D-VLAM: Differential Vision and Language Mixing for Rehearsal Free Continual Learning

M. Anwar Ma'sum[1*], Mohsen Guizani[2], and Waseem Ullah[2]

[1] Adelaide University, Adelaide SA 5000, Australia
[2] Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE
muhammadanwar.masum@adelaide.edu.au
{mohsen.guzani,waseem.ullah}@mbzuai.ac.ae

Abstract. Language-guided prompt-based continual learning emerges as a promising approach to tackle catastrophic forgetting in a dynamic environment under rehearsal-free constraint. However, existing state-of-the-art (SOTA) requires additional resources, such as LLM-generated descriptors and an LLM decoder that may not be available in a real application. Meanwhile, the more straightforward method experiences suboptimality due to the high similarity between language text embeddings. To address this problem, we propose a resource-efficient novel language-guided approach and algorithm that incorporates two novel ideas: (1) mixed vision and language modality for prompt generation, and (2) Differential prefix tuning for the model training process. Our experimental analysis shows that our method outperforms existing language-guided prompt-based methods, i.e., up to 30%, 26%, and 10% for final average accuracy, cumulative average accuracy, and final forgetting measure, respectively. The historical analysis confirms our method's stability-plasticity balance in every task. Our extended analysis shows that our method consistently achieves better performance in various prompt lengths and ViT layers. For further study and reproducibility, we also provide rigorous analysis, details, and source code of our method in the supplementary document. The implementation of the proposed method is available at `https://github.com/anwarmaxsum/D-VLAM`.

Keywords: continual learning · catastrophic forgetting · vision and language mixing · differential · rehearsal-free

1 Introduction

Continual learning (CL) has become a new focus in the field of artificial intelligence (AI) to address catastrophic forgetting in a dynamic environment [49,60]. CL has proven its impact and promising benefits in various studies, such as NLP, computer vision, graph [2,27,46] and applications such as medicine, energy, and smart city [4,38,55]. However, previous state-of-the-art (SOTA) CL methods [3,36,53] require previous task exemplars for a rehearsal process to minimize forgetting of previously learned knowledge [1,3,3,5,7,15,36,53]. This approach is not practical, as the old exemplars could be unavailable forever due to the privacy policy. In addition, the rehearsal process incurs additional memory and

* Corresponding

computational expense. Along with the advancement of foundation models [56], a prompt-based approach emerges as the new effective, yet efficient solution for rehearsal-free CL. Leveraging a frozen pre-trained backbone, the approach demands only small-sized learnable parameters (prompt) to adapt to a new task. More specifically, the language-guided prompt methods offers new innovation for a better stability-plasticity guided by text embedding discrimination.

Despite its promising performance on rehearsal-free CL, language-guided prompt methods face critical issues related to resources: First, the latest SOTAs, e.g., LAEPGen [30] and ConvPrompt [37], require LLM-generated class descriptors as language text modality rather than class names. The need for LLM access is arguably considered unfair and an overkill resource for language-guided CL methods. In another study, the utilization of the LLM decoder by GMM [6] is considered similarly. On the other hand, a more straightforward approach, as in LGCL [18] that utilizes encoded text embedding as loss anchors, falls short of its full potential due to the high similarities between class embeddings. Second, the need for LLM access to generate class descriptors incurs additional computational expenses, especially in the training phase. Third, the latest SOTAs require larger trainable parameters compared to the existing methods. Specifically, ConvPrompt that utilizes growing prompt generators accumulates more trainable parameters (prompt generators) than task-wise prompt methods such as DualPrompt [51] and CPrompt [12] or pool-based prompting such as L2P [52]. Similarly, LEAPGen, which utilizes a larger generator, requires higher trainable parameters even though it manages its structure as a task-wise generator.

Reflecting on the mentioned drawbacks, we propose a novel resource-efficient language-guided prompt-based method for rehearsal-free CL. Our method incorporates two novel ideas: (1) mixed vision and language modality for prompt generation and (2) differential prefix tuning for the model training process. Our vision and language (VL) mixing is a novel approach that is unique to existing language-guided mechanisms [6, 18, 30, 37] that process both modalities in a more detachable way. Our VL mixing is conducted through our proposed differential prefix tuning (DPT) for better cohesion between VL modalities. Unlike [57] which performs differential attention to tune the Visual Transformer (ViT) weight, our DPT tunes prompt generators that accommodate dual modalities. In addition, the trainable lambda coefficients are structured in a task-wise manner, which adds to the uniqueness of our method. To satisfy efficient resource and parameters, our method incorporates lightweight prompt generators and class names as a text modality. Thus, our method does not utilize LLM-generated class descriptors or an LLM decoder. Our contributions are as follows:

- We highlight resource fairness on language-guided continual learning and propose a novel prompt-based method named differential vision and language mixing (D-VLAM) for the rehearsal-free CL problem that carries out the three main principles mentioned above.
- We demonstrate 3 strategies of VL mixing mechanisms, i.e., complementary-VL, complementary-LV and aggregate-VL that outperform existing SOTAs.
- Our rigorous experiment and analysis prove the superiority of our method over existing SOTAs in final, cumulative, and historical perspectives in terms

of accuracy and forgetting measure. Our extended analysis proves the robustness of the proposed method in various settings, i.e., the number of layers and prompt lengths. We analyze the tradeoff between performance and running time as well as the number of parameters of our method in comparison with the existing state-of-the-art methods.
– We provide complete numerical results, technical details of our method, and the code of our proposed method for future study and reproducibility. Please see our supplementary.

2 Related Work

a). Parameter Efficient Fine Tuning (PEFT)-based CL: The PEFT-based CL approach emerges along with the advancement of foundation models that carry generalization capability. The prompt-based approach is one of the PEFT types that is well known for its scalability and performance on rehearsal-free CL. The prompt approach prepends a small-sized parameter (prompt) into the ViT multi-head self-attention (MSA) [10, 22, 24]. The prompt could be latent trainable parameters or a vector generated by trainable generator networks. Prompts are organized in several types of structures, i.e., pool-based manner, such as in L2P [52], task-wise structure as in DualPrompt [51], PGP [34], S-Prompt [50], HiDePrompt [48] and CPrompt [12], evolving component as in CODA-P [41], evolving generator as in ConvPrompt [37] and EvoPrompt [20] or task-wise generator as in LEAPGen [30]. The other approach utilizes Low-Rank Adaptation (LoRA) that extends frozen ViT weight W with pair of learnable matrices (A,B) as in LAE [11], CLoRA [40], InfLoRA [26], and CL-LoRA [13]. The LoRA-based method is popular for its intuitive expression and explainability. The newest PEFT approach i.e., adapter approach extends ViT backbone with a set of adapters such as in EASE [59] and MOS [42]. Rather than extending the inner level, such as ViT weight, adapters work on the feature level by projecting the ViT feature into a more discriminative form. The other strategy are adding a trainable projection layer after the ViT layers to improve the feature separability as performed by RanPAC [31], RanDumb [33], and FOAL [61].
b). Language guided CL: Language-guided becomes a new innovative solution to catastrophic forgetting by improving class discriminant via text embedding discrimination. One of the approaches utilizes encoded text embeddings as loss anchors, as in [18]. The other approach utilizes inter-task embedding similarity to compute the number of required new prompt generators [37], while the recent study utilizes the text embeddings as input of prompt generators [30]. The generative language-guided approach utilizes an LLM decoder to generate a text embedding combined with a vision embedding [6]. The use of an LLM encoder or LLM-generated descriptors, as in the aforementioned studies, is considered an overdose of resources that may be unaffordable in the real world. Another approach utilizes a pre-trained vision-language model (VLS), such as CLIP, as the backbone rather than just a text encoder [16, 17, 45]. This approach is arguably considered an overdose of resources for continual learning in a simple task such as image classification. Recent studies show that the language-guided approach is also applied for continual learning under data scarcity [8,23,28,29,32,35,43,58]

and advanced vision task [9, 25, 39, 44, 54]. This development demonstrates the extended prospect of the language-guided approach.

3 Preliminary

3.1 Problem Formulation

Class incremental learning (CIL) problem is defined as the problem of learning a sequence of fully supervised tasks $\mathcal{T}^1$,$\mathcal{T}^2$,...$\mathcal{T}^{T-1}$,$\mathcal{T}^T$ where after finishing learning a task $\mathcal{T}^T$, a model must recognize all learned tasks, i.e., $\{\mathcal{T}^t\}_{t=1}^T$. Symbol T represents the number of consecutive tasks. Each task carries pairs of training samples $\mathcal{T}^t = \{(x_i^t, y_i^t)\}_{i=1}^{N^t}$ where $x_i \in \mathcal{X}^t$ and $y_i \in \mathcal{Y}^t$ denotes the input image and its label respectively, $N^t = |\mathcal{T}^t|$ denotes cardinality of $\mathcal{T}^t$. $\mathcal{C}^t$ represents unique class labels in $\mathcal{Y}^t$ i.e. $\mathcal{C}^t = unique(\mathcal{Y}^t)$, and $|\mathcal{C}^t|$ denotes the number of classes in $\mathcal{T}^t$. Each task t is disjoint from another task t' i.e $\forall t, t' \neq t, (\mathcal{T}^t \cap \mathcal{T}^{t'} = \emptyset)$. In this study, we focus on CIL as the most challenging problem in CL. In a CIL setting, the task ID is available only in the training phase, different from task-incremental learning (TIL), where the task ID is visible both in the training and in the inference phase.

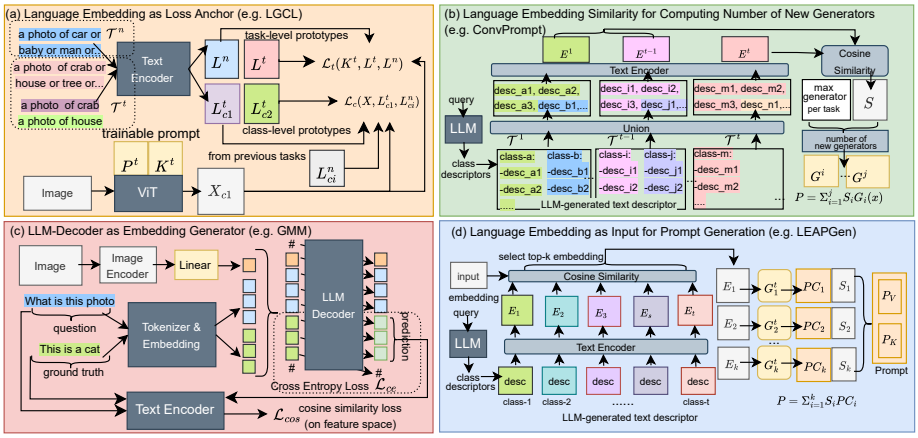

Fig. 1: Visualization of existing language-guided prompt-based CL methods (a) Language embedding as loss anchors (b) Language embedding similarity for computing number of new prompt generators (c) LLM decoder as embedding generator (d)Language embedding as input for prompt generation.

3.2 Language-Guided Approaches and Drawbacks

Figure 1 shows the existing language-guided approach for the CL problem. Figure 1(a) shows the most straightforward approach, which utilizes language embedding as loss anchors [18]. The method utilizes a class-level embedding L_{ci}^t encoded from string "a photo of "class name" and task-level embedding L^t that is similarly encoded from the unification of class names within a task t. As shown

in Figure 1(b), the second approach measures the similarity between the text embedding of the currently learned task and the text embedding of previous tasks to compute the number of required new prompt generators; i.e., $G_i(.)$,...,$G_j(.)$ [37]. Note that the method has generators from previous tasks; i.e., $G_1(.)$,...,$G_{i-1}(.)$, and the final prompt is computed by all generators; i.e., $P = \Sigma_{i=1}^{j} S_i G_i(x)$. Unlike the first approach, this approach utilizes LLM-generated class descriptors instead of class name as its language modality; e.g., the descriptor for class "apple" is ["round shape", "red or green color", "sweet"]. Figure 1(c) shows the generative approach that deploys the LLM decoder to generate the prediction embedding [6]. In the training phase, this approach concatenates image embedding and a pair of question-and-response text embeddings, then feeds it into the LLM decoder. In the inference phase, only the image and question embeddings are given to the decoder. The last approach, as visualized by figure 1(d) utilizes language embedding as input for prompt generators [30]. The method selects top-k descriptor embeddings as input to k prompt generators $(G_1^t(.), G_2^t(.), ..., G_k^t(.))$. The final prompt is computed as the weighted sum of its output with respective similarity (S_i); i.e., $P = \Sigma_{i=1}^{k} S_i PC_i(x) = \Sigma_{i=1}^{k} S_i G_i(x)$.

All the mentioned language-guided approaches suffer from particular drawbacks. The first approach suffers from high similarity between class embeddings; i.e., $\forall L_{ci}^t \neq L_{cj}^n, 0.73 \leq cos(L_{ci}^t, L_{cj}^n) \leq 0.96$ [30]. With the high similarities between the anchors, the class-wise loss $\mathcal{L}(x, L_{ci}^t, L_{cj}^n)$ confronts a challenging path to discriminate the classes. The second and fourth approaches send queries to LLM to generate class descriptors. The LLM access is arguably an overkill resource for a simple task such as CL in image classification. In addition, generating descriptors incurs additional training time, even though the LLM access is conducted in an online mode. In a real-world application, the LLM access is not always affordable. From the perspective of executed parameters, the growing generator approach increases both its training and inference time for every upcoming task. The fourth approach may experience lower increase on its training and testing time for each new task, but a greater increase in its storage utilization. The third approach explicitly utilizes the LLM decoder as a compulsory component in its pipeline. Aside from the overuse of the LLM decoder, this approach relies too much on the language modality. Figure 1(c) shows that the join embedding has far bigger portion of text embedding than vision embedding. In summary, there is no approach that effectively blends vision and language modality with reasonable resources for CL problem. These intriguing drawbacks motivate us to develop a resource-efficient CL method with a more cohesive vision and language mixing.

4 Proposed Method

Overview: We propose Differential Vision and LAnguage Mixing (D-VLAM) that incorporates two novel ideas: (1) mixed vision and language modality for prompt generation, and (2) Differential prefix tuning for the model training. From the perspective of language-guided prompt-based CL, our novel method is different from existing methods that generate prompts based-on vision modality only [18, 37] or language modality only [30]. In addition, our mixing process is

conducted in the prefix tuning mechanism, which is the center of the training and inference phases. Our proposed differential prefix tuning tunes prompt generators and learnable parameters that are different from the existing Differential Transformer [57] that tunes the ViT weights. Our method is resource-efficient since it does not require any resources or access to an LLM. The following subsections explain the details of our method.

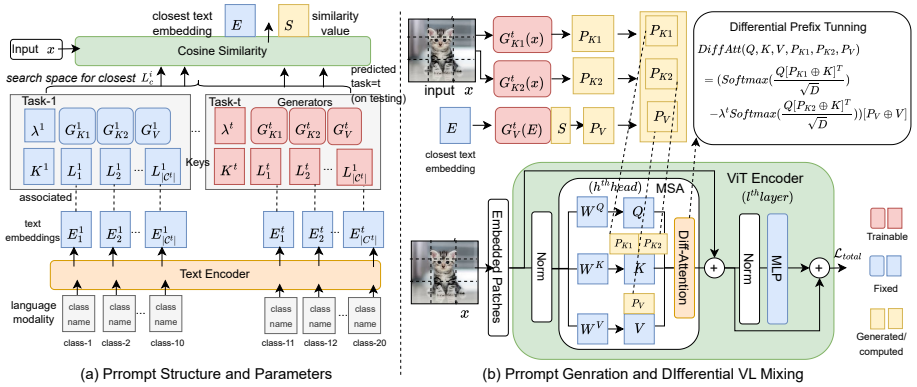

Fig. 2: Visualization of the proposed Differential Vision and LAnguage Mixing (D-VLAM) (a) Structure and learnable parameters. (b) Prompt generation and VL mixing through differential prefix tuning.

4.1 Structure and Learnable Parameters

Figure 2 shows the structure and learning process of our methods. As shown in sub-figure (a), our method is structured into task-wise prompt generators, where each task carries a set of prompt generators $G^t(,)$, a task-level learnable key K^t, class-level learnable keys L_c^t, and a learnable coefficient λ^t for differential prefix tuning. These task-associated parameters are trainable (red color) only within the training phase of task t, and frozen afterward (blue color). During the inference phase, the generators and parameters are selected based on the predicted task ID.

a). Prompt Generator: We design a task-associated prompt generator $G^t(.)$ as tuple of three elements $(G^t_{K1}(.), G^t_{K2}(.), G^t_V(.))$ that generate 3-elements prompt $P = (P_{K1}, P_{K2}, P_V)$. The prompt and generator are designed following our differential prefix tuning mechanism that takes 3-elements, different from standard prefix tuning that takes 2-elements prompt $P = (P_K, P_V)$ [24]. P_{K1} and P_{K2} are generated from the vision embedding while P_V is generated from the text embedding. The prompt $P = (P_{K1}, P_{K2}, P_V)$ is prepended to $K = W^K X$ and $V = W^v X$ of ViT layer's MSA, as shown in sub-figure (b). Following previous generator methods [30, 37], the generator G^t is distributed for all MSA heads of ViT layers. Therefore, each task-associated generator G^t is expressed as $G^t(.) = (G^t_{K1}(.), G^t_{K1}(.), G^t_V(.)) = \{(G^t_{K1,l,h}(.), G^t_{K2,l,h}(.), G^t_{V,l,h}(.))\}, \forall l \in [1..L], \forall h \in [1..H]$, where L and H are the depth of ViT layers and number of MSA heads per layer, respectively. To achieve our efficient-resource principle, $G^t_{K1,l,h}(.)$, $G^t_{K2,l,h}(.)$ or $G^t_{V,l,h}(.)$ is a Conv1D network with kernel size of 3, thus

only saving 4 parameters (3 kernel weights + 1 bias). Therefore, the total parameters of each task generator are $H \times L \times 3 \times 4$, which is generally less than 2.5K if we prefix-tune all the ViT/B layers with 16 MSA heads.

b). Learnable Keys and Coefficient: Along with the prompt generator, each task t carries learnable keys and a coefficient. The task-level key $K^t \in \mathbb{R}^D$ and the class-level key $L_c^t \in \mathbb{R}^D, \forall c \in \mathcal{C}^t$, where D is the embedding size, are important elements for predicting the task ID in the inference phase. As mentioned above, the predicted task ID is utilized to select the prompt generator $G^t(.)$. Each class-level L_c^t is associated with a text embedding E^c that is encoded from the string "class name" of c. In the inference phase, L_c^t is utilized to match E_c^t, i.e., based on the highest similarity between input x and L_c^t for $\forall t \in [1..T]$, where T is the predicted task ID. The coefficient $\lambda^t \in \mathbb{R}^1$ is a steering coefficient for differential prefix tuning. The coefficient λ is organized in a task-wise manner, so that each task flexibly adjusts its optimum value independent to other tasks.

4.2 Prompt Generation

We propose a novel prompt generation that takes both vision and language modalities as input. Given an input image x, closest class-level key L_c^t, text embedding E_c^t associated to L_c^t, and selected prompt generator $G^t(.)$, then prompt $P = (P_{K1}, P_{K2}, P_V)$ is generated by executing equation 1.

$$\begin{aligned} &(P_{K1}, P_{K2}, P_V) = \{(P_{K1,l,h}, P_{K2,l,h}, P_{V,l,h})\}, \\ &\forall l \in [1..L], \forall h \in [1..H], \text{where} \\ &P_{K1,l,h} = G^t_{K1,l,h}(x), P_{K2,l,h} = G^t_{K2,l,h}(x) \\ &P_{V,l,h} = S.G^t_{V,l,h}(E_c^t) = cos(x, L_c^t).G^t_{V,l,h}(E_c^t) \end{aligned} \tag{1}$$

where L and H are ViT layers and MSA heads respectively, $S = cos(x, L_c^t)$ is the cosine similarity between input and closest class-level key. Following the best practice in previous studies [37, 51], the input x can be an average patched embedding, class feature, or class token. Equation 1 shows that prompt P already contains mixed VL modalities. The following sub-section presents more details on how we mix 2 modalities in a prefix tuning mechanism.

4.3 Differential Prefix Tuning (DPT) for VL Mixing

a). Idea and Formulation: The idea of prefix tuning enables flexible adaptation across the network while preserving the output length, keeping it identical to the input length [21, 24]. Originally, it tunes learnable P_K and P_V to K and V, respectively, so that the attention function is performed to Q,$[P_K \oplus K]$ and $[P_V \oplus V]$ instead of Q,K and V, where $Q = W^Q X$,$K = W^K X$ and $V = W^V X$ are produced by multiplication of frozen MSA weights (W^Q, W^K, W^V) and MSA input X. The adaptation is achieved by tuning P_K and P_V that produce the minimal loss during the tuning process. On the other hand, the softmax value in the attention function is noisy, so it produces sub-optimality for the class recognition [57]. One of the solutions to the noise is refining the softmax value by subtracting some of its portion (λ). In previous study, the minuend

(Softmax($(Q_1K_1^T)/\sqrt{D}$)) and subtrahend (Softmax($(Q_2K_2^T)/\sqrt{D}$)) come from different vectors, i.e., $W^QX = [Q_1; Q_2]$, $W^KX = [K_1; K_2]$. In language-guided CL that deals with dual modalities, vision embedding is considered to have more noise than language embedding. Specifically, different samples from the same class label produce different vision embeddings but have the same language embedding. Thus, bringing the insights from the original prefix tuning and the differential transformer [57], we design a differential prefix tuning (DPT) that prepends P_{K1}, P_{K2}, and P_V to the MSA weights. $P_{K2} = G^t_{K2}(x)$ is considered an alternate (noisy) form of $P_{K1} = G^t_{K1}(x)$ that is generated from the vision modality, while $P_V = G^t_V(E^t_c)$ is generated from the language modality associated with x. Our differential prefix tuning is formulated in equation 2.

$$
\begin{aligned}
DiffAttn(Q, K, V, P_{K1}, P_{K2}, P_V) &= (S_1 - \lambda^t.S_2)[V \oplus P_V] \\
S_1 = \text{Softmax}(\frac{Q[P_{K1} \oplus K]^T}{\sqrt{D}}), S_2 &= \text{Softmax}(\frac{Q[P_{K2} \oplus K]^T}{\sqrt{D}})
\end{aligned} \tag{2}
$$

where D and $\oplus$ denote the dimension of the embedding and concatenation operation, respectively. Symbol λ^t denotes the task-wise trainable coefficient, as our proposed DPT tunes the task-wise prompt generator $G^t(.) = (G^t_{K1}(.), G^t_{K2}(.), G^t_V(.))$. Our DPT is focused on flexibility on mixing V-L portions during the prefix tuning process, that is notably different from differential attention in [57] that is focused on tunes Transformer weights (W^Q, W^K, W^V). Thus, our DPT utilize a single learnable coefficient λ^t rather than several learnable coefficients as in [57].

b). 3-Types of Mixing: Figure 3 shows 3-types of proposed mixing strategies. All of the types require the same number of trainable generators, i.e., $G^t_{K1}(.), G^t_{K2}(.)$, and $G^t_V(.)$. The Complementary-VL generates P_{K1}, and P_{K2} from the input image and generates P_V from the closest text embedding, while The Complementary-LV does the opposite, i.e., generates P_{K1}, and P_{K2} from the closest text embedding and generates P_V from the input image x. The Aggregate type generate P_{K1}, P_{K2}, P_{K1} from both input image and closest text embedding. Each of the prompt element is formed by concatenating $G^t(x)$ and $G^t(E)$ on the respective prompt elements $K1$, $K2$, and V.

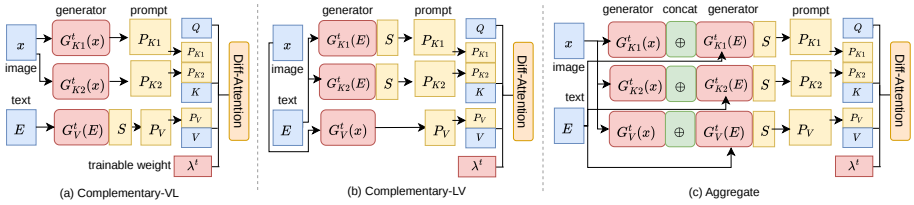

Fig. 3: Variations of the proposed Differential Vision and LAnguage Mixing (D-VLAM) (a) Complementary-VL (b) Complementary-LV and (c) Aggregate-VL.

4.4 Learning, Losses, and Final Objectives

This subsection explains the loss formulation and learning objective of our method. Given a pre-trained ViT $f_\theta(.)$ with a frozen weight θ, the trainable

parameters $\phi = \{G^t, \lambda^t, K^t, L_c^t\}_{t=1}^T$, and MLP classifier $g_\psi(.)$ Then, the leoss and learning objective of our method is formulated as in the following explanation.

a). Tuning generators $G^t(,)$ and coefficient λ^t: The generator $G^t(,)$ generate 3-elements prompt and prepends it into ViT MSA weight, then the model perform DPT with λ^t, and the output is classified by the MLP classifier. Therefore, along with the MLP classifier, generator G^t and coefficient λ^t are tuned by using the cross entropy loss. The loss is derived into two components named intra-task loss ($\mathcal{L}_{intra}$) and inter-task loss $\mathcal{L}_{inter}$. Intra-task improves class recognition within the currently learned task, while the inter-task loss improves model discrimination for currently learned classes to preciously learned classes. The two components of cross entropy losses are formulated by equations 3 and 4.

$$\mathcal{L}_{intra} = -\sum_{c \in C^t} log \frac{exp(g_\psi(f_{\theta;P}(x))[c])}{\sum_{c' \in \mathcal{C}^t} exp(g_\psi(f_{\theta;P}(x))[c'])} \tag{3}$$

$$\mathcal{L}_{inter} = -\sum_{t=1}^{t=T} \sum_{c \in \mathcal{C}^t} log \frac{exp(g_\phi(f_{\theta;P}(x))[c])}{\sum_{t=1}^{t=T} \sum_{c' \in \mathcal{C}^t} exp(g_\phi(f_{\theta;P}(x))[c'])} \tag{4}$$

Note that $f_{\theta;P}$ is the output of the ViT layer that performs DPT by the generated prompt $P = (P_{K1}, P_{K2}, P_V)$ on top of the frozen ViT weight θ.

b). Tuning Task-Level and class-level learnable key: Task-level key K^t and class-level L_c^t have a crucial role in predicting task ID in the inference phase, where the predicted task ID that later on selects generator $G^t(.)$, DPT coefficient λ^t, and text embedding E_c^t. The task-level key K^t is tuned closer to the input sample within task t, i.e., $x_i \in \mathcal{T}^t$, while the class-level key L_c^t is tuned closer to the input sample of class c only. Both of the keys are tuned by a similarity loss as expressed in equation 5 and 6, respectively.

$$\mathcal{L}_K = -(x.K^t)/(max(||x||_2.||K^t||_2, \epsilon)) \tag{5}$$

$$\mathcal{L}_L = -(x.L_c^c)/(max(||x||_2.||L_c^t||_2, \epsilon)) \tag{6}$$

c). Joint Loss and Final Objective: Finally, accommodating the training losses for all components, i.e., generators $G^t(.)$, DPT coefficient λ^t, task-level keys K^t, class-level keys L_c^t, for $\forall t \in [1..T]$ and MLP head $g_\psi(.)$, we define a new joint loss function as shown in equation 7. To enhance the flexibility of our model, we add the coefficient losses of $\lambda_1, \lambda_2, \lambda_3,$ and, λ_4.

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{intra} + \lambda_2 \mathcal{L}_{inter} + \lambda_3 \mathcal{L}_K + \lambda_4 \mathcal{L}_L \tag{7}$$

d). Ensuring P_{K1} and P_{K2} Dissimilarity: As an alternate representation of P_{K1}, P_{K2} should be ensured to be different. Thus, after an update iteration, our method ensures the dissimilarity of ϕ_{K1} and ϕ_{K2}^t as the parameters of $G_{K1}^t(.)$ and $G_{K2}^t(.)$, respectively, by executing equation 8. The symbol ϵ is a random noise drawn from a Gaussian distribution.

$$\phi_{K2}^t = \phi_{K1}^t + \epsilon, \text{ if}(||\phi_{K2}^t - \phi_{K1}^t|| = 0) \tag{8}$$

The procedures for D-VLAM training and inference are presented in algorithms 1 and 2 respectively.

Algorithm 1 D-VLAM Training

Input: A sequence of tasks $\mathcal{T}^1$, $\mathcal{T}^2$,...,$\mathcal{T}^T$, a frozen L-layered (with H MSA heads/layer) pre-trained ViT $f_{\theta(.)}$, trainable MLP head $g_\psi(.)$ training epochs E, and batch size B.
for $t = 1 : T$ **do**
  Initiate trainable parameters for task t i.e. $(G^t(.), K^t, \{L^t_c\}_{c=1}^{|C^t|})$
  $\mathcal{B} \leftarrow$ Split $\mathcal{T}^t$ into B sized batches
  for $e = 1 : E$ **do**
    for $b = 1 : \mathcal{B}$ **do**
      Find top-1 L^j_c where $j \in [1..t]$, $c \in [1..|C^j|]$
      Generate VL prompt $P = \{(P_{K1,l,h}, P_{K2,l,h}, P_{V,l,h})\}_{l\in[1..L],h\in[1..H]}$ Eq. 1
      Compute logits by forwarding the input i.e. $g_\psi(f_{\theta,P}(x))$
      Compute $\mathcal{L}_{intra}$ as in equation 3
      Compute $\mathcal{L}_{inter}$ as in equation 4
      Compute $\mathcal{L}_K$ as in equation 5
      Compute $\mathcal{L}_L$ as in equation 6
      Compute $\mathcal{L}_{total}$ as in equation 7
      Update parameters $(G^t(.), K^t, \{L^t_c\}_{c=1}^{|C^t|})$ based on $\mathcal{L}_{total}$
    end for
  end for
end for
Output: Optimum parameters $\{(G^t(.), K^t, \{L^t_c\}_{c=1}^{|C^t|})\}_{c=1}^{|C^t|}\}_{t=1}^{T}$

5 Experiment Results and Analysis

5.1 Experiment Setup

a). Datasets: We evaluate our method in three CIL benchmarks, i.e., CIFAR100 (100 classes) [14], ImageNet-R (200 classes) [19], and CUB (200 classes) dataset [47]. We follow 5, 10, and 20 task splits as in the SOTAs [30, 37].
b). Baselines and Metrics: We compare D-VLAM to both language-guided and non-language-guided prompt-based SOTAs, i.e., L2P [52], DualPrompt [51], CODA-P [41], LGCL [18], EvoPrompt [20], CPrompt [12], ConvPrompt [37], MOS [42], and LEAPGen [30]. We also compare D-VLAM to the low-Rank Adaptation (LoRA)-based methods [11, 13, 26, 40], generative approach [6], and CLIP-based methods [16,17,45]. Following [37], we measure the final average accuracy (FAA), cumulative average accuracy (CAA), and final forgetting measure (FFM) (supplementary).
c). Implementation Details: Our method is implemented in Pytorch, utilizing the pre-trained ViT-B/16 backbone, Adam optimizer, set 128 batch-size, and a cosine learning scheduler. The initial learning rate is set to 0.01, 0.05, and 0.005 for CIFAR100, ImageNet-R, and CUB, respectively, with 20 maximum epochs. To ensure fairness, all the methods are run in a rehearsal-free setting, with 3 different seed numbers, under the same resources and environment, i.e., a single NVIDIA A100 GPU with 40 GB RAM, and class name as language modality. The hyper-parameter settings follow their official settings, see supplementary.

Algorithm 2 D-VLAM Inference

Input: An input x, a frozen L-layered (with H MSA heads/layer) pre-trained ViT $f_{\theta(.)}$, and optimized parameters $\{(G^t(.), K^t, \{L_c^t\}_{c=1}^{|C^t|})\}_{c=1}^{|C^t|}\}_{t=1}^{T}$, and optimized MLP head $g_\psi(.)$
//Predict task-id t using soft task-id prediction
$t = \arg\max_{t\in[1..T], c\in C^t}\{(x.K^t)(x.L_c^t)/((max(||x||_2.||K^t||_2, \epsilon))(max(||x||_2.||K^t||_2, \epsilon))$
Find top-1 L_c^j where $j \in [1..t]$, $c \in [1..|C^j|]$
Generate prompt $P = \{(P_{K1,l,h}, P_{K2,l,h}, P_{V,l,h})\}_{l\in[1..L],h\in[1..H]}$ as in equation 1
Compute logits by forwarding the input i.e. $logits = g_\psi(f_{\theta,P}(x))$
Compute Pedicted label $\hat{y} = argmax(logits)$
Output: Predicted label $\hat{y}$

Table 1: Summarized numerical result on CIFAR100 and ImageNet-R datasets with 5-task, 10-task, and 20-task settings, ± 0.00 indicates the method is run 1 time due to crash or poor performance

Method	ImageNet-R 5-Task			CIFAR100 5-Task		
	FAA	**CAA**	**FFM**	**FAA**	**CAA**	**FFM**
L2P	64.62 ± 0.32	68.01 ± 0.42	3.94 ± 0.16	84.77 ± 0.48	88.67 ± 0.3	6.18 ± 0.57
DualPrompt	69.71 ± 0.11	72.78 ± 0.14	3.32 ± 0.16	86.41 ± 0.21	89.95 ± 0.1	5.37 ± 0.21
CODA-P	74.89 ± 0.36	79.71 ± 1.27	8.89 ± 0.65	88.22 ± 1.06	92.25 ± 1.28	7.05 ± 2.18
LGCL	69.93 ± 0.21	72.91 ± 0.19	3.04 ± 0.36	86.90 ± 0.40	90.45 ± 0.18	5.01 ± 0.35
EvoPrompt	77.27 ± 0.40	81.67 ± 0.18	1.79 ± 0.31	89.07 ± 0.38	92.32 ± 0.26	5.25 ± 0.65
CPrompt	78.65 ± 0.00	82.44 ± 0.00	6.00 ± 0.00	89.22 ± 0.05	93.09 ± 0.06	5.02 ± 0.17
ConvPrompt	78.94 ± 0.15	82.95 ± 0.24	4.05 ± 0.75	90.17 ± 0.16	93.54 ± 0.05	3.68 ± 0.31
MOS	29.33 ± 0.00	52.16 ± 0.00	72.09 ± 0.00	76.99 ± 0.00	86.08 ± 0.00	25.66 ± 0.00
LEAPGen	82.24 ± 1.22	84.10 ± 1.07	0.68 ± 1.07	96.98 ± 0.05	97.09 ± 0.12	0.05 ± 0.02
D-VLAM	**83.81 ± 0.18**	**85.41 ± 0.43**	**0.18 ± 0.09**	**97.09 ± 0.06**	**97.24 ± 0.12**	**0.03 ± 0.01**
	ImageNet-R 10-Task			**CIFAR100 10-Task**		
	FAA	**CAA**	**FFM**	**FAA**	**CAA**	**FFM**
L2P	62.50 ± 0.51	67.05 ± 0.47	5.01 ± 0.40	83.84 ± 0.32	88.67 ± 0.16	6.55 ± 0.34
DualPrompt	68.59 ± 0.24	72.18 ± 0.20	4.61 ± 0.07	85.36 ± 0.2	89.77 ± 0.20	5.41 ± 0.33
CODA-P	73.77 ± 0.50	79.38 ± 1.48	7.94 ± 0.08	86.44 ± 0.16	91.27 ± 0.56	6.38 ± 1.46
LGCL	68.65 ± 0.25	72.57 ± 0.19	4.75 ± 0.33	85.68 ± 0.43	90.16 ± 0.29	5.46 ± 0.22
EvoPrompt	76.00 ± 0.26	80.97 ± 0.30	4.22 ± 0.42	88.17 ± 0.51	92.18 ± 0.49	5.39 ± 0.45
CPrompt	76.32 ± 0.53	81.50 ± 0.30	6.10 ± 0.75	86.92 ± 1.04	91.73 ± 0.66	5.43 ± 0.74
ConvPrompt	77.48 ± 0.31	81.41 ± 0.34	4.33 ± 0.45	88.51 ± 0.26	92.71 ± 0.04	3.79 ± 0.06
MOS	27.07 ± 0.00	46.21 ± 0.00	63.31 ± 0.00	67.70 ± 0.00	80.26 ± 0.00	33.66 ± 0.00
LEAPGen	83.18 ± 1.11	85.06 ± 0.43	2.21 ± 1.42	98.00 ± 0.57	97.38 ± 1.52	0.16 ± 0.13
D-VLAM	**86.54 ± 0.13**	**87.97 ± 0.15**	**0.35 ± 0.14**	**98.48 ± 0.06**	**98.63 ± 0.08**	**0.11 ± 0.03**
	ImageNet-R 20-Task			**CIFAR100 20-Task**		
	FAA	**CAA**	**FFM**	**FAA**	**CAA**	**FFM**
L2P	57.40 ± 0.31	63.33 ± 0.21	10.76 ± 0.45	81.89 ± 0.38	87.16 ± 0.33	8.81 ± 0.1
DualPrompt	65.19 ± 0.17	70.31 ± 0.29	7.30 ± 0.18	82.32 ± 0.22	87.47 ± 0.24	6.88 ± 0.35
CODA-P	70.55 ± 0.71	77.08 ± 1.02	8.23 ± 0.86	81.29 ± 0.16	87.72 ± 0.44	6.82 ± 1.6
LGCL	64.96 ± 0.67	70.18 ± 0.37	7.35 ± 0.65	83.18 ± 0.4	88.63 ± 0.18	7.22 ± 0.47
EvoPrompt	74.93 ± 0.64	79.92 ± 0.13	6.72 ± 0.90	84.63 ± 0.21	89.47 ± 0.21	9.19 ± 0.41
CPrompt	74.23 ± 0.17	79.82 ± 0.51	5.98 ± 0.24	83.60 ± 0.00	90.10 ± 0.00	6.47 ± 0.00
ConvPrompt	74.37 ± 0.37	79.32 ± 0.12	5.34 ± 0.49	86.51 ± 0.59	91.29 ± 0.38	5.81 ± 0.35
MOS	41.23 ± 0.00	45.53 ± 0.00	45.52 ± 0.00	19.41 ± 0.00	36.61 ± 0.00	82.93 ± 0.00
LEAPGen	85.48 ± 1.02	85.15 ± 0.48	2.43 ± 0.28	87.81 ± 1.21	97.52 ± 0.84	0.29 ± 0.01
D-VLAM	**87.78 ± 0.30**	**89.19 ± 0.27**	**0.95 ± 0.48**	**98.93 ± 0.06**	**99.17 ± 0.06**	**0.29 ± 0.09**

5.2 Result and Analysis

a). Overall Performance: Table 1 shows the summarized numerical results of consolidated algorithms in the split ImageNet-R and CIFAR100 datasets.

Table 2: Comparison of D-VLAM to LoRA, Generative, and CLIP-based methods on CIFAR100 and ImageNet-R.

Method	CIFAR100 10-Task		ImageNet-R 10-Task	
	FAA	**CAA**	**FAA**	**CAA**
C-LoRA	82.97	87.06	71.89	75.33
LAE	88.81	91.59	71.70	76.71
InfLoRA	89.84	91.70	75.65	80.82
CL-LoRA	-	91.85	-	-
GMM	87.59	-	80.72	-
Cont-CLIP	66.68	75.15	71.35	78.65
CLAP4CLIP	63.45	74..19	75.80	81.22
MG-CLIP	79.00	85.63	80.85	87.13
D-VLAM	**98.48**	**98.63**	**86.54**	**87.97**

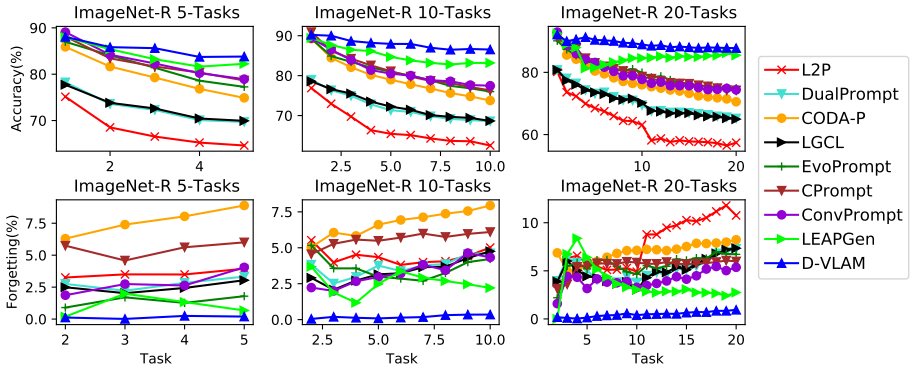

Fig. 4: Plot of historical performance on ImageNet-R dataset with 5-task, 10-task, and 20-task settings.

The detailed numerical results are presented in the supplementary. In the split ImageNet-R dataset, D-VLAM achieves the highest performance in all settings. Compared to LEAPGen as the closest competitor, D-VLAM achieves a higher performance with 1.5-3.5% FAA and 1.3-4% CAA margin. The gap is even higher in comparison to the other SOTAs, i.e., 5-30% for FAA and 3-26% for CAA. The table shows that a higher gap is found in the higher number of tasks. For the forgetting measure, D-VLAM consistently achieves the lowest FFM in all settings of ImageNet-R. In addition, the magnitude of its FFM and CFM is lower than 0.5%, while the LEAPGen suffers from 1-2.4%, FFM and the other SOTAs generally suffer from more than 3% FFM. In CIFAR100 dataset, D-VLAM achieves a higher performance than LEAPGen with 1-10% margin, while in comparison to the rest of the SOTAs, D-VLAM achieves higher margins, i.e., 7-15% for FAA and 4-12% for CAA. Our method also attains the lowest forgetting than all the competitors. Similarly, the higher gap of accuracy and forgetting occurs in a higher number of tasks. Note that MOS can not perform well in a rehearsal-free setting, as shown by Table 1, thus, we exclude it from the margin above. Table 2 shows that the proposed method outperforms LoRA-based, generative-

based and CLIP-based methods in both datasets with 10 task settings. Please see comparison or the methods in CUB dataset in our supplementary.

b). Historical Performance: Figure 4 shows the historical performance, i.e., average accuracy and average forgetting of all learned classes, after finishing learning on each task t on ImageNet-R dataset.The D-VLAM's gentle slope shows that our proposed method loses the least performance drop compared to the existing SOTAs. In the CIFAR100 dataset, our method even manages to increase its performance in the later tasks. The existing methods suffer from a significant performance drop i.e up 15%. For the average forgetting measure, our method once again demonstrates a stable forgetting rate on every task. The figure shows that the D-VLAM suffers from the lowest average forgetting in all tasks and all settings. D-VLAM experiences less than 1% forgetting measure from the first until the last task. On the contrary, the competitor methods suffer from an increasing average forgetting during incremental learning, as shown by the rising curves. The methods suffer from a high average forgetting, i.e., for up to 8%. This analysis confirms how our method handles catastrophic forgetting better than the existing methods. Please see the supplementary for historical analysis on CIFAR100 and CUB datasets.

c). Ablation Study: We evaluate the impact of each component of our method in an ablation study that is presented in Table 3. The table shows that fine-tuning (FT) alone achieves poor performance, showing it can not handle the catastrophic forgetting problem. Adding inter-task loss $\mathcal{L}_{inter}$ into intra-task loss $\mathcal{L}_{intra}$ improves the accuracy by approximately 2% but also increases the forgetting. The prompt generation with vision modality $G(x)$ boosts the model accuracy tremendously i.e. 12%, and reduces the forgetting by 2% magnitude. The learnable task-level key K^t and its loss $\mathcal{L}_K$ that restructure the model into task-wise prompt generator improve the FAA by 5% and CAA by 1% margin. The FFM also decreases by 1% with the new configuration. Adding class-level key L_c^t, its loss $\mathcal{L}_L$, and including L_c^t into the task prediction mechanism improves the accuracy significantly and reduces forgetting by 2%. Adding a new prompt component from language modality $G^T(E)$ improves the model performance by at least 2% in both accuracy and forgetting. At this configuration, our method already achieves better performance than existing SOTAs. Finally, mixing the vision and language modality through differential prefix tuning (DPT) with trainable coefficient λ^t improves the accuracy by 1% margin, and reduces a slight forgetting rate.

Table 3: Ablation study on ImageNet-R dataset

Component	Loss	FAA	CAA	FFM
FT	$\mathcal{L}_{intra}$	58.5	63.3	5.9
FT	$\mathcal{L}_{intra}$,$\mathcal{L}_{inter}$	59.5	65.3	7.1
FT+$G(x)$	$\mathcal{L}_{intra}$,$\mathcal{L}_{inter}$	71.9	78.5	5.7
FT+$G^t(x)$+K^t	$\mathcal{L}_{intra}$,$\mathcal{L}_{inter}$,$\mathcal{L}_K$	76.8	79.5	4.8
FT+$G^t(x)$+K^t+L_c^t	$\mathcal{L}_{intra}$,$\mathcal{L}_{inter}$,$\mathcal{L}_K$,$\mathcal{L}_L$	82.2	85.9	2.2
FT+$G^t(x)$+$G^t(E)$+K^t+L_c^t	$\mathcal{L}_{intra}$,$\mathcal{L}_{inter}$,$\mathcal{L}_K$,$\mathcal{L}_L$	85.5	87.1	0.7
FT+$G^t(x)$+$G^t(E)$+K^t+L_c^t +λ^t+DPT	$\mathcal{L}_{intra}$,$\mathcal{L}_{inter}$,$\mathcal{L}_K$,$\mathcal{L}_L$	86.5	87.9	0.2

d). Variation on VL Mixing: Table 4 shows the performance of various mixing mechanisms on ImageNet-R. Complement types generate P_K and P_V from different modalities, while the aggregate type concatenates ($\oplus$) generated prompts from both modalities, see "Prompt structure" column. Table 4 shows all 3 types of mixing configuration gain a comparable performance with $\leq 0.5\%$ difference both in accuracy and forgetting. In addition, the table confirms DPT improves all configurations' performance.

Table 4: The Performance of various types of VL mixing on ImageNet-R dataset, "Comp-" and $\oplus$ denote complementary and concatenation, respectively.

Conf.	DPT	Prompt structure	FAA	CAA	FFM
Comp-VL	x	$P_K = G_K^t(x), P_V = G_V^t(E)$	85.5	87.1	0.7
Comp-LV	x	$P_K = G_K^t(E), P_V = G_V^t(x)$	85.4	86.7	0.3
Aggregate	x	$P_K = G_K^t(x) \oplus G_K^t(E)$, $P_V = G_V^t(x) \oplus G_V^t(E)$	85.5	87.0	0.3
Comp-VL	v	$P_K1 = G_{K1}^t(x), P_K2 = G_{K2}^t(x)$, $P_V = G_V^t(E)$	86.5	87.9	0.2
Comp-LV	v	$P_{K1} = G_{K1}^t(E), P_{K2} = G_{K2}^t(E)$, $P_V = G_V^t(x)$	86.1	87.7	0.4
Aggregate	v	$P_{K1} = G_{K1}^t(x) \oplus G_{K1}^t(E)$, $P_{K2} = G_{K2}^t(x) \oplus G_{K2}^t(E)$, $P_V = G_V^t(x) \oplus G_V^t(E)$	86.4	87.8	0.7

e). Sensitivity Analysis: Figure 5 shows the sensitivity of D-VLAM compared to existing SOTAs, at different prompt lengths and layer depths. The figure shows that D-VLAM successfully maintains its performance in different prompt lengths and layer depths that perform prefix tuning. D-VLAM achieves better performance than the existing SOTAs even in the smallest number of prompt length and ViT layers. In addition, it maintains the performance gap to LEAPGen as the closest competitor with a compelling margin, i.e., more than 2% in both settings.

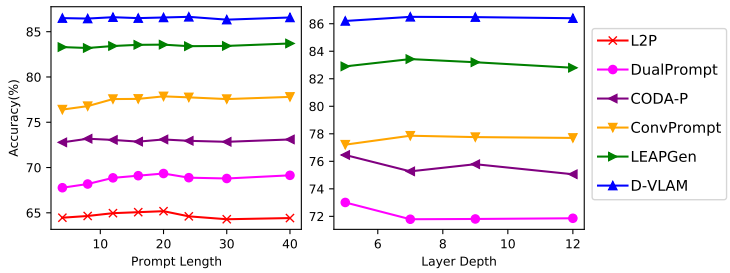

Fig. 5: Sensitivity on prompt lengths and layer depths.

f).Trade-off Between Running Time and Performance Table 5 shows the trade-off between running time and performance on ImageNet-R dataset for

the most performing methods, i.e., D-VLAM, LEAPGen, and ConvPrompt. Table 5 shows that D-VLAM required the least number of learnable parameters compared to LEAPGen and ConvPrompt. In terms of running time, D-VLAM requires slightly longer training time than LEAPGen, but shorter than ConvPrompt. This is the logical impact caused by the differential prefix tuning executed by D-VLAM. In differential prefix tuning, the method computes the softmax value in the attention function 2 times, while in standard prefix tuning, which is performed by LEAPGen, the method computes the softmax value once. However, D-VLAM requires a comparable inference time to LEAPGen, and is shorter than ConvPrompt's inference time. In spite of the small running time drawbacks, D-VLAM achieves the highest performance compared to the competitors.

Table 5: Tradeoff between performance and running time on ImageNet-R dataset with 10 tasks setting.

Method	#Params	Running Time (h)			Acc. IM-R 10T	
		Inference	Training	Total	FAA	CAA
ConvPrompt	1.28 M	0.033	1.007	1.040	77.48	81.43
LEAPGen	6.35 M	0.025	0.655	0.680	83.18	85.06
D-VLAM	0.16 M	0.025	0.725	0.750	86.54	87.97

6 Concluding Remark

We highlight resource fairness of language-guided methods for rehearsal-free continual learning and propose a resource-efficient novel language-guided approach and algorithm that incorporates two novel ideas: (1) mixed vision and language modality for prompt generation, and (2) Differential prefix tuning for the model training process. Our experimental analysis shows that our method outperforms existing language-guided prompt-based methods, i.e., up to 30%, 26%, and 10% final average accuracy, cumulative average accuracy, and forgetting measure, respectively. The historical analysis confirms our method's stability-plasticity balance in every task. Our extended analysis shows that our method consistently achieves better performance in various prompt lengths and ViT layers. We demonstrate several options for VL mixing that perform effectively. Our method requires a comparable training and inference time compared to the existing state-of-the-art methods.

Acknowledgement

Prof. Guizani and Dr. Ullah would like to acknowledge the support of the Mohamed Bin Zayed University of Artificial Intelligence (MBZUAI). This work was conducted during Ma'sum's visiting research program at Department of Machine Learning, MBZUAI on 2025.

References

1. Belouadah, E., Popescu, A.: Il2m: Class incremental learning with dual memory. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 583–592 (2019)
2. Biesialska, M., Biesialska, K., Costa-Jussa, M.R.: Continual lifelong learning in natural language processing: A survey. arXiv preprint arXiv:2012.09823 (2020)
3. Boschini, M., Bonicelli, L., Buzzega, P., Porrello, A., Calderara, S.: Class-incremental continual learning into the extended der-verse. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(5), 5497–5512 (2022)
4. Bruno, P., Quarta, A., Calimeri, F.: Continual learning in medicine: A systematic literature review. Neural Processing Letters **57**(1), 2 (2025)
5. Buzzega, P., Boschini, M., Porrello, A., Abati, D., Calderara, S.: Dark experience for general continual learning: a strong, simple baseline. Advances in neural information processing systems **33**, 15920–15930 (2020)
6. Cao, X., Lu, H., Huang, L., Liu, X., Cheng, M.M.: Generative multi-modal models are good class incremental learners. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 28706–28717 (2024)
7. Castro, F.M., Marín-Jiménez, M.J., Guil, N., Schmid, C., Alahari, K.: End-to-end incremental learning. In: Proceedings of the European conference on computer vision (ECCV). pp. 233–248 (2018)
8. Chen, Y., Ding, T., Wang, L., Huo, J., Gao, Y., Li, W.: Enhancing few-shot class-incremental learning via training-free bi-level modality calibration. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 9881–9890 (2025)
9. Cheraghian, A., Hayder, Z., Ramasinghe, S., Rahman, S., Jafaryahya, J., Petersson, L., Harandi, M.: Canonical shape projection is all you need for 3d few-shot class incremental learning. In: European Conference on Computer Vision. pp. 36–53. Springer (2024)
10. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
11. Gao, Q., Zhao, C., Sun, Y., Xi, T., Zhang, G., Ghanem, B., Zhang, J.: A unified continual learning framework with general parameter-efficient tuning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11483–11493 (2023)
12. Gao, Z., Cen, J., Chang, X.: Consistent prompting for rehearsal-free continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 28463–28473 (2024)
13. He, J., Duan, Z., Zhu, F.: Cl-lora: Continual low-rank adaptation for rehearsal-free class-incremental learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 30534–30544 (2025)
14. Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., Gilmer, J.: The many faces of robustness: A critical analysis of out-of-distribution generalization. International Conference on Computer Vision (2021)
15. Hou, S., Pan, X., Loy, C.C., Wang, Z., Lin, D.: Learning a unified classifier incrementally via rebalancing. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (IEEE/CVF Conference on Computer Vision and Pattern Recognition) pp. 831–839 (2019)

16. Huang, L., Cao, X., Lu, H., Meng, Y., Yang, F., Liu, X.: Mind the gap: Preserving and compensating for the modality gap in clip-based continual learning. arXiv preprint arXiv:2507.09118 (2025)
17. Jha, S., Gong, D., Yao, L.: Clap4clip: Continual learning with probabilistic finetuning for vision-language models. Advances in neural information processing systems **37**, 129146–129186 (2024)
18. Khan, M.G.Z.A., Naeem, M.F., Van Gool, L., Stricker, D., Tombari, F., Afzal, M.Z.: Introducing language guidance in prompt-based continual learning. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 11463–11473 (2023)
19. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
20. Kurniawan, M.R., Song, X., Ma, Z., He, Y., Gong, Y., Qi, Y., Wei, X.: Evolving parameterized prompt memory for continual learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 13301–13309 (2024)
21. Le, M., Nguyen, C., Nguyen, H., Tran, Q., Le, T., Ho, N.: Revisiting prefix-tuning: Statistical benefits of reparameterization among prompts. In: International Conference on Learning Representations (2025), https://openreview.net/forum?id=QjTSaFXg25
22. Lester, B., Al-Rfou, R., Constant, N.: The power of scale for parameter-efficient prompt tuning. arXiv preprint arXiv:2104.08691 (2021)
23. Li, S., Liu, X., Liu, F., Jiao, L., Wang, J., Huang, X., Ma, Y., Chen, P., Li, L., Liu, X., et al.: Imagining vision from language for few-shot class-incremental learning. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 7395–7404 (2025)
24. Li, X.L., Liang, P.: Prefix-tuning: Optimizing continuous prompts for generation. arXiv preprint arXiv:2101.00190 (2021)
25. Li, X., Huang, L., An, Z., Feng, W., Yang, C., Diao, B., Wang, F., Xu, Y.: Geometric feature embedding for effective 3d few-shot class incremental learning. In: Forty-second International Conference on Machine Learning
26. Liang, Y.S., Li, W.J.: Inflora: Interference-free low-rank adaptation for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23638–23647 (2024)
27. Liu, H., Zhou, Y., Liu, B., Zhao, J., Yao, R., Shao, Z.: Incremental learning with neural networks for computer vision: a survey. Artificial intelligence review **56**(5), 4557–4589 (2023)
28. Liu, Y., Yang, M.: Sec-prompt: Semantic complementary prompting for few-shot class-incremental learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 25643–25656 (2025)
29. Liu, Y., Chen, Z., Wang, D., Luo, X., Chen, B., Lu, G.: Pet-gpra: Rethinking pet with gradient-aware prompting and router-free adapters for few-shot class-incremental learning. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 8303–8312 (2025)
30. Ma'sum, M.A., Pratama, M., Ramasamy, S., Liu, L., Habibullah, H., Kowalczyk, R.: Vision and language synergy for rehearsal free continual learning. In: International Conference on Learning Representations (2025)
31. McDonnell, M.D., Gong, D., Parvaneh, A., Abbasnejad, E., Van den Hengel, A.: Ranpac: Random projections and pre-trained models for continual learning. Advances in Neural Information Processing Systems **36**, 12022–12053 (2023)

32. Park, K.H., Song, K., Park, G.M.: Pre-trained vision and language transformers are few-shot incremental learners. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23881–23890 (2024)
33. Prabhu, A., Sinha, S., Kumaraguru, P., Torr, P., Sener, O., Dokania, P.: Randumb: Random representations outperform online continually learned representations. Advances in Neural Information Processing Systems **37**, 37988–38006 (2024)
34. Qiao, J., Tan, X., Chen, C., Qu, Y., Peng, Y., Xie, Y., et al.: Prompt gradient projection for continual learning. In: The Twelfth International Conference on Learning Representations (2023)
35. Ran, H., Gao, X., Li, L., Li, W., Tian, S., Wang, G., Shi, H., Ning, X.: Brain-inspired fast-and slow-update prompt tuning for few-shot class-incremental learning. IEEE Transactions on Neural Networks and Learning Systems (2024)
36. Rebuffi, S.A., Kolesnikov, A., Sperl, G., Lampert, C.H.: icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2001–2010 (2017)
37. Roy, A., Moulick, R., Verma, V.K., Ghosh, S., Das, A.: Convolutional prompting meets language models for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23616–23626 (2024)
38. Sayed, A.N., Himeur, Y., Varlamis, I., Bensaali, F.: Continual learning for energy management systems: A review of methods and applications, and a case study. Applied Energy **384**, 125458 (2025)
39. Senbao, Z., Shucheng, H., Pengyi, L., Mingxing, L.: A multimodal framework for 3d few-shot class-incremental learning. Multimedia Systems **31**(5), 1–13 (2025)
40. Smith, J.S., Hsu, Y.C., Zhang, L., Hua, T., Kira, Z., Shen, Y., Jin, H.: Continual diffusion: Continual customization of text-to-image diffusion with c-lora. arXiv preprint arXiv:2304.06027 (2023)
41. Smith, J.S., Karlinsky, L., Gutta, V., Cascante-Bonilla, P., Kim, D., Arbelle, A., Panda, R., Feris, R., Kira, Z.: 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. pp. 11909–11919 (2023)
42. Sun, H.L., Zhou, D.W., Zhao, H., Gan, L., Zhan, D.C., Ye, H.J.: Mos: Model surgery for pre-trained model-based class-incremental learning. arXiv preprint arXiv:2412.09441 (2024)
43. Sun, H., Zhou, J., He, X., Xu, J., Peng, Y.: Finefmpl: Fine-grained feature mining prompt learning for few-shot class incremental learning. IJCAI (2024)
44. Sur, T., Mukherjee, S., Rahaman, K., Chaudhuri, S., Khan, M.H., Banerjee, B.: Hyperbolic uncertainty-aware few-shot incremental point cloud segmentation. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 11810–11821 (2025)
45. Thengane, V., Khan, S., Hayat, M., Khan, F.: Clip model is an efficient continual learner. arXiv preprint arXiv:2210.03114 (2022)
46. Tian, Z., Zhang, D., Dai, H.N.: Continual learning on graphs: A survey. arXiv preprint arXiv:2402.06330 (2024)
47. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset (2011)
48. Wang, L., Xie, J., Zhang, X., Huang, M., Su, H., Zhu, J.: Hierarchical decomposition of prompt-based continual learning: Rethinking obscured sub-optimality. Advances in neural information processing systems **36** (2024)

49. Wang, L., Zhang, X., Su, H., Zhu, J.: A comprehensive survey of continual learning: theory, method and application. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)
50. Wang, Y., Huang, Z., Hong, X.: S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. Advances in neural information processing systems **35**, 5682–5695 (2022)
51. Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C.Y., Ren, X., Su, G., Perot, V., Dy, J., et al.: Dualprompt: Complementary prompting for rehearsal-free continual learning. In: European Conference on Computer Vision. pp. 631–648. Springer (2022)
52. Wang, Z., Zhang, Z., Lee, C.Y., Zhang, H., Sun, R., Ren, X., Su, G., Perot, V., Dy, J., Pfister, T.: Learning to prompt for continual learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 139–149 (2022)
53. Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., Fu, Y.: Large scale incremental learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 374–382 (2019)
54. Xu, W., Huang, T., Qu, T., Yang, G., Guo, Y., Zuo, W.: Filp-3d: Enhancing 3d few-shot class-incremental learning with pre-trained vision-language models. Pattern Recognition **165**, 111558 (2025)
55. Yang, L., Luo, Z., Zhang, S., Teng, F., Li, T.: Continual learning for smart city: A survey. IEEE Transactions on Knowledge and Data Engineering (2024)
56. Yang, Y., Zhou, J., Ding, X., Huai, T., Liu, S., Chen, Q., Xie, Y., He, L.: Recent advances of foundation language models-based continual learning: A survey. ACM Computing Surveys **57**(5), 1–38 (2025)
57. Ye, T., Dong, L., Xia, Y., Sun, Y., Zhu, Y., Huang, G., Wei, F.: Differential transformer. arXiv preprint arXiv:2410.05258 (2024)
58. Zhao, Y., Li, J., Song, Z., Tian, Y.: Language-inspired relation transfer for few-shot class-incremental learning. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)
59. Zhou, D.W., Sun, H.L., Ye, H.J., Zhan, D.C.: Expandable subspace ensemble for pre-trained model-based class-incremental learning. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23554–23564 (2024)
60. Zhou, D.W., Wang, Q.W., Qi, Z.H., Ye, H.J., Zhan, D.C., Liu, Z.: Class-incremental learning: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)
61. Zhuang, H., Liu, Y., He, R., Tong, K., Zeng, Z., Chen, C., Wang, Y., Chau, L.P.: F-oal: Forward-only online analytic learning with fast training and low memory footprint in class incremental learning. Advances in Neural Information Processing Systems **37**, 41517–41538 (2024)