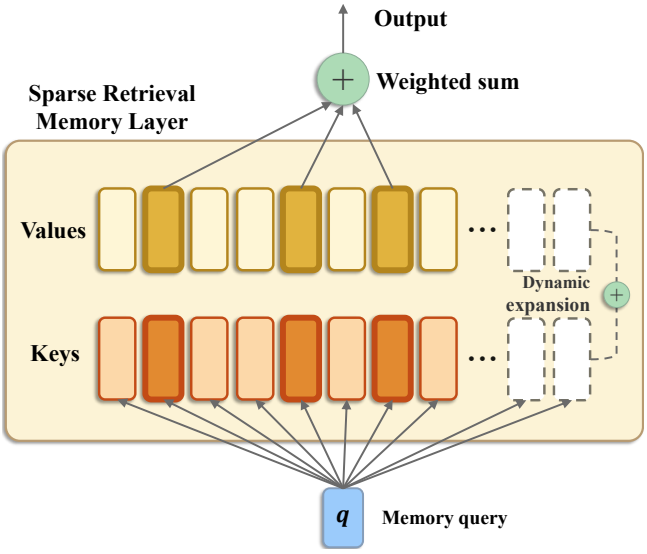


Figure 2: Internal structure of a Sparse Retrieval Memory layer. Each layer maintains M learnable Key-Value slot pairs. Input tokens query the keys via cosine similarity, and only the Top-K most relevant slots (highlighted) are retrieved for a weighted combination. Dormant slots (dashed) are initially inactive and can be dynamically activated via coverage-based expansion to accommodate novel distributions.

3.4 Coverage-based Dynamic Expansion

The memory pool's capacity dictates the model's continual plasticity: a fixed budget saturates as tasks grow, forcing slot overwriting, while over-provisioning wastes capacity and hinders utilization. To adapt, we manage the pool through an active mask vector $A \in \{0,1\}^M$ over a unified pre-allocated parameter pool and design a coverage-based dynamic expansion mechanism (dormant slots in Figure 2).

Concretely, for any incoming sample i within a batch, we quantify the coverage gap d_i of its [CLS] token $\boldsymbol{x}_i^{\text{cls}} \in \mathbb{R}^C$ against all currently active memory keys. The [CLS] token aggregates global image information and thus serves as a stable, compact proxy of the input distribution for expansion decisions, avoiding spurious activations triggered by local patch-level patterns:

$$d_i = 1 - \max_{j \in \{m | A_m = 1\}} \cos(\boldsymbol{x}_i^{\text{cls}}, \boldsymbol{K}_j) \tag{8}$$

When d_i surpasses a predefined empirical threshold τ, it signifies that the existing active pool insufficiently represents the input distribution. A dormant slot is then activated with its key initialized from the novel input feature $\boldsymbol{K}_{\text{new}} = \boldsymbol{x}_i^{\text{cls}}$ and its active state toggled to $A_{\text{new}} = 1$.

Zero Value Initialization. A critical design choice for newly activated slots is the initialization of their value vectors. A natural strategy is to copy the input features into the new value ($\boldsymbol{V}_{\text{new}} = \boldsymbol{x}_i^{\text{cls}}$); however, this injects an unsupervised signal into the memory output before the slot has received gradient supervision. Instead, we initialize new slot values to zero ($\boldsymbol{V}_{\text{new}} = \boldsymbol{0}$), so freshly activated slots contribute nothing until shaped by task-specific gradients, while the key, initialized from $\boldsymbol{x}_i^{\text{cls}}$, lets the slot participate in routing competition immediately.

Per-Task Expansion Budget. To prevent unbounded growth, we enforce a per-task budget β and clamp the maximum new activations per batch (to 3 in all experiments) for representational stability. Newly activated slots inherit the mean existing usage count, preventing the penalty from unfairly boosting zero-count slots.

In all main experiments, we start from M=100 slots per layer with a total pool capacity of 300, comparing with simpler alternatives in Section 4.3.

4 Experiments

4.1 Experimental Setup

Datasets. To comprehensively evaluate the robustness of our proposed method, we conduct experiments on diverse continual learning benchmarks:

- **ImageNet-R** [8]: Contains 200 classes with diverse artistic renditions (e.g., sketches, paintings, sculptures). We evaluate it under varying sequence lengths, including 10, 20, 50, and 100 tasks (denoted as 10S, 20S, 50S, and 100S).
- **CIFAR-100** [17]: Consists of 100 classes of natural images, split into 10 tasks (10S).
- **DomainNet** [28]: A highly challenging dataset drawn from 6 distinct visual domains, evaluated under 10, 50, and 100 task splits.

Evaluation Protocols. We follow the **Exemplar-Free Class-Incremental Learning (CIL)** paradigm. The memory buffer size for historical data is strictly set to zero ($\mathcal{E} = \emptyset$). After learning all T tasks, the model is evaluated on each task k's test set across all seen classes without task identities. Let $a_{T,k}$ denote the Top-1 accuracy on task k after the completion of all T tasks. We report the following two metrics:

Average Accuracy (↑): the mean per-task accuracy after learning all tasks:

$$A_T = \frac{1}{T} \sum_{k=1}^{T} a_{T,k} \tag{9}$$

Average Forgetting (↓): the mean accuracy degradation relative to each task's historical best:

$$F_T = \frac{1}{T-1} \sum_{k=1}^{T-1} \left(\max_{t \in \{1,\ldots,T-1\}} a_{t,k} - a_{T,k} \right) \tag{10}$$

Higher Accuracy and lower Forgetting indicate better continual learning performance.

Baselines. We compare against state-of-the-art exemplar-free continual learning methods: the frozen backbone baseline (**Seq-FT**), prompt-based methods (**L2P**, **DualPrompt**, **CODA-Prompt**, **EvoPrompt** [18, 33, 40, 41]), adapter-based (**EASE** [48]), and PEFT methods (**InfLoRA**, **SD-LoRA**, **BiLoRA** [21, 42, 51]).

Implementation Details. We employ a ViT-B/16 pre-trained on ImageNet-21K as our foundation model, keeping its weights entirely frozen. Our Sparse Retrieval Memory Modules are integrated in parallel at intermediate layers 3–7 (5 layers) for ImageNet-R and DomainNet, and layers 3–6 (4 layers) for CIFAR-100. Each layer is initialized with M=100 memory slots, which are dynamically expanded via the coverage-based strategy during training (reaching up to approximately 300 active slots). During routing, we retrieve the Top-K slots (K=10 for ImageNet-R and CIFAR-100, K=5 for DomainNet). We use L_2-normalized cosine attention for ImageNet-R and CIFAR-100, and standard dot-product attention for DomainNet. We empirically find that under the strictly frozen backbone constraint, cosine similarity exhibits limited learning capacity and flexibility on DomainNet; switching to standard dot-product attention ensures sufficient gradient flow and learning effectiveness in this domain-diverse setting. The channel-wise gate is initialized such that $\tanh(\boldsymbol{g}) \approx 0.1$ to ensure conservative memory injection initially. The memory modules introduce approximately 2.3M trainable parameters (2.7% of the frozen ViT-B/16 backbone); however, due to Top-K sparse routing (K=10 of up to 300 active slots per layer), the per-step activated parameters (and hence the incremental FLOPs and inference latency) are negligible compared to the dense self-attention layers.

Optimization Strategy. The learnable memory parameters are optimized using AdamW with a base learning rate of 3×10^{-4} and a cosine annealing schedule. Key normalization and an orthogonal regularization loss (weight 0.1) maintain key diversity, and the logarithmic usage penalty is integrated before the Top-K operation.

Classification. During training, standard task-isolation techniques (logit slicing, Weight Alignment [46]) are applied. For inference, the SGD-trained linear head is replaced by an analytical Ridge Regression classifier [27] that projects memory-augmented [CLS] features to high dimensions via a fixed random matrix and solves the weights in closed form from cumulative cross-task statistics. To ensure fair comparison, we note that the Ridge Regression classifier is an additional design choice orthogonal to the memory architecture. On 10S-ImageNet-R, InfLoRA achieves 74.96% with its standard SGD CNN head but drops to 68.88% when using the RR classifier, while our method achieves 78.41% (SGD head) and

Table 1: Comparison with existing methods. The standard deviation (i.e., the value after ±) is reported when available. 'Seq-FT' denotes the frozen pre-trained backbone baseline evaluated with an analytical Ridge Regression classifier. Missing results are denoted by "-".

Method	Venue	10S-ImageNet-R		20S-ImageNet-R		10S-DomainNet		10S-CIFAR-100	
		Acc.↑	Forgetting↓	Acc.↑	Forgetting↓	Acc.↑	Forgetting↓	Acc.↑	Forgetting↓
L2P [41]	CVPR'22	$61.57_{\pm 0.66}$	$9.73_{\pm 0.47}$	$59.38_{\pm 0.50}$	$5.89_{\pm 0.36}$	$81.17_{\pm 0.83}$	$8.98_{\pm 1.25}$	$83.83_{\pm 0.04}$	$7.63_{\pm 0.30}$
DualPrompt [40]	ECCV'22	$68.13_{\pm 0.49}$	$4.68_{\pm 0.20}$	$63.21_{\pm 0.49}$	$5.28_{\pm 0.45}$	$81.70_{\pm 0.78}$	$8.04_{\pm 0.31}$	$86.51_{\pm 0.33}$	$5.16_{\pm 0.09}$
CODA-Prompt [33]	CVPR'23	$75.45_{\pm 0.56}$	$1.64_{\pm 0.10}$	$72.37_{\pm 1.19}$	$0.96_{\pm 0.15}$	$80.04_{\pm 0.79}$	$10.16_{\pm 0.35}$	$86.25_{\pm 0.74}$	$1.67_{\pm 0.26}$
APG [35]	ICCV'23	73.27	8.59	71.22	7.39	-	-	89.35	6.01
ESN [39]	AAAI'23	$71.07_{\pm 0.29}$	$4.99_{\pm 0.49}$	$64.77_{\pm 0.71}$	$6.65_{\pm 1.24}$	$79.22_{\pm 2.04}$	$10.62_{\pm 2.12}$	$86.34_{\pm 0.52}$	$4.76_{\pm 0.14}$
OS-Prompt++ [15]	ECCV'24	$75.67_{\pm 0.40}$	$\mathbf{1.27}_{\pm 0.10}$	$73.77_{\pm 0.19}$	$\mathbf{0.79}_{\pm 0.07}$	-	-	$86.68_{\pm 0.67}$	$\mathbf{1.18}_{\pm 0.21}$
EvoPrompt [18]	AAAI'24	$76.83_{\pm 0.08}$	$2.78_{\pm 0.06}$	$74.41_{\pm 0.23}$	$2.56_{\pm 0.22}$	$79.50_{\pm 0.29}$	$3.81_{\pm 0.36}$	$87.97_{\pm 0.30}$	$2.60_{\pm 0.42}$
PGP [29]	ICLR'24	$69.34_{\pm 0.05}$	$4.53_{\pm 0.04}$	-	-	$80.41_{\pm 0.25}$	$8.39_{\pm 0.18}$	$86.92_{\pm 0.05}$	$5.35_{\pm 0.19}$
OVOR-Deep [11]	ICLR'24	$76.11_{\pm 0.21}$	$7.16_{\pm 0.34}$	$73.85_{\pm 0.29}$	$6.80_{\pm 0.65}$	$79.61_{\pm 0.86}$	$4.77_{\pm 0.94}$	$85.99_{\pm 0.89}$	$6.42_{\pm 2.03}$
ConvPrompt [31]	CVPR'24	$77.86_{\pm 0.25}$	$4.33_{\pm 0.24}$	$75.10_{\pm 0.39}$	$4.10_{\pm 0.29}$	$79.47_{\pm 0.35}$	$6.49_{\pm 0.43}$	$88.87_{\pm 0.33}$	$4.75_{\pm 0.15}$
InfLoRA [21]	CVPR'24	$75.65_{\pm 0.14}$	$5.73_{\pm 0.44}$	$71.01_{\pm 0.45}$	$6.83_{\pm 0.44}$	$81.45_{\pm 0.68}$	$5.35_{\pm 0.52}$	$87.06_{\pm 0.25}$	$6.22_{\pm 0.39}$
EASE [48]	CVPR'24	76.17	7.82	73.27	8.51	78.89	7.89	87.76	5.94
CPrompt [5]	CVPR'24	$77.14_{\pm 0.11}$	$5.97_{\pm 0.68}$	$74.79_{\pm 0.28}$	$7.34_{\pm 0.65}$	$82.97_{\pm 0.34}$	$7.45_{\pm 0.93}$	$87.82_{\pm 0.21}$	$5.06_{\pm 0.50}$
VPT-CPG [25]	AAAI'25	$78.63_{\pm 0.52}$	$7.18_{\pm 0.62}$	$75.33_{\pm 0.49}$	$7.08_{\pm 0.52}$	$83.21_{\pm 0.67}$	$7.09_{\pm 0.82}$	$\mathbf{90.63}_{\pm 0.44}$	$3.98_{\pm 0.65}$
CAPrompt [19]	AAAI'25	$71.67_{\pm 0.45}$	$5.24_{\pm 0.48}$	$71.24_{\pm 0.80}$	$6.45_{\pm 0.78}$	$80.97_{\pm 0.98}$	$4.21_{\pm 0.34}$	$87.27_{\pm 0.39}$	$4.90_{\pm 0.22}$
SD-LoRA [42]	ICLR'25	$77.34_{\pm 0.35}$	-	$75.26_{\pm 0.37}$	-	$81.01_{\pm 0.42}$	-	$88.01_{\pm 0.31}$	-
BiLoRA [51]	CVPR'25	$77.95_{\pm 0.14}$	-	$72.41_{\pm 0.76}$	-	$76.56_{\pm 0.41}$	-	$87.46_{\pm 0.76}$	-
LoRA-DRS [23]	CVPR'25	$75.94_{\pm 0.46}$	-	$73.78_{\pm 0.44}$	-	$71.19_{\pm 0.55}$	-	$89.14_{\pm 0.23}$	-
SEMA [36]	CVPR'25	78.00	-	74.53	-	80.92	-	86.98	-
Seq-FT		72.77	4.04	72.61	5.27	82.66	**3.03**	81.28	6.89
Seq-LoRA		$62.30_{\pm 0.68}$	$30.06_{\pm 0.74}$	52.63	39.93	49.44	46.85	70.41	31.48
Ours	**This work**	$\mathbf{80.50}_{\pm 0.09}$	$4.85_{\pm 0.14}$	$\mathbf{79.92}_{\pm 0.05}$	$5.77_{\pm 0.23}$	$\mathbf{84.68}_{\pm 0.11}$	$4.25_{\pm 0.15}$	$88.82_{\pm 0.07}$	$6.37_{\pm 0.18}$

80.50% (RR head), showing that the performance gain stems from the sparse memory representations rather than the classifier choice.

Hyperparameters. For reproducibility, we list all hyperparameters explicitly: scaling factor γ=10.0, initial memory capacity M=100, retrieval count K=10 (ImageNet-R/CIFAR-100) and K=5 (DomainNet), coverage threshold τ=0.02, per-task expansion budget β=20, per-batch activation limit of 3, total pool capacity of 300 slots, learning rate 3×10^{-4} with AdamW optimizer and cosine annealing, orthogonal regularization weight 0.1, weight decay 0.05 (ImageNet-R/CIFAR-100) and 0.01 (DomainNet), gradient clipping at 5.0, random projection dimension 16,384 for the RR classifier, and gating initialization $\tanh(\boldsymbol{g})\approx0.1$.

4.2 Main Results

Table 1 summarizes Accuracy and Forgetting across the primary benchmarks, where our method significantly outperforms existing state-of-the-art methods across various sequence settings.

On ImageNet-R, our method achieves **80.50%** (10S) and **79.92%** (20S) CIL accuracy, surpassing VPT-CPG by 1.87% and 4.59% respectively. On DomainNet, our approach reaches **84.68%**, outperforming CPrompt by 1.71% and VPT-CPG by 1.47%. On CIFAR-100, our method attains a competitive 88.82%. Methods with dedicated prompt growing (e.g., VPT-CPG at **90.63%**) excel on this homogeneous benchmark, whereas our sparse memory design shows greater advantage on domain-diverse benchmarks. As shown in Figure 3, our method maintains substantially higher accuracy than the frozen baseline throughout the task sequence. Our forgetting rates (4.25%–6.37%) are moderate, reflecting the stability-plasticity trade-off of sparse routing where only K out of M slots receive gradients per token. Notably, the Seq-FT baseline exhibits the lowest forgetting rates because its backbone parameters are not updated, giving it perfect stability at the severe cost of limited plasticity (72.77%). Our method introduces trainable parameters to absorb new task knowledge, trading a marginal stability reduction for substantial accuracy gains. As reported in Table 1, a naive sequential LoRA fine-tuning baseline (Seq-LoRA, rank 8 in the same layers as SRM, under the identical RR classifier protocol) consistently underperforms our method across all benchmarks (e.g., 62.30±0.68% vs. 80.50% on 10S-ImageNet-R), with severe forgetting and its initially learned task accuracy collapsing from 91.6% to 51.8%. Increasing the rank from 8 to 16 yields no improvement (63.59%). These results show that vanilla PEFT without sparse routing suffers from catastrophic forgetting, whereas our advantage stems from the sparse retrieval mechanism rather than the mere presence of trainable parameters.

4.3 Ablation Study

To validate the algorithmic designs within our framework, we conduct extensive ablation studies on the ImageNet-R benchmark.

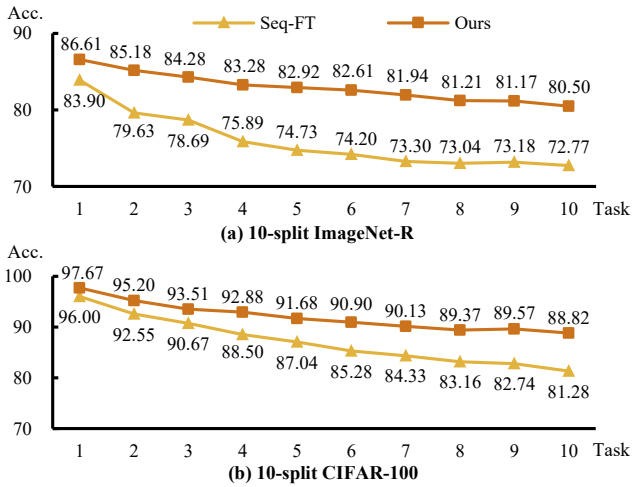


Figure 3: Per-task CIL accuracy curves on 10S-ImageNet-R (top) and 10S-CIFAR-100 (bottom). Our method (orange) consistently maintains higher accuracy than the frozen backbone baseline (golden) as tasks accumulate, with the gap widening over the task sequence.

Table 2: Component ablation study on ImageNet-R (20 tasks). Starting from the frozen ViT backbone, we incrementally add each proposed component.

Configuration	CIL (%)
Frozen ViT Backbone	72.61
+ Sparse Retrieval Memory (Fixed M=100)	79.68
+ Coverage-based Dynamic Expansion	**79.92**

As shown in Table 2, Sparse Retrieval Memory is the dominant contributor (**+7.07%** CIL over the frozen backbone), and coverage-based expansion adds **+0.24%** over the fixed-capacity baseline; disabling the logarithmic usage penalty reduces accuracy from 79.92% to 79.60% (−0.32%). While the expansion gain is modest on 20 tasks, it becomes decisive on longer sequences: on 100S-ImageNet-R, disabling expansion drops accuracy from 76.70% to 73.24% (a 3.46% degradation), confirming that fixed-capacity pools saturate and force slot overwriting in long sequences.

Memory Capacity Management Strategies. We compare three capacity management strategies: fixed capacity (M=100 unchanged), task-wise fixed expansion (a fixed number of new slots per task), and coverage-based expansion (adaptive growth via feature coverage analysis). As shown in Table 3, coverage-based expansion achieves the highest accuracy with only ∼300 slots (vs. task-wise expansion's ∼1050, which performs *worse* than fixed capacity by 0.23% as blindly adding slots degrades Top-K precision).

Hyperparameter Sensitivity. We examine the sensitivity to initial capacity M and retrieval count K. As shown in Table 4, M=100 achieves the optimal balance on ImageNet-R: smaller pools restrict representational bandwidth while larger pools introduce excessive sparsification. The optimal K differs across datasets (K=10 for ImageNet-R/CIFAR-100, K=5 for DomainNet), yet our method stays robust across all tested values (CIL variation ≤0.31% for M, ≤0.18% for K).

Table 3: Comparison of memory expanding strategies on ImageNet-R (20 tasks).

Expanding Strategy	Acc. (%)	Total Slots
Fixed Capacity (M=100)	79.68	100
Task-wise Fixed Expansion	79.45	∼1050
Coverage-based Expansion	**79.92**	∼300

Table 4: Hyperparameter sensitivity analysis on 10S-ImageNet-R.

Hyperparameter	Tested Values			
Memory Size M (initial)	50	**100**	150	200
Acc. (%)	80.25	**80.50**	80.37	80.19
Top-K	5	**10**	15	20
Acc. (%)	80.40	**80.50**	80.34	80.32

Table 5: Memory layer configuration ablation on 10S-ImageNet-R.

Layer Position	CIL (%)	Layer Count	CIL (%)
3-4-5-6-7	**80.50**	4 (layers 3-6)	80.32
4-5-6-7-8	79.96	**5 (layers 3-7)**	**80.50**
5-6-7-8-9	78.95	6 (layers 3-8)	80.05
6-7-8-9-10	77.93	7 (layers 3-9)	80.14

Memory Layer Configuration. We further ablate the placement and number of memory layers. As shown in Table 5, memory augmentation at earlier intermediate layers is more effective: shifting the 5-layer window from the default (layers 3–7) toward deeper layers causes progressive CIL degradation, with the deepest configuration (layers 6–10) dropping 2.57% relative to the default. This confirms that earlier intermediate layers where semantic features are actively formed are the most effective positions for memory augmentation. Regarding the number of memory layers, 5 layers achieve the best balance. Using only 4 layers restricts capacity, while adding more layers (6–7) yields diminishing returns with increased forgetting (+1.04% and +1.22% respectively), suggesting that deeper memory layers interfere with the backbone's upper-level abstract features.

4.4 Robustness and Scalability

Long-Sequence Continual Learning. A critical limitation of many CL methods is their inability to sustain performance over long task sequences. We stretch our evaluation to long-sequence scenarios by dividing ImageNet-R and DomainNet into **50 tasks** and **100 tasks**. As shown in Table 6, even under extreme sequential fragmentation, our memory-augmented architecture retains

Table 6: Performance on long-sequence continual learning experiments using 50 and 100 splits of ImageNet-R and DomainNet.

Method	Venue	50S-ImageNet-R		50S-DomainNet		100S-ImageNet-R		100S-DomainNet	
		Acc.↑	Fgt.↓	Acc.↑	Fgt.↓	Acc.↑	Fgt.↓	Acc.↑	Fgt.↓
L2P [41]	CVPR'22	51.38	12.34	63.13	11.19	41.51	14.48	54.83	14.95
OVOR-Deep [11]	ICLR'24	63.25	5.23	68.29	6.85	43.02	7.30	52.09	8.66
ConvPrompt [31]	CVPR'24	64.61	7.12	71.76	6.37	44.32	8.97	56.21	6.27
InfLoRA [21]	CVPR'24	62.81	10.37	71.87	9.20	42.23	14.17	48.06	15.43
EASE [48]	CVPR'24	70.27	6.73	65.34	9.64	51.56	7.56	37.26	29.12
CPrompt [5]	CVPR'24	70.75	7.44	70.74	9.01	59.90	9.52	57.60	9.42
VPT-CPG [25]	AAAI'25	73.08	12.12	72.27	19.93	64.63	12.18	60.81	27.43
Ours	**This work**	**77.51**	9.00	**82.16**	9.15	**76.70**	8.96	**80.84**	11.13

Table 7: Results using DINO and iBOT pre-trained weights (ImageNet-1k) on ImageNet-R and DomainNet (10 tasks each).

Weights	Method	10S-ImageNet-R		10S-DomainNet	
		Acc.↑	Fgt.↓	Acc.↑	Fgt.↓
DINO-1k	L2P [41]	56.71	4.60	50.72	16.89
	CODA-P [33]	64.02	7.70	66.74	8.91
	EASE [48]	65.97	9.24	70.23	9.64
	CPrompt [5]	64.66	9.26	75.31	9.93
	InfLoRA [21]	68.31	8.81	75.05	7.70
	VPT-CPG [25]	68.32	2.78	77.46	10.18
	Ours	**71.15**	9.04	**78.71**	7.76
iBOT-1k	L2P [41]	60.80	6.65	53.32	17.12
	CODA-P [33]	68.02	6.23	65.80	12.35
	EASE [48]	69.91	9.22	72.39	10.25
	CPrompt [5]	69.14	8.82	77.45	9.28
	InfLoRA [21]	71.84	9.05	76.60	8.51
	VPT-CPG [25]	72.16	5.80	77.79	13.02
	Ours	71.69	10.64	**82.38**	**6.17**

remarkable robustness. On 50S-ImageNet-R, our method achieves **77.51%** accuracy with 9.00% forgetting, substantially outperforming all baselines in accuracy. On 50S-DomainNet, our method reaches **82.16%** with 9.15% forgetting. Under the most extreme 100-task setting, our method attains **76.70%** and **80.84%** on ImageNet-R and DomainNet respectively, outperforming VPT-CPG by 12.07% and 20.03%. The graceful degradation from 10 to 100 tasks highlights our scalability: accuracy drops by only 3.80% on ImageNet-R and 3.84% on DomainNet, contrasting sharply with VPT-CPG's steep decline of 14.00% and 22.40%, confirming that coverage-based dynamic expansion effectively accommodates ever-growing task sequences.

Generalization Across Pre-training Paradigms. To verify that our sparse retrieval memory mechanism is not rigidly overfitted to a specific pre-training regime (i.e., supervised IN-21K), we evaluate our method using backbones pre-trained with self-supervised learning frameworks: **DINO-1k** [2] and **iBOT-1k** [50]. With DINO-1k, our method attains 71.15% on ImageNet-R and 78.71% on DomainNet, and with iBOT-1k it attains 71.69% and 82.38%, respectively. As detailed in Table 7, our method achieves the highest CIL accuracy in three out of four backbone/dataset combinations and remains competitive on the fourth (iBOT-IR, only 0.47% behind VPT-CPG), confirming the generality of our sparse retrieval mechanism across different pre-training paradigms. On DomainNet with iBOT-1k, our method reaches 82.38%, outperforming VPT-CPG by 4.59%, a margin even larger than that observed under IN-21K, suggesting that external memory is particularly effective at compensating for weaker backbone features by learning complementary task-specific representations. Sparse Retrieval Memory thus acts as a general-purpose plug-in that strengthens frozen features regardless of pre-training paradigm, with the largest gains on the more domain-diverse DomainNet benchmark, where the frozen features are less transferable and external memory compensates most.

5 Conclusion

We presented Sparse Retrieval Memory, a parameter-efficient framework for class-incremental learning that augments a frozen pre-trained Vision Transformer with sparsely routed external memory modules. Unlike prior sparse memory approaches requiring costly re-pre-training, our method deploys learnable key-value memory as parallel residual branches alongside frozen attention layers. Through Top-K sparse routing and coverage-based dynamic expansion with zero-initialized values, our architecture significantly mitigates cross-task gradient interference through sparse activation, and adaptively grows memory capacity. Extensive experiments on ImageNet-R, CIFAR-100, and DomainNet demonstrate state-of-the-art performance with robust scalability to sequences of up to 100 tasks, while training fewer than 3% of the backbone parameters and storing zero historical exemplars.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62576282 and 62476223, and in part by the Natural Science Basic Research Program of Shaanxi (Program No. 2026JC-YXQN-237).

References

[1] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. 2020. Dark Experience for General Continual Learning: a Strong, Simple Baseline. In *Conference on Neural Information Processing Systems*.

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. 2021. Emerging Properties in Self-Supervised Vision Transformers. In *IEEE/CVF International Conference on Computer Vision*.

[3] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. 2022. AdaptFormer: Adapting Vision Transformers for Scalable Visual Recognition. In *Conference on Neural Information Processing Systems*.

[4] 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. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*.

[5] Zhanxin Gao, Jun Cen, and Xiaobin Chang. 2024. Consistent Prompting for Rehearsal-Free Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 28463–28473.

[6] Zijian Gao, Wangwang Jia, Xingxing Zhang, Dulan Zhou, Kele Xu, Dawei Feng, Yong Dou, Xinjun Mao, and Huaimin Wang. 2025. Knowledge Memorization and Rumination for Pre-trained Model-based Class-Incremental Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20523–20533.

[7] Jiangpeng He, Zhihao Duan, and Fengqing Zhu. 2025. CL-LoRA: Continual Low-Rank Adaptation for Rehearsal-Free Class-Incremental Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 30534–30544.

[8] 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. 2021. The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization. In *IEEE/CVF International Conference on Computer Vision*. 8320–8329.

[9] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-Efficient Transfer Learning for NLP. In *International Conference on Machine Learning*.

[10] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

[11] Wei-Cheng Huang, Chun-Fu Chen, and Hsiang Hsu. 2024. OVOR: OnePrompt with virtual outlier regularization for rehearsal-free class-incremental learning. In *International Conference on Learning Representations*. 39355–39374.

[12] Paul Janson, Wenxuan Zhang, Rahaf Aljundi, and Mohamed Elhoseiny. 2022. A Simple Baseline that Questions the Use of Pretrained-Models in Continual Learning. In *NeurIPS Workshop*.

[13] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual Prompt Tuning. In *European Conference on Computer Vision*.

[14] Li Jiao, Qiuxia Lai, Yu Li, and Qiang Xu. 2024. Vector Quantization Prompting for Continual Learning. In *Conference on Neural Information Processing Systems*.

[15] Youngeun Kim, Yuhang Li, and Priyadarshini Panda. 2024. One-Stage Prompt-Based Continual Learning. In *European Conference on Computer Vision*, Vol. 15071. 163–179.

[16] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Oriol Vinyals, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences* 114, 13 (2017), 3521–3526.

[17] Alex Krizhevsky. 2009. *Learning Multiple Layers of Features from Tiny Images*. Technical Report. University of Toronto.

[18] Muhammad Rifki Kurniawan, Xiang Song, Zhiheng Ma, Yuhang He, Yihong Gong, Yang Qi, and Xing Wei. 2024. Evolving Parameterized Prompt Memory for Continual Learning. In *AAAI Conference on Artificial Intelligence*. 13301–13309.

[19] Qiwei Li and Jiahuan Zhou. 2025. CAPrompt: Cyclic Prompt Aggregation for Pre-Trained Model Based Class Incremental Learning. In *AAAI Conference on Artificial Intelligence*. 18421–18429.

[20] Zhizhong Li and Derek Hoiem. 2018. Learning without Forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 12 (2018), 2935–2947.

[21] Yan-Shuo Liang and Wu-Jun Li. 2024. InfLoRA: Interference-Free Low-Rank Adaptation for Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23638–23647.

[22] Jessy Lin, Luke Zettlemoyer, Gargi Ghosh, Wen-Tau Yih, Aram Markosyan, Vincent-Pierre Berges, and Barlas Oğuz. 2025. Continual Learning via Sparse Memory Finetuning. *arXiv preprint arXiv:2510.15103* (2025).

[23] Xuan Liu and Xiaobin Chang. 2025. LoRA Subtraction for Drift-Resistant Space in Exemplar-Free Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15308–15318.

[24] Yue Lu, Jie Tan, Shizhou Zhang, Yinghui Xing, Guoqiang Liang, and Yanning Zhang. 2026. Nearest-Neighbor Class Prototype Prompt and Simulated Logits for Continual Learning. *Pattern Recognition* 169 (2026), 111933.

[25] Yue Lu, Shizhou Zhang, De Cheng, Guoqiang Liang, Yinghui Xing, Nannan Wang, and Yanning Zhang. 2025. Training Consistent Mixture-of-Experts-Based Prompt Generator for Continual Learning. In *AAAI Conference on Artificial Intelligence*. 19152–19160.

[26] Yue Lu, Shizhou Zhang, De Cheng, Yinghui Xing, Nannan Wang, Peng Wang, and Yanning Zhang. 2024. Visual Prompt Tuning in Null Space for Continual Learning. In *Conference on Neural Information Processing Systems*. 7878–7901.

[27] Mark D. McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. 2023. RanPAC: Random Projections and Pre-trained Models for Continual Learning. In *Conference on Neural Information Processing Systems*. 12022–12053.

[28] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. 2019. Moment Matching for Multi-Source Domain Adaptation. In *IEEE/CVF International Conference on Computer Vision*.

[29] Jingyang Qiao, Zhizhong Zhang, Xin Tan, Chengwei Chen, Yanyun Qu, Yong Peng, and Yuan Xie. 2024. Prompt Gradient Projection for Continual Learning. In *International Conference on Learning Representations*. 3856–3877.

[30] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. iCaRL: Incremental Classifier and Representation Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2001–2010.

[31] Anurag Roy, Riddhiman Moulick, Vinay Kumar Verma, Saptarshi Ghosh, and Abir Das. 2024. Convolutional Prompting meets Language Models for Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23616–23626.

[32] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations*.

[33] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. 2023. CODA-Prompt: COntinual Decomposed Attention-Based Prompting for Rehearsal-Free Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11909–11919.

[34] Yuwen Tan, Qinhao Zhou, Xiang Xiang, Ke Wang, Yuchuan Wu, and Yongbin Li. 2024. Semantically-Shifted Incremental Adapter-Tuning is A Continual ViTransformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23252–23262.

[35] Yu-Ming Tang, Yi-Xing Peng, and Wei-Shi Zheng. 2023. When Prompt-based Incremental Learning Does Not Meet Strong Pretraining. In *IEEE/CVF International Conference on Computer Vision*. 1706–1716.

[36] Huiyi Wang, Haodong Lu, Lina Yao, and Dong Gong. 2025. Self-Expansion of Pre-trained Models with Mixture of Adapters for Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10087–10098.

[37] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. 2023. Hierarchical Decomposition of Prompt-Based Continual Learning: Rethinking Obscured Sub-optimality. In *Conference on Neural Information Processing Systems*. 69054–69076.

[38] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. 2022. S-Prompts Learning with Pre-trained Transformers: An Occam's Razor for Domain Incremental Learning. In *Conference on Neural Information Processing Systems*. 5682–5695.

[39] Yabin Wang, Zhiheng Ma, Zhiwu Huang, Yaowei Wang, Zhou Su, and Xiaopeng Hong. 2023. Isolation and Impartial Aggregation: A Paradigm of Incremental Learning without Interference. In *AAAI Conference on Artificial Intelligence*. 10209–10217.

[40] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. 2022. DualPrompt: Complementary Prompting for Rehearsal-Free Continual Learning. In *European Conference on Computer Vision*, Vol. 13686. 631–648.

[41] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022. Learning to Prompt for Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 139–149.

[42] Yichen Wu, Hongming Piao, Long-Kai Huang, Renzhen Wang, Wanhua Li, Hanspeter Pfister, Deyu Meng, Kede Ma, and Ying Wei. 2025. SD-LoRA: Scalable Decoupled Low-Rank Adaptation for Class Incremental Learning. In *International Conference on Learning Representations*.

[43] Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. 2024. Boosting Continual Learning of Vision-Language Models via Mixture-of-Experts Adapters. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23219–23230.

[44] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. 2023. SLCA: Slow Learner with Classifier Alignment for Continual Learning on a Pre-trained Model. In *IEEE/CVF International Conference on Computer Vision*. 19091–19101.

[45] Shizhou Zhang, Yue Lu, De Cheng, Yinghui Xing, Nannan Wang, Peng Wang, and Yanning Zhang. 2026. VPT-NSP2++: Importance-Aware Visual Prompt Tuning in Null Space for Continual Learning. *IEEE Transactions on Pattern Analysis and*

Machine Intelligence 48, 4 (2026), 4318–4335.

[46] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. 2020. Maintaining Discrimination and Fairness in Class Incremental Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13208–13217.

[47] Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. 2025. Revisiting Class-Incremental Learning with Pre-Trained Models: Generalizability and Adaptivity are All You Need. *International Journal of Computer Vision* 133, 3 (2025), 1012–1032.

[48] Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. 2024. Expandable Subspace Ensemble for Pre-Trained Model-Based Class-Incremental Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 23554–23564.

[49] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. 2024. Class-Incremental Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46, 8 (2024), 5765–5784.

[50] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. 2022. iBOT: Image BERT Pre-Training with Online Tokenizer. In *International Conference on Learning Representations*.

[51] Hao Zhu, Yifei Zhang, Junhao Dong, and Piotr Koniusz. 2025. BiLoRA: Almost-Orthogonal Parameter Spaces for Continual Learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 25613–25622.

[52] Huiping Zhuang, Yuchen Liu, Run He, Kai Tong, Ziqian Zeng, Cen Chen, Yi Wang, and Lap-Pui Chau. 2024. F-OAL: Forward-only Online Analytic Learning with Fast Training and Low Memory Footprint in Class Incremental Learning. In *Conference on Neural Information Processing Systems*.