Is Parameter Isolation Better for Prompt-Based Continual Learning?

Supplementary Material

1. Training Algorithm

We adopt the HiDe-Prompt framework [17] and adjust it to suit our shared prompt pool setting. During the Within-Task Prediction (WTP) phase, we jointly optimize the classification head ψ and the global prompt pool $\mathcal{P} = \{p_1, \ldots, p_K\}$, where each training instance dynamically activates a sparse subset of prompts via a learned task-specific router. The selected prompt composition $\tilde{p}$ is used in the encoder $f_\theta(x, \tilde{p})$ to obtain the instructed representation. All past prompt parameters remain active and shared, enabling dynamic reuse instead of freezing as in static prompt allocation.

To further enhance representation stability, we adopt the same contrastive regularization term as in HiDe-Prompt. Specifically, for each previously encountered class $c \in \mathcal{Y}^{(i)}, i = 1, \ldots, t-1$, its instructed and uninstructed representations are approximated by class-wise Gaussian distributions $\mathcal{G}_c$ and $\hat{\mathcal{G}}_c$. Let $\tilde{P}_t = \{\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_{N_t}\}$ denote the set of instance-specific prompts dynamically selected by the task-specific router $\mathcal{R}_t$ for all training samples in $\mathcal{D}_t$. We define the current task embeddings as $\mathcal{H}_t = \{f_\theta(\boldsymbol{x}_i^{(t)}, \tilde{p}_i) \mid i = 1, \ldots, N_t\}$ and $\boldsymbol{\mu}_c$ be the mean of $\mathcal{G}_c$. The contrastive loss is computed as:

$$\mathcal{L}_{\mathrm{CR}} = \sum_{h \in \mathcal{H}_t} \sum_{i=1}^{t-1} \sum_{c \in \mathcal{Y}^{(i)}} \log \left(\frac{\exp(\boldsymbol{h} \cdot \boldsymbol{\mu}_c / \tau)}{Z(h)} \right), \tag{1}$$

$$Z(h) = \sum_{\boldsymbol{h}' \in \mathcal{H}_t} \exp(\boldsymbol{h} \cdot \boldsymbol{h}' / \tau) + \sum_{i=1}^{t-1} \sum_{c' \in \mathcal{Y}^{(i)}} \exp(\boldsymbol{h} \cdot \boldsymbol{\mu}_{c'} / \tau) \tag{2}$$

Here, $Z(h)$ acts as the partition function, collecting all positive and negative pairs for normalization and τ is the temperature that is set to 0.8. The overall loss for WTP combines classification and contrastive objectives:

$$\mathcal{L}_{\mathrm{WTP}}(\psi, \tilde{P}_t) = \mathcal{L}_{\mathrm{CE}}(\psi, \tilde{P}_t) + \lambda \mathcal{L}_{\mathrm{CR}}, \tag{3}$$

where λ is a balancing hyperparameter. Following the WTP stage, we further refine the classifier parameters ψ through a dedicated objective known as task-adaptive prediction (TAP). This stage aims to mitigate classifier bias by accounting for the distributional properties of all previously observed classes. Specifically, TAP optimizes h_ψ using pseudo features sampled from Gaussian approximations of prior class representations. The TAP loss is defined as: The TAP loss is defined as:

$$\mathcal{L}_{\mathrm{TAP}}(\psi) = \sum_{i=1}^{t} \sum_{c \in \mathcal{Y}^{(i)}} \sum_{\boldsymbol{h} \in \mathcal{H}_{i,c}} -\log \left(\frac{\exp\big(h_\psi(\boldsymbol{h})[c]\big)}{Z_{\mathrm{TAP}}(\boldsymbol{h})} \right) \tag{4}$$

where $Z_{\mathrm{TAP}}(\boldsymbol{h}) = \sum_{j=1}^{t} \sum_{c' \in \mathcal{Y}^{(j)}} \exp\big(h_\psi(\boldsymbol{h})[c']\big)$ is the normalization factor (partition function) over all previously observed classes and $\mathcal{H}_{i,c}$ contains pseudo representations sampled from the Gaussian prototype $\mathcal{G}_c$ of class c from task $\mathcal{T}_i$. This step enhances the alignment of the classifier with the evolving feature space, thus improving robustness against forgetting.

At test time, our method employs a lightweight auxiliary task predictor $\hat{h}_\omega : \mathbb{R}^D \to \mathbb{R}^T$, trained with the Task-Identity Inference (TII) objective, to infer the task identity from the uninstructed representation $f_\theta(x)$. This module is trained using a cross-entropy loss over pseudo features sampled from the approximate Gaussian distributions $\hat{\mathcal{G}}_c$ of previously encountered classes. Formally, the TII objective is defined as:

$$\mathcal{L}_{\mathrm{TII}}(\omega) = \sum_{c \in \mathcal{Y}_t} \sum_{\hat{\boldsymbol{h}} \in \hat{\mathcal{H}}_c} -\log \left(\frac{\exp(\hat{h}_\omega(\hat{\boldsymbol{h}})[c])}{\sum_{c' \in \mathcal{Y}_t} \exp(\hat{h}_\omega(\hat{\boldsymbol{h}})[c'])} \right), \tag{5}$$

where $\hat{\mathcal{H}}_c$ contains pseudo representations sampled from the uninstructed class prototype $\hat{\mathcal{G}}_c$ for $c \in \mathcal{Y}^{(i)}$ and $i = 1, \ldots, t$..

During inference, the predicted task index from $\hat{h}_\omega$ determines the corresponding router $\mathcal{R}_t$, which computes prompt relevance scores based on input features and selects the top-k prompts $\mathcal{K}(h)$ from the shared pool. These prompts are then aggregated into a composite vector $\tilde{p}$, which conditions the encoder to produce the final prediction: $\hat{y} = h_\psi(f_\theta(x, \tilde{p}))$. Please refer to Algorithm 1 for more details.

2. Experimental Details

2.1. Additional Benchmark Details

In this section, we provide detailed information on the benchmarks used in our work.

Split CIFAR-100 : CIFAR-100 [9] is a widely used image classification benchmark consisting of images from 100 classes. The dataset contains diverse natural images, including animals, vehicles, and various everyday objects. Each image is annotated with both fine and coarse labels, enabling hierarchical classification tasks. For class-incremental learning settings, the 100 classes are randomly divided into 10 incremental tasks, each comprising a unique set of classes.

Split ImageNet-R : ImageNet-R [8] is a variant of the ImageNet benchmark, containing images from 200 classes

Algorithm 1 Training Algorithm of Hash

Input: Pre-trained transformer backbone f_θ, training sets $\mathcal{D}_1, ..., \mathcal{D}_T$, number of tasks T, number of epochs E, hyperparameters τ and λ.

Output: Parameters $R_1, ..., R_T, p_1, ..., p_K, \omega$ and ψ

Initialize $R_1, p_1, ..., p_K, \omega$ and ψ
for $t = 1, ..., T$ **do**
  for $c \in \mathcal{Y}_t$ **do**
    Obtain $\hat{\mathcal{G}}_c$ from f_θ and $\mathcal{D}_t$ ▷ Uninstructed Representations
  Initialize empty prompt batch: $\tilde{P}_t$
  for $x_i \in \mathcal{D}_t$ **do**
    Construct routing score: $s_i = \mathcal{R}_t(x_i)$
    Construct top-k index set: $\mathcal{K}(s_i) = \text{TopK}(s_i)$
    Construct attention weights: $\alpha_k = \exp(s_{i,k}) / \sum_{j \in \mathcal{K}(h_i)} \exp(s_{i,j})$
    Construct instance prompt: $\tilde{p}_i = \sum_{k \in \mathcal{K}(s_i)} \alpha_k p_k$
    Append $\tilde{p}_i$ to $\tilde{P}_t$
  for $epoch = 1, ..., E$ **do**
    Optimize $\boldsymbol{p}_t$ and ψ with $\mathcal{L}_{\text{WTP}}$ in Eq. (3) ▷ Within-Task Prediction
    Optimize ω with $\mathcal{L}_{\text{TII}}$ in Eq. (5) ▷ Task-Identity Inference
    Optimize ψ with $\mathcal{L}_{\text{TAP}}$ in Eq. (4) ▷ Task-Adaptive Prediction
  for $c \in \mathcal{Y}_t$ **do**
    Obtain $\mathcal{G}_c$ from f_θ, $\tilde{P}_t$ and $\mathcal{D}_t$ ▷ Instructed Representations
return $(R_1, ..., R_T, p_1, ..., p_K, \omega, \psi)$

that are selected to emphasize robustness to distribution shifts. The dataset includes diverse renditions of objects such as paintings, cartoons, sketches, and other non-photographic styles, covering a broad range of natural and man-made categories. These classes are also randomly divided into 10 distinct incremental tasks.

Split CUB-200 : CUB-200 [16] is a fine-grained image classification dataset containing images from 200 bird species. The dataset features a diverse collection of natural bird photographs, capturing various poses, backgrounds, and environmental conditions. Each image is annotated with a species-level class label, supporting detailed visual recognition research. For class-incremental learning, the 200 categories are randomly split into 10 sequential tasks, each with a distinct subset of bird species.

5-Datasets : 5-Datasets[4] is a composite continual learning dataset that integrates five widely used image classification datasets: CIFAR-10 [9], MNIST [11], Fashion-MNIST [21], SVHN [14], and notMNIST [1]. Each component dataset presents distinct visual characteristics, ranging from natural images and street view numbers to handwritten and typewritten digits or fashion items. In this benchmark, each dataset is treated as a separate incremental task, enabling the evaluation of methods under substantial distributional shifts and task heterogeneity.

Table 1. Performance comparison of **adapter-based** continual learning methods using ViT-B/16 with Sup-21K weights. Here we present Final Average Accuracy (FAA).

Method	Split CIFAR-100	Split CUB-200
C-ADA	88.25	76.13
LAE	85.33	80.97
ADAM + Adapter	87.29	85.84
EASE	87.76	84.65
InfLoRA	88.31	80.78
Hash (Ours)	**95.02**	**91.34**

Table 2. Performance comparison of pre-trained **model-based** continual learning methods using ViT-B/16 with Sup-21K weights. Here we present Final Average Accuracy (FAA).

Method	Split CIFAR-100	Split CUB-200
ADAM + VPT-D	85.04	85.28
ADAM + SSF	85.27	85.67
ADAM + Adapter	87.29	85.84
RanPAC	92.20	90.30
Hash (Ours)	**95.02**	**91.34**

2.2. Additional Comparison Methods Details

In this section, we provide a detailed description of the methods compared in our work.

Table 3. Overall performance comparison on Split CIFAR-100 and Split ImageNet-R. We present Final Average Accuracy (FAA), Cumulative Average Accuracy (CAA), and Average Forgetting Measure (FM) of all methods under different pre-trained models.

PTM	Method	Split CIFAR-100			Split Imagenet-R		
		FAA (↑)	**CAA**(↑)	FM(↓)	**FAA** (↑)	**CAA**(↑)	FM(↓)
iBOT-21K	L2P	79.13 ± 1.25	85.13 ± 0.05	7.50 ± 1.21	61.31 ± 0.50	68.81 ± 0.52	10.72 ± 0.40
	DualPrompt	78.84 ± 0.47	86.16 ± 0.02	8.84 ± 0.67	58.69 ± 0.61	66.61 ± 0.67	11.75 ± 0.92
	CODA-Prompt	80.83 ± 0.27	87.02 ± 0.20	7.50 ± 0.25	61.22 ± 0.35	66.76 ± 0.37	9.66 ± 0.20
	CPrompt	82.14 ± 0.32	88.09 ± 0.09	7.02 ± 0.24	74.42 ± 0.18	79.19 ± 0.27	7.02 ± 0.36
	HiDe-Prompt	93.02 ± 0.15	94.56 ± 0.05	1.26 ± 0.13	70.83 ± 0.17	73.23 ± 0.08	**6.77** ± 0.23
	NoRGa	94.76 ± 0.15	95.86 ± 0.31	1.34 ± 0.14	73.06 ± 0.26	77.46 ± 0.42	6.88 ± 0.49
	Hash (Ours)	**95.12** ± 0.24	**95.97** ± 0.21	**1.22** ± 0.17	**76.97** ± 0.18	**81.87** ± 0.29	6.84 ± 0.39
iBOT-1K	L2P	75.51 ± 0.88	82.53 ± 1.10	6.80 ± 1.70	59.43 ± 0.28	66.83 ± 0.92	11.33 ± 1.25
	DualPrompt	76.21 ± 1.00	83.54 ± 1.23	9.89 ± 1.81	60.41 ± 0.76	66.87 ± 0.41	9.21 ± 0.43
	CODA-Prompt	79.11 ± 1.02	86.21 ± 0.49	7.69 ± 1.57	66.56 ± 0.68	73.14 ± 0.57	7.22 ± 0.38
	CPrompt	83.12 ± 0.39	89.53 ± 0.08	6.42 ± 0.27	72.42 ± 0.18	75.98 ± 0.23	7.05 ± 0.21
	HiDe-Prompt	93.48 ± 0.11	95.02 ± 0.01	1.63 ± 0.10	71.33 ± 0.21	73.62 ± 0.13	7.11 ± 0.02
	NoRGa	94.01 ± 0.04	95.11 ± 0.35	**1.61** ± 0.30	72.77 ± 0.20	76.55 ± 0.46	7.10 ± 0.39
	Hash (Ours)	**94.59** ± 0.07	**95.98** ± 0.24	1.72 ± 0.19	**75.01** ± 0.18	**78.45** ± 0.22	**6.99** ± 0.19
DINO-1K	L2P	72.23 ± 0.35	79.71 ± 1.26	8.37 ± 2.30	57.21 ± 0.69	64.09 ± 0.74	7.47 ± 0.96
	DualPrompt	73.95 ± 0.49	81.85 ± 0.59	9.32 ± 1.42	57.98 ± 0.71	65.39 ± 0.27	9.32 ± 0.69
	CODA-Prompt	77.50 ± 0.64	84.81 ± 0.30	8.10 ± 0.01	63.15 ± 0.39	69.73 ± 0.25	6.86 ± 0.11
	CPrompt	81.98 ± 0.52	89.82 ± 0.43	9.14 ± 0.66	71.92 ± 0.38	76.23 ± 0.32	6.77 ± 0.64
	HiDe-Prompt	92.51 ± 0.11	94.25 ± 0.01	1.67 ± 0.20	68.11 ± 0.18	71.70 ± 0.01	6.45 ± 0.58
	NoRGa	93.43 ± 0.33	94.65 ± 0.62	1.65 ± 0.25	71.77 ± 0.44	75.76 ± 0.49	6.42 ± 0.68
	Hash (Ours)	**94.12** ± 0.09	**95.21** ± 0.17	**1.59** ± 0.24	**73.98** ± 0.23	**77.74** ± 0.12	**6.25** ± 0.44
MoCo-1K	L2P	77.24 ± 0.69	83.73 ± 0.70	5.57 ± 0.75	54.13 ± 0.67	62.09 ± 0.76	**4.88** ± 0.42
	DualPrompt	77.56 ± 0.63	84.37 ± 0.51	6.54 ± 0.50	54.45 ± 0.30	62.92 ± 0.41	5.34 ± 0.41
	CODA-Prompt	77.83 ± 0.34	84.97 ± 0.23	12.60 ± 0.02	55.75 ± 0.26	65.49 ± 0.36	10.46 ± 0.04
	CPrompt	84.12 ± 0.29	89.23 ± 0.33	6.92 ± 0.15	64.79 ± 0.26	70.64 ± 0.14	9.73 ± 0.33
	HiDe-Prompt	91.57 ± 0.20	93.70 ± 0.01	**1.51** ± 0.17	63.77 ± 0.49	68.26 ± 0.01	9.37 ± 0.71
	NoRGa	93.52 ± 0.06	94.94 ± 0.29	1.63 ± 0.13	64.52 ± 0.16	70.21 ± 0.64	9.06 ± 0.19
	Hash (Ours)	**94.37** ± 0.09	**95.62** ± 0.21	1.54 ± 0.13	**67.09** ± 0.13	**72.12** ± 0.09	8.14 ± 0.25

L2P : L2P [20] is a continual learning framework that leverages a pool of learnable prompts to guide a frozen pre-trained transformer model. During each incremental task, L2P dynamically selects and updates prompts relevant to the current data, enabling the model to adapt to new tasks without revisiting previous data or altering the backbone weights. This approach effectively mitigates catastrophic forgetting and facilitates efficient knowledge integration across tasks.

DualPrompt : DualPrompt [19] is a prompt-based continual learning method designed for transformer architectures. It introduces two types of prompts: task-shared prompts that capture general knowledge across all tasks, and task-specific prompts that focus on information unique to each task. By jointly optimizing both prompt types, DualPrompt achieves a balance between knowledge retention and task adaptability, significantly improving performance in class-incremental learning scenarios.

CODA-Prompt : CODA-Prompt [15] introduces a set of learnable prompt components, which are assembled into input-conditioned prompts using an attention-based scheme. Unlike previous methods, all prompting parameters are optimized end-to-end with the task loss, enabling greater capacity and adaptability for rehearsal-free continual learning.

C-Prompt : CPrompt [7] addresses the training-testing inconsistency in prompt-based continual learning by introducing classifier consistency learning (CCL) and prompt consistency learning (PCL). CCL exposes prompts to all classifiers during training, while PCL improves prediction robustness and prompt selection using random prompt sampling and a multi-key mechanism. This design achieves more consistent and effective rehearsal-free continual learning.

HidePrompt : HiDe-Prompt [17] proposes a hierarchical decomposition framework for prompt-based continual learn-

ing under self-supervised pretraining. It disentangles the continual learning objective into three components—within-task prediction, task-identity inference, and task-adaptive prediction—and optimizes them jointly using task-specific prompts and structured contrastive regularization. This approach enhances robustness to pretraining paradigms and improves generalization in task-incremental scenarios.

NoRGa : NoRGa [10] is a prompt-based continual learning framework that reinterprets prefix tuning as adding task-specific experts to a pre-trained mixture-of-experts model. By embedding non-linear residual gating into the prefix scoring mechanism, NoRGa improves parameter estimation and sample efficiency while preserving model compactness and mitigating catastrophic forgetting.

2.3. Evaluation Metric

Evaluation Metrics. We adopt three standard metrics to assess continual learning performance: Final Average Accuracy (FAA), Cumulative Average Accuracy (CAA), and Forgetting Measure (FM). Let $a_{i,t}$ denote the accuracy on task $\mathcal{T}_i$ after learning task $\mathcal{T}_t$, and define the average accuracy after learning t tasks as $A_t = \frac{1}{t}\sum_{i=1}^{t} a_{i,t}$. Then, FAA is computed as $A_T = \frac{1}{T}\sum_{i=1}^{T} a_{i,T}$, CAA is given by $\frac{1}{T}\sum_{t=1}^{T} A_t = \frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{t}\sum_{i=1}^{t} a_{i,t}\right)$, and FM is calculated as $\frac{1}{T-1}\sum_{i=1}^{T-1} \max_{1\le t<T}\left(a_{i,t} - a_{i,T}\right)$. FAA is the primary metric to evaluate final performance, CAA reflects performance across the entire learning process, and FM quantifies the extent of forgetting.

2.4. Implementation Details

For each setting, we report the average performance over three independent runs using different random seeds to account for training variance. In more detail, L2P [20] is configured with a total of $N = 30$ prompts, a prompt length of $L_p = 5$, and Top-K key selection with $K = 5$. For DualPrompt [19], we use a prompt length of $L_g = 5$ for the task-shared prompts $\boldsymbol{g}$, which are inserted into layers 1 and 2, and a prompt length of $L_e = 20$ for the task-specific prompts $\boldsymbol{e}$, inserted at layers 3 through 5. S-Prompt++ [18] follows a similar configuration to DualPrompt, but replaces all task-shared prompts with task-specific ones; specifically, the task-specific prompts are inserted into layers 1-5 with a prompt length of $L_e = 20$. For CODA-Prompt [15], we use $N = 100$ prompts, each of length $L_p = 8$, inserted into layers 1-5. HiDe-Prompt [17] and NoRGa [10] shares the overall architectural design of S-Prompt++, yet differs by substituting the task-specific keys with an auxiliary classifier $\hat{h}_\omega$ for task identification. Hash modifies the HiDe-Prompt architecture by adopting a mixture-of-experts framework, where the Top-K experts with $K = 2$ are selected for each input. We note that our reported results are based on inference with batch size 1. We have verified that performance remains consistent across different inference batch sizes, but omit these additional configurations for brevity.

3. Extended Results

3.1. Long Sequence Incremental Analysis

We also present results under the challenging 50-session setting. As shown in Figure 1, our method maintains strong performance even with 50 sessions, exhibiting only a minor drop in accuracy compared to the 20-session setting. Notably, the performance degradation is slower than other prompt-sharing methods, highlighting the anti-forgetting advantage of our history-aware module.

3.2. Comparison with Adapter-based Methods

We compare our method with several recent adapter-based continual learning approaches. LAE [5] and ADAM [23] propose unified frameworks for parameter-efficient tuning (PET) via model ensembling and modular adaptation. C-ADA [6] introduces Continual Adapter Layers with a Scale-and-Shift module to enable task-specific attention and feature transformation. EASE [22] achieves rehearsal-free continual learning by iteratively adapting multiple adapter modules through repeated forward propagation, while InfLoRA [12] leverages low-rank adaptation with an online class-specific memory to stabilize task transitions.

As shown in Table 1, our method achieves a Final Average Accuracy (FAA) of 95.02% on Split CIFAR-100, surpassing the next best method, InfLoRA, by 6.71%. On the fine-grained Split CUB-200 benchmark, our approach attains 91.34% FAA, 5.5% higher than ADAM + Adapter. Adapter-based methods provide parameter efficiency and modularity in continual learning, but often require careful placement. In contrast, our prompt-based method uses dynamic prompt composition and instance-adaptive routing, allowing for more flexible and precise knowledge sharing and task adaptation..

3.3. Comparison with Pre-trained Model-based Methods

Recent research has demonstrated that continual learning can greatly benefit from pre-trained models (PTMs), especially when paired with parameter-efficient tuning (PEFT) strategies. Methods such as ADAM [23] and RanPAC [13] typically adapt the backbone model only during the first task, employing techniques like FiLM or prompt tuning. By leveraging strong pre-trained representations, these approaches often attain high accuracy on the initial task sequence.

However, these methods exhibit notable limitations when presented with new tasks, as the backbone remains fixed after the initial adaptation. This constraint can hinder the model's ability to acquire and disentangle features specific

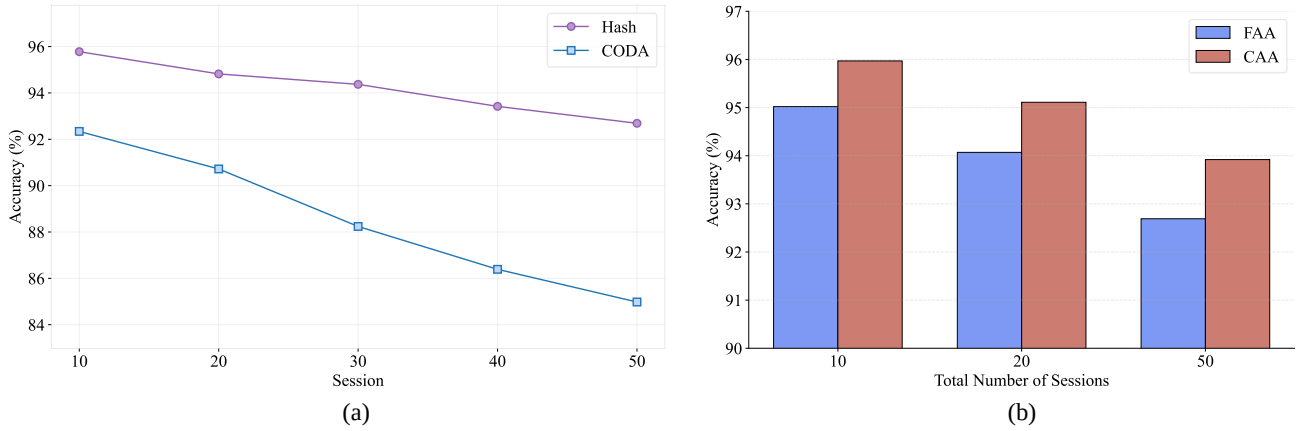

Figure 1. Long sequence incremental analysis: (a) Test accuracy after every 10 sessions in the 50-session setting on Split CIFAR-100, comparing Hash and CODA. (b) Test accuracy across different total session numbers.

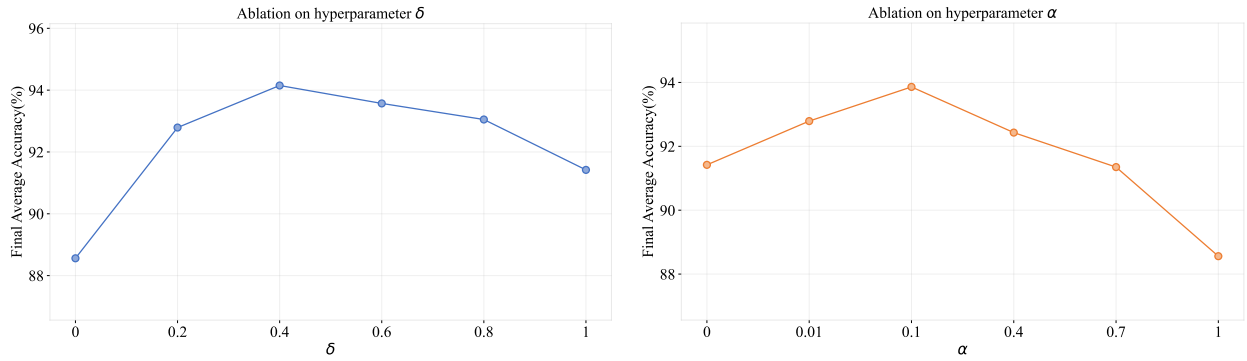

Figure 2. Ablation study on the hyperparameter of history-aware module.

to novel tasks, particularly in the presence of significant distribution shift.

To address this, we propose a method that maintains a global prompt pool and employs a dynamic routing mechanism to select task-relevant prompts during both training and inference. As shown in Table 2, our approach consistently outperforms first-task adaptation baselines on both Split CIFAR-100 and Split CUB-200 benchmarks. Specifically, we achieve a Final Average Accuracy (FAA) of 95.02% on CIFAR-100, exceeding ADAM and RanPAC by 2.82%, with similar improvements observed on CUB-200 (91.34% FAA). These results highlight the effectiveness of continual prompt adaptation in overcoming the rigidity of static backbone tuning.

Table 4. Effect of different history-aware dynamic routing strategies.

HDR (ψ)	Split CIFAR-100		Split ImageNet-R	
	FAA	CAA	FAA	CAA
None	88.56	90.12	75.92	79.53
Logarithmic	91.02	93.21	76.84	80.91
Polynomial	93.12	94.38	76.95	81.05
Stepwise (Ours)	**94.15**	**95.03**	**77.03**	**81.21**

3.4. Results under Alternative Pre-training Paradigms

To complement the main paper, which reports results based on supervised ImageNet-21K (Sup-21K) pre-training, we provide additional experimental results for all compared

Table 5. Effect of different history-aware gradient modulation strategies.

HGM (γ)	Split CIFAR-100		Split ImageNet-R	
	FAA	CAA	FAA	CAA
None	88.56	90.12	75.92	79.53
Inverse	92.45	93.88	76.23	80.24
Exponential	93.24	94.19	76.37	80.45
Piecewise (Ours)	**93.86**	**94.77**	**76.49**	**80.82**

methods under alternative pre-training paradigms, including iBOT [24], DINO [2], and MoCo [3]. These results serve to evaluate the robustness and generalizability of various prompt-based continual learning approaches across diverse pre-training strategies. All experiments are conducted under the same experimental settings as described in the main paper, ensuring fair and consistent comparisons.

Across all alternative pre-training paradigms, our method consistently achieves high accuracy and maintains low forgetting rates. This demonstrates that the shared prompt and history-aware routing strategies not only ensure robust performance under different pre-trained backbones, but also offer strong scalability to diverse continual learning scenarios.

4. Additional Ablation Results

4.1. Ablation on History-Aware Modulator

We study several alternative penalty and modulation functions to validate the design choices in our history-aware routing and gradient modulation components.

Dynamic Routing. We compare three variants: (i) Logarithmic penalty $\psi(h) = \log(1 + h)$, which applies a smooth and conservative decay that favors stable reuse of frequently activated prompts; (ii) Polynomial penalty $\psi(h) = h^\gamma$, where $\gamma > 1$, enforcing stronger suppression on heavily used prompts to encourage diversity; and (iii) our adopted Stepwise penalty, which subtracts a constant value δ for top-k prompts, offering a practical trade-off between simplicity and balancing. Logarithmic tends to under-penalize, while Polynomial may over-penalize and hurt performance. Stepwise yields better load balancing without sacrificing reuse.

Gradient Modulation. We examine: (i) Inverse Scaling $\gamma(h) = 1/(1+\beta h)$, offering gradual decay and flexible control; (ii) Exponential Decay $\gamma(h) = \exp(-\beta h)$, enabling stronger suppression but more sensitive to β; and (iii) our adopted Piecewise Constant strategy, where only top-k experts are modulated with a constant factor $\alpha < 1$. Compared to continuous modulation, our approach achieves better balance between stability and plasticity while being easier to tune.

Table 6. Comparison of training times for NoGRa and Hash. All experiments were conducted on a single NVIDIA A100 GPU.

Method	CIFAR-100	ImageNet-R	CUB-200	5-Datasets
Hide	2.80h	2.67h	1.04h	24.06h
NoRGa	2.85h	2.70h	1.10h	24.23h
Hash	2.66h	2.49h	0.96h	23.52h

As illustrated in Figure 2, we select the optimal hyperparameters $\delta = 0.4$ and $\alpha = 0.1$ based on the ablation study. Empirical results (Table 4 and Table 5) are consistent with our analysis: our piecewise and stepwise strategies achieve superior performance and stability across datasets, validating their effectiveness and ease of tuning.

5. Training Cost Analysis

All experiments are conducted on a single A100 GPU, and the corresponding training times are reported in Table 6. Compared to HiDe-Prompt and NoRGa, our approach not only requires less training time but also achieves superior performance across all evaluated benchmarks. This computational efficiency stems from our architectural design choices: we employ a smaller total number of prompts through shared pooling, use shorter prompt lengths per instance due to sparse top-k selection, and inject prompts into fewer layers compared to Hide-Prompt. These design decisions substantially reduce the computational overhead during both forward and backward passes, while our dynamic routing mechanism ensures that the reduced capacity is allocated more effectively. As a result, our method offers a more efficient optimization process without sacrificing accuracy, demonstrating the advantages of parameter sharing and selective activation in achieving both computational efficiency and strong continual learning performance.

References

[1] Yaroslav Bulatov. Notmnist dataset. http://yaroslavvb.blogspot.com/2011/09/notmnist-dataset.html, 2011. Google (Books/OCR), Tech. Rep. 2

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 6

[3] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9640–9649, 2021. 6

[4] Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 386–402. Springer, 2020. 2

[5] Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11483–11493, 2023. 4

[6] Xinyuan Gao, Songlin Dong, Yuhang He, Qiang Wang, and Yihong Gong. Beyond prompt learning: Continual adapter for efficient rehearsal-free continual learning. In *European Conference on Computer Vision*, pages 89–106. Springer, 2024. 4

[7] Zhanxin Gao, Jun Cen, and Xiaobin Chang. Consistent prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28463–28473, 2024. 3

[8] 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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021. 1

[9] Alex Krizhevsky. Learning multiple layers of features from tiny images. *Citeseer*, 2009. 1, 2

[10] Minh Le, Huy Nguyen, Trang Nguyen, Trang Pham, Linh Ngo, Nhat Ho, et al. Mixture of experts meets prompt-based continual learning. *Advances in Neural Information Processing Systems*, 37:119025–119062, 2024. 4

[11] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278–2324, 1998. 2

[12] 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. 4

[13] Mark D McDonnell, Dong Gong, Amin Parvaneh, Ehsan Abbasnejad, and Anton Van den Hengel. Ranpac: Random projections and pre-trained models for continual learning. *Advances in Neural Information Processing Systems*, 36:12022–12053, 2023. 4

[14] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 4. Granada, 2011. 2

[15] 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. 3, 4

[16] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. *California Institute of Technology*, 2011. 2

[17] Liyuan Wang, Jingyi Xie, Xingxing Zhang, Mingyi Huang, Hang Su, and Jun Zhu. Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. *Advances in Neural Information Processing Systems*, 36: 69054–69076, 2023. 1, 3, 4

[18] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *Advances in Neural Information Processing Systems*, 35:5682–5695, 2022. 4

[19] 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. 3, 4

[20] 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. 3, 4

[21] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 2

[22] Da-Wei Zhou, Hai-Long Sun, Han-Jia Ye, and De-Chuan Zhan. Expandable subspace ensemble for pre-trained model-based class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23554–23564, 2024. 4

[23] Da-Wei Zhou, Zi-Wen Cai, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need. *International Journal of Computer Vision*, 133:1012–1032, 2025. 4

[24] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. Image bert pre-training with online tokenizer. In *International Conference on Learning Representations*, 2022. 6