Teaching Prompts to Coordinate: Hierarchical Layer-Grouped Prompt Tuning for Continual Learning

Shengqin Jiang, Tianqi Kong, Yuankai Qi, Haokui Zhang, Lina Yao, Quan Z. Sheng, Qingshan Liu, Ming-Hsuan Yang

Abstract—Prompt-based continual learning methods fine-tune only a small set of additional learnable parameters while keeping the pre-trained model's parameters frozen. It enables efficient adaptation to new tasks while mitigating the risk of catastrophic forgetting. These methods typically attach one independent task-specific prompt to each layer of pre-trained models to locally modulate its features, ensuring that the layer's representation aligns with the requirements of the new task. However, although introducing learnable prompts independently at each layer provides high flexibility for adapting to new tasks, this overly flexible tuning could make certain layers susceptible to unnecessary updates. As all prompts till the current task are added together as a final prompt for all seen tasks, the model may easily overwrite feature representations essential to previous tasks, which increases the risk of catastrophic forgetting. To address this issue, we propose a novel hierarchical layer-grouped prompt tuning method for continual learning. It improves model stability in two ways: (i) Layers in the same group share roughly the same prompts, which are adjusted by position encoding. This helps preserve the intrinsic feature relationships and propagation pathways of the pre-trained model within each group. (ii) It utilizes a single task-specific root prompt that learns to generate sub-prompts for each layer group. In this way, all sub-prompts are conditioned on the same root prompt, enhancing their synergy and reducing independence. Extensive experiments across four benchmarks demonstrate that our method achieves favorable performance compared with several state-of-the-art methods.

Index Terms—continual learning, prompt tuning, position encoding, catastrophic forgetting.

1 Introduction

Continual learning (CL) seeks to emulate the human ability to dynamically adapt to new environments while gradually accumulating knowledge from continuously evolving data streams. This task addresses the limitations of traditional static models [2], [3], which struggle to cope with non-stationary data distributions. Yet, it introduces a central challenge: catastrophic forgetting. During incremental learning, model parameters often shift toward new tasks, leading to rapid loss of previously acquired knowledge. This challenge requires the model to maintain sufficient plasticity to acquire new information while preserving enough stability to retain prior knowledge.

Recently, prompt tuning has emerged as an efficient and effective way for CL [4]. It introduces a small number of trainable parameters to guide the pre-trained model in adapting to new tasks, preserving the model's representational capacity while enabling efficient adaptation at low computational cost. Early prompt tuning methods [4]–[6] typically build a task-specific prompt pool and select prompts for each input via key–value matching. Unfortunately, this hard selection could lead to suboptimal prompt choices when matching is inaccurate, degrading prediction accuracy. To improve the adaptability of pre-trained models on incremental tasks, more recent works [1], [7] typically assign independent task-specific prompts to several consecutive layers, allowing the model to flexibly absorb new knowledge, as illustrated in Fig. 1(a). Building on this layered design, they further adopt a weighted fusion strategy to combine prompts from previously learned tasks into a unified prompt representation for all seen tasks. However, the high flexibility introduced by layer-wise independent prompts is a double-edged sword. Although this design improves the model's ability to adapt to new tasks, it may trigger excessive updates to prompts in certain layers. Because prompts from all previous tasks are ultimately fused into a single prompt for all seen tasks, these uncontrolled updates may overwrite features essential for earlier tasks, thereby increasing the risk of catastrophic forgetting.

To tackle this issue, we propose a hierarchical layer-grouped prompt tuning method for CL. Instead of adding prompts independently to each layer, our method dynamically generates sub-prompts, as shown in Fig. 1. It seeks to promote coordinated updates across layer groups and limit abrupt shifts of a single layer within a group, thereby reliably preserving the crucial representations learned from previous tasks. Specifically, we first construct task-specific root prompts for each task and use intermediate adapters to generate shared implicit prompts for each group of layers. Second, the position incentive embedding is introduced,

- *S. Jiang and T. Kong are with the School of Computer Science, Nanjing University of Information Science and Technology, Nanjing, 210044, China.*
- *Y. Qi and Q. Z. Sheng are with the School of Computing, Macquarie University, Sydney 2113, Australia.*
- *H. Zhang is with School of Cybersecurity, Northwestern Polytechnical University, Xi'an 710072, China.*
- *L. Yao is with University of New South Wales, Sydney, NSW, Australia and CSIRO's Data61, Sydney, NSW, Australia.*
- *Q. Liu is with School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China.*
- *Ming-Hsuan Yang is with the University of California at Merced, Merced, CA 95343 USA.*

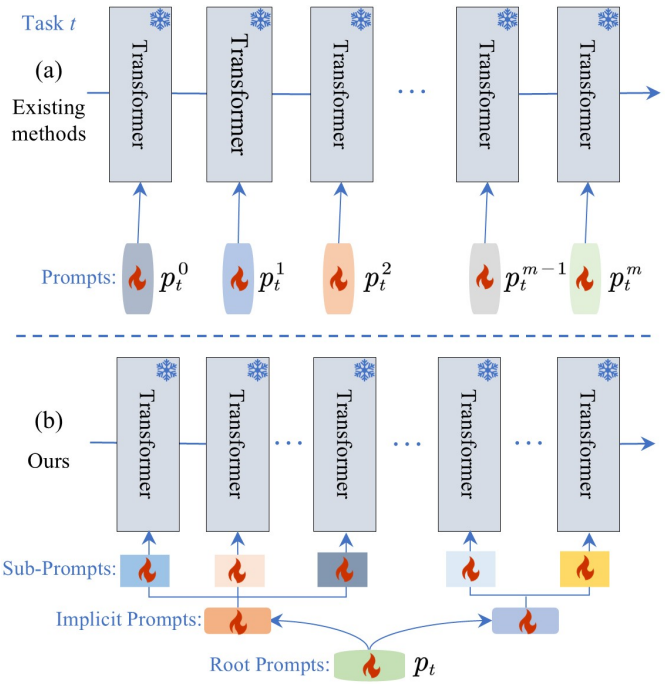

Fig. 1: Comparison between conventional independent prompt tuning (e.g., [1]) and our grouped prompting. Existing methods typically fine-tune pre-trained Transformer layers with independent prompts. Although this design offers high flexibility for learning new tasks, it can make critical representations in certain layers of old tasks susceptible to unnecessary updates, thereby exacerbating catastrophic forgetting. In contrast, our method addresses this issue by generating shared implicit prompts.

which enables prompts to be aware of the positional structure of layers within a group during fine-tuning. Following [1], we employ a soft task matching strategy to perform a weighted fusion of sub-prompts. Extensive experiments demonstrate the effectiveness of our method, consistently outperforming state-of-the-art (SoTA) methods across four public datasets. For instance, our method achieves a final average accuracy of 97.59%, and an average accuracy of 98.24% across all tasks on CIFAR-100, setting new SoTA. The contributions of this paper are threefold.

- We propose a hierarchical layer-grouped prompt tuning method for CL that derives sub-prompts from task-specific root prompts. This method offers a relatively consistent cross-layer prompt modulation scheme. It helps mitigate the over-updating of prompts in a single layer, reducing the risk of forgetting knowledge from previous tasks.
- We introduce position incentive embedding to strengthen prompts' awareness of layer order within each group, further improving prompt-tuning performance.
- Extensive experiments on four CL benchmarks demonstrate that our method performs favorably against several SOTA methods. Moreover, compared to the baseline, our method not only boosts model performance but also reduces the number of parameters that need training.

2 Related Work

2.1 Continual Learning

In CL, the models continually acquire new knowledge while minimizing the loss of previously learned information, a challenge known as catastrophic forgetting. Traditional methods tackle this by implementing strategies that balance the plasticity and stability of models, which can be generally categorized into three types: replay-based methods, regularization-based methods, and network structure-based methods. Replay-based methods alleviate forgetting by retaining a subset of representative samples from previous tasks [8], [9]. Regularization-based methods mitigate forgetting by introducing constraint terms to the loss function, thereby regulating parameter updates [10], [11]. While both approaches stabilize model performance, their stringent parameter constraints often limit the model's adaptability and flexibility in learning new tasks. Network structure-based methods, on the other hand, dynamically expand the network architecture to accommodate new tasks, preventing the overwriting of old knowledge [12]. Although these methods effectively reduce forgetting, the linear increase in model parameters with the number of tasks results in significant storage and computational costs.

2.2 Prompt-based Continual Learning

In recent years, fine-tuning techniques for pre-trained models have gained significant attention due to their strong generalization capabilities [13]. In this context, prompt-based continual learning has emerged as a prominent approach [14], [15], effectively guiding pre-trained models to adapt to incoming data streams by inserting elaborately designed prompts into Transformer modules. This method enables favorable performance without the need to replay previous samples, thereby driving significant advancements in the field of continual learning.

Initially, visual prompt fine-tuning [4] was applied to pretrained models by building a learnable prompt pool for dynamically selecting the most suitable prompt for different samples. DualPrompt [16] introduced general prompts and expert prompts on this basis to capture task-shared and task-related knowledge, respectively. Then, S-Prompt [17] learned prompts independently across domains, allowing the prompts to be optimized for each domain individually. ConvPrompt [7] introduced layer-specific prompts by applying convolution to task-shared embeddings, thereby enhancing knowledge transfer across tasks. More recently, the dynamic weighted fusion is employed [1] to merge historical frozen prompts and learnable prompts in the current task via a cyclic weighted aggregation.

Although these methods have consistently driven improvements in network performance, their designs still have limitations: they typically insert mutually independent prompts into Transformer layers. While independently learned prompts offer substantial flexibility for new tasks, this flexibility can also trigger undesired updates in certain layers during task adaptation. Such over-updating risks overwriting feature representations essential for previous tasks, thereby increasing the risk of catastrophic forgetting. To address this, we propose a hierarchical layer-grouped prompt tuning method. It maintains the intrinsic feature

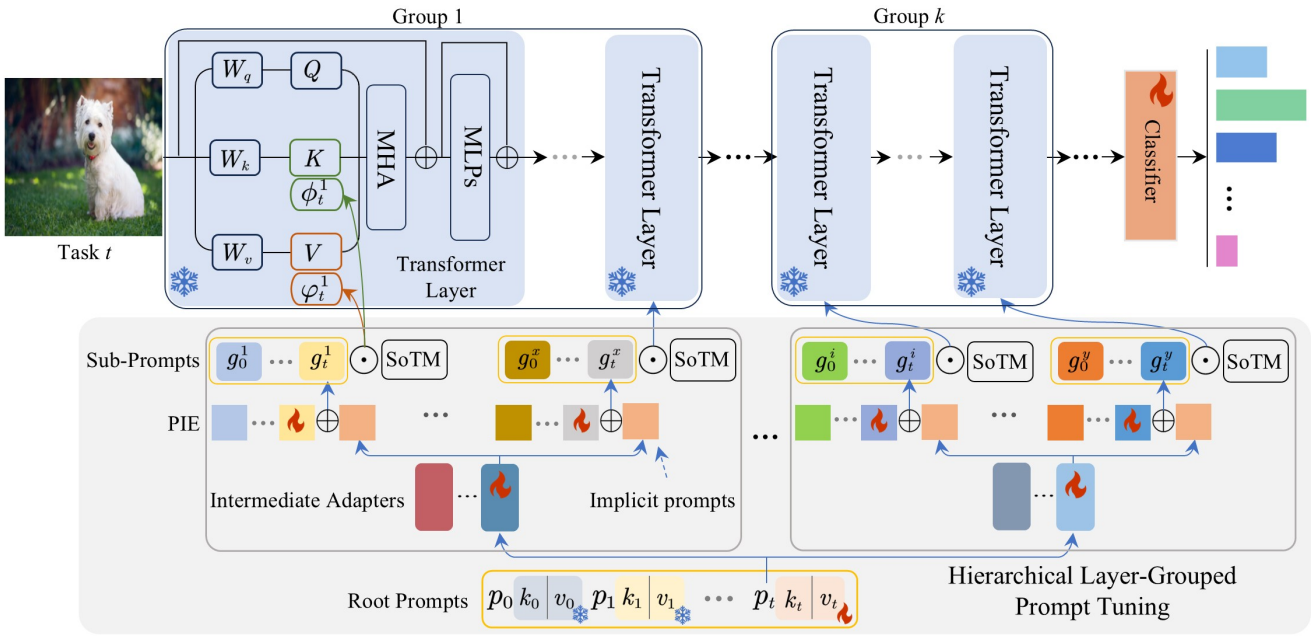

Fig. 2: Overview of our method for CL. The continual learning framework adapts a pre-trained model through prompt tuning, allowing it to acquire new knowledge while preserving previously learned information. Instead of the conventional approach of constructing prompts independently, we propose a hierarchical layer-grouped prompt tuning method (Sec. 3.1) that dynamically produces a set of shared implicit prompts for each group of layers from a task-specific prompt using intermediate adapters, enhancing the synergy of prompts across layer groups and within each group. Building on this, we introduce positional incentive embeddings (PIE) (Sec. 3.2), which enable prompts to recognize their sequential order within a group. Finally, a soft task matching (SoTM) method (Sec. 3.3) is employed to weight the sub-prompts, improving overall network performance.

relationships and propagation pathways of the pre-trained model by fostering coordinated updates across layers, enhancing overall model stability.

3 Method

Task Formulation. In CL, data typically arrive as a stream composed of $\mathcal{D}$ sequential tasks, i.e., $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_t\}$. Each task $\mathcal{D}_i$ consists of a training set $\mathcal{D}_i = \{(x_j, y_j)\}_j^{N_i}$, where N_i denotes the number of instances in task i, and $x_j \in \mathcal{X}^i$, $y_j \in \mathcal{Y}^i$ represent the sample and corresponding label, respectively. Note that the label spaces of different tasks are disjoint, i.e., $\mathcal{Y}^n \cap \mathcal{Y}^m = \varnothing$. At each incremental stage, the model can only access data from the current task, and the task identity is unknown during testing.

We follow the existing CL methods [1], [18] with a pre-trained visual transformer (ViT) as the feature extractor Ψ_t. A classifier W_t and a set of task-specific learnable prompt tokens $p = \{p_0, p_1, ..., p_t\}$ are also included to be competent for learning incremental tasks, where each $p_i \in \mathbb{R}^{L \times D}$ denotes the prompt associated with task i, with L and D representing the prompt length and dimension, respectively. The learnable prompts consist of both key tokens $p_i^k \in \mathbb{R}^{L \times D}$ and value tokens $p_i^v \in \mathbb{R}^{L \times D}$ as shown in Fig. 2, which are injected into the multi-head attention layers.

Method Overview. In this paper, we propose a simple yet effective prompt tuning method to improve model performance in CL scenarios. The overall framework is illustrated in Fig. 2. Our method divides consecutive layers into several groups, and generates a shared implicit prompt for each group from a task-specific root prompt. This design coordinates prompt updates across groups to prevent large shifts of a single layer within each group from overwriting representations critical to prior knowledge, thereby preserving information from previous tasks more effectively. Moreover, we introduce a positional incentive embedding based on the layer order within each group, which is added to shared implicit prompts and yields the final sub-prompts for each layer. This enables the sub-prompts to be aware of the order of layers and, correspondingly, adjust prompts to introduce mild diversity. The following sections describe each component of the proposed method in detail.

3.1 Hierarchical Layer-Grouped Prompt Tuning

In CL tasks, a promising approach is to introduce learnable prompts that interact with features at various depths of a pre-trained model, dynamically modulating their representations to direct the model toward task-relevant semantic information. This strategy enables the model to preserve previously acquired knowledge while continuously adapting to new tasks. These satisfactory results stem from its ability to flexibly adapt to new tasks through layer-wise prompts. However, such flexibility could drive the model to over-update certain layers to fit the current task, which risks overwriting features from previous tasks and increasing the likelihood of forgetting. To overcome this, we introduce a hierarchical layer-grouped prompt tuning method. At its core, we leverage single task-specific prompts to dynami-

cally generate shared implicit prompts for each group of layers.

For simplicity, we illustrate our approach using a single task-specific prompt, referred to as the root prompt p_t. This root prompt is further decoupled into a key prompt k_t and a value prompt v_t, which are respectively injected into the key and value pathways of each frozen Transformer layer. This design aims to dynamically guide the model to learn feature representations highly relevant to the current task from its pre-trained knowledge. In contrast to existing approaches that typically concatenate or simply add prompt vectors directly to hidden states, our method introduces learnable intermediate adapters that project root prompts into distinct Transformer layers. Here, we partition the network layers into distinct groups. For instance, a three-group partitioning mirrors the hierarchical structure of networks, comprising low-, mid-, and high-level stages. Critically, rather than employing the conventional practice of assigning independent prompts to each layer, we share a common implicit prompt, derived from a root prompt, among all layers within the same group. Consequently, the implicit prompts for each group can be formulated as follows:

$$\theta_t^l = \Psi_t^l\left(k_t\right), \chi_t = \Upsilon_t^l\left(v_t\right), \tag{1}$$

where $l = 1, 2, ..., n(< m)$ denotes the index of groups, $\Psi_t^l\left(\cdot\right)$ and $\Upsilon_t^l\left(\cdot\right)$ denote the projection functions in intermediate adapters. Inspired by [19], we employ a parameter-efficient adapter to project the root prompt.

This design offers three advantages: First, from the perspective of parameter efficiency, the sharing mechanism significantly reduces the number of parameters in the prompts, alleviating the optimization difficulty when fine-tuning the model on incremental data. Second, from the viewpoint of representation learning, the prompt shared across layers forces the model to dynamically adjust its transfer strategy for pre-trained knowledge during the learning process, based on the feature granularity extracted by different layers (e.g., shallow-level textures and deep-level semantics), thereby enhancing the model's ability to learn cross-layer, multi-granularity feature structures for the current task. This could prevent a certain layer in the network from overwriting old knowledge due to over-optimization, thereby effectively reducing the risk of catastrophic forgetting. Finally, from the aspect of information flow, such a tree-like prompt architecture, which originates from the root prompt and branches out to each layer via intermediate adapters, establishes an implicit communication bridge between layers, facilitating the overall optimization consistency of the model during the transfer learning process.

3.2 Position Incentive Embedding

To strengthen the model's ability to characterize task-specific data, we introduce a position incentive embedding. This embedding assigns independent positional encodings to the implicit prompts in each layer, allowing the model to explicitly capture the relative positional relationships among prompts. Such a design not only preserves the intrinsic positional-awareness structure of the pre-trained model but also alleviates distribution shifts introduced by the additional prompts, thereby enhancing the model's efficiency in leveraging pre-trained knowledge for new tasks. It is worth noting that although each layer contains two distinct implicit prompts, they share the same positional encoding β_t^i in this study. Accordingly, the implicit prompt $g_t^i = \left\{\phi_t^i, \varphi_t^i\right\}$ for each layer is defined as bellow:

$$\phi_t^i = \theta_t^l + \beta_t^i, \varphi_t^i = \chi_t^l + \beta_t^i, \tag{2}$$

where $i = 1, 2, ..., m$ denotes the index of Transformer layers.

3.3 Soft Task Matching

Due to the use of a task-specific prompts, the model is unable to directly identify the task ID during inference, which may lead to decreased prediction accuracy. To mitigate this limitation, we adopt a soft task matching method that dynamically adjusts the prompt allocation weights based on the model's inference outputs. Unlike [1], which directly weight the explicit prompts, our method emphasizes weighting the sub-prompts. Specifically, we first initialize the historically frozen sub-prompts and the current prompts using an average-weighted scheme determined by the total number of tasks. Next, we aggregate the network's inference probability distributions across the task dimension to compute task weights. These weights are then used to perform a weighted fusion of the sub-prompts, yielding the final inference results. With this prompt strategy, the model is capable of achieving state-of-the-art performance through an efficient two-stage inference process.

4 Experiment

4.1 Experimental Setup

Datasets. Our experiments are conducted on standard incremental learning benchmark widely used for pre-trained ViT–based continual learning, including CIFAR-100 [20], ImageNet-R (IN-R) [21], ImageNet-A (IN-A) [22], and VTAB [23]. Among them, VTAB contains 50 classes, CIFAR-100 includes 100 classes, and both IN-R and IN-A consist of 200 classes. The classes for each task are divided according to the total number of tasks. Furthermore, we evaluate several pre-training paradigms on ImageNet-21K [24] and ImageNet-1K [5], including Sup-21K, iBOT-21K [25], and DINO-1K [26].

Baselines. Our method is compared against several existing prompt-based methods, including L2P [4], Dual-P [16], CODA-P [5], CA-P [1], S-P [17], and HiDe-P [27]. Furthermore, we include comparisons with adapter-based methods, such as SCIL [28], ADAM [28], O-LoRA [29], InfLoRA [30], PLAN [31], ACMap [32], and SEMA [19].

Implementation Details. All experiments are conducted using the PyTorch framework, following the experimental setups in [1]. A pre-trained Vision Transformer (ViT) [33] is adopted as backbone, whose parameters are initialized from ImageNet-21K pre-training. Model optimization is performed using the Adam optimizer with an initial learning rate of 3e-3 and a batch size of 24. The prompt length L_p is set to 20 for the ImageNet-R dataset and 10 for all other datasets. The rank r of the low-rank matrix in the intermediate adapter is fixed at 16. For each new task, the prompt

TABLE 1: Performance of our method compared to SOTA methods on different datasets.

Method	CIFAR100		5-Task IN-R		10-Task IN-R		20-Task IN-R		5-Task IN-A		VTAB	
	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA
SCIL [28]	81.26	87.13	54.33	59.70	54.33	61.12	54.33	61.92	49.44	60.50	84.38	85.99
ADAM [28]	87.47	92.18	70.58	77.28	69.18	76.71	67.30	75.08	49.57	60.53	84.35	85.95
L2P [4]	85.02	89.51	65.83	72.90	69.75	74.55	65.82	74.49	39.30	46.67	63.56	79.17
Dual-P [16]	85.63	90.39	68.81	73.91	67.18	73.10	68.88	73.67	48.78	58.35	77.58	88.11
CODA-P [5]	86.20	91.01	74.68	79.78	73.05	79.15	65.32	70.36	37.06	50.73	85.85	85.13
O-LoRA [29]	83.41	89.05	73.12	77.33	65.74	72.89	59.94	68.92	-	-	-	-
InfLoRA [30]	86.73	91.71	76.77	81.75	74.72	81.38	69.65	76.97	41.61	56.84	86.52	89.61
PLAN [31]	87.54	92.21	77.79	81.93	75.25	80.41	71.06	77.93	-	-	-	-
ACMap [32]	87.81	92.04	74.92	80.00	70.49	77.31	70.49	77.31	**56.19**	65.19	87.56	91.21
SEMA [19]	87.84	92.23	77.13	83.27	74.82	81.39	69.69	77.84	51.35	62.50	90.86	91.99
CA-P [1]	95.35	97.03	79.16	81.74	80.22	83.38	78.27	83.19	-	-	-	-
Ours	**97.59**	**98.24**	**82.68**	**83.29**	**84.10**	**85.49**	**82.70**	**85.00**	55.16	**67.50**	**93.00**	**93.67**

TABLE 2: Performance of our method of different pre-trained weights compared to SOTA methods.

Method	CIFAR-100						IN-R					
	Sup-21K		iBOT-21K		DINO-1K		Sup-21K		iBOT-21K		DINO-1K	
	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA	FAA	CAA
L2P [4]	83.06	88.27	79.00	85.13	70.63	79.71	63.65	67.25	55.35	68.81	57.40	64.09
Dual-P [16]	86.60	90.64	78.76	86.16	74.90	81.85	68.79	71.96	54.55	66.61	58.57	65.39
CODA-P [5]	86.94	91.57	80.83	87.02	77.50	84.81	70.03	74.26	61.22	66.76	63.15	69.73
S-P [17]	88.81	92.25	79.14	85.85	74.97	81.60	69.68	74.25	55.16	66.43	57.64	64.90
HiDe-P [18]	92.61	94.03	93.02	94.56	92.51	94.25	75.06	76.60	70.83	73.23	68.11	71.70
CA-P [1]	95.35	97.03	92.48	95.58	91.57	93.95	80.22	83.38	77.09	80.19	72.21	77.07
Ours	**97.59**	**98.24**	**96.29**	**96.91**	**95.03**	**96.26**	**84.10**	**85.49**	**77.30**	**80.19**	**74.70**	**78.24**

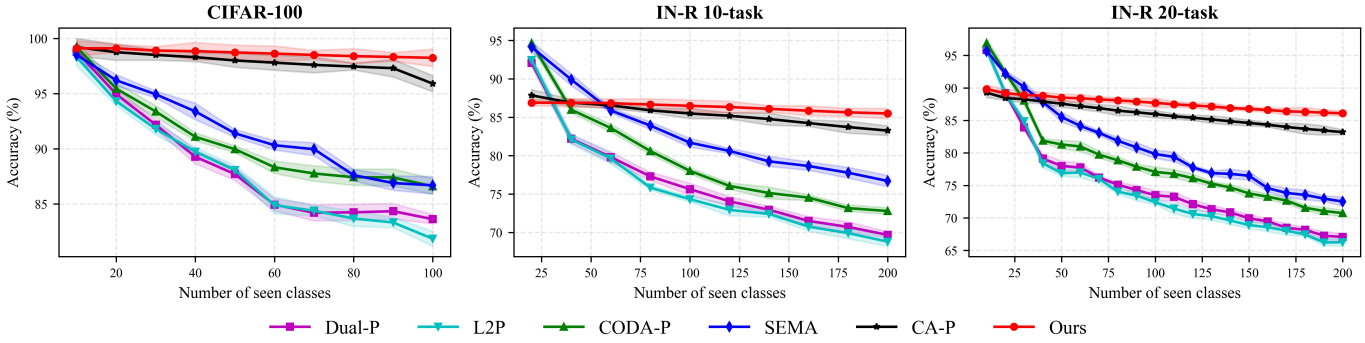

Fig. 3: Final average accuracy of different methods after each incremental task.

parameters and the weights of the generation network are initialized using the corresponding prompt and mapping network weights from the preceding task. During training, sub-prompts are embedded in all Transformer layers, with parameter sharing implemented across four layers.

Evaluation Metrics. Following existing studies [19], we adopt the final average accuracy (FAA) and continual average accuracy (CAA) as evaluation metrics to enable a comprehensive comparison of different methods. In the ablation studies, we further introduce the average forgetting (AF) [27] to quantify performance degradation across tasks. In addition, we report the number of parameters (Param) to evaluate the computational complexity introduced by additional parameters.

4.2 Comparison with SOTA Methods

In this section, we conduct a series of experiments to validate the effectiveness of our proposed method. Table 1 presents the comparison results of our method with some recent SOTA methods across four benchmarks. Whether compared with prompt-based methods or adapter-based methods, our method demonstrates superior performance in terms of FAA and CAA. Taking the CIFAR-100 dataset as an example, when compared with CA-P [1], the second-best performing method (which achieves a final accuracy exceeding 95%), our method outperforms it by 2.35% in FAA and 1.25% in CAA. On the more challenging ImageNet-R dataset, our method not only significantly surpasses the baseline model but also outperforms the second-best

method by margins of 4.45%, 4.84%, and 5.66% in final average accuracy under the 5-, 10-, and 20-task settings, respectively, while consistently maintaining a leading position in incremental accuracy.

This advantage primarily arises from the fundamental difference between our method and existing methods in prompt design. Existing methods typically employ independent prompts and adapt to new tasks through layer-by-layer updates. In contrast, our method uses a root prompt to generate a set of shared implicit prompts for each group of layers, enabling coordinated optimization across layers. This design effectively reduces the risk of catastrophic forgetting caused by the over-updating of single-layer prompts. In addition, the proposed positional incentive embedding guides the features at each network layer to adapt more effectively to the current task, enabling a more robust knowledge representation. Overall, these results demonstrate that the proposed method exhibits superior stability and robustness in addressing evolving data distributions.

To assess the impact of different pretraining paradigms on model performance, we conduct additional evaluations, summarized in Table 2. The results indicate that our method consistently delivers performance gains across three widely used pretrained models. Moreover, we notice that the choice of pretrained weights significantly affects downstream task performance. On the CIFAR-100 dataset, most methods achieve the best results with Sup-21K pre-trained weights, followed by iBOT-21K, while DINO-1K produces the weakest performance. This pattern is even more pronounced on the more challenging IN-R dataset. Although our method is also affected by the pretrained weights, its performance on CIFAR-100 shows relatively minor fluctuations. On IN-R, despite some variation, our method continues to substantially outperform all other methods. These results demonstrate that our proposed method has strong adaptability.

To verify the effectiveness of the proposed method, we further perform qualitative visualizations of the final accuracy rates after each incremental task on the CIFAR-100 and IN-R datasets under various settings. As shown in Fig. 3, it demonstrates that our method achieves consistently strong performance across all incremental tasks. Specifically, on CIFAR-100, although our method performs similarly to other methods on the base task, the accuracy of competing methods drops significantly in subsequent tasks, whereas our method remains stable. This difference arises primarily from two factors: forgetting of previously learned knowledge and limited capacity to acquire new knowledge. In contrast, our method effectively preserves old knowledge while integrating new information. On the more challenging IN-R dataset, our method does not attain the highest accuracy on the base task, but maintains relatively stable performance in subsequent tasks. This further highlights its ability to balance stability and plasticity throughout incremental learning.

4.3 Ablation Study

Effectiveness of Key Components. In our method, two key components are incorporated: the hierarchical layer-grouped prompt (HLGP) and position incentive embedding (PIE). To examine their respective contributions, we conduct ablation studies on CIFAR-100 and IN-R , with results presented in Table 3. We employ CA-P [1] as the baseline, which does not use concave and linear constraints. It utilizes task-dependent prompts for incremental task fine-tuning. Building upon this baseline, we first replace independent prompts with HLGP in the second row of the table. The results show that, using only about 15% of the baseline's fine-tuning parameters, our proposed method achieves performance gains on both datasets. Notably, on IN-R, the forgetting rate is reduced to 0.18%, indicating the excellent stability of our incremental model. This improvement can be attributed to the proposed prompt's ability to explicitly produce a hierarchy of implicit prompts linked through a root prompt, thereby enabling collaborative learning of robust task representations and effectively mitigating catastrophic forgetting.

Next, we introduce PIE on top of HLGP, which models the explicit positional relationships among prompts. This strategy further guides the model to fine-tune specific layer-wise features more accurately. With this addition, the model achieved a 5.3% increase in average accuracy on IN-R and an even lower forgetting rate, further validating its high stability. Although this component introduces some additional fine-tuning parameters, the total remains 78% of the baseline. In summary, our method markedly reduces the number of trainable parameters while achieving higher accuracy and enhancing resistance to forgetting, demonstrating both efficiency and robustness in continual learning scenes.

TABLE 3: Ablation studies on the key components in our method. HLGP and PIE denote the layer-grouped prompt and position incentive position, respectively.

Method	CIFAR-100			IN-R		
	FAA	AF	Params	FAA	AF	Params
Baseline	95.9	1.48	1.92M	79.12	3.95	2.92M
+HLGP	96.2	1.04	0.28M	79.86	0.81	0.44M
+HLGP+PIE	97.59	0.38	1.28M	84.10	0.58	2.28M

Effectiveness of the Number of Shared Layers in Hierarchical Layer-Grouped Prompt. To assess the impact of the number of shared layers in HLGP on network performance, we conduct ablation studies on the CIFAR-100 and IN-R datasets. As shown in Table 4, it shows that as the number of shared layers decreases, the model parameters grow approximately linearly. This trend arises because the mapping of root prompts across different layers is achieved solely by increasing the number of intermediate adapters. In terms of performance, the fluctuations in average accuracy remain relatively small. When all 12 layers share the same sub-prompt, the model attains the smallest parameters but also the lowest accuracy. With 6 shared layers, the model achieves a favorable balance between accuracy and parameter efficiency, but its forgetting rate is slightly higher than that of the 4-layer configuration. Further reducing the number of shared layers to 1 or 2 results in a substantial increase in parameters without noticeable performance gains. These findings indicate that sharing implicit prompts can significantly reduce fine-tuning parameters while having minimal impact on model accuracy, thus facilitating faster adaptation to downstream tasks. Accordingly, we adopt a configuration

of 4 shared layers in all subsequent experiments.

TABLE 4: Ablation results of different shared layer numbers on CIFAR-100 and IN-R.

Nums	CIFAR-100			IN-R		
	FAA	AF	Params	FAA	AF	Params
1	97.60	0.29	1.59M	84.20	0.48	2.66M
2	97.53	0.22	1.33M	84.07	0.53	2.41M
4	97.59	0.38	1.20M	84.10	0.58	2.28M
6	97.53	0.39	1.16M	84.46	0.62	2.24M
12	97.40	0.33	1.12M	83.85	0.65	2.19M

Effectiveness of the Intermediate Feature Dimension of Adapters. The intermediate adapter adopts a bottleneck architecture that uses two linear layers to map root prompts to implicit prompts. We further analyze the effect of the intermediate feature dimensions on network performance, as illustrated in Fig. 4. The results show that the model achieves optimal performance when the intermediate output dimension is set to 16. When the dimension is smaller than 16, the network performance exhibits a decreasing trend, primarily because a lower dimension restricts the model's capacity to generate richer feature representations. When the dimension is further increased to 32, the network performance on the IN-R dataset declines noticeably. Therefore, we set the intermediate feature dimension of adapters to 16.

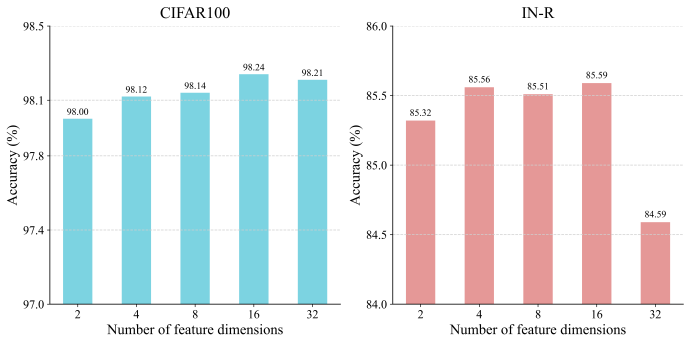

Fig. 4: Ablation study on the intermediate feature dimension of adapters.

TABLE 5: Comparison of different position embeddings. NSP, SPE, and CPE denote the non-shared position incentive embedding, sinusoidal positional encoding, and conditional positional encoding, respectively.

Method	CIFAR-100			IN-R		
	FAA	AF	Params	FAA	AF	Params
NSP	96.80	0.24	2.97M	84.40	0.40	5.81M
SPE [33]	96.07	0.48	0.28M	79.40	0.74	0.44M
CPE [34]	83.80	15.20	0.65M	62.16	22.60	0.81M
Ours	97.69	0.38	1.20M	84.10	0.58	2.28M

Effectiveness of Position Incremental Embedding. To assess the effectiveness of PIE, we compare it with several existing methods, as summarized in Table 5. We first compare PIE against a non-shared positional strategy, which assigns independent positional embeddings to each prompt. Experimental results show that this strategy slightly improves the final-stage average accuracy on the IN-R dataset but reduces performance by 0.89% on CIFAR-100. This discrepancy likely arises because, on the large-scale IN-R dataset, additional positional prompts facilitate the modeling of richer semantic information, thereby improving performance. However, on the small-scale CIFAR-100 dataset, the extra parameters may increase optimization difficulty and consequently degrade performance. In contrast, our method achieves consistently better overall results while using only half as many parameters. We further compare PIE with other positional embedding schemes, such as SPE [33] and CPE [34]. Although these methods employ even fewer parameters, they fail to produce satisfactory performance improvements. It shows that such strategies are primarily effective for augmenting positional information at the feature level and do not generalize well to the prompt level.

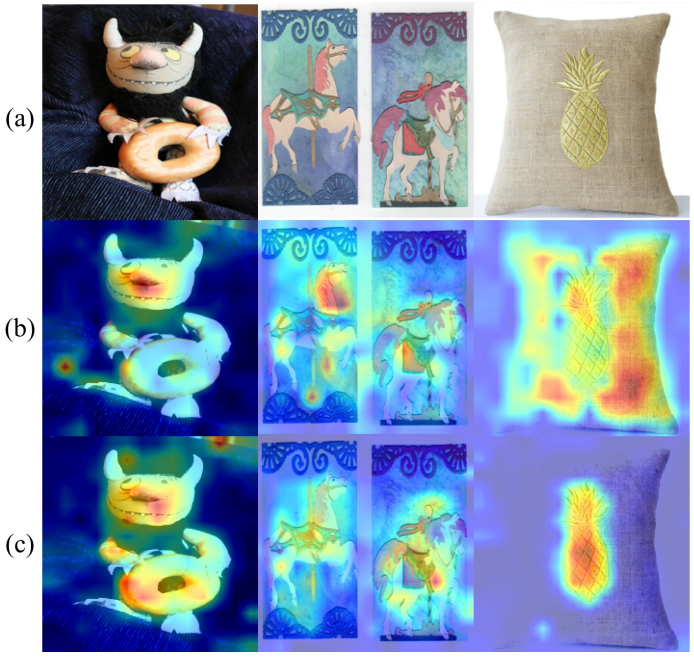

Fig. 5: Comparison of feature visualization of our method with and without PIE. We visualize the high-level semantic explanation for our method without/with PIE by attention guided CAM [35]. (a) denotes the input images, whereas (b) and (c) present the corresponding cases without/with PIE.

To assess the impact of PIE on feature extraction, we visualize three representative examples in Fig. 5. The results show that incorporating positional embedding enables deeper layers of the network to more accurately focus on regions of interest. Specifically, in the first column, without PIE, the model primarily attends to background regions and fails to recognize the bagel. In contrast, with the proposed method, the model exhibits a markedly improved focus on the target of interest, thereby reducing the likelihood of misclassification. In the second and third columns, the model without PIE simultaneously attends to both the target and background areas, whereas the inclusion of PIE directs the network's attention more precisely toward the most discriminative parts of the target. These results demonstrate

that PIE provides more effective attention guidance for feature learning across layers, thereby enhancing the model's discriminative capability.

5 Conclusion

In this paper, we propose a hierarchical layer-grouped prompt tuning method for continual learning. Different from adding prompts independently to each layer, our method generates sub-prompts and learns them coordinately across layers. This method can prevent the prompts of each layer from being over-updated, effectively protecting the acquired knowledge from catastrophic forgetting. We first construct task-specific root prompts for each task and use intermediate adapters to generate shared implicit prompts for each group of layers. Then, we introduce a position incentive embedding that increases the prompts' awareness of the order of layers within a group during fine-tuning. At last, extensive experiments confirm the effectiveness of our method, which consistently outperforms SOTA methods across several public datasets.

References

[1] Q. Li and J. Zhou, "Caprompt: Cyclic prompt aggregation for pre-trained model based class incremental learning," in *AAAI Conference on Artificial Intelligence*, T. Walsh, J. Shah, and Z. Kolter, Eds., 2025, pp. 18 421–18 429.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[3] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, J. Burstein, C. Doran, and T. Solorio, Eds., 2019, pp. 4171–4186.

[4] Z. Wang, Z. Zhang, C. Lee, H. Zhang, R. Sun, X. Ren, G. Su, V. Perot, J. G. Dy, and T. Pfister, "Learning to prompt for continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 139–149.

[5] J. S. Smith, L. Karlinsky, V. Gutta, P. Cascante-Bonilla, D. Kim, A. Arbelle, R. Panda, R. Feris, and Z. Kira, "Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11 909–11 919.

[6] Z. Gao, J. Cen, and X. Chang, "Consistent prompting for rehearsal-free continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 28 463–28 473.

[7] A. Roy, R. Moulick, V. K. Verma, S. Ghosh, and A. Das, "Convolutional prompting meets language models for continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23 616–23 626.

[8] A. Douillard, M. Cord, C. Ollion, T. Robert, and E. Valle, "Podnet: Pooled outputs distillation for small-tasks incremental learning," in *European Conference on Computer Vision*, vol. 12365. Springer, 2020, pp. 86–102.

[9] T. Kim, J. Park, and B. Han, "Cross-class feature augmentation for class incremental learning," in *AAAI Conference on Artificial Intelligence*. AAAI Press, 2024, pp. 13 168–13 176.

[10] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2935–2947, 2018.

[11] S. Magistri, T. Trinci, A. Soutif-Cormerais, J. van de Weijer, and A. D. Bagdanov, "Elastic feature consolidation for cold start exemplar-free incremental learning," in *International Conference on Learning Representations*, 2024.

[12] H. Yan, L. Wang, K. Ma, and Y. Zhong, "Orchestrate latent expertise: Advancing online continual learning with multi-level supervision and reverse self-distillation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2024, pp. 23 670–23 680.

[13] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, no. 9, pp. 195:1–195:35, 2023.

[14] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, vol. 139. PMLR, 2021, pp. 8748–8763.

[15] A. Villa, J. L. Alcázar, M. Alfarra, K. Alhamoud, J. Hurtado, F. C. Heilbron, A. Soto, and B. Ghanem, "PIVOT: prompting for video continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2023, pp. 24 214–24 223.

[16] Z. Wang, Z. Zhang, S. Ebrahimi, R. Sun, H. Zhang, C. Lee, X. Ren, G. Su, V. Perot, J. G. Dy, and T. Pfister, "Dualprompt: Complementary prompting for rehearsal-free continual learning," in *European Conference on Computer Vision*, S. Avidan, G. J. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., vol. 13686, 2022, pp. 631–648.

[17] Y. Wang, Z. Huang, and X. Hong, "S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., 2022.

[18] L. Wang, J. Xie, X. Zhang, H. Su, and J. Zhu, "Hide-pet: Continual learning via hierarchical decomposition of parameter-efficient tuning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 8, pp. 6687–6702, 2025.

[19] H. Wang, H. Lu, L. Yao, and D. Gong, "Self-expansion of pre-trained models with mixture of adapters for continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 10 087–10 098.

[20] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[21] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer, "The many faces of robustness: A critical analysis of out-of-distribution generalization," in *IEEE/CVF International Conference on Computer Vision*. IEEE, 2021, pp. 8320–8329.

[22] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song, "Natural adversarial examples," in *IEEE Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2021, pp. 15 262–15 271.

[23] X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruyssen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy *et al.*, "A large-scale study of representation learning with the visual task adaptation benchmark," *arXiv preprint arXiv:1910.04867*, 2019.

[24] T. Ridnik, E. B. Baruch, A. Noy, and L. Zelnik, "Imagenet-21k pre-training for the masses," in *Neural Information Processing Systems Track on Datasets and Benchmarks*, J. Vanschoren and S. Yeung, Eds., 2021.

[25] J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. L. Yuille, and T. Kong, "Image BERT pre-training with online tokenizer," in *International Conference on Learning Representations*, 2022.

[26] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *IEEE/CVF International Conference on Computer Vision*. IEEE, 2021, pp. 9630–9640.

[27] L. Wang, J. Xie, X. Zhang, M. Huang, H. Su, and J. Zhu, "Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., 2023.

[28] D. Zhou, Z. Cai, H. Ye, D. Zhan, and Z. Liu, "Revisiting class-incremental learning with pre-trained models: Generalizability and adaptivity are all you need," *International Journal of Computer Vision*, vol. 133, no. 3, pp. 1012–1032, 2025.

[29] X. Wang, T. Chen, Q. Ge, H. Xia, R. Bao, R. Zheng, Q. Zhang, T. Gui, and X. Huang, "Orthogonal subspace learning for language model continual learning," in *Findings of the Association for Computational Linguistics: EMNLP*, 2023, pp. 10 658–10 671.

[30] Y. Liang and W. Li, "Inflora: Interference-free low-rank adaptation for continual learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 23 638–23 647.

[31] X. Wang, Z. Zhuang, and Y. Zhang, "Plan: Proactive low-rank allo-

cation for continual learning," in *IEEE/CVF International Conference on Computer Vision*, 2025, pp. 2909–2918.
[32] T. Fukuda, H. Kera, and K. Kawamoto, "Adapter merging with centroid prototype mapping for scalable class-incremental learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Computer Vision Foundation / IEEE, 2025, pp. 4884–4893.
[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.
[34] X. Chu, Z. Tian, B. Zhang, X. Wang, and C. Shen, "Conditional positional encodings for vision transformers," in *International Conference on Learning Representations*, 2023.
[35] S. Leem and H. Seo, "Attention guided CAM: visual explanations of vision transformer guided by self-attention," in *AAAI Conference on Artificial Intelligence, AAAI 2024*, 2024, pp. 2956–2964.