Prompt Customization for Continual Learning

Yong Dai
Pengcheng laboratory
China
chd-dy@foxmail.com

Xiaopeng Hong
Harbin Institute of Technology
China
hongxiaopeng@hit.edu.cn

Yabin Wang
Xi'an Jiaotong University
China

Zhiheng Ma
Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
China

Dongmei Jiang
Pengcheng laboratory
China

Yaowei Wang
Pengcheng laboratory
China

ABSTRACT

Contemporary continual learning approaches typically select prompts from a pool, which function as supplementary inputs to a pre-trained model. However, this strategy is hindered by the inherent noise of its selection approach when handling increasing tasks. In response to these challenges, we reformulate the prompting approach for continual learning and propose the prompt customization (PC) method. PC mainly comprises a prompt generation module (PGM) and a prompt modulation module (PMM). In contrast to conventional methods that employ hard prompt selection, PGM assigns different coefficients to prompts from a fixed-sized pool of prompts and generates tailored prompts. Moreover, PMM further modulates the prompts by adaptively assigning weights according to the correlations between input data and corresponding prompts. We evaluate our method on four benchmark datasets for three diverse settings, including the class, domain, and task-agnostic incremental learning tasks. Experimental results demonstrate consistent improvement (by up to 16.2%), yielded by the proposed method, over the state-of-the-art (SOTA) techniques. The codes are released on https://github.com/Yong-DAI/PC.

CCS CONCEPTS

• **Computing methodologies → Neural networks**.

KEYWORDS

Continual learning, incremental learning, prompting, prompt customization, prompt generation, prompt modulation.

1 INTRODUCTION

Continual learning pertains to the seamless integration of new learning tasks into a unified model while preventing catastrophic forgetting of previously acquired information [9, 28, 51, 65]. Continual learning has been involved in numerous applications including image classification, audio-visual learning, and vision-language pretraining in the image, multimedia, and multimodal domains [7, 8, 15, 19, 35, 54, 64, 68]. Early continual learning methods [1, 63] safeguarded entrenched knowledge against new information through the application of regularization-based constraints to new task loss functions. The efficacy of these strategies was substantially influenced by the interplay between former and latter tasks. Subsequent methods [12, 36, 38] have endeavored to retain a part of representative old data to reinforce previous knowledge during new task learning. Nonetheless, using data buffers has issues in data privacy and memory constraints [39].

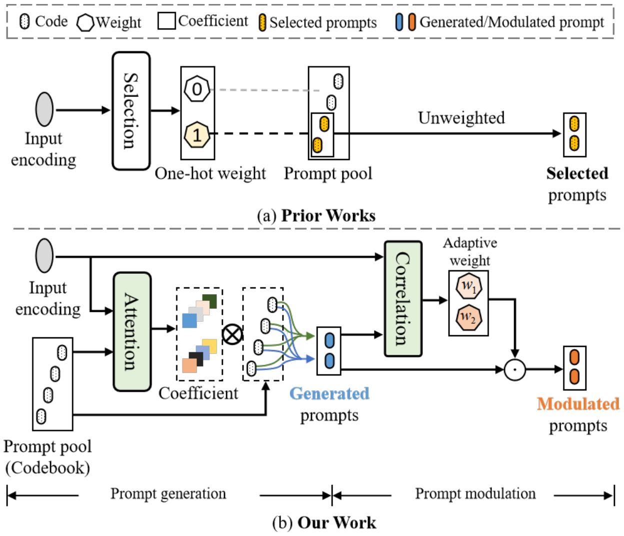


Figure 1: *Prior works* [52, 57, 58] employ a deterministic selection of prompts. *Our work* tailors specific prompts for each instance through a process of *prompt generation* from a fixed-size codebook and *soft prompt modulation.* The key differences lie in the generated prompts by a linear combination of prompts based on instance-specific coefficients and modulated prompts by further assigning adaptive weights.

Currently, top-performing methods are prompt-based methods [48, 52, 57, 58]. Prompt learning is regarded as a flexible way to adapt models by solely training additional inputs while keeping the model frozen. As Fig. 1(a) shows, they usually learn to directly select

ACM MM, 2024, Melbourne, Australia

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Table 1: Consistent performance Increase (Inc.) in terms of average accuracy (A_a, %) using a simple prompt weighting operation on L2P [58].

Datasets	Split CIFAR-100	Split ImageNet-R	CORe50	DomainNet
Inc.	1.90 ↑	3.90 ↑	1.51 ↑	6.20 ↑

prompts from a prompt pool. However, the performance of the above methods is highly dependent on this hard selection strategy, which becomes increasingly erratic as the number of tasks and the size of the prompt pool grows [14]. Moreover, these methods typically push to select the same prompt package for instances from a common task. Treating each task as a cohesive unit poses challenges due to the significant variability observed within tasks. In addition, it is also hard to provide more suitable instructions for each instance, which leads to prompts homogeneity problems [14].

To handle the above problems caused by *hard prompt selection*, we reformulate the prompting approach for continual learning. In contrast to conventional methods that employ hard prompt selection, as Fig. 1(b) shows, we propose to generate instance-specific prompts, in which the generation strategy eliminates the selection stage and instance-specific prompts ease the homogeneity problems. The instance-specific prompt generation is achieved by assigning specific soft weights (coefficients) to prompts from a fixed-sized pool of prompts (codebook), which are then linearly combined. Additionally, in our investigation, we have noted that even simple prompt weighting[1] leads to consistent performance improvement, as evidenced in Tab. 1. This observation further catalyzes the design of an adaptive prompt modulation scheme to accommodate the generated prompts with the continually evolving codebook.

Based on this understanding, we propose the prompt customization (PC) method, which comprises two key modules: a prompt generation module (PGM) and a prompt modulation module (PMM). PGM adapts a predefined codebook to individual instances and generates corresponding prompts based on the predicted generation coefficient vectors derived from attention mechanisms between inputs and the codebook. Following this step, PMM adaptively modulates these generated prompts by assigning weights predicted by their correlations between inputs and respective prompts. The entire model is optimized using a straightforward yet highly effective loss function. Experimental results on four mainstream datasets demonstrate that the proposed method outperforms the SOTA techniques significantly (up to 16.2%) across diverse tasks, including class, domain, and task-agnostic incremental tasks.

The contributions of the proposed approach are threefold:

- We propose prompt customization, a novel prompting method for continual learning. The proposed method leads to more variation and less homogeneity for prompts than hard selection and circumvents the need for prompt selection during inference, thereby mitigating the potential errors for prompt selection when dealing with challenging samples.
- We design a prompt generation module that generates finer instance-specific prompts through a linear combination of prompts from a designated codebook. The generated prompts are more distinguishable and expressive than generic task-specific prompts used in previous methods.
- We devise a prompt modulation module that further modulates the corresponding prompts with adaptive weights capitalizing on the correlations between instances and prompts and makes prompts more flexible.

2 RELATED WORK

2.1 Continual Learning

The field of continual learning has developed quickly in recent years, leading to the development of numerous methods [9, 28]. The landscape of architecture-based methodologies is characterized by their propensity to extract features from numerous intermediate layers [20, 33, 44, 55], or to extend models and fine-tune classification layers to learn new tasks [25, 41, 43, 46, 60, 61]. However, their efficacy is curtailed due to the significant forgetting of preceding tasks, coupled with the expansion in network complexity [41, 60].

In contrast, regularization-based methodologies operate by inculcating constraints into the loss functions of new tasks, thereby averting the subjugation of previous knowledge by novel information [1, 21, 27, 63]. Although the methods seem to be potential, these constraints are markedly reliant on the interrelation between old and new tasks, rendering them less suitable for scenarios involving plenty of incremental tasks or intricate data distributions in challenging datasets [56].

The contemporary favor towards rehearsal-based techniques is underscored by their setting of preserving a subset of representative data of previous tasks during the training of new tasks, facilitating the revisitation of prior knowledge [3–6, 12, 38, 40, 59]. Nevertheless, these strategies evince a pronounced susceptibility to the selection of previous data, a vulnerability that escalates linearly with the burgeoning task count [3–5, 16, 59, 62]. However, these augmentations remain inadequate in circumventing concerns pertaining to data privacy and memory constraints [47].

2.2 Prompt Tuning for Continual Learning

Prompt tuning has garnered much attention as a versatile technique for model adaptation, involving the exclusive training of supplementary inputs in a parameter-efficient way while keeping the model frozen [18, 24, 26, 29, 30, 66, 67]. This ingenious approach facilitates the model's effective reuse of learned representations, thereby circumventing the need for arduous and costly relearning from scratch [2, 11, 17, 19, 48, 52, 57, 58].

L2P stands as a pioneering endeavor in integrating prompts into the domain of continual learning, a stride marked by the elimination of rehearsal requirements. It introduces prompts as trainable parameters, enabling the direct selection of multiple prompts from a designated prompt pool [58]. DualPrompt introduces a novel partitioning of the prompt pool into two distinct prompt spaces: task-agnostic and task-specific [57]. A similar approach is adopted by S-Prompt, which independently learns task-specific prompts [52]. This segregation of tuning strategies significantly mitigates the issue of catastrophic forgetting, albeit at the cost of prompt proliferation as the number of tasks increases. The prompts for the current task are directly selected based on an index, employing a

[1] The detailed structure is given in supplementary materials.

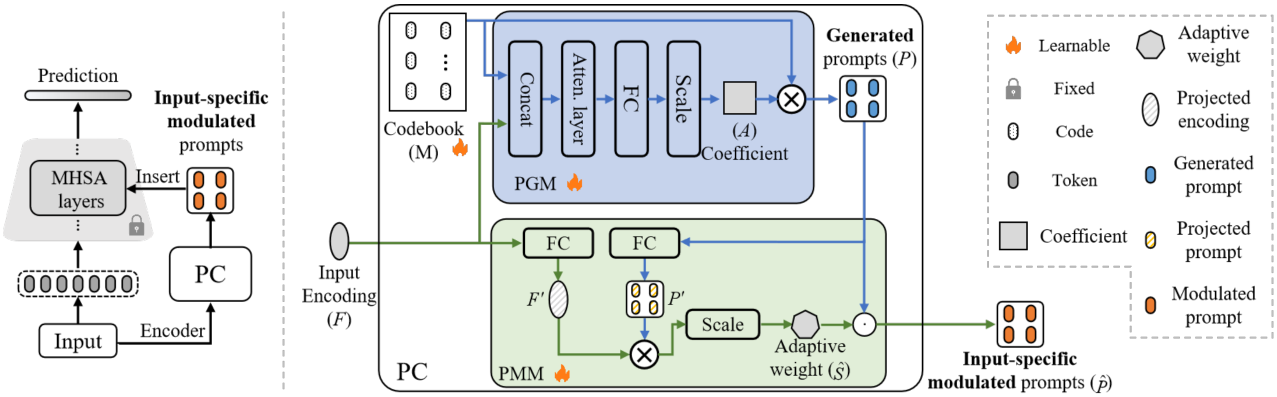


Figure 2: The framework of the proposed PC. The PC comprises two integral modules: the PGM and PMM. PGM takes input encoding and codebook as input and generates input-specific prompts through a linear combination of prompts from a designated codebook. PMM takes input encoding and the above generated prompts as input to quantize their correlations which are further utilized to modulate the corresponding prompts. The final input-specific modulated prompts will be inserted into the corresponding MHSA layers to assist the frozen backbone in performing classification tasks. Unlike the previous works, our PC circumvents the need for rigorous prompt selection during inference and generates finer instance-specific prompts which are more distinguishable and expressive. 'Atten.' and 'MHSA' mean attention and multi-head self-attention layer, respectively. For simplicity, all symbols are present without subscripts.

one-hot weighting technique. However, this selection strategy is constrained by the growing noise due to the expanding prompt pool with the incorporation of additional tasks [14]. And the design of task-level prompts also limits the adaptivity of a model to each instance without finer instructions.

Moreover, CODA-Prompt [48] enhances the one-hot weighting mechanism by introducing an attention-based prompt-assemblage scheme, yielding promising performance. Nevertheless, it treats all the assembled prompts uniformly without further adjustment, which limits the adaptiveness of prompts to instruct the model. This motivates us to design the prompt modulation module to adaptively adjust the weight for each prompt. DAP [19] lies in the scope of generating instance-level prompts and achieves superior performance. However, it also involves a selection step to search for the domain-relevant knowledge of a target task, in which the superior performance is only obtained in the batch-wise selection setup, and the performance drops sharply in the instance-wise selection setup.

To address the previously delineated concerns, the proposed approach eschews the conventional prompt selection strategy. Instead, it undertakes the task of soft prompt generation and modulation on an instance-specific basis, facilitated by a predefined fixed-size codebook. This innovative approach circumvents issues associated with noisy selection and the linear proliferation of prompts and leads to more variation and less homogeneity for prompts.

3 PROPOSED APPROACH

3.1 Problem Definition

Assuming $D = \{D_1, D_2 \cdots D_T\}$ be a sequence of tasks with T incremental tasks, where t-th task $D_{\mathrm{t}} = \left\{\left(x_i^t, y_i^t\right)\right\}_{i=1}^{n_t}$ contains tuples of data sample $x_i^t \in X$ and the corresponding ground-truth labels $y_i^t \in Y$. The continual learning is defined to train a single model $f_\theta : X \longrightarrow Y$ parameterized by θ to handle the T incremental tasks. In the inferring phase, f_θ would predict the corresponding label $y \in Y$ for the given sample x which is unseen from arbitrary tasks. Note that, the data from previous tasks may become inaccessible during the training of the current task.

3.2 Overall Framework

In this work, we introduce a novel approach called PC, which caters to the customization of instance-level prompts. This process is facilitated by the novel soft generation and modulation of prompts, underpinned by a pre-established fixed-size codebook. Specifically, PC comprises two integral modules: the PGM and PMM.

As illustrated in Fig. 2, the PGM primarily generates tailored instance-specific prompts through a linear combination of prompts from a designated codebook. This generation is executed based on a coefficient vector, which is predicted through an attention mechanism integrating inputs and the codebook. Conversely, the PMM achieves adaptive prompt modulation by assigning dynamic weights to the generated prompts, capitalizing on the correlations between instances and prompts.

Subsequently, the modulated instance-specific prompts are seamlessly integrated into multiple layers of multi-head self-attention (MHSA). These integrated prompts culminate in the generation of final predictions, accomplished through a fixed Vision Transformer (VIT) as the backbone coupled with a learnable classifier.

3.3 Prompt Generation Module

We first create a learnable **codebook** in advance of prompt generation. This codebook is denoted as M which manifests as a matrix $\mathbb{R}^{N \times L}$ composed of N fundamental codes of length L.

With the codebook in place, the PGM is formulated to facilitate codebook adaptation for each task, achieved through the generation of instance-specific prompts guided by the coefficients. The concatenation of input encoding $F_{i,t}$ and the codebook M constitute the input for PGM, where $F_{i,t}$ stems from a previously frozen pre-trained vision model, predicated on the input data x_i^t. Leveraging attention layers, PGM establishes the link between $F_{i,t}$ and M, subsequently employing one FC layer and a scaling operation to predict and normalize the corresponding coefficient vector $A_{i,t}$ for the input data x_i^t. Finally, the corresponding generated instance-specific prompts $P_{i,t}$ are obtained through the matrix product of the coefficient vector $A_{i,t}$ and the codebook M as:

$$P_{i,t} = A_{i,t} \times \mathrm{M} \tag{1}$$

where the coefficient vector $A_{i,t}$ is with the size of $\mathbb{R}^{n \times N}$, so the corresponding generated prompts $P_{i,t}$ are with the size of $\mathbb{R}^{n \times L}$ which contains n prompts of length L.

3.4 Prompt Modulation Module

Upon obtaining the n instance-specific prompts via generation, a uniform allocation of contributions is present if the prompts are employed directly. This would imply an equal distribution of influence across the current instance prediction task. However, our perspective differs from this uniform allotment, believing that the expected distribution should be dynamically adjusted. Hence, we propose a modulation strategy based on the correlations between input encoding and the aforementioned n prompts, which also makes the prompts adaptive.

Referencing Fig. 2, the PMM modulates the generated prompts through quantized correlations between input encoding and the generated prompts. In a precise breakdown, PMM initiates its operations by deploying a pair of fully connected layers with the projection dimension of $\mathbb{R}^{L \times L}$. This strategic configuration serves to delicately project the instance encoding $F_{i,t}$ and the instance-specific prompts $P_{i,t}$ into a shared representation space. The projections are respectively denoted as $F'_{i,t}$ and $P'_{i,t}$. Then PMM calculates their correlations by a matrix product operation as:

$$S_{i,t} = F'_{i,t} \times P'_{i,t} \tag{2}$$

In order to quantize the correlations, PMM employs a scale operation by a sigmoid and a linear scale operation as:

$$\hat{S}_{i,t} = \sigma((S_{i,t} - S_{i,t}^{\min}) \Big/ (S_{i,t}^{\max} - S_{i,t}^{\min})) \tag{3}$$

where $\hat{S}_{i,t}$ is the quantized correlations, σ means the sigmoid non-linear operation. $S_{i,t}^{\max}$ and $S_{i,t}^{\max}$ mean the maximum and minimum value of $S_{i,t}$. The linear scale quantizes the correlations between 0 and 1. We hold the view that the quantized correlations for the generated prompts should not be too small, hence the sigmoid operation is further set after the linear scale operation to achieve non-linear operation and make the quantized correlations bigger than 0.5. Finally, PMM modulates the prompts according to the quantized correlations:

$$\hat{P}_{i,t} = \hat{S}_{i,t} \cdot P_{i,t} \tag{4}$$

3.5 Update Strategy

The codebook serves as a universal and overarching foundation for all instance-specific prompts, effectively consolidating pivotal insights gleaned from incremental tasks. In order to aggregate key information to alleviate catastrophic forgetting, we suppose that the codebook ought to undergo stable updates. Specifically, we employ a momentum optimization [49]. We define $\mathrm{M_{t-1}}'$ at task $t-1$ as an ensemble of the current version at task $t-1$ and earlier versions of codebook M:

$$\mathrm{M}'_{t-1} = \alpha \mathrm{M}'_{t-2} + (1-\alpha)\, \mathrm{M}_{t-1} \tag{5}$$

where M_{t-1} is the learned codebook at task $t-1$, α is a smoothing coefficient which is set at 0.99 referring to [49]. We further use a regularization item to further constrain the update of M_t under the guidance of M'_{t-1}, thus, to encourage the M_t to aggregate key information of past tasks to alleviate catastrophic forgetting as:

$$\mathcal{L}_{re} = \frac{1}{N} \left\| \mathrm{M}'_{t-1} - \mathrm{M_t} \right\|_2^2 \tag{6}$$

What's more, to reduce interference between each code in the codebook, we also add an orthogonality constraint as:

$$\mathcal{L}_{or} = \left\| \mathrm{M}\mathrm{M}^{\mathrm{T}} - \mathrm{I} \right\|_2 \tag{7}$$

In contrast to conventional methods that employ a hard prompt selection approach and utilize a matching loss to optimize the distance between selected prompts and input from a common task, such a matching loss is not suitable for the proposed PC as PC eliminates the need for prompt selection. As for PC, the classification task is optimized with the cross-entropy loss $\mathcal{L}_{ce}$ as:

$$\mathcal{L}_{ce} = -\frac{1}{n_t} \sum_{i=0}^{n_t - 1} y_i^t \log\left(\hat{y}_i^t\right) \tag{8}$$

where $\hat{y}_i^t$ means final predicted labels for each input x_i^t. Thus, the overall loss function is computed as:

$$\mathcal{L} = \mathcal{L}_{ce} + \lambda \mathcal{L}_{or} + \beta \mathcal{L}_{re} \tag{9}$$

β is a scalar to weight the regularization item, and β is set at 1 here referring to [49]. The loss function is straightforward yet highly effective which is validated through extensive experiments.

4 EXPERIMENTAL SETTING AND RESULTS

4.1 Datasets and Experimental Preparation

4.1.1 Datasets. In this work, we would like to verify the validity of the proposed PC in multiple aspects, including the class, domain, and task-agnostic incremental learning settings[2]. Therefore, we follow the prior work, i.e. L2P [58], SPrompt [52], and ESN [53], and pick the representative datasets for a fair comparison. For the class-incremental task, we consider the CIFAR-100 [22] and ImageNet-Rendition (ImageNet-R) [42, 57] datasets as the base datasets and randomly split them into 5, 10, and 20 incremental classification tasks respectively for each task. For the domain-incremental task, we conduct experiments on Core50 [31] to predict the unseen domains based on incremental trained domains. And for the task-agnostic incremental learning settings, we

[2] The details are also given in supplementary materials.

Table 2: Performance comparison on the split CIFAR-100 dataset for class incremental learning setting. B_s means buffer size. * suggests results copied from the original paper, † suggests results copied from ([57]), ‡ suggests results using the corresponding codebases and calculating by the normal equation. Results for the prompt-based methods are in *instance-wise* prompts setup.

Method	B_s	5 Tasks		10 Tasks		20 Tasks	
		$A_a \uparrow$	$F \downarrow$	$A_a \uparrow$	$F \downarrow$	$A_a \uparrow$	$F \downarrow$
DER++† [3]	1000	-	-	61.06	39.87	-	-
BiC† [59]	1000	-	-	66.11	35.24	-	-
ER† [5]	1000	-	-	67.87	33.33	-	-
Co^{2}L† [4]	1000	-	-	72.15	28.5	-	-
EWC† [21]	0	-	-	47.01	33.27	-	-
LwF‡ [27]	0	61.55	25.13	60.69	27.77	56.46	28.87
L2P‡ [58]	0	83.44	6.11	82.72	7.19	79.75	7.61
DAP‡ [19]	0	-	-	83.26	8.27	-	-
DualPrompt‡ [57]	0	85.92	5.62	85.33	5.71	82.38	6.11
ESN* [53]	0	-	-	86.34	**4.76**	-	-
Coda-P‡ [48]	0	87.14	6.02	86.41	7.17	83.77	7.78
PC	0	**88.04**	**4.66**	**87.20**	5.61	**84.65**	**6.03**
Upper-bound	0	90.97	-	90.97	-	90.97	-

Table 3: Performance comparison on the ImageNet-R dataset for class incremental learning setting. B_s means buffer size. * suggests results copied from the original paper, † suggests results copied from ([57]), ‡ suggests results using the corresponding codebases and calculating by the normal equation. Results for the prompt-based methods are in *instance-wise* prompts setup.

Method	B_s	5 Tasks		10 Tasks		20 Tasks	
		$A_a \uparrow$	$F \downarrow$	$A_a \uparrow$	$F \downarrow$	$A_a \uparrow$	$F \downarrow$
DER++† [3]	1000	-	-	55.47	34.64	-	-
BiC† [59]	1000	-	-	52.14	36.7	-	-
ER† [5]	1000	-	-	55.13	35.38	-	-
Co^{2}L† [4]	1000	-	-	53.45	37.3	-	-
EWC† [21]	0	-	-	35.00	56.16	-	-
LwF‡ [27]	0	40.62	50.69	38.54	52.37	32.05	53.42
L2P‡ [58]	0	62.61	8.01	61.21	8.65	57.36	9.07
DualPrompt‡ [57]	0	67.83	**4.79**	66.47	**5.75**	63.25	**6.13**
Coda-P‡ [48]	0	75.25	6.86	74.26	7.91	71.16	8.49
PC	0	**75.41**	6.42	**74.34**	7.35	**71.44**	7.62
Upper-bound	0	79.31	-	79.31	-	79.31	-

apply an incremental training process and present the task-agnostic test average results on DomainNet [37].

4.1.2 Compared Methods. Task identity is not available in the context of class-incremental learning, thus we resort to selecting the most established and high-performing methods without prior knowledge of task identity during the testing phase. The compared methods contain regularization-based methods (EWC [21] and LwF [27]), rehearsal-based methods (ER [5], BiC [59], DER++ [3] and Co^{2}L [4]), as well as the prompt-based methods (L2P [58],S-Prompt [52], DualPrompt [57], CODA [48], ESN [53], and DAP [19]) .

To make a fair comparison, the above methods as well as the proposed PC[3] employ the pre-trained ViT-B/16 as the backbone. Specifically, the methods (L2P, DualPrompt, and DAP) involving selection strategy usually present the inferring results in the batch-wise prompt setup where a single set of prompts is chosen for the entire batch. In this setup, the frequently used prompts are selected to correct potential errors when dealing with challenging samples, which is not realistic. We hold the view that results should be given in *instance-wise* prompts setup where prompts are selected on a

[3] Implementation details and training schedule of the proposed PC are given in the supplementary materials.

Table 4: Performance comparison on CORe50 for domain incremental learning setting. B_s means buffer size. * suggests results copied from original paper, † and ‡ suggest results copied from [52] and [6], respectively.

Method	B_s	$A_a \uparrow$
DER++† [3]	2500	79.70
BiC† [59]	2500	79.28
ER† [5]	2500	80.10
Co^{2}L† [4]	2500	79.75
L2P* [58]	2500	81.07
EWC† [21]	0	74.82
LwF† [27]	0	75.45
L2P* [58]	0	78.33
S-iPrompt* [52]	0	83.13
DualPrompt‡ [57]	0	87.20
PC	0	**91.35**
Upper-bound	-	92.20

Table 5: Performance (task-agnostic domain-incremental Learning) comparison on DomainNet. B_s means buffer size. * suggests results copied from the original paper, † suggests results copied from ([52]), ‡ suggests results using the corresponding codebases and calculating by the normal equation.

Method	B_s	$A_a \uparrow$
EWC† [21]	0	47.62
LwF† [27]	0	49.19
L2P† [58]	0	40.15
S-iPrompt* [52]	0	50.62
DualPrompt‡ [57]	0	49.30
PC	0	**58.82**
Upper-bound	-	63.22

per-instance basis. Hence, in this paper, we give the performance for the prompt-based methods in *instance-wise* prompts setup.

Note that, the upper bound is typically achieved through supervised fine-tuning on the i.i.d. data from all tasks, which is commonly considered as the best performance a continual learning method can attain. Besides, we utilize two widely-used metrics[4]: Average Accuracy (A_a, the higher, the better") and Forgetting (F the lower, the better) [32]. Unless otherwise specified, we report the performance in *percentile* and omit the subscript.

4.2 Experimental Results

4.2.1 Results on Class-incremental Learning. Tab. 2 and Tab. 3 report the performance of the datasets of Split Cifar-100 and Split ImageNet-R. As for the common task setting (10 tasks), we observe that the proposed PC achieves 87.20% and 74.34% of A_a measure, respectively, which achieves remarkable improvement over the regularization-based methods (an average of roughly 64.6% and 102.6% on Split Cifar-100 and Split ImageNet-R, respectively) and the rehearsal-based methods (an average of roughly 31.0% and 37.6% on Split Cifar-100 and Split ImageNet-R, respectively) even though they use many buffers. The proposed PC also outperforms all the compared prompt-based methods (an average of roughly 2.4% and 7.0% on Split Cifar-100 and Split ImageNet-R, respectively) in terms of A_a measure. The kinds of methods prove to be insufficiently effective due to the significant variability across tasks, thereby limiting the precision of prompt *instance-wise* selection. Consequently, these limitations hinder the optimal performance they are supposed to achieve through their prompts-separate optimization.

Table 6: Performance comparison. T_t means the training time (hour). ‡ suggests results using the corresponding codebases.

Method	A_a	$T_t \uparrow$
L2P‡ [58]	83.00	0.87
DualPrompt‡ [57]	85.80	0.94
CODA-P‡ [48]	86.41	3.60
PC-light	86.68	1.06
PC	**87.20**	1.78
Upper-bound	90.85	-

Among these methods, only CODA-P[5] performs comparably to our method. However, PC outperforms CODA-P by a margin of 0.8% and 0.1% in terms of average accuracy due to our instance-level design, which effectively mitigates intra-task variance and eliminates noisy selection strategies. Although the proposed PC achieves slightly higher Forgetting values on the two datasets and secures the second-best position, it remains highly competitive.

In addition to the 10-task setting, we also provide results with other settings to compare the scalability of each method based on a smaller (5 tasks) and a larger number of tasks (20 tasks), respectively. The results show that the proposed PC achieves the best results in terms of A_a measure no matter what the task setting is. And PC only decreases slowly when handling more tasks, which shows that our method scales stably to different task number settings.

4.2.2 Results on Domain-incremental Learning. We further consider the DIL tasks on CORe50 with the corresponding results shown in Tab. 4. It is worth noting that none of the tested domains in CORe50 were included in the incremental training process. The Forgetting measure is not suitable for the unseen tested domain setting, thus only the results of A_a are given. The rehearsal-based methods demonstrate superior performance compared to EWC, LwF, and even L2P, suggesting that incorporating large buffers may yield marginal improvements over supervised training. However, they still fall short of the exceptional performance achieved by prompt-based methods such as S-iPrompt, DualPrompt, and PC, highlighting the clear advantage of rehearsal-free approaches. S-iPrompt is a variant of S-Prompt, it achieves significant advances compared to L2P due to easing the inter-task interference problem of L2P. DualPrompt further improves the performance compared to

[4] The details are given in supplementary materials.

[5] We implement the official codes of CODA-P and present the results of each task in supplementary materials.

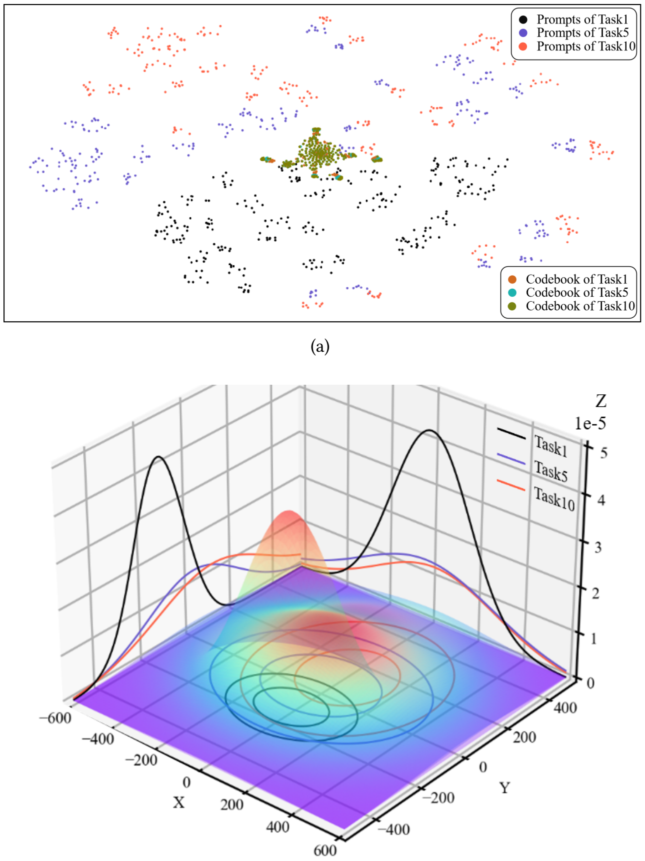
L2P and S-iPrompt with the benefits of multi-layer prompts. Moreover, the limited inter-domain variance in the CORe50 dataset minimizes the disparities among prompts from different tasks, thereby promoting the superior performance of S-iPrompt and DualPrompt even when wrong prompts are selected. Nevertheless, our PC attains an impressive 91.35% performance on the A_a measure, a result that closely approaches the upper-bound performance threshold. Moreover, the proposed PC consistently outperforms all other compared methods, showcasing the exceptional generalization capabilities of our PC on previously unseen domains. This can be attributed to the utilization of a global and comprehensive codebook, as well as the corresponding prompt generation and modulation modules.

4.2.3 Results on Task-Agnostic Domain-incremental Learning. For DIL tasks, the task-agnostic setting is generally regarded as more challenging [45]. The Forgetting measure is also not suitable for the task-agnostic setting and thus is omitted. Experiments of the setting are conducted on DomainNet with the results shown in Tab. 5. Note that, 345 classes are selected for each task in this experiment, which is highly challenging. The variance between domains in DomainNet is significantly larger compared to CORe50, which greatly benefits the precision of the selection strategy in L2P, S-iPrompt, and DualPrompt. However, the proposed PC still exhibits its superiority and achieves 58.82% in terms of A_a measure with an approximately 16.2% improvement compared to the second best exemplar-free DIL method S-iPrompt. The results also demonstrate the effectiveness of the PMM in significantly enhancing the capacity of prompts to handle domain shifts.

4.3 Analysis of The Proposed Method

4.3.1 Training time. To further demonstrate the effectiveness of the PC, we also report the results of prompt-based methods in terms of A_a and training time (T_t) based on the split CIFAR-100 dataset[6]. As Tab. 6 shows, the DualPrompt model surpasses L2P in terms of performance by employing prompts on multiple layers with only a little more training time. Additionally, CODA-P enhances the performance further by incorporating attention calculation based on DualPrompt. However, CODA-P requires almost 4 times the training time of DualPrompt according to the corresponding default settings. We propose to employ a dual-attention mechanism in the context of PC, aiming to generate and modulate prompts. Notably, even with a near training time, the light PC (PC-light) consistently outperforms DualPrompt by an absolute 0.88% and even outperforms CODA-P by 0.26% with only an appropriate quarter training time. Furthermore, increasing the number of attention layers in PC leads to additional advantages with an absolute 0.79% improvement with only an appropriate half training time compared to CODA-P.

4.3.2 Diversity of the prompts generated. As previously detailed in Section 3, the iterative augmentation of task numbers concurrent updates to both the codebook and prompts. We visualize the codebook and prompts corresponding to the initial, fifth, and tenth tasks by t-SNE [10] depicted in Fig. 3 (a). Additionally, Fig. 3 (b) portrays the 2D Gaussian distribution of the prompts' t-SNE projections for the corresponding tasks.



(a)

(b)

Figure 3: The visualization of codebook and prompts for the first, fifth, and tenth tasks. (a) t-SNE visualization of the updated codebook and prompts; (b) Gaussian distribution of prompts. The figure is best viewed in color.

The center of Fig. 3 (a) depicts the codes in the codebook after the learning of tasks 1, 5, and 10. The remaining segments of Fig. 3 display the prompts of the test data after the completion of the training process. Our observations are threefold: Firstly, the prompts associated with each task exhibit a notably greater diversity compared to the codebook, thus affirming the efficacy of the proposed prompt customization approach; Secondly, the iterative updates of the prompts across tasks are discernible, which enables the prompts are adaptive to the instances of different tasks. A more in-depth exploration by examining the 2D Gaussian distribution of the t-SNE projections of prompts is depicted in Fig. 3 (b). Remarkably, it is evident that the proposed PC adeptly tailors prompts across a varied range for each task, effectively managing the data distribution shifts; Thirdly, the update of the codebook remains relatively stable, primarily attributed to the momentum optimization with the imposed orthogonality constraint, forming a strong basis for forgetting alleviation.

[6] We train each method with the default setting referring to the official paper, detailed training setting and platform are given in the supplementary materials.

Table 7: Performance of ablation study.

Method	Split Cifar-100		Split ImageNet-R		Core50	DomainNet
	$A_a \uparrow$	$F \downarrow$	$A_a \uparrow$	$F \downarrow$	$A_a \uparrow$	$A_a \uparrow$
Baseline	69.93	8.44	53.55	9.20	71.28	36.54
+select prompts	85.51	5.91	67.75	9.03	86.89	49.05
+PGM	87.03	5.68	73.51	7.79	90.54	57.53
+PGM+spw	87.12	5.65	73.67	7.65	90.83	57.56
+PGM+PMM	**87.20**	**5.61**	**74.34**	**7.35**	**91.35**	**58.82**

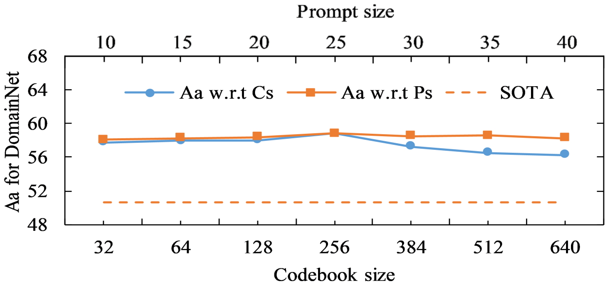


Figure 4: The performance with regard to different codebook sizes (Cs) and prompt sizes (Ps). The results demonstrate the stability of the proposed PC with small fluctuations concerning different parameters.

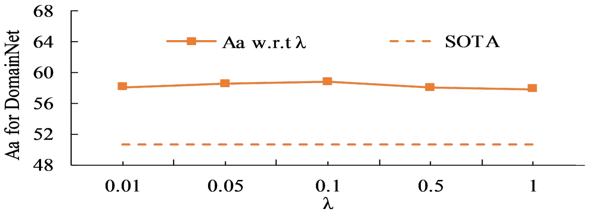


Figure 5: The performance respects to the weight λ.

4.4 Effectiveness Analysis

4.4.1 Codebook Related learning. The establishment of the codebook serves as a foundational element for prompt generation, playing a pivotal role in shaping the method's capacity to handle incremental tasks. Fig. 4 offers insights into the impact of codebook size on outcomes. Our observations reveal that augmenting the codebook size from 32 to 256 yields a marginal improvement in performance. The most optimal results, however, materialize when the codebook size is set at 256.

Remarkably, our approach consistently delivers competitive performance even with a more modest codebook size of 32. This underscores the method's adaptability and its capacity to achieve commendable results even under such conditions.

Moreover, to enhance the diversity of the codebook and mitigate interference among its components, an orthogonality constraint is introduced. The impact of its weight λ on performance is depicted in Fig. 5. The results illustrate that incrementally augmenting λ up to 0.1 yields consistent enhancements in performance. However, as λ is further increased, there is a concurrent reduction in the influence of the classification loss term, $\mathcal{L}_{ce}$, potentially leading to a detrimental effect on overall performance.

4.4.2 Prompt Related Learning. The proposed PGM takes input and codebook as input to compute their attention and further linearly combines the codebook for the corresponding input.

In the field of prompt-based methods, a common approach is to enlarge the size of the prompt to attain improved performance. However, as depicted in Fig. 4, our PC approach distinguishes itself by showcasing a notable degree of insensitivity to prompt size variations. This insensitivity highlights the robustness of the PC method, as it manages to maintain competitive performance levels even when the prompt size is reduced. This reduction in prompt size not only serves to enhance parameter efficiency but also ensures the method's capability to deliver commendable results.

In light of these findings, in this paper, we have determined an optimal prompt size of 25. This choice reflects a balance between maintaining effective performance and optimizing parameter efficiency within the context of the PC approach.

4.4.3 Ablation Study. The proposed method adopts prompt generation and modulation steps to obtain instance-specific prompts. The ablation studies in Tab. 7 justify the necessity of generation and modulation steps. Firstly, we set a baseline where the backbone is the same as that of the PC and we only update the final classification layer while keeping the backbone frozen. The performances (e.g., 69.93% on Split Cifar-100) are with much forgetting (e.g., 8.44% on Split Cifar-100) since the update process of prompts for the new tasks has a negative effect on the old tasks. Then we utilize the directly select strategy in L2P [58] to select 25 prompts ("+selected prompts") from the codebook, the corresponding results (e.g., 85.51% on Split Cifar-100) improve a lot compared to the baseline which demonstrates the effectiveness of prompt tuning approach. However, the results are still not satisfied, this may be due to the inter-task interference problem by the noisy select strategy for each instance. This variant is similar to DualPrompt while this variant has a larger prompt pool. This variant is also not better than DualPrompt with a smaller prompt size, which is also imagined according to Fig. 4 since increasing the prompt size has saturated or even negative returns. Next, the performance (e.g., 87.03% on Split Cifar-100) is better than the above two variants when we only employ the PGM ("+PGM"), which demonstrates the effectiveness of the soft generation strategy of PGM compared to the direct selection strategy in prior works. What is more, we further add the simple prompt weighting operation ("+PGM+spw") which still obtains a little performance improvement, as evidenced in Tab. 1. Finally, the difference between the variant "+PGM" and the proposed PC ("+PGM+PMM") also shows that the PMM is able to increase the capacity of prompts, thus being effective in further improving performance.

5 CONCLUSION

In this paper, we propose a prompt customization method for continual learning. Specifically, we propose PGM and PMM techniques to enable prompt customization by generating and modulating instance-specific prompts. The proposed method replaces the hard selection of prompts with flexible and efficient prompt generation and modulation. The strategy leads to more variation and less homogeneity for prompts and avoids the inherent noise of the selection strategy when handling increasing tasks. We perform extensive experiments on four benchmark datasets across three distinct settings, encompassing class, domain, and more challenging task-agnostic incremental learning scenarios. The experimental results demonstrate the general applicability and robustness of the proposed method for incremental classes and even unseen domains. In the future, we will adapt the proposed method to other multimedia or multimodal datasets and explore more sophisticated and generalized methods for prompt generation without using a prompt pool or codebook, which holds promise for enhancing prompt adaptation across different modalities.

A APPENDICES

In this part, we additionally present the detailed framework of the Plus_W, details setting of the benchmark datasets, thorough implementation details and train schedule of PC, other related analyses, and the detailed experiment results of CODA-P. The full implementation code has given in the supplementary materials and will be made publicly available.

B DATAILED PIPELINE OF PLUS_W

We investigate that a simple weighting operation for prompt boosts the corresponding performance to some extent. The weighting operation is given in Fig. 6. We first employ normalization to make the query and keys at the same scale. Then we concatenate them and employ a convolution operation to encourage the query to interact with each key. We further set a FC layer and a normalization operation to predict and normalize the corresponding weight W. The W is then utilized to weight the prompts by a dot product. The weighted prompts are denoted as P_w which is then selected ($\mathrm{P}_\mathrm{w}^\mathrm{id}$) according to the top K indexes (Id) of ranked W. Note that, we calculate the weight between the query and keys instead of prompts, which decreases the potential negative effect to the prompt optimization. Plus_W is a straightforward but effective prompt weighting method, leading to consistent performance improvement.

C DETAILED SETTING OF THE BENCHMARK DATASETS AND EVALUATION MATRICES

C.1 Datasets

In this work, for the class-incremental task, we consider the CIFAR-100 and ImageNet-Rendition (ImageNet-R) datasets as the base datasets and randomly split them into 10 incremental classification tasks with 10 classes or 20 classes respectively for each task. And for the domain-incremental task, we conduct experiments on Core50 and DomainNet. Details are given as follows.

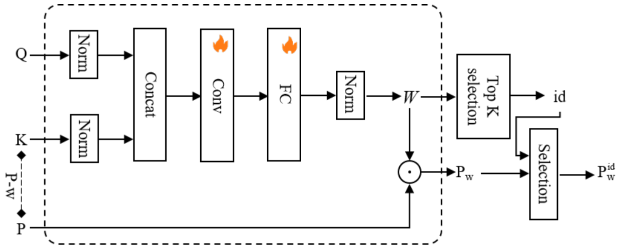


Figure 6: The pipeline of Plus_W. Q, K, and P mean query, keys, and prompts, respectively, given L2P. 'P-w' denotes keys and prompts are pair-wise tuples. 'Norm', 'Concat' are normalization and concatenation. 'Conv' and 'FC' mean convolution and fully connected layers.

Split CIFAR-100 The CIFAR-100 dataset has 100 classes, each containing 600 images, including 500 training pictures and 100 test pictures [22]. The 100 classes in CIFAR-100 are divided into 20 superclasses. Each image has a "fine" label (the class to which it belongs) and a "rough" label (the super-class to which it belongs). Dozens of related pieces of literature usually split the original CIFAR-100 according to the "fine" label into 10 separate tasks (Split CIFAR-100) which is regarded as a commonly used benchmark in the field of continual learning. It divides the original CIFAR-100 into 10 separate tasks, each containing 10 classes. Although it may seem like a relatively simple task for image classification under independent and identically distributed (i.i.d.) conditions, it is challenging and effectively exposes significant forgetting rates in class-incremental learning when advanced incremental learning methods are applied [57].

Split ImageNet-R ImageNet-R is an extension of the ImageNet dataset [42], which includes various renditions of multiple styles such as art, cartoon, DIYART, graffiti, embroidery, graphics, origami, painting, pattern, plastic, plush sculpture drawing, tattoo, toy, video game, etc. The Imagenet-R dataset has undergone a deduction process that resulted in 30k images by removing 200 categories from the original ImageNet. Like the way of getting Split CIFAR-100, splitting ImageNet-R [57] is also challenging due to the variance intra-class styles. The Split ImageNet-R benchmark randomly partitions the 200 classes into 10 tasks, each consisting of 20 classes. The dataset is then split into a training set and a test set under the portion of 8:2.

CORe50 The CORe50 dataset [31], cited as a standard for continual object recognition, comprises 50 categories from 11 distinct domains. In the continual learning setting, incremental training is performed on data from eight domains (120K images), while the remaining (unseen) domains are used as test sets. Following the methodology of [52] and [13] citations, we present average forward classification accuracy for CORe50 in this paper.

DomainNet DomainNet [37] is a dataset consisting of 345 categories and approximately 600,000 images that can be utilized for domain adaptation and Domain Incremental Learning (DIL). The

Table 8: Performance A_a of CODA-P tested on the previous tasks after completing the training phase of each task on Split Cifar-100 dataset. 'Tt' means t-th task.

T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
99.3	93.3	89.8	88.5	86.8	83.9	84.2	84.2	82.8	81.9
	96.6	94.4	85.2	82.5	80.8	79.7	78.2	78.0	78.0
		96.1	95.8	94.0	92.7	91.7	91.0	90.4	90.5
			95.4	93.7	91.4	91.1	90.1	90.0	87.1
				93.3	93.1	92.7	92.9	92.5	90.8
					84.6	83.3	82.1	81.9	80.1
						88.6	87.7	86.2	85.0
							90.8	90.4	89.1
								94.3	91.9
									89.8

Table 9: Performance A_a of CODA-P tested on the previous tasks after completing the training phase of each task on Split ImageNet-R dataset. 'Tt' means t-th task.

T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
94.5	90.6	87.3	86.4	82.3	79.9	77.7	75.7	75.1	74.2
	81.8	79.9	78.5	74.6	73.0	72.5	69.6	68.0	67.3
		86.6	85.5	86.1	84.4	82.5	81.9	80.7	79.7
			77.4	74.6	73.5	72.2	69.7	69.2	67.8
				82.9	82.3	78.7	77.7	77.7	76.5
					80.0	78.6	76.5	75.4	74.8
						81.1	80.4	79.2	77.7
							80.8	79.7	78.0
								73.9	71.9
									74.7

images from DomainNet are divided into six domains, with the DIL setup on this platform being identical to that of CaSSLe [13]. In accordance with [52] and [13], we present task-agnostic test average results for forward classification accuracy.

C.2 Evaluation matrices

For settings that involve task boundaries and where each task is associated with a test set, we utilize two widely-used metrics: Average Accuracy (higher values indicate better performance) and Forgetting (lower values indicate better performance) [32]. After the training phase is completed for task t, we assess the current learner's performance on all previous tasks using their respective test sets. Let $a_{t,j} \in [0, 1]$ denote the accuracy achieved on task j-th test set after training on task t. The Average Accuracy (A_a) measures the average value of $a_{t,j}$ as:

$$A_a = \frac{1}{t}\sum_{j=1}^{t} a_{t,j} \tag{10}$$

And $f_{t,j}$ denotes the forgetting transfer calculated on task j-th test set after training on task t which is define as:

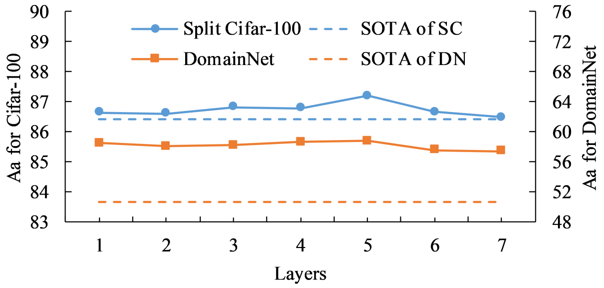


Figure 7: The performance with regard to the number of attention layers of PGM.

$$f_j^t = \max_{i \in \{1, \cdots t-1\}} a_{i,j} - a_{t,j} \tag{11}$$

Hence, the Forgetting (F_t) measures the average value of f_j^t as:

$$F_t = \frac{1}{t-1}\sum_{j=1}^{t-1} f_j^t \tag{12}$$

D IMPLEMENTATION DETAILS AND TRAINING SCHEDULE

We apply task-agnostic prompts in the first 2 blocks like DualPrompt [57] inspired by Complementary Learning Systems (CLS) [23, 34] and the proposed modulated instance-specific prompts in the third to fifth blocks like DualPrompt. As for the prompting function to combine the modulated instance-specific prompts with input embedding, we employ the Prefix Tuning (Pre-T) referring to the DualPrompt [57].

While considering applying prompts in different blocks, the PC establishes an independent PRM for each block and assigns a shared codebook and a shared PMM to all the PRMs of all blocks. Accordingly, the instance-specific prompts are divided into two parts $\hat{P}^K_{x_i^t}$ and $\hat{P}^V_{x_i^t}$ for key and values of M_{SA} layer in each VIT block, respectively. $\hat{P}^K_{x_i^t}$ and $\hat{P}^V_{x_i^t}$ are all with the size of $\mathbb{R}^{n\times L}$. Specifically, the application of a prompting function can be interpreted as modifying the inputs of the M_{SA} layers [50]. Let $h \in \mathbb{R}^{D\times L}$ denote the input to the M_{SA} layer, and let h^Q, h^K, and h^V represent its input query, key, and values, respectively. Hence, the corresponding prompts are combined by Pre-T in each corresponding M_{SA} layer as follows:

$$f_P(p^\ell, h^\ell) = M_{SA}(h^{Q,\ell}, [\hat{P}^{K,\ell}_{x_i^t}, h^{K,\ell}], [\hat{P}^{V,\ell}_{x_i^t}, h^{V,\ell}]) \tag{13}$$

where ℓ means ℓ-th block.

The model is trained for 5 epochs for each task by iteratively minimizing the loss $\mathcal{L}$. Adam optimization is employed for training with a batch size of 64 on NVIDIA A100 GPU and a momentum of 0.9. The initial learning rate is set as 0.007.

E HYPERPARAMETER ANALYSES

The attention computation employs several ViT blocks to explore the relationships between each other. Fig. 7 reports the performance

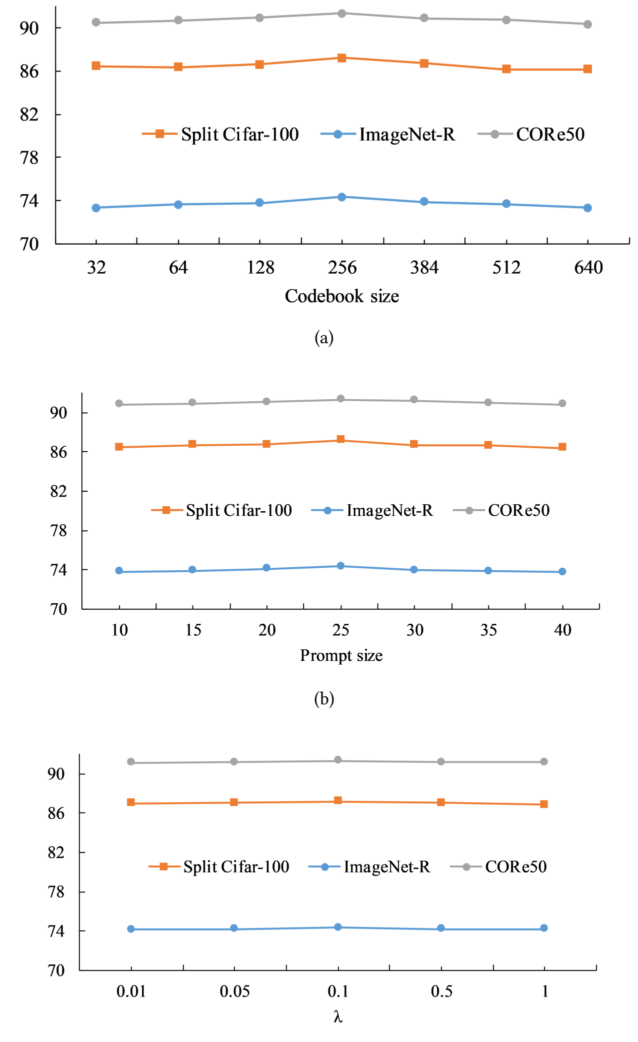


Figure 8: The performance with regard to the proposed codebook size, prompt size, and the weight λ.

with regard to the number of attention layers of PGM, the performance shows small fluctuations when the layer number is less than 5 and drops with the number further increasing.

Besides, we also give the performance results with regard to the proposed codebook size, prompt size, and the weight λ in Fig. 8. The results also demonstrate the stability of the proposed PC with small fluctuations concerning different parameters.

F DETAILED EXPERIMENT RESULTS OF CODA-P

We train CIFAR-100 for 20 epochs, and ImageNet-R for 50 epochs according to the original paper. Hence, it requires almost 4 times the training time of DualPrompt (5 epochs) on CIFAR-100. Tab .8 and Tab. 9 give the performance of CODA-P tested on the previous tasks after completing the training of each task on Split CIFAR-100 dataset and Split ImageNet-R dataset, respectively. The forgetting results of CODA-P in the original paper are different from the experimental results, we report the corresponding performances of A_a and F for CODA-P according to the above tables referring to the eq. 10 and 12.

REFERENCES

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. *Proceedings of the European conference on computer vision (ECCV)* (2018), 139–154.

[2] Hyojin Bahng, Ali Jahanian, Swami Sankaranarayanan, and Phillip Isola. 2022. Visual prompting: Modifying pixel space to adapt pre-trained models. *arXiv preprint arXiv:2203.17274* (2022).

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

[4] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. 2021. Co2l: Contrastive continual learning. *Proceedings of the IEEE/CVF International conference on computer vision* (2021), 9516–9525.

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

[6] Haoran Chen, Zuxuan Wu, Xintong Han, Menglin Jia, and Yu-Gang Jiang. 2023. PromptFusion: Decoupling Stability and Plasticity for Continual Learning. *arXiv preprint arXiv:2303.07223* (2023). arXiv:2303.07223 [cs.CV]

[7] Kexin Chen, Yuyang Du, Tao You, Mobarakol Islam, Ziyu Guo, Yueming Jin, Guangyong Chen, and Pheng-Ann Heng. 2024. LLM-Assisted Multi-Teacher Continual Learning for Visual Question Answering in Robotic Surgery. *arXiv preprint arXiv:2402.16664* (2024).

[8] Yu Cheng, Kin-Yeung Wong, Kevin Hung, Weitong Li, Zhizhong Li, and Jun Zhang. 2018. Deep nearest class mean model for incremental odor classification. *IEEE Transactions on Instrumentation and Measurement* 68, 4 (2018), 952–962.

[9] Wei Cong, Yang Cong, Gan Sun, Yuyang Liu, and Jiahua Dong. 2023. Self-paced Weight Consolidation for Continual Learning. *IEEE Transactions on Circuits and Systems for Video Technology* (2023).

[10] LV der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[11] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. 2022. Dytox: Transformers for continual learning with dynamic token expansion. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), 9285–9295.

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

[13] Enrico Fini, Victor G Turrisi Da Costa, Xavier Alameda-Pineda, Elisa Ricci, Karteek Alahari, and Julien Mairal. 2022. Self-supervised models are continual learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9621–9630.

[14] Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. 2023. A Unified Continual Learning Framework with General Parameter-Efficient Tuning. *arXiv preprint arXiv:2303.10070* (2023).

[15] Alvin Heng and Harold Soh. 2023. Selective Amnesia: A Continual Learning Approach to Forgetting in Deep Generative Models. *arXiv preprint arXiv:2305.10120* (2023).

[16] Qinghua Hu, Yucong Gao, and Bing Cao. 2022. Curiosity-driven class-incremental learning via adaptive sample selection. *IEEE Transactions on Circuits and Systems for Video Technology* 32, 12 (2022), 8660–8673.

[17] Tony Huang, Jack Chu, and Fangyun Wei. 2022. Unsupervised prompt learning for vision-language models. *arXiv preprint arXiv:2204.03649* (2022).

[18] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual prompt tuning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*. Springer, 709–727.

[19] Dahuin Jung, Dongyoon Han, Jihwan Bang, and Hwanjun Song. 2023. Generating Instance-level Prompts for Rehearsal-free Continual Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 11847–11857.

[20] Zixuan Ke, Bing Liu, and Xingchang Huang. 2020. Continual learning of a mixed sequence of similar and dissimilar tasks. *Advances in neural information processing systems* 33 (2020), 18493–18504.

[21] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural

networks. *Proceedings of the national academy of sciences* 114, 13 (2017), 3521–3526.

[22] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases* (2009).

[23] Dharshan Kumaran, Demis Hassabis, and James L McClelland. 2016. What learning systems do intelligent agents need? Complementary learning systems theory updated. *Trends in cognitive sciences* 20, 7 (2016), 512–534.

[24] Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691* (2021).

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

[26] Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190* (2021).

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

[28] Huiwei Lin, Shanshan Feng, Xutao Li, Wentao Li, and Yunming Ye. 2022. Anchor assisted experience replay for online class-incremental learning. *IEEE Transactions on Circuits and Systems for Video Technology* (2022).

[29] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *Comput. Surveys* 55, 9 (2023), 1–35.

[30] Jochem Loedeman, Maarten C Stol, Tengda Han, and Yuki M Asano. 2022. Prompt Generation Networks for Efficient Adaptation of Frozen Vision Transformers. *arXiv preprint arXiv:2210.06466* (2022).

[31] Vincenzo Lomonaco and Davide Maltoni. 2017. Core50: a new dataset and benchmark for continuous object recognition. In *Conference on Robot Learning*. PMLR, 17–26.

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

[33] Arun Mallya and Svetlana Lazebnik. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition* (2018), 7765–7773.

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

[35] Shentong Mo, Weiguo Pian, and Yapeng Tian. 2023. Class-Incremental Grouping Network for Continual Audio-Visual Learning. *arXiv preprint arXiv:2309.05281* (2023).

[36] Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. 2020. Latent replay for real-time continual learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 10203–10209.

[37] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. 2019. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*. 1406–1415.

[38] Quang Pham, Chenghao Liu, and Steven Hoi. 2021. Dualnet: Continual learning, fast and slow. *Advances in Neural Information Processing Systems* 34 (2021), 16131–16144.

[39] Ameya Prabhu, Hasan Abed Al Kader Hammoud, Puneet K Dokania, Philip HS Torr, Ser-Nam Lim, Bernard Ghanem, and Adel Bibi. 2023. Computationally Budgeted Continual Learning: What Does Matter?. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 3698–3707.

[40] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. 2020. Gdumb: A simple approach that questions our progress in continual learning. *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16* (2020), 524–540.

[41] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. 2019. Random path selection for continual learning. *Advances in Neural Information Processing Systems* 32 (2019).

[42] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, and Michael Bernstein. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision* 115, 3 (Dec. 2015), 211–252.

[43] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *arXiv preprint arXiv:1606.04671* (2016).

[44] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. 2018. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*. PMLR, 4548–4557.

[45] Murray Shanahan, Christos Kaplanis, and Jovana Mitrović. 2021. Encoders and ensembles for task-free continual learning. *arXiv preprint arXiv:2105.13327* (2021).

[46] Chao Shang, Hongliang Li, Fanman Meng, Qingbo Wu, Heqian Qiu, and Lanxiao Wang. 2023. Incrementer: Transformer for Class-Incremental Semantic Segmentation With Knowledge Distillation Focusing on Old Class. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7214–7224.

[47] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 1310–1321.

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

[49] Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems* 30 (2017).

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[51] Andrés Villa, Juan León Alcázar, Motasem Alfarra, Kumail Alhamoud, Julio Hurtado, Fabian Caba Heilbron, Alvaro Soto, and Bernard Ghanem. 2023. Pivot: Prompting for video continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24214–24223.

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

[53] Yabin Wang, Zhiheng Ma, Zhiwu Huang, Yaowei Wang, Zhou Su, and Xiaopeng Hong. 2023. Isolation and impartial aggregation: A paradigm of incremental learning without interference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 10209–10217.

[54] Yinjun Wang, Liling Zeng, Liming Wang, Yimin Shao, Yongxiang Zhang, and Xiaoxi Ding. 2021. An Efficient Incremental Learning of Bearing Fault Imbalanced Data Set via Filter StyleGAN. *IEEE Transactions on Instrumentation and Measurement* 70 (2021), 1–10.

[55] Zifeng Wang, Tong Jian, Kaushik Chowdhury, Yanzhi Wang, Jennifer Dy, and Stratis Ioannidis. 2020. Learn-prune-share for lifelong learning. *2020 IEEE International Conference on Data Mining (ICDM)* (2020), 641–650.

[56] Zifeng Wang, Zheng Zhan, Yifan Gong, Yucai Shao, Stratis Ioannidis, Yanzhi Wang, and Jennifer Dy. 2023. DualHSIC: HSIC-Bottleneck and Alignment for Continual Learning. *arXiv preprint arXiv:2305.00380* (2023).

[57] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. 2022. Dualprompt: Complementary prompting for rehearsal-free continual learning. *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI* (2022), 631–648.

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

[59] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. 2019. Large scale incremental learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2019), 374–382.

[60] Shipeng Yan, Jiangwei Xie, and Xuming He. 2021. Der: Dynamically expandable representation for class incremental learning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), 3014–3023.

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

[62] Da Yu, Mingyi Zhang, Mantian Li, Fusheng Zha, Junge Zhang, Lining Sun, and Kaiqi Huang. 2023. Contrastive Correlation Preserving Replay for Online Continual Learning. *IEEE Transactions on Circuits and Systems for Video Technology* (2023).

[63] Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. *International conference on machine learning* (2017), 3987–3995.

[64] Yaohua Zha, Jinpeng Wang, Tao Dai, Bin Chen, Zhi Wang, and Shu-Tao Xia. 2023. Instance-aware Dynamic Prompt Tuning for Pre-trained Point Cloud Models Supplementary File. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14161–14170.

[65] Gengwei Zhang, Liyuan Wang, Guoliang Kang, Ling Chen, and Yunchao Wei. 2023. SLCA: Slow Learner with Classifier Alignment for Continual Learning on a Pre-trained Model. *arXiv preprint arXiv:2303.05118* (2023).

[66] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Conditional prompt learning for vision-language models. *IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), 16816–16825.

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

[68] Hongguang Zhu, Yunchao Wei, Xiaodan Liang, Chunjie Zhang, and Yao Zhao. 2023. CTP: Towards Vision-Language Continual Pretraining via Compatible Momentum Contrast and Topology Preservation. *arXiv preprint arXiv:2308.07146*

(2023).