Self-Expansion of Pre-trained Models with Mixture of Adapters for Continual Learning

Huiyi Wang[1,2], Haodong Lu[1], Lina Yao[2,1], Dong Gong[1*]
[1]University of New South Wales, [2]CSIRO's Data61
{huiyi.wang, haodong.lu, dong.gong}@unsw.edu.au; lina.yao@data61.csiro.au

Abstract

*Continual learning (CL) aims to continually accumulate knowledge from a non-stationary data stream without catastrophic forgetting of learned knowledge, requiring a balance between stability and adaptability. Relying on the generalizable representation in pre-trained models (PTMs), PTM-based CL methods perform effective continual adaptation on downstream tasks by adding learnable adapters or prompts upon the frozen PTMs. However, many existing PTM-based CL methods use restricted adaptation on a fixed set of these modules to avoid forgetting, suffering from limited CL ability. Periodically adding task-specific modules results in linear model growth rate and impaired knowledge reuse. We propose **S**elf-**E**xpansion of pre-trained models with **M**odularized **A**daptation (SEMA), a novel approach to enhance the control of stability-plasticity balance in PTM-based CL. SEMA automatically decides to reuse or add adapter modules on demand in CL, depending on whether significant distribution shift that cannot be handled is detected at different representation levels. We design modular adapter consisting of a functional adapter and a representation descriptor. The representation descriptors are trained as a distribution shift indicator and used to trigger self-expansion signals. For better composing the adapters, an expandable weighting router is learned jointly for mixture of adapter outputs. SEMA enables better knowledge reuse and sub-linear expansion rate. Extensive experiments demonstrate the effectiveness of the proposed self-expansion method, achieving state-of-the-art performance compared to PTM-based CL methods without memory rehearsal. Code is available at* https://github.com/huiyiwang01/SEMA-CL.

1. Introduction

With the development of deep neural networks, deep learning models have achieved significant success in various fields, such as computer vision [15, 24]. However, real-world scenarios often present learning tasks in a dynamic data stream with non-stationary distributions [50]. Considering the need for efficient model updating and restricted budgets on storage and computation [35], it is not guaranteed to store all the historical data and repeatedly re-train the model. Continual learning (CL) is investigated to learn incrementally and accumulate knowledge efficiently from the non-stationary data stream without *catastrophic forgetting* [46, 54] of previously learned knowledge [14, 59, 65, 71]. It requires CL approaches to achieve a balance between knowledge expansion (*i.e.*, plasticity) and knowledge retention (*i.e.*, stability) [22, 55, 71]. Many CL approaches have been studied to tackle the challenge relying on different strategies, such as experience replay (ER) [7, 8, 77], regularization on parameters or representations [6, 39, 77], and architectures with modularization or isolation [55, 66, 70, 75, 78].

Given the progress in the pre-trained models (PTMs) with reliable representation, recent works explore the potential of using PTMs, such as Vision Transformer (ViT) [15], as the starting point of CL, unlike the "training-from-scratch" paradigm. PTM-based CL approaches [73, 74] usually keep the PTMs frozen to enable stable representation and alleviate forgetting. The PTMs are continually adapted to downstream tasks through parameter-efficient fine-tuning with newly expanded parameters as prompts and/or adapters [13, 51, 68, 73, 74, 83, 90, 91]. On the other hand, some methods enable continual fine-tuning of PTMs on real-world downstream tasks arriving in a streaming manner. Many PTM-based CL approaches mainly add and learn a *fixed* set/pool of prompts [33, 93] or adapters [9] shared by all downstream tasks in the stream [51, 73, 74, 90]. To alleviate forgetting caused by the interference on the newly added parameters, they restrict the parameter updating only on the first task seen in stream [51, 90] or use various regularization on the shared parameters [73, 74]. Their continual adaptation potentials are limited by the fixed and static size of prompt and adapter parameters. Some recent methods expand the PTMs with task-specific parameters to produce input-conditioned prompts [68] or ensemble of adapters [92].

*D. Gong is the corresponding author. This project was partially supported by an ARC DECRA Fellowship (DE230101591) to D. Gong, and PhD scholarship support from UNSW and CSIRO Data61.

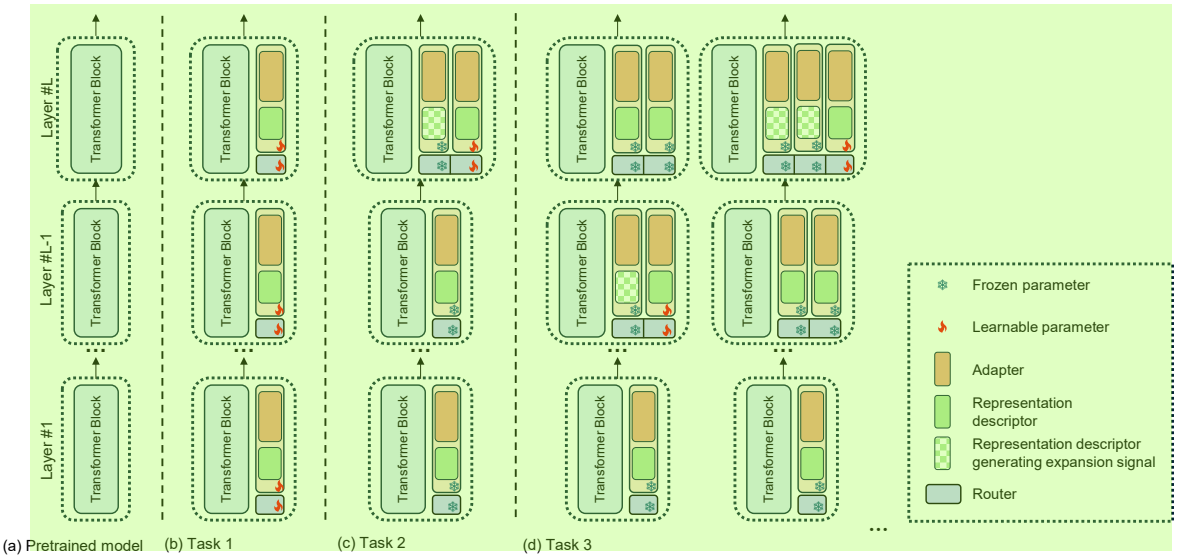


Figure 1. An example of the self-expansion process. (a) The PTM (*i.e.*, ViT) with L transformer layers at the initial point of CL. (b) The first session adaptation – at Task 1, a modular adapter and a (dummy) router is added and trained in each transformer layer. (c) The modular adapters and routers added in the previous step (Task 1) are frozen to alleviate forgetting. When Task 2 arrives, *only* the representation descriptor in the L-th layer detects feature distribution shift (with novel patterns) and generates *expansion signal*. A new module is added and trained in the L-th layer, with the router expanded and updated. (d) At Task 3, new adapter is added at $L-1$-th layer after the expansion signal is firstly generated. In this demo example, the expansion is triggered and produced again in the L-th layer, following the expansion in the $L-1$-th layer. If a task does not trigger expansion signal in any layer (implying no significantly different pattern), expansion would not happen, and existing adapters would be reused. More discussions are in Appendix A.1.

The task-specifically added modules can help reduce the interference but cause a *linearly*-scaled model size (w.r.t. number of tasks) and restrained knowledge sharing and reuse.

Considering that the PTM and the newly added parameters in expansion can provide a *stable* representation and a knowledge *extension* mechanism for CL, respectively, we focus on how to further enhance the control of the *stability-plasticity* balance during continual expansion. Although task-specific expansion of PTMs [68, 92] directly reduces the cross-task conflicts, it causes undesired *linear* scaling of model size and may impair knowledge transfer/reuse [55, 65, 70]. To address these issues, we propose SEMA, a CL approach with **S**elf-**E**xpansion of pre-trained models with **M**odularized **A**daptation. It automatically expands PTMs with modularized adapters on demand and continually learns them to accommodate the distribution shifts without overwriting previously learned knowledge. Unlike existing methods that expand PTMs with a pre-defined fixed-size pool [51, 74, 83, 90] or task-specific components [68, 73, 92], we design modularized adapters to enable SEMA automatically decide *when* and *where* (*i.e.*, which layer) to expand the PTM (*i.e.*, a pre-trained ViT) on demand for tackling new requirements with sufficient and flexible plasticity, as shown in Fig. 1. The model continually learns how to *compose* the learned adapters. With the enhanced knowledge transfer and reuse, SEMA can thus perform better by only expanding the parameter size *sub-linearly*.

We introduce *modular/modularized adapters* that can be identified and reused to solve new tasks, selectively adding and learning a subset of new adapters for unseen knowledge. Specifically, we design the modular adapter as a pair of a functional *adapter* and a *representation descriptor* (RD). The functional adapters produce specific feature representations to adapt to the different requirements of different tasks. The RDs are jointly trained to capture the *feature distribution* relevant to the coupled adapter at corresponding layers, serving as indicators of distribution shift at the representation level of intermediate layers. SEMA can use the representation descriptors for self-expansion – a new modular adapter is added at a specific layer if and only if all the representation descriptors indicate the input feature as a unseen pattern; otherwise, the existing frozen adapters are reused, resulting in *sub-linear* expansion. They can be implemented as a model with density estimation or novelty detection ability, such as autoencoder (AE) [27] or variational autoencoder (VAE) [38]. The module expansion at each layer can happen flexibly to supplement existing representation space, leading to sufficient plasticity. The on-demand expansion strategy strengthens the knowledge transfer and reuse, compared to the task-specific expansion [68, 92]. For example, cat images and dog images have more shared features than food images; the SEMA model trained only on cat images tends to expand more new adapters when training on food images than on dog images. To effectively *compose* the adapters, we design an *expandable weighting router* to produce layer-wise weighted mixture of the adapters in a form of mixture

of experts (MoE), which are expanded and learned in the self-expansion process. Despite the RDs may be used for adapter assignment by hard selection, the learned soft mixture router can perform more effectively (Appendix C.3). We summarize our contributions as follows:

- We propose a novel continual learning approach via self-expansion of PTMs with modularized adapters, *i.e*. SEMA. In CL, it automatically determines the expansion necessity and location for new adapters, adding them at specific layers to accommodate new patterns in samples. The model enhances the control of stability-plasticity trade-off through adapter reuse and flexible expansion performed only on demand. SEMA enables *sub-linear* expansion and operates without the need for rehearsal.
- To achieve SEMA, we introduce modular adapters comprising a functional adapter and a representation descriptor. The representation descriptor maintains the distribution of pertinent input features, serving as a local novel pattern detector for expansion during training. The expandable weighting router is maintained simultaneously for *composing* the adapters via weighted mixture.
- Extensive experiments are conducted to validate the effectiveness and analyze the behavior of the proposed method, which demonstrates the model's ability on alleviating forgetting and knowledge transfer as well as the plausibility of the automated process.

2. Related Work

Continual Learning (CL). The mainstream taxonomy classifies continual learning methods into three categories: replay-based methods, regularization-based methods and architecture-based methods [14, 71]. Replay-based methods aim to alleviate catastrophic forgetting by retaining a memory buffer to store the information from old tasks for future replay [6, 8, 48, 59]. With simple intuition and effectiveness in preventing forgetting, these methods are limited by the size of the memory buffer and may also raise privacy concerns. An alternative approach is to implicitly maintain a generative model for producing pseudo-samples with similar distribution to old classes [11, 37, 60, 61, 67]. Regularization-based methods penalize significant changes to important parameters for seen tasks [2, 4, 39, 53, 84, 85], or consolidate the knowledge learnt from previous tasks with knowledge distillation [28, 41, 46, 88]. Instead of using all available parameters for all tasks, architecture-based methods allocate a subset of parameters dedicated to each task, which can be performed with task masking [36, 49, 66, 75] or dynamic architecture [3, 31, 43, 44, 55, 70, 78–81]. These methods tend to achieve optimal performance with less forgetting as isolating parameters and growing capacity for novel tasks reduce task interference during training, however, they are mostly restricted to simple applications due to the complex model design.

Parameter-Efficient Fine-Tuning (PEFT). Parameter-efficient fine-tuning methods train a small set of additional parameters rather than the entire pre-trained model, which reduces the demands placed upon computational resources. Prompt tuning modifies input tokens/prefixes via learnable prompts [33, 45]. LoRA [30] injects low-rank matrices to approximate weight updates and avoids additional inference latency via re-parameterization, which has been further utilized as experts with mixture modeling in recent works [16, 21, 72, 76]. Adapters introduced by [29], along with its variants [9, 34], insert lightweight learnable modules into the transformer. To enhance the efficacy of adapter learning, [23] investigates different insertion forms, and [12, 57, 63] explores the potential of adapter compositions.

PTM-based CL. Recent works adopt PTMs, such as ViT and CLIP, as the backbone in the CL system to exploit its robust representational ability and enable further adaptation on downstream tasks [32, 62, 87, 89]. PTM can serve as a feature extractor for prototypes, which can be used for classification with distance measurement [51, 52, 56, 90]. PEFT techniques are also widely used to adapt PTMs in CL, including adaptation and prompting. L2P [74] and DualPrompt [73] apply a pool of prompts in CL through visual prompt tuning [33]. The prompt learning process is further improved by [68] with an attention mechanism and input-conditioned weights. ConvPrompt [62] adds parameter per task using linguistic knowledge from a large language model. Similar to prompt tuning in CL, some works also explore the use of a fixed set of adapters [13, 17, 82] or task-oriented expansion [47, 92] for better transfer of ViT to downstream CL tasks. [19] builds a unified framework incorporating both prompt and adapter-based methods. [10] adds experts in the pre-training of large language models (LLMs).

3. Methodology

3.1. Problem Definition

Continual learning constructs a scenario where the model is required to learn from sequentially arriving tasks [14]. Consider a sequence of T tasks $(\mathcal{D}^1, \mathcal{D}^2, ..., \mathcal{D}^T)$ with distribution shift, where $\mathcal{D}^t = \{(x_i^t, y_i^t)\}_{i=1}^{n_t}$ is the dataset containing n_t data samples for the t-th task. Only the training samples from $\mathcal{D}^t$ are accessible while seeing the t-th task [74], if without additional ER process [8]. In a typical class-incremental learning (CIL) scenario [14], the classes in different tasks are non-overlapping, specifically, with the label space of the t-th task denoted by Y_t, $Y_t \cap Y_{t'} = \emptyset$ for $t \neq t'$. Let $F_\theta : X \rightarrow Y$ (with X and Y denoting the domain of input and label) be a model parameterized with θ. The goal of CL is to learn one model F_θ that can minimize the objective on each task t in the stream: $\mathbb{E}_{(x,y)\in D^t}\mathcal{L}_{\text{CE}}(F_\theta(x), y)$, where $\mathcal{L}_{\text{CE}}(\cdot, \cdot)$ denotes the cross entropy loss in CIL.

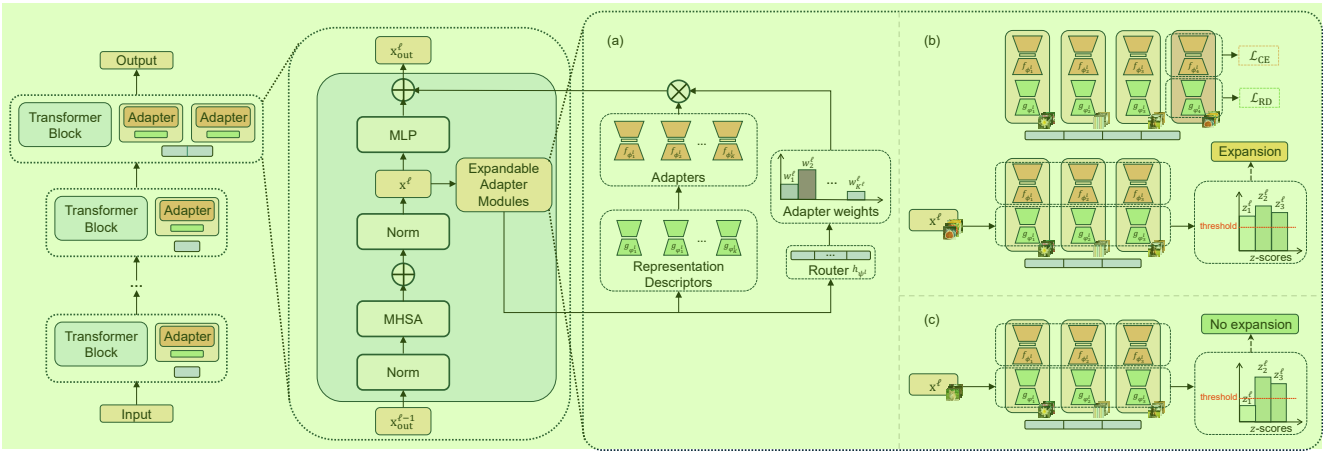


Figure 2. Overview of the model architecture. (a) shows the structure of expandable adapter modules with adapters, RDs and router. (b) shows the scenario where expansion is triggered by representations with distribution different to previous tasks, estimated by RD. RDs are trained to align with the feature distribution of the corresponding task via only $\mathcal{L}_{\text{RD}}$, unaffected by gradients from the classification loss. (c) shows the scenario where incoming distribution can be handled by previously added modules, resulting in no expansion and adapter reuse.

3.2. Overview

We propose a PTM-based CL approach (*i.e.*, SEMA) with a self-expansion mechanism to automatically add modularized adapters at arbitrary layers of the PTM (*i.e.*, a pre-trained ViT with frozen parameters) on demand for handling automatically detected novel patterns in CL task stream, as shown in Fig. 1 and 2. The proposed method simultaneously learns a weighted mixture router for composing the adapters for different inputs. The design enhances the balance of knowledge transfer/reuse and plasticity for handling novelty, with only *sub-linear* expansion rate [5, 55].

To achieve the modularized design of SEMA, we introduce the *modular adapters* containing a pair of functional adapter $f_\phi(\cdot)$ and representation descriptor $g_\varphi(\cdot)$, as defined in Sec. 3.3. Each added functional adapter works as a branch of a specific layer of the pre-trained transformer; and the representation descriptor indicates the feature distribution that can be handled by the paired $f_\phi(\cdot)$. In CL, when new tasks arrive, $g_\varphi(\cdot)$'s of the already-added adapters are used to detect novel feature patterns layer-by-layer. Only when novel pattern (*i.e.*, representation-level distribution shifts) are detected, new adapters, *i.e.*, pairs of $(f_\phi(\cdot), g_\varphi(\cdot))$, are added and trained. After trained sufficiently, the adapters are kept frozen to alleviate forgetting and can be reused in future tasks. The details of the *self-expansion strategy* are in Sec. 3.6. At each layer of the PTM, an *expandable weighting router* is continually maintained and updated for composing the adapters via weighted mixture, as introduced in Sec. 3.4. When no adapters are added, the existing frozen adapters are retrieved and reused.

3.3. Representation-Aware Modular Adapter

The modular adapter $(f_\phi(\cdot), g_\varphi(\cdot))$ is designed as a pair of *functional adapter* $f_\phi(\cdot)$ and a *representation descriptor* $g_\varphi(\cdot)$, which enables the module to be aware of the distribution of the local representation. One or more adapters can be added at arbitrary blocks/layers of the transformer.

Functional adapter. In a (pre-trained) ViT, there are L layers of transformer blocks, where each of them mainly contains a multi-head self-attention (MHSA) module and a multi-layer perceptron (MLP) module [15], as shown in Fig. 2. We keep all the parameters in the ViT frozen and perform adaptation through the learnable parameters in the continually added adapters. As a commonly used solution [9, 90], the functional adapter with learnable parameters is added as a side branch of the MLP in any layer of the ViT.

Let $\mathbf{x}^l \in \mathbb{R}^d$ denote the feature input of the MLP at l-th layer/block of ViT. In the proposed method, there can be different numbers (*i.e.*, K^l) of adapters added at each layer through the self-expansion process. The k-th functional adapter at l-th layer is denoted as $f_{\phi_k^l}(\cdot)$. Each $f_{\phi_k^l}(\cdot)$ takes $\mathbf{x}^l$ as input to bridge the representation gap between the pre-trained model and the downstream tasks. By default, we implement $f_{\phi_k^l}(\cdot)$ as a lightweight adapter [9] containing a down-projection layer with parameters $\mathbf{W}^l_{\text{down},k} \in \mathbb{R}^{d\times r}$, an up-projection layer with parameters $\mathbf{W}^l_{\text{up},k} \in \mathbb{R}^{r\times d}$, and a non-linear ReLU activation [1] in between. By taking $\mathbf{x}^l$ as input, the output of each functional adapter is formulated as

$$f_{\phi_k^l}(\mathbf{x}^l) = \text{ReLU}(\mathbf{x}^l \cdot \mathbf{W}^l_{\text{down},k}) \cdot \mathbf{W}^l_{\text{up},k}, \quad (1)$$

where $\phi_k^l \equiv \{\mathbf{W}^l_{\text{up},k}, \mathbf{W}^l_{\text{down},k}\}$ and $\mathbf{x}^l$ is treated as row vector for notation simplicity. If there is only one adapter at the l-th layer (*i.e.*, $K^l = 1$), the output representation of the MLP is adjusted as $\mathbf{x}^l_{\text{out}} = \text{MLP}(\mathbf{x}^l) + f_{\phi_k^l}(\mathbf{x}^l)$. SEMA can continually expand the model with more than one adapters if needed. The number of adapters at each layer is automatically determined on demand, with a rate that is sub-linear w.r.t. number of tasks. Although similar adapter formulation

have been used to handle CL, they only perform adaptation on the first task using only one adapter [51, 90] or periodically expand the PTM using task-specific adapters *linearly* [92]. In addition to Eq. 1, the functional adapters can also be implemented as other forms, such as LoRA [30], as discussed in Sec. 4.3.

Representation descriptor. The representation descriptor (RD) $g_{\varphi_k^l}(\cdot)$ is paired with the functional descriptor $f_{\phi_k^l}(\cdot)$ to capture the characteristics of the local representation. It is designed and trained to indicate what kind of input representation can be handled by the corresponding functional adapter at each specific layer. Representation descriptors can be implemented as any model with density estimation or novelty detection ability. For simplicity, we implement them as AE [27], containing an encoder and a decoder. When a new pair of modular adapter is added at layer l, the RD $g_{\varphi_k^l}(\cdot)$ is trained by minimizing the reconstruction loss on all the features fed to $f_{\phi_k^l}(\cdot)$, *i.e.*, $\mathcal{X}_k^l$:

$$\mathcal{L}_{\text{RD},k}^l(x) = \sum\nolimits_{\mathbf{x}\in\mathcal{X}_k^l} ||\mathbf{x} - g_{\varphi_k^l}(\mathbf{x})||_2^2. \tag{2}$$

In our expansion strategy (in Sec. 3.6), when a new task t arrives, at each l-th layer, if all existing RDs detect significantly novel distributions (based on the z-score of reconstruction errors), the expansion signal is triggered. $f_{\phi_k^l}(\cdot)$ and $g_{\varphi_k^l}(\cdot)$ are trained on this task t and then kept frozen in the future. $\mathcal{X}_k^l$ represents the input feature $\mathbf{x}^l$ of all the samples in this new expansion-triggering task t.

3.4. Expandable Weighting Router for Mixture Usage of Adapters

By definition, the representation descriptor can be used to compose the adapters, as in similar modular networks. However, it heavily relies on the statistics of similar inputs in a batch [55] and can be unreliable for individual inputs. We thus directly maintain and learn an *expandable weighting router* for a weighted mixture of the functional adapters.

For any l-th layer with K^l adapters, the routing function is defined as $h_{\psi^l}(\cdot) : \mathbb{R}^d \to \mathbb{R}^{K^l}$. Similar to [16], we implement $h_{\psi^l}(\cdot)$ as a linear mapping function followed by a softmax operation $\mathbf{w}^l = h_{\psi^l}(\mathbf{x}^l) \equiv \text{softmax}(\mathbf{x}^l \cdot \mathbf{W}_{\text{mix}}^l)$, where $\mathbf{W}_{\text{mix}}^l \in \mathbb{R}^{d\times K^l}$ is the parameter of ψ^l. As shown in Fig. 2, the weights $\mathbf{w}^l \in \mathbb{R}^{K^l}$ can produce the mixture of the added functional adapters to produce the output representation of the MLP in the transformer:

$$\mathbf{x}_{\text{out}}^l = \text{MLP}(\mathbf{x}^l) + \sum\nolimits_{k=1}^{K^l} w_k^l \cdot f_{\phi_k^l}(\mathbf{x}^l). \tag{3}$$

When new adapter is added at any layer l, the router $h_{\psi^l}(\cdot)$, *i.e.*, $\mathbf{W}_{\text{mix}}^l$, is expanded for producing weights with one more dimension. The expanded router is trained together with the added adapters. While expanding the router, the parameters corresponding to the existing adapters remain frozen and only the newly added ones (*i.e.*, a newly added column in $\mathbf{W}_{\text{mix}}^l$) are trained. This approach, similar to the common practice for training classification heads in CL [47, 68], controls and restricts forgetting in the expandable router (shown in Fig. 5), though it cannot fully eliminate it.

3.5. Continual Learning Objective of SEMA

In SEMA, the model $F_\theta(\cdot)$ for solving the tasks consists of learnable parameters from the functional adapters and router with learnable parameters, *i.e.*, $\{\phi_k^l\}$ and $\{\psi^l\}$. The learnable parameters are dynamically added and learned. The representation descriptors are learned jointly for maintaining a state of the local representation. The overall objective in SEMA optimizes all these parameters:

$$\min_{\{\phi_k^l\},\{\psi^l\},\{\varphi_k^l\}} \sum\nolimits_{t=1}^{T} \mathbb{E}_{(x,y)\in D^t} \Big[\mathcal{L}_{\text{CE}}(F_{\{\phi_k^l\},\{\psi^l\}}(x),\ y) + \sum\nolimits_{l=1}^{L}\sum\nolimits_{k=1}^{K^l} \mathcal{L}_{\text{RD},k}^l(x;\varphi_k^l)\Big]. \tag{4}$$

Learning of modular adapters is executed only when new modules are added. The learned modules are kept frozen to prevent forgetting. Optimization of RDs can be parallel to other parameters. If no module is added in a specific task due to no significant pattern being identified by RDs, the existing modules can be reused without training.

3.6. Self-Expansion Strategy

The RDs provide the capacity to decide when and where to expand the model. We designed a more specific strategy to achieve the reliable self-expansion in the CL task stream.

Task-oriented expansion. The expansion may occur at any time as new samples are seen during training. To incorporate the task identification prior knowledge in CL, especially CIL, we improve parameter efficiency and expansion stability with task-oriented expansion. We restrict the addition to at most one adapter per layer for each task. When a new task t arrives, the method scans all samples in the *first epoch* to decide whether to expand the model. If the expansion signal is triggered, only one adapter is added and then trained for the whole task; otherwise, the task t data can reuse learned modules and the learning process moves to the next task.

z-score based expansion signal. When scanning through the new task data, an expansion signal at layer l is triggered when significantly new patterns are identified. It reflects that a $\mathbf{x}^l$ is out of the scope of all RDs, *i.e.*, reconstruction error is high with each $g_{\varphi_k^l}(\mathbf{x})$ [20], as illustrated in Fig. 4. However, it is impractical to directly use reconstruction error due to the perturbation and heterogeneous characteristics of each task and adapter. We thus compute and maintain the running statistics μ_k^l and standard deviation σ_k^l of reconstruction error on all relevant inputs used in training. Given any x^l in the

Method	CIFAR-100		5-Task IN-R		10-Task IN-R		20-Task IN-R		ImageNet-A		VTAB	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
FT Adapter	47.88	30.9	53.91	41.23	45.31	30.93	38.51	24.22	29.78	17.64	59.98	43.50
L2P	84.77	77.87	77.40	73.59	66.97	62.72	70.67	62.90	47.16	38.48	81.19	80.83
DualPrompt	86.60	80.43	76.39	72.29	72.83	66.75	62.33	61.97	59.54	50.23	82.89	79.79
CODA-P	**91.55**	86.11	81.63	76.98	81.11	75.25	75.00	70.02	47.29	35.02	79.88	81.58
SimpleCIL	82.31	76.21	65.83	61.31	67.09	61.35	67.59	61.35	60.05	49.24	85.29	83.61
ADAM	90.55	85.62	79.91	74.25	79.11	73.15	75.84	69.10	60.15	49.24	85.29	83.61
InfLoRA	90.51	85.05	78.58	72.58	81.39	75.32	78.87	72.60	59.71	46.21	88.90	87.63
SEMA	91.37	**86.98**	**84.75**	**79.78**	**83.56**	**78.00**	**81.75**	**74.53**	**64.53**	**53.32**	**91.26**	**89.64**

Table 1. Comparison with ViT-based CL methods in CIL. All models adopt ViT-B/16-IN1K as the backbone.

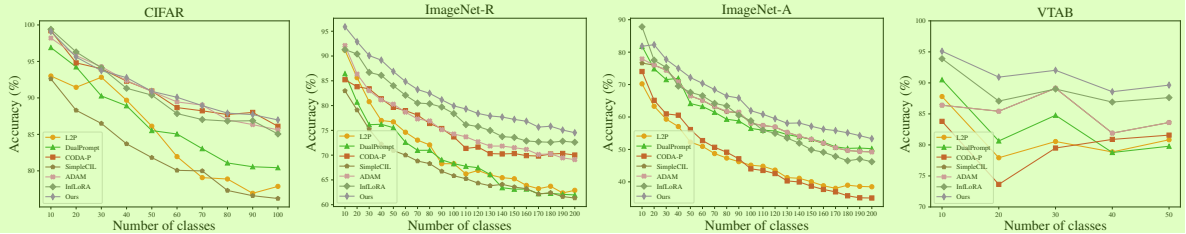


Figure 3. Incremental performance of different methods on class-incremental learning benchmarks.

scanning process for a future task, the z-score corresponding to each existing RD can be calculated as $z_k^l = (r_k^l - \mu_k^l)/\sigma_k^l$ with r_k^l as reconstruction error. If all z_k^l's for $k = 1, ..., K^l$ are larger than a threshold, the expansion signal is triggered. Considering that the z-score has normalized out perturbation and scale, the process can be very robust to the threshold setting, as shown in Sec. 4.3.

Multi-layer expansion. We facilitate self-expansion across multiple layers through distinct decision processes. Upon encountering a new task, self-expansion operations are executed sequentially from shallow layers to deeper layers. As new adapters are introduced at shallow levels, training ensures representations are aligned accordingly. Subsequently, the model determines whether to continue expanding into subsequent layers. The adaptable multi-layer expansion facilitates the accommodation of various distribution shifts and enables flexible inter-class knowledge sharing [18, 42].

4. Experiments

4.1. Setting and Implementation Details

Datasets. Experiments are conducted on common datasets used for pre-trained ViT-based CIL: CIFAR-100 [40], ImageNet-R (IN-R) [25], ImageNet-A [26] and VTAB [86].

Baselines. We validate our method by comparing with PTM-based rehearsal-free CL approaches using similar backbone (*e.g.*, ViT) and methodology, including fully fine-tuning of the adapter, L2P [74], DualPrompt [73], CODA-P [68], SimpleCIL [90], ADAM with Adapter [90] and InfLoRA [47].

Training details. We use the commonly used ViT-B/16 model [15] weights pre-trained on ImageNet-1K [64] as the PTM weights. We also conducted experiments with other pre-trained weights and left discussions in Appendix C.1. The batch size is set to 32. SGD is used as the optimizer with the initial learning rate set to 0.005 and 0.01 for adapters and RDs, respectively, decaying with cosine annealing. The hidden dimension of adapter is 16. In experiments, by default, we enable self-expansion in the last three transformer layers for simplicity without losing generality.

4.2. Experimental Results

We validate the proposed method by comparing with previous related state-of-the-art methods and reporting the average accuracy of all tasks $\mathcal{A}_N$ [7] and average incremental accuracy $\bar{\mathcal{A}}$ [59] metrics in Tab. 1. It shows that our method performs better than other related methods in terms of the average accuracy at the last step $\mathcal{A}_N$, which reflects the final goal of CL. Fig. 3 shows the variation in accuracy during the continual learning process.It shows the consistently superior performance of SEMA in the process. Although most previous approaches exhibit strong performance on CIFAR-100, the proposed methods shows more improvements on datasets containing adversarial samples similar to those found in ImageNet, due to its better stability-plasticity balance.

4.3. Ablation Studies and Analyses

Ablation studies on module expansion and adapter composing. We conduct ablation studies to demonstrate the effectiveness of the self-expansion process and investigate the influence of different adapter composing strategies, with the results reported in Tab. 2. We first conduct an experiment by removing the self-expansion process and only keeping the first-session adaptation (No Exp.), which is similar to ADAM [90] with slight difference on implementation. The

results show that the self-expansion can work reliably to continually improve the adaptation results.

Method	ImageNet-A		VTAB	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
SEMA	**64.53**	**53.32**	**91.26**	**89.64**
No Exp.	61.20	49.90	86.21	83.66
Avg. W.	56.88	44.31	90.84	89.14
Rand. W.	62.95	49.77	88.87	85.17
Top-1 Sel.	62.00	50.56	90.83	88.61
Rand. Sel.	61.70	50.36	90.82	88.51
Top-1 Sel. Inf.	61.96	50.36	90.95	88.84

Table 2. Ablation studies on adapter expansion and composing.

To demonstrate the benefits of the weighted mixture routing, we investigate several variants of SEMA with different adapter composing strategies. Firstly, we study two variants with a soft mixture of adapters relying average weighting (Avg. W.) and random weighting (Rand. W.), respectively. Tab. 2 shows that the expandable weighting router learns an effective weighting function. We further study the variants that perform routing by selecting only a single adapter indicated by the highest value from the learned weighting router (Top-1 Sel.) or through random drawing (Rand. Sel.). Additionally, we evaluate SEMA trained with mixture routing, using an inference strategy that selects only the adapter with the highest weight (Top-1 Sel. Inf.). The results show that the weighted soft mixture of the learned adapters works more effectively by encouraging the better usage of the learned adapters. More experiments about adapter composing using representation descriptor are in Appendix C.3.

Analysis on dynamic expansion process. To demonstrate how the representation descriptors are learned and how they work for self-expansion in CL, we visualize the reconstruction error of each AE-based RD corresponding to each sample seen during training, *i.e.*, their representation features at specific layer, in Fig. 4. For more intuitive visualization and simplified experiment, in this analysis, we restrict the automatic self-expansion only to the last layer of transformer. The analysis is conducted on VTAB dataset. In this case shown in Fig. 4, the reconstruction error of each RD decreases and converges after training on the corresponding task, after the RD is added for handling this task. When a new task arrives, the reconstruction errors for the existing RDs are calculated and used to detect novelty. The expansion signal is generated when significantly high reconstruction errors (scaled as z-scores) are detected from all the previous RDs (in Task 2 and 3). In Task 4 and 5, all samples can be well covered by at least one previous RD, which implies no significant distribution shift is detected and results in no expansion. Note that the z-score (*i.e.*, a normalized version of reconstruction error) is used for expansion in SEMA.

Analysis on adapter usage. Fig. 5 demonstrates the average adapter usage of each task from VTAB. This analysis is produced by restricting self-expansion to the last layer, as in

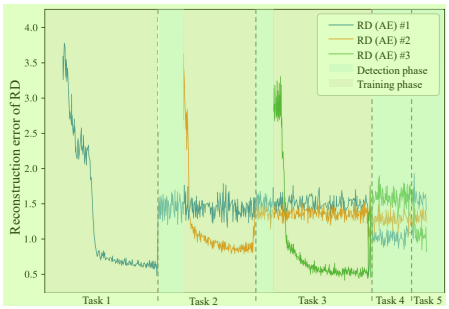


Figure 4. Reconstruction error during training to show the dynamic expansion process. Expansion occurs for Tasks 1, 2, and 3, while no expansion is triggered for Tasks 4 and 5 due to no detected distribution shift.

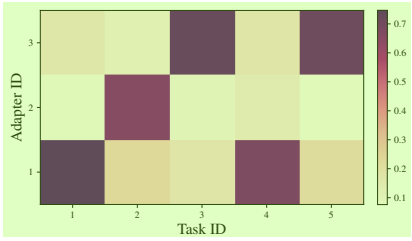
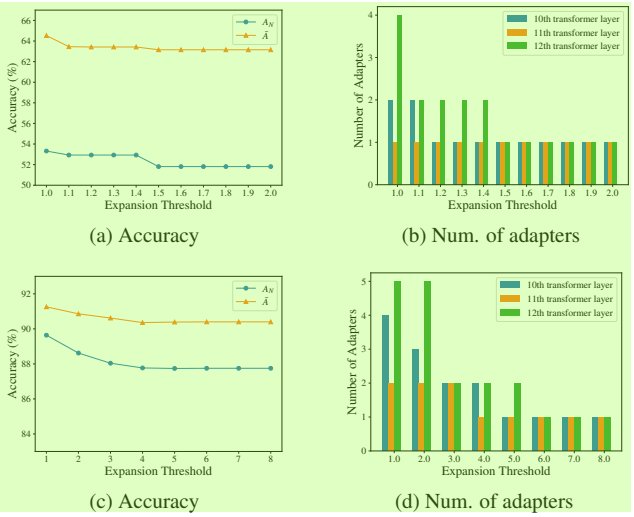


Figure 5. Visualization of adapter usage on VTAB. Adapters 1, 2, and 3 are added and trained on Tasks 1, 2, and 3, respectively. Tasks 4 and 5 primarily reuse Adapters 1 and 3 due to similar feature distributions with Tasks 1 and 3.

Fig. 4. Self-expansion is automatically produced for Task 1, 2 and 3. For tasks that triggered expansion, the adapters used are primarily those they were trained with, as shown in the figure. Task 4 and 5 share a similar selection pattern with the tasks they are similar with (Task 1 and 3 respectively), showing that added adapters are effectively reused for new tasks. More details are in Appendix C.3.

Study of expansion threshold. We investigate the impact of the expansion threshold on accuracy and the number of added adapters using ImageNet-A and VTAB. Firstly, the results in Fig. 6 show that the proposed method is not sensitive to the setting of the threshold, benefiting from the z-score-based expansion signal. Fig. 6b and 6d show how the threshold influences the number of added adapters (at each layer), displaying trends consistent with those in Fig. 6a and 6c. Fig. 6a and 6b show that a smaller expansion threshold leads to more frequent expansion, which could boost the performance at some level through more parameters. A threshold that is too large (*e.g.*, values over 1.5) minimizes the chance for expansion, which may lead to insufficient adaptation. In SEMA, a proper expansion threshold within a wide range can lead to a balance between the performance gain and the parameter size.

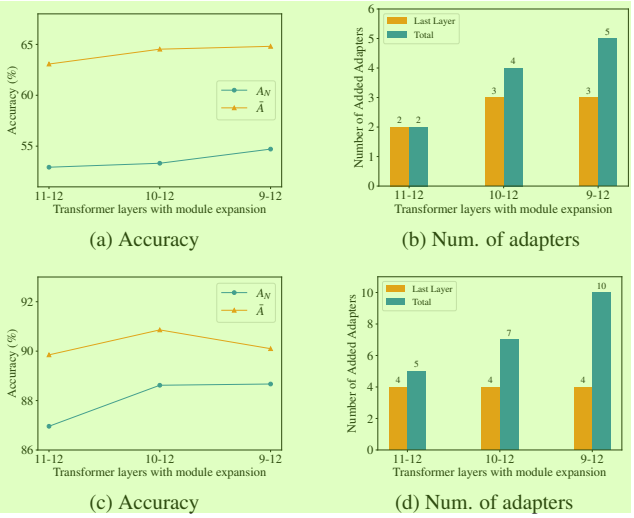
Analysis of multi-layer expansion. In Fig. 7, we explore the effects on accuracy by implementing expansion across varying numbers of layers, ranging from the last 2 layers (#11-#12) to the last 4 layers (#9-#12). Intuitively, allowing expansion in deeper layers enables better adaptation



(a) Accuracy (b) Num. of adapters

(c) Accuracy (d) Num. of adapters

Figure 6. Analysis of the impact of expansion threshold with (a)(b) ImageNet-A and (c)(d) VTAB. (a) and (c) show that SEMA can produce good accuracy stably with slight variation w.r.t. varying expansion threshold. (b) and (d) report how the number of added adapters (on the specific Transformer layers #10, #11, #12) changes with the varying threshold values, corresponding to (a) and (c), respectively. The proposed method is insensitive to the threshold. Adding more adapters may lead to higher accuracy, a proper threshold can achieve a balance between performance and model size.



(a) Accuracy (b) Num. of adapters

(c) Accuracy (d) Num. of adapters

Figure 7. Analysis of the effect of multi-layer expansion, with (a)(b) ImageNet-A and (c)(d) VTAB. By enabling automatic self-expansion on multiple transformer layers, SEMA can achieve better performance than restricting that on a single layer.

to different tasks. However, as shown in Fig. 7b and Fig. 7d, permitting expansion in early transformer layers also increases the overall number of added adapters, without a significant boost in performance as earlier layers tend to behave similarly despite distribution shifts. Also, enforcing addition of too many adapters may cause difficulty in training, especially in early transformer layers.

Method	ImageNet-A		VTAB	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
Adapter[9]	**64.53**	**53.32**	91.26	**89.64**
LoRA[30]	63.50	52.67	**91.85**	88.53
Convpass[34]	63.48	51.74	90.68	88.62

Table 3. Different adapter variants.

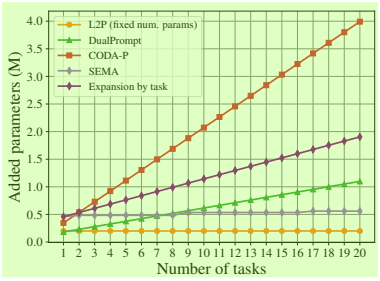


Figure 8. Analysis on added parameters (in Millions) during model deployment on ImageNet-A.

Ablation studies on adapter variants. Apart from Adapter [9], we extend our evaluation to other variants, namely LoRA [30] and Convpass [34]. As shown in Tab. 3, our proposed approach is robust to the choice of adapter methods, showing the broad applicability and effectiveness of our dynamic expansion strategy across different adapter methods.

Sub-linear growth of parameters. In Fig. 8, instead of expanding w.r.t. number of tasks, SEMA adds parameters at a sub-linear rate, showing the efficiency of the self-expansion mechanism. Further analysis is provided in Appendix C.2.

5. Conclusion

In this paper, we propose a novel self-expandable modularized adaptation approach for continual learning. SEMA learns to reuse and add modules in an automated manner without memory rehearsal. We incorporate an efficient expansion strategy with detection for feature distribution shifts in different layers of transformer-based models, successfully mitigating the forgetting problem of jointly using the fixed set of parameters. Experimental results demonstrate the outstanding performance of SEMA to datasets with different levels of distribution shifts.

Limitations and future work. We perform the task-oriented expansion at most once per layer for each task considering the CIL characteristics and parameter efficiency. The design can be more flexible to enable fully online dynamic expansion, which could open possibility in better adaptation for data with intra-task diversity and enable online CL. Moreover, the expansion of SEMA is based on the distribution shift detection ability from RDs, which could be further enhanced by elevating the optimization of RDs and expansion protocol to a meta level with a closed loop.

References

[1] Abien Fred Agarap. Deep learning using rectified linear units (relu). arxiv 2018. *arXiv preprint arXiv:1803.08375*, 1803. 4

[2] Hongjoon Ahn, Sungmin Cha, Donggyu Lee, and Taesup Moon. Uncertainty-based continual learning with adaptive regularization. *Advances in neural information processing systems*, 32, 2019. 3

[3] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3366–3375, 2017. 3

[4] 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. 3

[5] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 39–48, 2016. 4

[6] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020. 1, 3

[7] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with A-GEM. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 1, 6

[8] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. *arXiv preprint arXiv:1902.10486*, 2019. 1, 3

[9] Shoufa Chen, Chongjian Ge, Zhan Tong, Jiangliu Wang, Yibing Song, Jue Wang, and Ping Luo. Adaptformer: Adapting vision transformers for scalable visual recognition. *Advances in Neural Information Processing Systems*, 35:16664–16678, 2022. 1, 3, 4, 8

[10] Wuyang Chen, Yanqi Zhou, Nan Du, Yanping Huang, James Laudon, Zhifeng Chen, and Claire Cui. Lifelong language pretraining with distribution-specialized experts. In *International Conference on Machine Learning*, pages 5383–5395. PMLR, 2023. 3

[11] WU Chenshen, L Herranz, LIU Xialei, et al. Memory replay gans: Learning to generate images from new categories without forgetting [c]. In *The 32nd International Conference on Neural Information Processing Systems, Montréal, Canada*, pages 5966–5976, 2018. 3

[12] Alexandra Chronopoulou, Matthew E. Peters, Alexander Fraser, and Jesse Dodge. Adaptersoup: Weight averaging to improve generalization of pretrained language models. In *Findings of the Association for Computational Linguistics: EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 2009–2018. Association for Computational Linguistics, 2023. 3

[13] Yawen Cui, Zitong Yu, Rizhao Cai, Xun Wang, Alex C Kot, and Li Liu. Generalized few-shot continual learning with contrastive mixture of adapters. *arXiv preprint arXiv:2302.05936*, 2023. 1, 3

[14] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021. 1, 3

[15] 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. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 1, 4, 6

[16] Shihan Dou, Enyu Zhou, Yan Liu, Songyang Gao, Jun Zhao, Wei Shen, Yuhao Zhou, Zhiheng Xi, Xiao Wang, Xiaoran Fan, et al. Loramoe: Revolutionizing mixture of experts for maintaining world knowledge in language model alignment. *arXiv preprint arXiv:2312.09979*, 2023. 3, 5

[17] Beyza Ermis, Giovanni Zappella, Martin Wistuba, Aditya Rawal, and Cedric Archambeau. Memory efficient continual learning with transformers. *Advances in Neural Information Processing Systems*, 35:10629–10642, 2022. 3

[18] Chongyang Gao, Kezhen Chen, Jinmeng Rao, Baochen Sun, Ruibo Liu, Daiyi Peng, Yawen Zhang, Xiaoyuan Guo, Jie Yang, and VS Subrahmanian. Higher layers need more lora experts, 2024. 6

[19] 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. *arXiv preprint arXiv:2303.10070*, 2023. 3

[20] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1705–1714, 2019. 5

[21] Yunhao Gou, Zhili Liu, Kai Chen, Lanqing Hong, Hang Xu, Aoxue Li, Dit-Yan Yeung, James T Kwok, and Yu Zhang. Mixture of cluster-conditional lora experts for vision-language instruction tuning. *arXiv preprint arXiv:2312.12379*, 2023. 3

[22] Raia Hadsell, Dushyant Rao, Andrei A Rusu, and Razvan Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020. 1

[23] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. 3

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[25] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, pages 8340–8349, 2021. 6

[26] 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*, pages 15262–15271, 2021. 6

[27] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313 (5786):504–507, 2006. 2, 5

[28] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Lifelong learning via progressive distillation and retrospection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 437–452, 2018. 3

[29] 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*, pages 2790–2799. PMLR, 2019. 3

[30] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. 3, 5, 8

[31] Ching-Yi Hung, Cheng-Hao Tu, Cheng-En Wu, Chien-Hung Chen, Yi-Ming Chan, and Chu-Song Chen. Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[32] Saurav Jha, Dong Gong, and Lina Yao. CLAP4CLIP: Continual learning with probabilistic finetuning for vision-language models. In *Thirty-eighth Conference on Neural Information Processing Systems*, 2024. 3

[33] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pages 709–727. Springer, 2022. 1, 3

[34] Shibo Jie and Zhi-Hong Deng. Convolutional bypasses are better vision transformer adapters. *arXiv preprint arXiv:2207.07039*, 2022. 3, 8

[35] Daniel Justus, John Brennan, Stephen Bonner, and Andrew Stephen McGough. Predicting the computational cost of deep learning models. In *2018 IEEE international conference on big data (Big Data)*, pages 3873–3882. IEEE, 2018. 1

[36] Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22443–22456, 2021. 3

[37] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *arXiv preprint arXiv:1711.10563*, 2017. 3

[38] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. 2

[39] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526, 2017. 1, 3

[40] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009. 6

[41] Kibok Lee, Kimin Lee, Jinwoo Shin, and Honglak Lee. Overcoming catastrophic forgetting with unlabeled data in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 312–321, 2019. 3

[42] Yoonho Lee, Annie S. Chen, Fahim Tajwar, Ananya Kumar, Huaxiu Yao, Percy Liang, and Chelsea Finn. Surgical fine-tuning improves adaptation to distribution shifts. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. 6

[43] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 3925–3934. PMLR, 2019. 3

[44] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, pages 3925–3934. PMLR, 2019. 3

[45] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021. 3

[46] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1, 3

[47] Yan-Shuo Liang and Wu-Jun Li. Inflora: Interference-free low-rank adaptation for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23638–23647, 2024. 3, 5, 6

[48] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. *Advances in neural information processing systems*, 30, 2017. 3

[49] Arun Mallya, Dillon Davis, and Svetlana Lazebnik. Piggyback: Adapting a single network to multiple tasks by learning to mask weights. In *Proceedings of the European conference on computer vision (ECCV)*, pages 67–82, 2018. 3

[50] 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

[51] Mark McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton van den Hengel. RanPAC: Random projections and pre-trained models for continual learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. 1, 2, 3, 5, 6

[52] Lu Mi, Hao Wang, Yonglong Tian, Hao He, and Nir N Shavit. Training-free uncertainty estimation for dense regression: Sensitivity as a surrogate. In *Proceedings of the AAAI Con-*

ference on Artificial Intelligence, pages 10042–10050, 2022. 3

[53] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017. 3

[54] Cuong V Nguyen, Alessandro Achille, Michael Lam, Tal Hassner, Vijay Mahadevan, and Stefano Soatto. Toward understanding catastrophic forgetting in continual learning. *arXiv preprint arXiv:1908.01091*, 2019. 1

[55] Oleksiy Ostapenko, Pau Rodriguez, Massimo Caccia, and Laurent Charlin. Continual learning via local module composition. *Advances in Neural Information Processing Systems*, 34:30298–30312, 2021. 1, 2, 3, 4, 5

[56] Francesco Pelosin. Simpler is better: off-the-shelf continual learning through pretrained backbones. *arXiv preprint arXiv:2205.01586*, 2022. 3

[57] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 487–503. Association for Computational Linguistics, 2021. 3

[58] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 7

[59] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 1, 3, 6

[60] Matthew Riemer, Tim Klinger, Djallel Bouneffouf, and Michele Franceschini. Scalable recollections for continual lifelong learning. In *Proceedings of the AAAI conference on artificial intelligence*, pages 1352–1359, 2019. 3

[61] Mohammad Rostami, Soheil Kolouri, and Praveen K Pilly. Complementary learning for overcoming catastrophic forgetting using experience replay. *arXiv preprint arXiv:1903.04566*, 2019. 3

[62] Anurag Roy, Riddhiman Moulick, Vinay K Verma, Saptarshi Ghosh, and Abir Das. Convolutional prompting meets language models for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23616–23626, 2024. 3

[63] Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. Adapterdrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 7930–7946. Association for Computational Linguistics, 2021. 3

[64] 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*, 115:211–252, 2015. 6

[65] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International conference on machine learning*, pages 4528–4537. PMLR, 2018. 1, 2

[66] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *International conference on machine learning*, pages 4548–4557. PMLR, 2018. 1, 3

[67] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017. 3

[68] 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*, pages 11909–11919, 2023. 1, 2, 3, 5, 6

[69] 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. 1

[70] Tom Veniat, Ludovic Denoyer, and Marc'Aurelio Ranzato. Efficient continual learning with modular networks and task-driven priors. *arXiv preprint arXiv:2012.12631*, 2020. 1, 2, 3

[71] 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*, 2024. 1, 3

[72] Yaqing Wang, Sahaj Agarwal, Subhabrata Mukherjee, Xiaodong Liu, Jing Gao, Ahmed Hassan Awadallah, and Jianfeng Gao. AdaMix: Mixture-of-adaptations for parameter-efficient model tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5744–5760, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. 3

[73] 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 *European Conference on Computer Vision*, pages 631–648. Springer, 2022. 1, 2, 3, 6

[74] 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*, pages 139–149, 2022. 1, 2, 3, 6

[75] Mitchell Wortsman, Vivek Ramanujan, Rosanne Liu, Aniruddha Kembhavi, Mohammad Rastegari, Jason Yosinski, and Ali Farhadi. Supermasks in superposition. *Advances in Neural Information Processing Systems*, 33:15173–15184, 2020. 1, 3

[76] Xun Wu, Shaohan Huang, and Furu Wei. MoLE: Mixture of loRA experts. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[77] Qingsen Yan, Dong Gong, Yuhang Liu, Anton van den Hengel, and Javen Qinfeng Shi. Learning bayesian sparse networks with full experience replay for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 109–118, 2022. 1

[78] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2021. 1, 3

[79] Fei Ye and Adrian G Bors. Task-free continual learning via online discrepancy distance learning. *Advances in Neural Information Processing Systems*, 35:23675–23688, 2022.

[80] Fei Ye and Adrian G Bors. Self-evolved dynamic expansion model for task-free continual learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22102–22112, 2023.

[81] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017. 3

[82] Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Ping Hu, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23219–23230, 2024. 3

[83] Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Dong Wang, Huchuan Lu, and You He. Boosting continual learning of vision-language models via mixture-of-experts adapters. In *CVPR*, 2024. 1, 2

[84] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pages 3987–3995. PMLR, 2017. 3

[85] Chen Zeno, Itay Golan, Elad Hoffer, and Daniel Soudry. Task agnostic continual learning using online variational bayes. *arXiv preprint arXiv:1803.10123*, 2018. 3

[86] Xiaohua Zhai, Joan Puigcerver, Alexander Kolesnikov, Pierre Ruyssen, Carlos Riquelme, Mario Lucic, Josip Djolonga, Andre Susano Pinto, Maxim Neumann, Alexey Dosovitskiy, et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019. 6

[87] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. Slca: Slow learner with classifier alignment for continual learning on a pre-trained model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19148–19158, 2023. 3

[88] Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1131–1140, 2020. 3

[89] Zangwei Zheng, Mingyuan Ma, Kai Wang, Ziheng Qin, Xiangyu Yue, and Yang You. Preventing zero-shot transfer degradation in continual learning of vision-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19125–19136, 2023. 3

[90] Da-Wei Zhou, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need, 2023. 1, 2, 3, 4, 5, 6

[91] Da-Wei Zhou, Yuanhan Zhang, Jingyi Ning, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Learning without forgetting for vision-language models, 2023. 1

[92] 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 *CVPR*, 2024. 1, 2, 3, 5, 6

[93] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 1

Self-Expansion of Pre-trained Models with Mixture of Adapters for Continual Learning

Supplementary Material

A. More Details about SEMA

A.1. More Details of SEMA Training

We discuss more details of SEMA training using a more detailed example in Fig. 9, which contains more details (*i.e.*, different types of cases and the distribution shift detection/scanning procedure) compared to that in Fig. 1. At the start of the training, each transformer block at different layers is equipped with one adapter module containing one adapter and one representation descriptor, as well as an expandable weighting router, as shown in Fig. 9 (b). They are added as the default adapters and trained on the first task. After the first task, for the incoming new tasks, SEMA monitors the representations of each batch of samples at each layer with the AE-based representation descriptor. As discussed in Sec. 3.6, the distribution shift is measured using the z-score computed from the mean and standard deviation of reconstruction errors stored in a buffer. This buffer is implemented as a fixed stack of 500 samples, maintaining reconstruction errors from the most recent batches. New adapters are added if a significant enough representation/distribution shift is detected at each layer. Adding the adapters expands the model's representation ability for handling the new patterns. As introduced in the main paper, SEMA performs task-oriented expansion (in the class-incremental learning setting given the task boundary in training), adding at most one adapter per layer. As shown in Fig. 1 and Fig. 9, the detection and expansion operation starts from the transformer layers closest to the input. Once a significant distribution shift is detected at a specific layer that could not be handled by *all* existing adapters (detected by RDs), an expansion signal is triggered in this layer/block. A new adapter module will be added to the layer where the expansion signal is triggered, along with an expansion of the weighting router, and activated for training. After sufficient training, the detection phase will be restarted for the later layers. If no distribution shift is reported for a task in any layers, as shown in Fig. 9 (c), no adapter module will be added, and no training of adapters is required for this task.

B. More Details about Implementation and Evaluation

B.1. Details of Datasets

CIFAR-100 contains 100 classes with 500 training samples and 100 testing samples per class.

ImageNet-R contains renditions of 200 ImageNet classes, which is a challenging CL benchmark introduced by with great intra-class diversity.

ImageNet-A contains real-world images filtered from ImageNet in an adversarial manner which are hard to be classified by models pre-trained with ImageNet.

VTAB consists of 50 classes from 5 domains with 10 classes from each domain.

To construct class-incremental setting, for results reported in Tab. 1, CIFAR-100, ImageNet-A and VTAB are split in a manner where each task consists of 10 distinct classes. ImageNet-R is reported with results for 5 tasks (40 classes per task), 10 tasks (20 classes per task), and 20 tasks (10 classes per task).

B.2. Implementations of Compared Methods

For SimpleCIL and ADAM, we use the official implementation at https://github.com/zhoudw-zdw/RevisitingCIL. For InfLoRA, we use the official implementation at https://github.com/liangyanshuo/InfLoRA. For other prompting methods, namely L2P, DualPrompt and CODA-P, we adopt the open-source implementation from PILOT toolbox [69], available at https://github.com/sun-hailong/LAMDA-PILOT. In our experiments, we adhere to the hyperparameter configurations as specified in the original publications for each of the compared methods, We use ViT-B/16-IN1K as the backbone with the same data shuffling as [90] for all methods.

B.3. Details on Evaluation Metrics

Denote the accuracy of the i-th task after training on the N-th task as $\mathcal{A}_{i,N}$. The average accuracy $\mathcal{A}_N$ represents the average accuracy of all seen tasks after training on the N-th task:

$$\mathcal{A}_N = \frac{1}{N}\sum_{i=1}^{N}\mathcal{A}_{i,N},$$

which is often considered as the most important evaluation metric in continual learning.

The average incremental accuracy $\bar{\mathcal{A}}$ is the average accuracy along incremental stages, defined as:

$$\bar{\mathcal{A}} = \frac{1}{N}\sum_{t=1}^{N}\mathcal{A}_t.$$

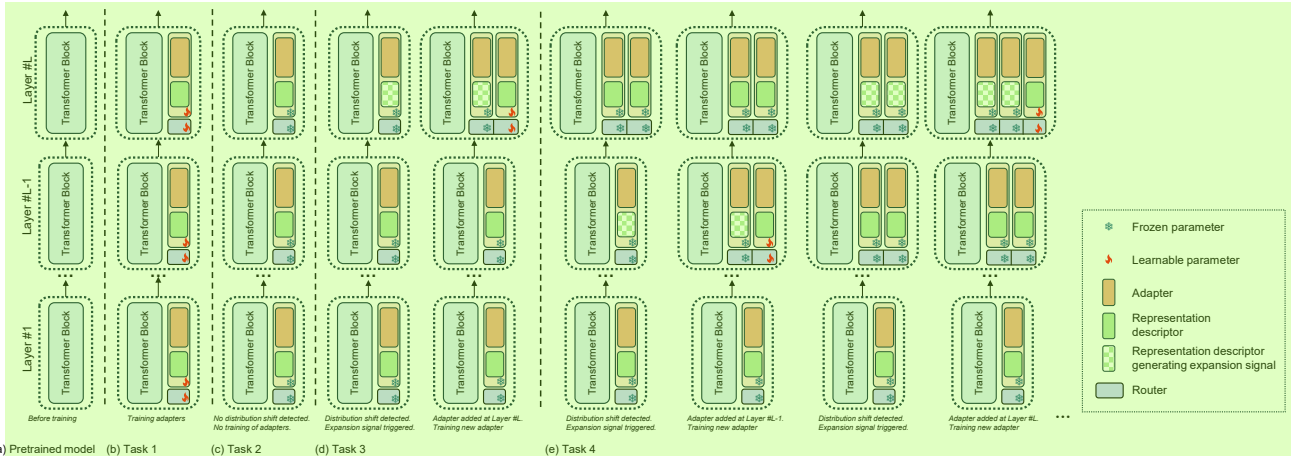


Figure 9. A more detailed example for the illustration of the learning process. (a) The pre-trained model with L transformer layers is provided for adaptation. (b) At the start of training, each transformer layer is equipped with one expandable weighting router and one adapter module, including one functional adapter and its paired representation descriptor. All modules are trainable at this stage. (c) All modules and routers are frozen after the training on Task 1. When Task 2 arrives, the detection of distribution shift is performed with all frozen representation descriptors in each transformer layer for all batches in Task 2. Since no distribution shift is observed, module addition is not performed and all modules are frozen. (d) As Task 3 arrives, the detection for the distribution shift is executed again and the distribution shift is observed in the L-th layer. Expansion signal is triggered and an adapter module is added in the L-th layer with the expanded router. Training for the newly added adapter and router is performed. Since the addition is performed at the last transformer layer, no further detection for distribution shift is required. (e) When Task 4 arrives, expansion signal is triggered in the $L-1$-th layer during the detection phase. After sufficient training, the newly added module is frozen and detection for distribution shift in later layers is executed. When both representation descriptors in the L-th layer consider the incoming feature as an outlier, expansion signal will be triggered. A new module is added for training in the L-th layer while all other modules are frozen.

Method	CIFAR-100		5-Task IN-R		10-Task IN-R		20-Task IN-R		ImageNet-A		VTAB	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
L2P	89.51	85.02	72.90	65.83	74.55	69.75	74.49	65.82	46.67	39.30	79.17	63.56
DualPrompt	90.39	85.64	73.91	68.81	73.10	67.18	73.67	68.88	58.45	48.78	88.11	77.58
CODA-P	91.01	86.20	79.78	74.68	79.15	73.05	70.36	65.32	50.73	37.06	85.13	85.85
SimpleCIL	87.13	81.26	59.70	54.33	61.12	54.33	61.92	54.33	60.50	49.44	85.99	84.38
ADAM	92.18	87.47	77.28	70.58	76.71	69.18	75.08	67.30	60.53	49.57	85.95	84.35
InfLoRA	91.71	86.73	81.75	76.77	81.38	74.72	76.97	**69.65**	56.84	41.61	89.61	86.52
SEMA	**92.23**	**87.84**	**83.27**	**77.13**	**81.39**	**74.82**	**77.84**	69.60	**62.50**	**51.35**	**91.99**	**90.86**

Table 4. Experiments on class-incremental learning benchmarks with ViT-B/16-IN21K weight.

C. More Experiments and Ablation Studies

C.1. Influence of Pre-trained Weights

In the main paper, we experiment SEMA and other methods with ViT-B/16-IN1K in Tab. 1. To study the influence of pre-trained weights, we further experiment SEMA with another commonly used pre-trained ViT weight, i.e., ViT-B/16-IN21K. We evaluate the performance using average accuracy $\mathcal{A}_N$ and average incremental accuracy $\bar{\mathcal{A}}$. As shown in Tab. 4, SEMA consistently outperforms prompting and adaptation methods in most class-incremental learning settings. This indicates that our model is robust in performance regardless of different choices of pre-trained weights.

C.2. Further Analyses on the Effectiveness of Self-Expansion

The proposed method SEMA enables the model to add parameters and expand its capacity on demand. It allows the model to handle samples that could not be handled before by adding a small number of parameters. In continual learning, this process helps to alleviate forgetting by avoiding interference from new patterns while still encouraging knowledge reuse and transfer. Unlike some methods [68, 73, 92] that continually adding task-specific modules by task with a *lin-*

Dataset	Expansion by Task Params (M)	$\mathcal{A}_N$	SEMA Params (M)	$\mathcal{A}_N$
CIFAR-100	1.066	86.86	0.645	86.98
ImageNet-R	1.904	74.08	0.617	74.53
ImageNet-A	1.904	52.80	0.560	53.32
VTAB	0.647	89.09	0.554	89.64

Table 5. Comparison of added parameters and accuracy with different expansion strategies. "Expansion by Task" is a *naive* implementation of SEMA's variant that adds one set of adapters (at all layers allowing expansion) for every new task. SEMA only expands if a distribution shift is detected by the representation descriptor.

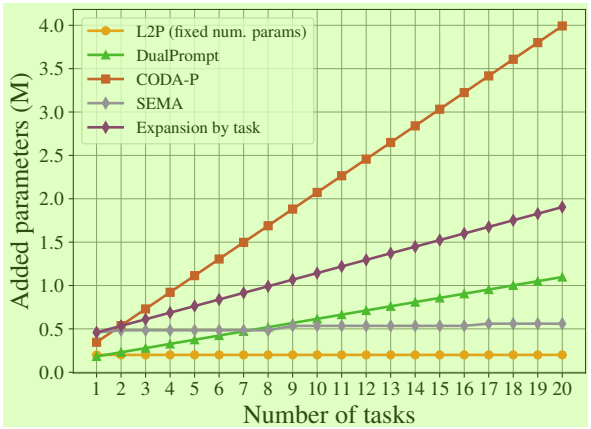


Figure 10. Analysis on added parameters (in Millions) during model deployment on ImageNet-A. We compare with methods using fixed number of prompts like L2P, and methods like DualPrompt and CODA-P that incrementally expand like SEMA but with prompts and on a linear basis according to tasks. Expansion by task adds adapters for every incoming task, whilst SEMA executes expansion on demand, which increments parameters on a sub-linear basis. Specifically, SEMA added more parameters (with expansions at more layers) at Task 9 than other steps with expansion.

ear parameter growth rate, SEMA produces a *sub-linear* expansion rate, w.r.t. number of seen tasks. To analyze and show the effectiveness of this self-expansion process, we conducted comparisons on four different settings where CIFAR-100, ImageNet-R, ImageNet-A and VTAB contain 10 tasks, 20 tasks, 20 tasks and 5 tasks respectively, corresponding to four settings reported in Fig. 3. We compare with other related methods and a *naive implementation* of the "expansion-by-task" variant of SEMA. This simple variant model incrementally adds adapters to the layers that allow expansion for each incoming task. The number of parameters and accuracy are reported in Tab. 5. Despite the naive implementation of "expansion-by-task", the results in Tab. 5 show that SEMA with flexible self-expansion can achieve better performance than that using more parameters. We demonstrate that our expansion strategy is efficient in both controlling the size of added parameters, regardless of the length of task sequence, encouraging knowledge reuse and reducing potential task interference in adapter weighting.

Tab. 6 reports the size of added parameters in several different PTM-based methods. While L2P uses a fixed size of prompt pool with small amount of added parameters, the fixed size of trainable parameters may limit its capability to adapt to more distribution shifts in continual learning and comes with a higher chance of forgetting. Compared to other methods (*i.e.*, CODA-P and DualPrompt) that incrementally add parameters (*i.e.*, prompts in these methods) for each task, SEMA involves much fewer added parameters in the model. Apart from the adaptation approach and expansion strategy, the compared methods in this part use similar techniques as the proposed method (such as the classifier and PTMs). Note that the added parameters for SEMA only consider the functional adapters that are used in deployment. The RDs are maintained for training and updating of the model, which can be handled in parallel to other parameters and do not influence the deployment of the model. As shown in Fig. 10 (also demonstrated in the main paper Fig. 8), SEMA can dynamically expand the model with a small *sub-linear* rate, while the other methods are usually with a *linear* rate.

C.3. Further Discussions on the Weighting Router

Routing relying on representation descriptor. In SEMA, we use the representation descriptors (RDs) to capture the distribution of the input representations corresponding to each modular adapter, which are used to detect novel patterns triggering the expansion signal. The RDs can be used to compose the adapters via hard selection, as in similar modular networks. Specifically, the reconstruction error of the AE-based RDs can provide the identity information of each inference sample, w.r.t. the adapters, at different layers. However, the RD-based adapter selection/routing can be unreliable for every single individual input, and related works usually rely on the statistics of a batch of samples [55], limiting the application. We thus propose directly learning the soft weighting router for mixture usage of the adapters. To analyze the behavior of the RDs in detail, we conduct the experiments that perform adapter composing relying on the RDs and show the results in Tab. 7. As shown in Tab. 7, the RD-based routing can achieve sound performance on most datasets, which validates the representation ability of RDs. SEMA with the soft weighting router can perform better, relying on the specifically learned router that is trained together with the adapters.

More discussions on adapter usage. Fig. 5 shows the average adapter usage of each task on VTAB. For clear visualization, we enable expansion to be performed only at the last layer and attach sample images from each task in Fig. 5. Adapter 1, Adapter 2, and Adapter 3 are automatically

Type	Method	CIFAR-100 Params (M)	CIFAR-100 $\mathcal{A}_N$	ImageNet-R Params (M)	ImageNet-R $\mathcal{A}_N$	ImageNet-A Params (M)	ImageNet-A $\mathcal{A}_N$	VTAB Params (M)	VTAB $\mathcal{A}_N$
Fixed Param Size	L2P	0.123	77.87	0.200	62.90	0.200	38.48	0.085	80.83
Expandable Param Size	DualPrompt	1.022	80.43	1.098	61.97	1.098	50.23	0.983	79.79
	CODA-P	3.917	86.11	3.994	70.02	3.994	35.02	3.878	81.58
	SEMA	**0.645**	**86.98**	**0.617**	**74.53**	**0.560**	**53.32**	**0.554**	**89.64**

Table 6. Number of added parameters used in model deployment, measured in Millions. L2P uses a fixed size of prompts. DualPrompt and CODA-P incrementally add parameters (*i.e.*, prompts) sequentially by task. SEMA adds a small number of parameters with its dynamic expansion strategy.

Method	CIFAR-100 $\bar{\mathcal{A}}$	CIFAR-100 $\mathcal{A}_N$	5-Task IN-R $\bar{\mathcal{A}}$	5-Task IN-R $\mathcal{A}_N$	10-Task IN-R $\bar{\mathcal{A}}$	10-Task IN-R $\mathcal{A}_N$	20-Task IN-R $\bar{\mathcal{A}}$	20-Task IN-R $\mathcal{A}_N$	ImageNet-A $\bar{\mathcal{A}}$	ImageNet-A $\mathcal{A}_N$	VTAB $\bar{\mathcal{A}}$	VTAB $\mathcal{A}_N$
SEMA	**91.37**	**86.98**	**84.75**	**79.78**	**83.56**	**78.00**	**81.75**	**74.53**	**64.53**	**53.32**	**91.26**	**89.64**
RD-based routing	90.91	83.61	84.46	79.50	82.76	76.63	81.02	74.13	61.80	50.36	90.83	88.53

Table 7. Comparison between routing with the expandable weighting router and RD-based routing.

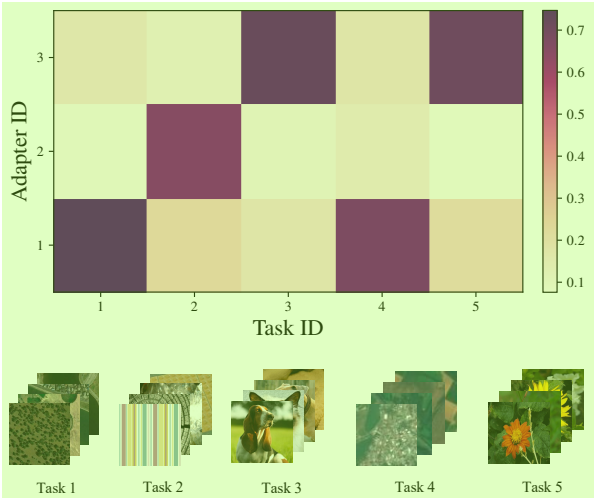


Figure 11. Adapter usage visualization on VTAB (same as Fig. 5). For clear and simplified visualization, we only allow expansion at the last transformer layer. We report the average adapter usage of each task. We also provide visual illustrations of sample images from each VTAB task.

Method	Train Time (s) CIFAR-100	ImageNet-R	ImageNet-A	VTAB
L2P	0.27	0.27	0.29	0.28
DualPrompt	**0.25**	0.25	0.27	**0.29**
CODA-P	0.31	0.32	0.35	0.36
SEMA (Overall)	**0.25**	**0.11**	**0.15**	0.31
- Adapter	0.13	0.10	0.12	0.20
- RD	0.12	0.01	0.03	0.11

Table 8. Average per-batch train time of each method on each task measured in seconds. SEMA (overall) denotes the training time used when adapter and representation descriptor (RD) are trained sequentially.

Method	Inference Time (ms)			
	CIFAR-100	ImageNet-R	ImageNet-A	VTAB
L2P	9.44	9.53	9.86	9.46
DualPrompt	9.44	9.51	9.84	9.44
CODA-P	9.45	9.47	9.85	9.43
ADAM	9.95	10.03	10.36	9.45
SEMA	**4.48**	**7.39**	**9.01**	**7.38**

Table 9. Per-image inference time of each method measured in milliseconds.

added and trained when Task 1, Task 2, and Task 3 arrive, respectively. Task 1, Task 2, and Task 3 all present high preference for choosing the adapters that were trained with them, showing the effectiveness of the router to direct samples to the adapter that is trained with a similar distribution. While adapter expansion is not triggered for Task 4, Task 4 data largely employs Adapter 1 during inference. As visualized in Fig. 11, the data distribution between Task 1 (remote sensing images) and Task 4 (land cover) is similar. Similarly, Task 3 (pets) and Task 5 (flowers) both comprise natural images with similar characteristics, hence have higher similarity in distribution than Task 1 (remote sensing images) and Task 2 (texture images), and exhibit a preference for Adapter 3. Thus, we show that our expandable weighting router can effectively select the proper mixture pattern of adapters with various data distributions.

C.4. Training and Inference Time

All experiments can be produced on a single NVIDIA GeForce RTX 3090 GPU. To compare the training efficiency, we report the per-batch training time averaged over the incremental learning process in Tab. 8. Similar to Tab. 5, ImageNet-R here is split into 20 tasks with 10 classes per task. Note that the training processes of adapter and representation descriptor in each adapter module of SEMA are in parallel after expansion, thus the training of these two components can be performed in parallel with multiple GPUs. We report the training time of adapters (*i.e.*, "Adapter" in Tab. 8) and representation descriptors (*i.e.*, "RD" in Tab. 8) separately, along with the overall time usage of SEMA training if adapters and representation descriptors are trained sequentially.

SEMA with components trained in a parallel manner is highly efficient. Even without the parallel setup, training the adapters and RDs in SEMA in sequence can still be faster than other PTM-based CL methods on most datasets. As SEMA only expands while encountering distribution shifts in incoming new tasks, for tasks that do not trigger expansion, no training of adapters and representation descriptors is performed and training time on these tasks is minimized, leading to training efficiency in the long term. Note that the scanning for distribution shifts is stopped as long as a batch of data triggers expansion behaviour, which is more efficient comparing to InfLoRA which requires processing through all data in the given task twice for LoRA initialization before training and post-training computation for gradient projection memory.

We evaluate the inference efficiency and report the average inference time of each image measured in milliseconds in Tab. 9. We show that SEMA is efficient compared to other methods on all datasets. The inference latency of the listed prompting continual learning methods is caused by the extra procedure of processing the image with a frozen pre-trained model for the query function. Similarly, ADAM requires extra feature extraction with a frozen pre-trained model for the concatenation of pre-trained features and adapted features. SEMA relieves the dependency on the frozen pre-trained model as we focus on the intermediate feature distribution of each transformer block.

C.5. Additional Results on Longer Task Sequence

We perform the 50-step experiment on ImageNet-R and ImageNet-A, where each task contains 4 classes, and report the performance in Tab. 10. SEMA outperforms all other methods in longer task sequences.

Method	ImageNet-R		ImageNet-A	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
L2P	69.11	63.53	40.77	33.31
DualPrompt	64.21	56.25	49.74	39.83
CODA-P	61.34	56.37	34.36	23.17
ADAM	69.59	62.58	59.44	48.58
InfLoRA	67.01	61.37	47.33	31.27
SEMA	**74.64**	**67.03**	**60.82**	**49.18**

Table 10. Evaluation on longer task sequence with 50 tasks.

C.6. Additional Results on Incremental Performance

We present a comparison of performance across incremental stages for CIFAR-100, 20-Task ImageNet-R, 20-Task

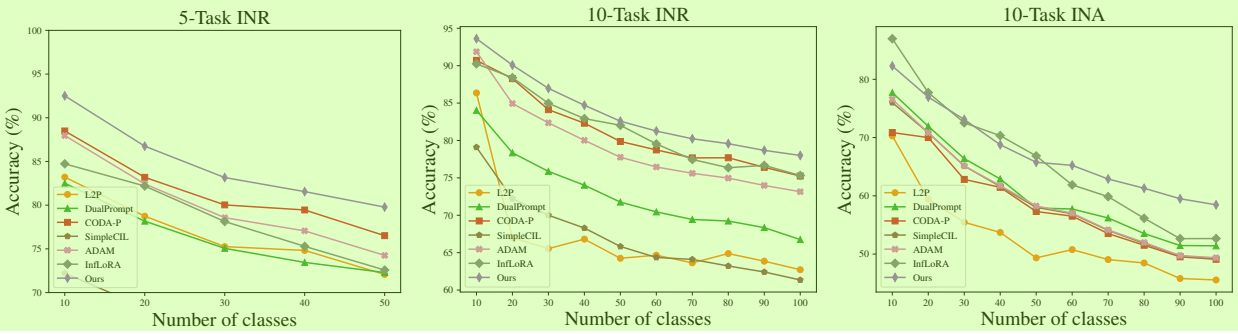


Figure 12. More results on incremental performance for ImageNet-R and ImageNet-A.

ImageNet-A and VTAB in Fig. 3 of the main paper. We further conduct experiments on ImageNet-A which is split into 10 tasks. We provide the incremental performance of 5-Task ImageNet-R, 10-Task ImageNet-R and 10-Task ImageNet-A in Fig. 12. Both figures show that SEMA performs consistently well with different dataset splits.

C.7. Analyses on Training with Less Data

We further conduct analyses on the scenario of training with less data. Benefiting from the better knowledge reuse/transfer ability, SEMA can achieve better performance with less data. We specifically compare with a state-of-the-art method, EASE [92], which expands task-specific adapters at all layers of the transformer. Unlike all other methods we compared with in the main paper, EASE also incrementally adds classification heads for all tasks and ensembles them in inference. In Tab. 11, we show the results of experiments on VTAB while removing 90% of samples in one and two tasks, respectively, denoted as VTAB-1 and VTAB-2. Although EASE uses a much stronger classification head, SEMA can perform better in this data efficiency learning experiment. We then further extend this data efficiency experiment to ImageNet-A by keeping only 10 or 20 percent of data for all tasks. As shown in Tab. 12, with sub-linear expansion, SEMA obtains performance comparable to EASE which requires task-oriented expansion at linear growth rate.

Method	VTAB-1		VTAB-2	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
SEMA	86.74	81.33	85.99	80.06
EASE	86.56	78.37	86.76	78.86

Table 11. Experiments on setting with limited data samples on VTAB. VTAB-1 and VTAB-2 randomly removes 90 percent of data in one and two task(s), respectively.

Method	ImageNet-A 10%		ImageNet-A 20%	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
SEMA	52.90	41.41	57.85	48.26
EASE	52.79	41.67	57.46	48.65

Table 12. Experiments on setting with limited data samples on ImageNet-A. ImageNet-A 10% contains only 10 percent of data in original ImageNet-A for all tasks and ImageNet-A 20% contains 20 percent.

C.8. Experimental Results with Different Seeds and Varying Class Orders

We conduct five independent runs with different seeds for SEMA on all datasets, and report the mean and standard deviation of accuracies over separate runs in Tab. 13. With different random seeds, each run is performed with different shuffling of class order and model initialization weights. This demonstrates the robustness of SEMA's performance with varying task/class orderings.

C.9. Ablation Study on the Hidden Dimension in AE

We test different values for hidden dimensions in the AE as representation descriptors. The AE-based representation descriptors enable the capture of the characteristics of the data for decision-making on whether to add a new adapter during continual training. According to Fig. 13, the proposed method can perform well with a wide range of settings on the AE's hidden dimension.

C.10. Results with Representation Enhancement

As discussed, different PTM-based continual learning methods focus on updating/adapting the backbone/representation (*e.g.*, SEMA, InfLoRA [47], CODA-P [68]) and continually conducting feature representation enhancement of frozen PTMs (*e.g.*, RanPAC [51]), respectively. These two types of methods are orthogonal and can work together. The pro-

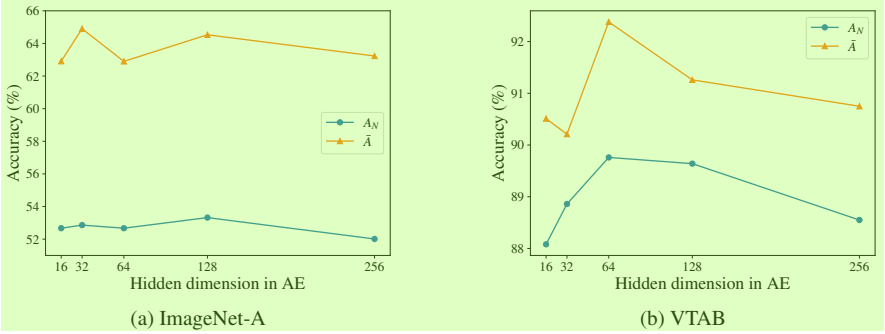


Figure 13. Ablation on representation descriptor.

Method		CIFAR-100	5-Task IN-R	10-Task IN-R	20-Task IN-R	ImageNet-A	VTAB
SEMA	$\bar{\mathcal{A}}$	91.37 ± 0.38	84.75 ± 0.84	83.56 ± 0.41	81.75 ± 1.00	64.53 ± 0.99	91.26 ± 0.47
	$\mathcal{A}_N$	86.98 ± 0.57	79.78 ± 0.46	78.00 ± 0.49	74.53 ± 0.92	53.32 ± 0.69	89.64 ± 0.63

Table 13. Accuracies with standard deviation over 5 independent runs.

Method	CIFAR-100		5-Task IN-R		10-Task IN-R		20-Task IN-R		ImageNet-A		VTAB	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
RanPAC	93.81	90.04	83.81	79.57	84.23	79.00	83.87	78.18	69.96	62.15	91.97	91.33
SEMA+RanPAC	**94.54**	**90.95**	**85.93**	**81.58**	**85.59**	**80.55**	**85.13**	**79.40**	**71.87**	**63.33**	**93.99**	**92.33**

Table 14. Results on different methods using random projection technique.

posed self-expansion learning in SEMA can also be combined with the statistical alignment techniques of RanPAC, *i.e.*, SEMA+RanPAC, to get better performance. Specifically, the feature enhancement with random projection and prototype classifiers in RanPAC is applied to the representations from SEMA's model. Tab. 14 demonstrates that the representations are benefited from the self-expansion strategy, as SEMA+RanPAC outperforms RanPAC implemented with a single adapter and first-session adaptation.

Method	CIFAR-100		10-Task IN-R	
	$\bar{\mathcal{A}}$	$\mathcal{A}_N$	$\bar{\mathcal{A}}$	$\mathcal{A}_N$
Zero-shot	76.36	66.96	79.17	77.08
ADAM	79.53	71.26	72.06	70.90
SEMA	**82.74**	**73.52**	**80.94**	**78.18**

Table 15. Performance on pre-trained CLIP model.

C.11. Experiments with CLIP.

We further conduct experiment with a pre-trained vision-language model, namely CLIP with a ViT-B/16 backbone [58], and report the performance in Tab. 15. SEMA outperforms zero-shot CLIP and ADAM which have no parameter expansion, highlighting the effectiveness of our dynamic expansion strategy and its broad applicability to pre-trained models.