

# CluMo: Cluster-based Modality Fusion Prompt for Continual Learning in Visual Question Answering

YULIANG CAI, University of Southern California, US

MOHAMMAD ROSTAMI, University of Southern California, US

Large vision-language models (VLMs) have shown significant performance boost in various application domains. However, adopting them to deal with several sequentially encountered tasks has been challenging because finetuning a VLM on a task normally leads to reducing its generalization power and the capacity of learning new tasks as well as causing catastrophic forgetting on previously learned tasks. Enabling using VLMs in multimodal continual learning (CL) settings can help to address such scenarios. To improve generalization capacity and prevent catastrophic forgetting, we propose a novel prompt-based CL method for VLMs, namely **Cluster-based Modality Fusion Prompt (CluMo)**. We design a novel **Key-Key-Prompt** pair, where each prompt is associated with a visual prompt key and a textual prompt key. We adopt a two-stage training strategy. During the first stage, the single-modal keys are trained via *K*-means clustering algorithm to help select the best semantically matched prompt. During the second stage, the prompt keys are frozen, the selected prompt is attached to the input for training the VLM in the CL scenario. Experiments on two benchmarks demonstrate that our method achieves SOTA performance. The code is publicly available at [here](#).

**JAIR Associate Editor:** Kai-Wei Chang

## JAIR Reference Format:

Yuliang Cai and Mohammad Rostami. 2025. CluMo: Cluster-based Modality Fusion Prompt for Continual Learning in Visual Question Answering. *Journal of Artificial Intelligence Research* 83, Article 24 (August 2025), 25 pages. DOI: [10.1613/jair.1.17940](https://doi.org/10.1613/jair.1.17940)

## 1 Introduction

Visual Question Answering (VQA) is a challenging multimodal task that involves answering natural language questions about a given image. Solving VQA problems requires the integration and understanding of both visual content and textual information to produce accurate and contextually appropriate answers. Recent progress in this field has been largely driven by the emergence of large pre-trained vision-language models (VLMs) such as CLIP [40] and ViLT [14], which are capable of jointly learning representations from both modalities and relating the semantic knowledge about an input image successfully to the text query and generalize well on unseen data [59, 62, 48]. However, most VQA models are trained under the assumption of static data distributions. In contrast, real-world applications often involve dynamic and evolving environments, where data arrives sequentially over time and exhibits non-stationary distributions [25, 38]. For instance, new questions and novel image types may appear that differ significantly from the original training data. In such scenarios, it becomes crucial for a VQA system to continuously adapt to new tasks while retaining previously acquired knowledge—a paradigm known as Continual Learning (CL).

Continual learning seeks to address the problem of catastrophic forgetting [9], where a model tends to forget previously learned information when trained on new tasks. To mitigate this issue, several categories of CL

---

Authors' Contact Information: Yuliang Cai, ORCID: [0009-0007-4628-225X](https://orcid.org/0009-0007-4628-225X), [caiyulia@usc.edu](mailto:caiyulia@usc.edu), University of Southern California, Los Angeles, California, US; Mohammad Rostami, ORCID: [0000-0001-7873-1004](https://orcid.org/0000-0001-7873-1004), [mrostami@isi.edu](mailto:mrostami@isi.edu), University of Southern California, Los Angeles, California, US.



This work is licensed under a [Creative Commons Attribution International 4.0 License](https://creativecommons.org/licenses/by/4.0/).

© 2025 Copyright held by the owner/author(s).

DOI: [10.1613/jair.1.17940](https://doi.org/10.1613/jair.1.17940)

algorithms have been proposed: Regularization-based methods [15, 25] introduce constraints during model updates to prevent drastic changes in parameters that are critical for past tasks. These methods often use importance weights to preserve key knowledge. Expansion-based methods [7, 4, 3] dynamically expand the model architecture by allocating additional parameters or modules for new tasks. This allows the model to learn new information with minimal interference to existing knowledge. Rehearsal-based methods [42, 47, 44] store a small set of representative samples from previous tasks in a memory buffer. These samples are replayed during training on new tasks to reinforce prior learning. Prompt-based methods [58, 57] utilize learnable prompts—task-specific or semantically-rich tokens appended to the model input—to guide the model’s attention and maintain task-specific representations. These methods adapt the model without modifying its core parameters, making them highly scalable and efficient for large VLMs. Integrating continual learning strategies into VQA systems is a promising direction toward building adaptive, lifelong learning agents that can operate robustly in open-world, evolving environments.

Most existing CL methods consider unimodal, that is, vision-only or language-only, settings and hence are sub-optimal to address VQA tasks. Applying uni-modal methods does not take the multimodal nature of VQA tasks and leads to limited view. Meanwhile, the uni-modal methods may ignore the rich and complex interaction between modalities and cause deteriorating performance [38, 52]. On the other hand, only a few CL methods are introduced to handle multimodal tasks. [17, 34] adopt knowledge distillation for CL-VQA, and [39, 61, 18] adopt prompt-style methods as solution. However, none of them explicitly focus on the interaction between different modalities to boost VQA tasks.

To tackle the mentioned shortcoming, we propose a novel two-stage prompt learning-based CL method, namely **Cluster-based Modality fusion prompt (CluMo)**, with the focus of modality interaction. Figure 1 visualizes the high-level idea of our approach. Our method adopts a pre-trained VLM as its backbone and benefits from a clustering-based modal-specific key strategy to boost generalization capacity and minimize catastrophic forgetting when the base VLM is trained on tasks encountered sequentially. More specifically, we use a clustering-based algorithm to train visual-prompt keys and textual-prompt keys during the first stage. During the second stage, we assign each input image-question pair with well-trained prompt keys to its corresponding visual keys and textual keys. We then use the combination of two modal-specific keys to find the best-matched prompt to adapt the model for the input task. We also benefit from knowledge distillation during training to further improve the performance. Our proposed method outperforms existing alternative methods. Our specific contribution includes:

- We propose a novel clustering-based prompt learning method for training VLMs in CL settings to address VQA tasks with vision-and-language inputs.
- We use a two-stage training strategy to train the prompt keys before training the whole model to guarantee optimal prompt selection that is necessary for generalization on the input.
- We offer extensive experiments to demonstrate that the proposed approach achieves SOTA performance against CL existing methods and offer insight about the reason of this improved performance.

## 2 Related Works

*Visual Question Answering.* Visual Question Answering (VQA) has been a pivotal task at the intersection of computer vision and natural language processing which led to advances on more complex tasks. Initially, VQA was formulated as a classification task in which answers are selected from a predefined set of answers [1] and was solved by using CNNs for image feature extraction and RNNs for text processing. With the development of transformer and BERT-like models [30, 23], performance in VQA tasks has significantly been improved due to the better capacity of capturing the intricate relationship between two modalities and generating the response in the form of meaningful and descriptive texts. In recent years, VQA task has become downstream task for pretrained large vision language model (VLM) [26, 41, 21, 31], which are initially pretrained on large-scale image-text dataset

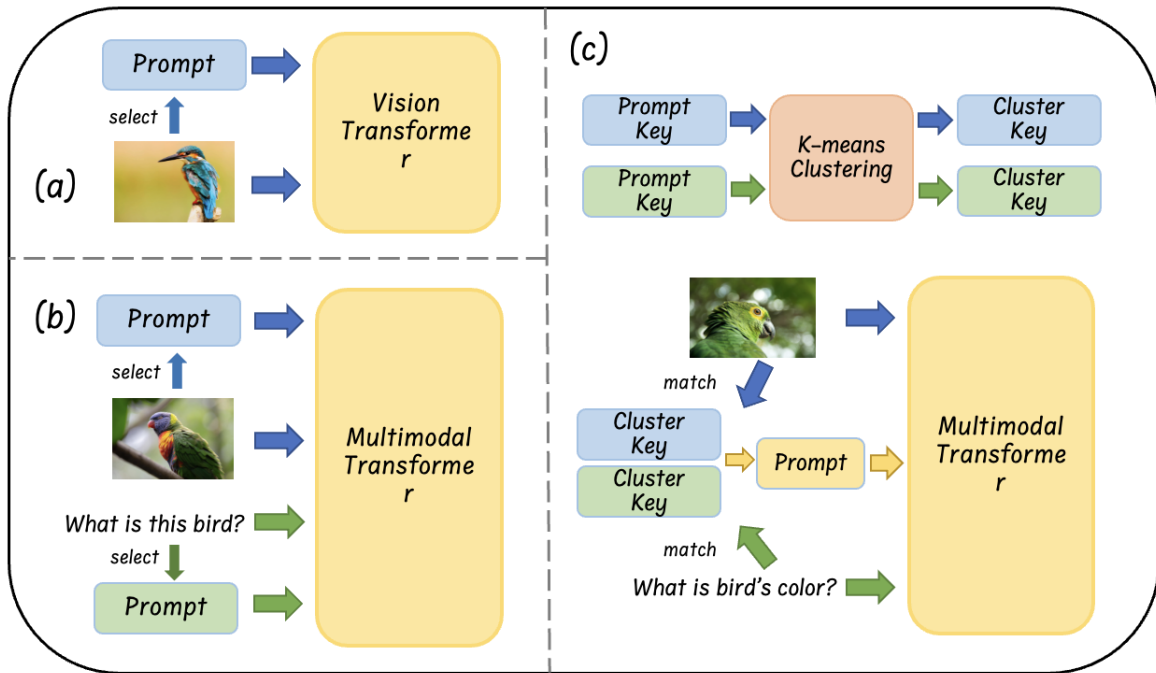


Fig. 1. Comparison between existing prompt-based CL methods and our proposed method: **(a)** Uni-modal based methods use image feature to select prompts from a prompt pool. **(b)** Multi-modal based methods use the image features to select image prompts and use the text features to select the text prompts. **(c)** We first train the prompt key using a clustering algorithm to form a cluster key and use the combination of the cluster key from both modalities to select the fusion prompt.

that are not necessary related to visual question answering. The pre-trained VLMs are essentially capable for zero-shot VQA, and can also be fine-tuned by VQA dataset for VQA downstream task specifically.

*Prompt-Based Learning.* Prompt learning, firstly introduced in NLP tasks [36], is based on providing a fixed function to condition a pre-trained model so that it gets extra information token which specializes it to perform the down-stream task. It is more memory-efficient than using Adapters [37] or LoRA [13] and has been used successfully to guide responses of VLMs for a particular task. In general, prompts are not only considered fixed texts, for example, “A photo of”, but also can be trainable soft-vectors for both image and text modalities [19, 24]. [63] introduces learnable text context, composed of trainable vectors, to replace the fixed textual input. [58] adopts trainable vectors append to image input to add additional knowledge to visual input. Our method adopts trainable vectors as prompts based on both image and text input.

*Continual Learning.* The central challenge in continual learning is mitigating catastrophic forgetting, where learning new tasks leads to the loss of knowledge from earlier ones [56, 7, 8]. A common mitigation strategy is experience replay, which interleaves past and current data during training [45, 27, 22, 29]. However, due to memory limits, only a subset of past examples can be stored, requiring intelligent sample selection methods. To bypass memory constraints altogether, generative models have been proposed. These models generate synthetic samples that approximate past task distributions, enabling generative rehearsal without storing actual data [49, 53, 46]. Another approach involves weight consolidation, where important parameters for old tasks are identified and

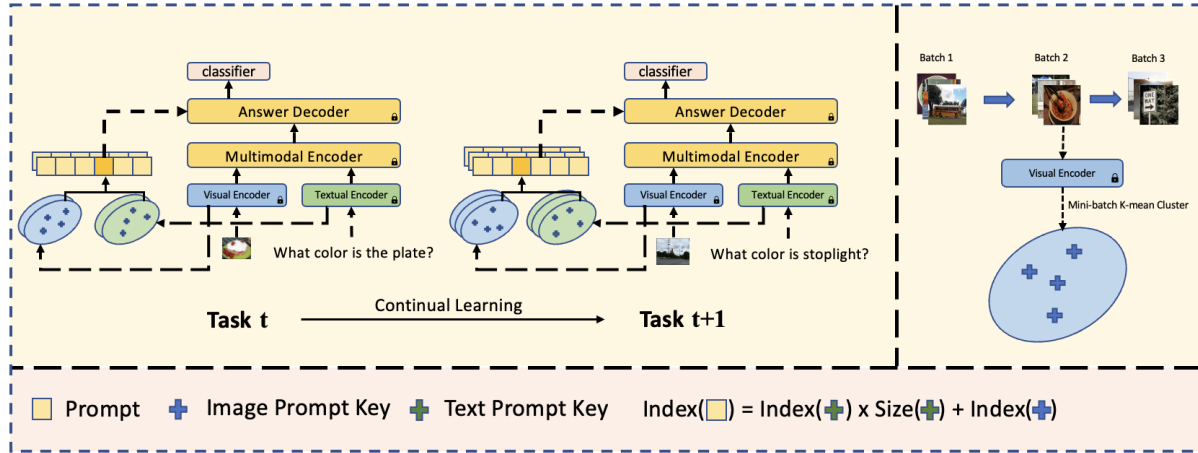


Fig. 2. Block diagram of the proposed approach: **Left**: the backbone contains a pre-trained frozen visual encoder, a textual encoder, and a multimodal encoder. The answer decoder shares the same architecture as multimodal encoder. During the training phase, the fixed number of visual prompt keys, textual prompt keys, and a prompt pool will be added for each new task. **Right**: the procedure of visual prompt key training consists of training the modal-specific prompt key by a sequence of randomly selected batches of training data from current task until convergence is reached. Same procedure for textual prompt key.

consolidated [33, 6], enforcing the model to learn new tasks using weights that are unimportant for preserving the past learned knowledge.

*Prompt Learning for Continual Learning.* Following classic CL, existing works for addressing CL with VLMs can be broadly categorized into replay-based methods and parameter-isolation approaches. Replay-based methods leverage episodic memory or synthetic data to retain prior knowledge [51, 60]. More recent directions include prompt-based continual learning, where task-specific prompts are learned or dynamically selected [58], and modular approaches that restructure the model architecture to accommodate new tasks efficiently [10]. Prompt learning has been used in CL to prevent catastrophic forgetting when a large pre-trained models is trained on a stream of sequentially encountered tasks. It allows a single model to quickly adapt to new tasks in the stream without needing extensive retraining. It allows scalability to learning a large number of tasks since each task primarily requires learning or generating new prompts rather than retraining the entire model. L2P [58] pioneered to connect prompt-based learning and CL. Instead of having a single shared prompt to learn all tasks, L2P introduced the concept of “prompt pool” to maintain prompts for different tasks independently from each other. DualPrompt [57] extended the idea of prompt pool in L2P by introducing task-specific prompt and task-agnostic prompt. S-Prompt [55] applied clustering to build the prompt pool with domain-specific prompts. These prompt learning methods for CL only consider single-modality, such as vision-only or text-only, and hence are sub-optimal for tasks with multi-modal inputs such VQA when the modalities are related. Our method benefits from the specific properties of multi-modal data to address VQA in CL settings using prompt learning and leads to performance improvements against these methods.

### 3 Problem Description

Consider a set of VQA tasks,  $\{\mathcal{T}_i\}_{i=1}^T$ , which are encountered sequentially and each of them is from different domain. For each of the tasks, a labeled training dataset  $\mathcal{D}^i = \{ \langle (I_i^j, T_i^j)^i, y_i^j \rangle_{j=1}^{N_i} \}$  is accessible,  $N_i$  denotes the

size of dataset,  $I_i^j \in \mathbb{R}^{H \times W \times C}$  denotes the input image,  $T_i^j \in \mathbb{R}^{L \times |V|}$  denotes the input text, and  $y_i^j$  denotes the text-typed discrete label. The order in which the VQA tasks are observed is not known in advance and the training data points are assumed to be drawn iid from a task-specific joint distribution  $p_i^t(\cdot, \cdot, \cdot)$ . Upon learning each task, the model moves forward to learn the next task. Since all the previously learned tasks can be encountered at any time during testing in the future, the model should learn new tasks such that its knowledge of previously learned tasks is maintained, by preventing catastrophic forgetting.

We formulate our problem in a domain-diverse **domain-incremental** learning setting [16] which assumes all the tasks are from different domains and the boundaries between them are known during training and inference time. However, the task identities are not known during inference. We consider that each task can be learnt individually by adapting a pre-trained large multimodal transformer  $f_{\theta_M}^i(\cdot, \cdot)$  via minimizing a suitable discrimination loss  $\mathcal{L}$ , for example, cross entropy. In our approach, all the model parameters, except the final classifier layer  $\theta_{cls}$ , are frozen during training to preserve the generalizability of the model. We benefit from prompt learning to enable using a single model to learn all tasks. To prevent catastrophic forgetting, a trainable task-specific prompt pool, which contains several task-specific prompts, is attached to the model  $f_{\theta_M}^i(\cdot, \cdot)$  such that the best-semantically-matched prompt is selected based on image and text inputs for task specialization. The prompt is then pre-pended to the input vectors so that the output is generated based on specialization. Our method is rehearsal-free and does not need any memory buffer similar to prior approaches [28, 42].

## 4 Proposed Architecture

Our architecture, named **Cluster-based Modality fusion prompt (CluMo)**, is a pool of key-key-prompt pair that contains two unimodal task-specific cluster-based prompt keys for vision and text embeddings and one prompt. The combination of the selections from both keys is then used to select the best matched prompt from the prompt pool. A high-level diagram of our approach is presented in Figure 2. In this section, we first introduce the preliminaries such as backbone model and prompt pool-based method in 4.1, and modality fusion prompt in Sec. 4.2, then the cluster-based prompt key is described in Sec. 4.3, and the training and the inference strategy is discussed in Sec. 4.4.

### 4.1 Preliminary

*Backbone.* The base multimodal transformer contains three encoders: the visual encoder  $E_v$ , the textual encoder  $E_t$ , and the multimodal fusion encoder  $E_f$ . Given a image input  $I$  (e.g. a single image), and a textual input  $T$  (e.g. a question), the data processing pipeline for the model is:

$$\hat{y}(I, T) = \mathcal{F}(E_f([E_v(I); E_t(T)])), \quad (1)$$

where  $\mathcal{F}(\cdot)$  is the classifier to predict the answer, and  $\hat{y}$  is the predicted answer.

*Prompt Pool.* As an adoption of prompt learning in continual learning, a prompt pool, in general format, is a set of trainable key-value (**k-p**) pair, in which  $\mathbf{k} \in R^{1 \times D}$  denotes the ‘‘prompt key’’, and  $\mathbf{p} \in R^{L_p \times D}$  is the prompt.  $L_p$  and  $D$  denote the length and dimension of the prompt. Given an input image  $I$ , we compute image feature vector  $V_I = E_v(I) \in R^{L_v \times D}$ , where  $L_v$  is the dimension of the features, after passing the image through the visual encoder.  $V_I[0]$  is matched with all the keys  $\mathbf{k}$  within the prompt pool via similarity score, such as L2 similarity, to find the most similar  $\mathbf{k}_i$ . In general condition, the corresponding  $\mathbf{p}_i$  is selected and preprend to  $I$  as  $I' = [\mathbf{p}_i; I]$ . Parameters of  $\mathbf{k}$  and  $\mathbf{p}$  are updated through back-propagation during the training. However, in our setting, **the prompt is preprended to the image feature vector**  $V_I' = [\mathbf{p}_i; V_I]$ , and we adopt a two stage training strategy that prompt keys are trained before model and prompts.

## 4.2 Modality Fusion Prompt

Previous prompt-based CL methods such as L2P [58] associate each prompt in the prompt pool with a single prompt key to form Key-Value pair. In practice, the prompt keys in prompt-based CL can be considered as cluster centers. These cluster encode a notion of similarity between the prompts. The input feature vectors that form a cluster in the feature space can be assigned to these cluster centers, which are prompt keys. The intuition behind this idea is that **feature vectors with small geometric distance in the feature space are semantically similar** [55].

However, such a key-value pair design considers only single modality without tasks with multimodal inputs. The reason is that different input modalities contain different or complementary semantic information. Hence, having prompt keys that associate with each modality can help guiding prompt selection, which is more comprehensive and representative in term of semantic properties of each modality. Thus, we propose a task-specific prompt pool architecture, namely **Modality Fusion Prompt**, which is composed of the **visual prompt keys**  $\mathbf{K}_v$ , the **textual prompt keys**  $\mathbf{K}_t$ , and the **prompt pool**  $\mathbf{P}$  as following:

$$\begin{aligned}\mathbf{K}_t &= [\mathbf{k}_{t_1}, \mathbf{k}_{t_2}, \dots, \mathbf{k}_{t_{S_t}}], \\ \mathbf{K}_v &= [\mathbf{k}_{v_1}, \mathbf{k}_{v_2}, \dots, \mathbf{k}_{v_{S_v}}], \\ \mathbf{P} &= [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_{S_p}], \\ \mathbf{k}_{t_i} &\in R^D, \mathbf{k}_{v_j} \in R^D, \mathbf{p}_k \in R^{L_p \times D},\end{aligned}\tag{2}$$

where  $S_t$ ,  $S_v$ , and  $S_p$  are the sizes of textual prompt key, the visual prompt key, and the prompt pool, respectively.  $L_p$  is the length of each prompt and  $D$  is the hidden dimension of the transformer backbone. The prompt pool size  $S_p$  is then determined as  $S_p = S_v \times S_t$ . Each prompt is associated with the unique combination of one visual prompt key and one textual prompt key. Given a specific visual prompt key  $K_{v_m}$  and a specific textual prompt key  $K_{t_n}$ , the Key-Key-Value pair is defined as the following:

$$(\mathbf{k}_{v_i}, \mathbf{k}_{t_j}) \rightarrow \mathbf{p}_{i*S_v+j}\tag{3}$$

As modality fusion prompt is **task-specific**, new visual prompt keys, textual prompt keys and a prompt pool will be initialized for each of the new coming task. The previous ones are frozen during training. Based on our experiment setting, the backbone contains 290.3M parameters, while the modality fusion prompt only contains 73728 parameters for every task, which is only 0.025% the size of original backbone model.

## 4.3 Cluster-based Prompt Key

$K$ -means clustering algorithm has been widely adopted in machine learning algorithms for semantic separation and understanding, where data from different domains can be explicitly separated via clustering in an unsupervised way [55][5]. However, even though the data from single task belong to the same domain, they can still be further divided into sub-domains based on the semantic property. To make each prompt key be the semantical cluster center of the sub-domains for both vision and text inputs, we adopt mini-batch  $K$ -means clustering algorithm on prompt keys of  $\mathbf{K}_v$  and  $\mathbf{K}_t$  to make each prompt key diverse and representative. Let  $\mathcal{B} = (I, T)$  be the random batch from the training dataset. We extract the image feature vector  $v_I$  and the text feature vector  $v_T$  as follows:

$$\mathbf{V}_I = E_v(I), \mathbf{V}_T = E_t(T),\tag{4}$$

where  $\mathbf{V}_I \in R^{B \times L_I \times D}$  and  $\mathbf{V}_T \in R^{B \times L_T \times D}$ ,  $B$  is the batch size,  $L_I$  and  $L_T$  are the length of vectors for image and text features, represent the embedded image and text input respectively. For visual prompt key clustering, each image feature vector,  $V_{I_n}$ , is set by taking mean along second the dimension such that  $\hat{V}_{I_n} \in R^{B \times D}$ , and  $\hat{V}_{I_n}$  is used to compare with every prompt key in  $\mathbf{K}_v$ :

$$\text{similarity}(n, m) = \|\hat{V}_{I_n} - \mathbf{K}_{v_m}\|_2,\tag{5}$$

**Algorithm 1** PromptKeyTraining

**Require:** Dataset  $\mathcal{D}$ , Image Prompt Key Pool  $\mathbf{K}_v$ , Text Prompt Key Pool  $\mathbf{K}_t$ , Image Prompt Size  $S_i$ , Text Prompt Size  $S_t$

```

while Not Converge do
  Random Select batch of image  $I$ , text  $T$  from  $\mathcal{D}$ 
   $\hat{\mathbf{V}}_{I_m} = \text{mean}(E_v(I), \text{dim} = 1)$ 
   $\hat{\mathbf{V}}_{T_m} = \text{mean}(E_t(T), \text{dim} = 1)$ 
   $\text{Cluster}_I = \text{dictionary}()$ 

   $\text{Cluster}_T = \text{dictionary}()$ 

  for  $i, t$  in  $\hat{\mathbf{V}}_{I_m}, \hat{\mathbf{V}}_{T_m}$  do
     $\text{Key}_{img} = \text{image key with top similarity}(i, \mathbf{K}_v)$ 

     $\text{Key}_{txt} = \text{text key with top similarity}(t, \mathbf{K}_t)$ 
     $\text{Cluster}_I[\text{Key}_{img}].\text{append}(i)$ 
     $\text{Cluster}_T[\text{Key}_{txt}].\text{append}(t)$ 
  end for
  for  $i$  in  $S_i$  do
     $\mathbf{K}_v[i] = \text{mean}(\text{Cluster}_I[i])$ 
  end for
  for  $i$  in  $S_t$  do
     $\mathbf{K}_t[i] = \text{mean}(\text{Cluster}_T[i])$ 
  end for
end while

```

and the prompt key with highest similarity is assigned to match  $\mathbf{V}_{I_m}$ . After calculation of the whole batch  $\mathcal{B}$ , the prompt keys are then updated by calculating the mean of all  $\hat{\mathbf{V}}_{I_m}$  assigned to that specific visual prompt key. We repeat the above step until the convergence. The procedure of updating the text prompt key  $\mathbf{K}_t$  is similar to updating the image prompt keys. Algorithm 1 summarizes our approach for prompt key training.

#### 4.4 Training and Inference

During training, we adopt a **two-stage training** strategy to ensure that the prompt keys are correctly settled before learning the current task. In the first stage of learning each task  $T_i$ , we random select batches from the current task's dataloader to train mini-batch  $k$ -means cluster on the visual and the textual prompt key  $\mathbf{K}_v$  and  $\mathbf{K}_t$  until reaching the convergence of the clustering algorithm. During the second stage, the trained  $\mathbf{K}_v$  and  $\mathbf{K}_t$  are frozen. Within the iteration of training dataloader, each training instance is assigned to its nearest prompt key using  $k$ -nearest neighbor (KNN) algorithm to find the best match prompt  $\mathbf{p}_k$  from the prompt pool  $\mathbf{P}$ .  $\mathbf{p}_k$  is then attached to the model pipeline. Inspired by [57], we also add a shared general prompt,  $\mathbf{p}_g$ , with has the same dimension of  $\mathbf{p}_k$  for maintaining cross-task knowledge:

$$\hat{y}(I, T) = \mathcal{F}(E_f([\mathbf{p}_g; \mathbf{p}_k; E_v(I); E_t(T)])) \quad (6)$$

During the second stage, we also use knowledge distillation to further boost the performance. Before the training of task  $T$ , we keep a frozen copy of model after finishing  $T - 1$ , denoted as  $\mathcal{M}_{T-1}$ . To prevent significant

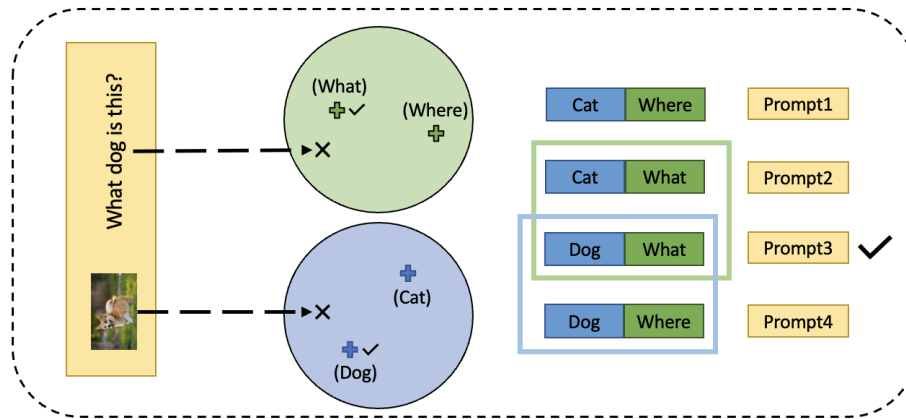


Fig. 3. Naive Example of Prompt Selection. Consider a naive animal VQA dataset which only contains dog and cat images and questions only about “what” and “where”. During the first stage training, the visual prompt keys and textual prompt keys are learnt to represent “dog”/“cat” and “where”/“what” respectively (in realistic settings the keys will learn the implicit cluster instead of explicit category). Given a test image-question pair, the image and question are projected to their modality-specific feature space through the encoders. The nearest prompt keys, which are keys represent “dog” and “what” are selected. The combination of the two prompt keys lead to “prompt3” which is finally selected.

---

### Algorithm 2 Continual Learning Procedure

---

**Require:** Datasets  $\mathbf{D}$ , Pretrained Model  $\mathbf{M}$

Freeze  $\mathbf{M}$  except  $\mathbf{M}$ .classifier

$\mathbf{M}$ .CluMoList = dict()

**for** dataset  $D_i$  in  $\mathbf{D}$  **do**

    Initialize new CluMo  $C_i$

$\mathbf{M}$ .CluMoList[ $D_i$ ] =  $C_i$

**PromptKeyTraining**( $D_i, C_i$ )

Freeze visual key and textual key of  $C_i$

**for** Batch  $\mathbf{B}$  in  $D_i$  **do**

**for** img  $i$ ,txt  $t$  in  $\mathbf{B}$  **do**

        Select  $Key_{img}$  based on  $i$  from  $C_i$

        Select  $Key_{txt}$  based on  $t$  from  $C_i$

        Find prompt  $P$  by  $Key_{img}$  and  $Key_{txt}$  from  $C_i$

**end for**

        loss = **Train**( $\mathbf{M}, \mathbf{B}, P$ )

        loss.backward()

**end for**

**end for**

---

parameter shift, we pass the same input image and question to both  $\mathcal{M}_T$  and  $\mathcal{M}_{T-1}$  and add the difference between the two model’s output to the loss:

$$\mathcal{L}_{KD}(\mathbf{I}, \mathbf{T}) = \text{MSE}(\hat{y}_{\mathcal{M}_T}(\mathbf{I}, \mathbf{T}), \hat{y}_{\mathcal{M}_{T-1}}(\mathbf{I}, \mathbf{T})). \quad (7)$$

The final objective loss function would be:

$$\mathcal{L} = \mathcal{L}_{ce}(\hat{y}(\mathbf{I}, \mathbf{T}), y) + \mathcal{L}_{KD} \quad (8)$$

Where  $\mathcal{L}_{ce}$  is the same cross entropy loss. The overall training procedure for all tasks come in sequence is presented in Algorithm 2.

During **inference**, the model is frozen and we follow a task-agnostic procedure to let the test instance select the best matched prompt across all prompt pools. The text image-text pair is sent to all the task-specific prompt pool and the image input is aligned with the best-matched image prompt key while the text input is aligned with the best-matched text prompt key within each pool. After getting the best-matched keys, we select the pair of image/text prompt key with the top sum of text similarity and image similarity. The combination of prompt keys is deployed to find the corresponding prompt, which is pre-pend to the output of multimodal encoder. To help better understand the procedure of prompt selection, an visualization example is provided in Figure 3.

## 5 Experiments

We evaluate the performance of **CluMo** on two continual learning VQA benchmarks and adopt 9 continual learning methods from different category as baselines. In this section, we introduce the experiment setup in Sec 5.1, conduct comparison experiments against other baselines in Sec 5.2, exam the effect of each component of **CluMo** via ablative study in Sec 5.3, and provide more comprehensive analysis of our method in Sec 5.4. Our implementation code is publicly available: <https://github.com/YuliangCai2022/CLUMO>.

### 5.1 Experiment Setup

*Backbone.* We used the public pre-trained large multimodal transformer, ALBEF [21], as our backbone for VQA task. It consists of an image encoder, a text encoder, a multimodal encoder, which uses cross-attention between the two modalities. Specifically for VQA tasks, an pre-trained answer decoder is append after the multimodal encoder, which has the same architecture as its multimodal encoder.

*Baselines for comparison.* We use nine methods for comparison. We include algorithms from major CL approaches. We include two regularization-based methods: **EWC** [15] and **LwF** [25], four rehearsal-based methods: **ER** [44], **GEM** [54], **MAFED** [35] and **VQACL** [61]. We also include three SOTA prompt-based continual learning methods, **L2P** [58], **DualPrompt** [57], and **S-Prompt** [55]. We also include finetuning to demonstrate the positive effect of CL. Following the original setting of each method, we leave the whole backbone model unfrozen for non-prompt-based methods and freeze the whole backbone model for prompt-based methods except for the classifier. To make the fair comparison, we fit all the continual learning methods into our backbone, ALBEF, instead of using the original model proposed in each method. Detailed description of each baseline method is presented in Appendix A.

*CL Tasks.* We evaluate our method on tasks built using the **CLOVE** [18] dataset which is a VQA-based continual learning dataset that has been used in the literature before. The benchmark contains two separate benchmarks for different scenario, including scene-incremental setting benchmark, **CLOVE-scene**, and function-incremental setting benchmark, **CLOVE-function**. Each of the task sets contains six tasks which are domain-specific and diverse from each other. For more details about **CLOVE** and the tasks we use, please refer to the Appendix B and Table 9. Furthermore, we introduce extra dataset as supplement experiments. We design VQA-continual learning setting with five datasets: **COCOQA** [43], **OKVQA** [32], **VQAv2** [11], **TEXTQA** [50], **GQA** [2]. We refer to this

Table 1. Comparative experimental results: the accuracy and forgetting rate for different task order. For each task sequence,  $A \uparrow$  indicates the accuracy of the method, while  $F \downarrow$  is the forgetting rate of each.

| Method       | CLOVE-scene  |                |              |                |              |                |              |                |              |                |              |                |
|--------------|--------------|----------------|--------------|----------------|--------------|----------------|--------------|----------------|--------------|----------------|--------------|----------------|
|              | abcdef       |                | dbafec       |                | bdcafe       |                | acbefd       |                | caefdb       |                | bafedc       |                |
|              | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ |
| Finetune     | 34.03        | 34.28          | 34.89        | 34.99          | 38.83        | 21.65          | 34.45        | 35.14          | 34.42        | 34.47          | 33.95        | 35.67          |
| EWC          | 37.49        | 28.04          | 37.00        | 29.10          | 37.95        | 27.46          | 37.99        | 29.68          | 37.13        | 28.63          | 37.83        | 27.97          |
| LwF          | 38.18        | 26.82          | 35.03        | 32.84          | 37.31        | 29.11          | 37.85        | 29.87          | 37.94        | 28.15          | 38.21        | 27.48          |
| ER           | 41.05        | 19.92          | 42.09        | 17.12          | 42.37        | 18.09          | 41.91        | 20.28          | 41.11        | 19.65          | 42.08        | 20.52          |
| GEM          | 41.52        | 18.33          | 43.14        | 14.73          | 42.89        | 17.43          | 42.54        | 20.13          | 41.90        | 20.88          | 43.11        | 19.86          |
| MAFED        | 42.72        | 17.23          | 43.48        | 16.53          | 44.22        | 16.90          | 43.53        | 18.12          | 43.03        | 17.21          | 44.68        | 18.04          |
| VQACL        | 43.46        | 16.94          | 44.03        | 14.18          | 44.87        | 16.24          | 44.85        | 17.91          | 43.77        | 16.90          | 45.01        | 16.85          |
| L2P          | 43.01        | 18.22          | 45.84        | 15.03          | 44.64        | 17.41          | 45.63        | 14.96          | 44.78        | 17.99          | 46.58        | 14.85          |
| DualPrompt   | 45.51        | 15.86          | 46.58        | 13.49          | 45.83        | 16.48          | 46.27        | 15.45          | 46.21        | 15.89          | 47.01        | 13.16          |
| S-Prompt     | 45.73        | 14.11          | 45.93        | 14.17          | 46.99        | 14.38          | 46.68        | 14.77          | 47.53        | 12.19          | 44.09        | 22.32          |
| <b>CluMo</b> | <b>48.23</b> | <b>9.69</b>    | <b>47.52</b> | <b>9.55</b>    | <b>48.18</b> | <b>9.57</b>    | <b>48.00</b> | <b>8.78</b>    | <b>47.97</b> | <b>11.17</b>   | <b>47.68</b> | <b>11.06</b>   |

| Method       | CLOVE-function |                |              |                |              |                |              |                |              |                |              |                |
|--------------|----------------|----------------|--------------|----------------|--------------|----------------|--------------|----------------|--------------|----------------|--------------|----------------|
|              | soarkl         |                | rsaolk       |                | osrlak       |                | oarlks       |                | skaolr       |                | ksoarl       |                |
|              | $A \uparrow$   | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ | $A \uparrow$ | $F \downarrow$ |
| Finetune     | 31.55          | 53.76          | 37.34        | 39.64          | 23.34        | 57.32          | 24.09        | 62.79          | 16.34        | 74.82          | 17.50        | 84.46          |
| EWC          | 35.70          | 47.92          | 37.82        | 41.55          | 38.92        | 40.48          | 40.74        | 33.89          | 37.53        | 37.22          | 40.85        | 32.32          |
| LwF          | 37.18          | 46.86          | 36.81        | 44.12          | 39.21        | 39.81          | 36.81        | 41.29          | 30.49        | 53.11          | 29.17        | 55.84          |
| ER           | 42.22          | 32.97          | 39.78        | 38.62          | 41.22        | 35.79          | 37.14        | 33.38          | 33.41        | 48.99          | 38.23        | 38.01          |
| GEM          | 44.58          | 30.87          | 41.43        | 29.46          | 40.87        | 32.98          | 39.81        | 28.77          | 36.88        | 39.14          | 40.26        | 31.87          |
| MAFED        | 43.48          | 31.01          | 40.62        | 34.12          | 42.70        | 30.17          | 40.23        | 26.41          | 35.15        | 44.39          | 40.08        | 34.74          |
| VQACL        | 44.36          | 18.28          | 42.89        | 27.45          | 43.58        | 26.86          | 41.25        | 23.53          | 38.67        | 40.31          | 41.84        | 28.65          |
| L2P          | 44.80          | 16.38          | 43.39        | 21.26          | 43.27        | 21.97          | 42.54        | 19.18          | 40.4         | 31.92          | 43.37        | 24.19          |
| DualPrompt   | 45.01          | 15.90          | 44.26        | 17.43          | 44.66        | 18.50          | 43.69        | 15.31          | 39.32        | 34.78          | 45.65        | 20.54          |
| S-Prompt     | 45.45          | 13.47          | 45.01        | 14.76          | <b>45.27</b> | 14.29          | 42.98        | 20.20          | 42.85        | 25.82          | 44.09        | 22.32          |
| <b>CluMo</b> | <b>46.72</b>   | <b>11.14</b>   | <b>46.29</b> | <b>11.28</b>   | 45.13        | <b>13.52</b>   | <b>46.55</b> | <b>8.21</b>    | <b>47.04</b> | <b>11.15</b>   | <b>45.77</b> | <b>11.69</b>   |

dataset as Merged-Based VQA continual learning dataset. Due to the computational limit and the unbalance of each task's size, we restrict each training dataset size to 20K, and each testing dataset size to 3K by selecting the top 20K/3K data from the dataset and thus form it in a way similar to the scale of CLOVE dataset.

*CluMo Hyper-parameter.* In all of our comparative experiments, for every task, we set the visual prompt key size and textual prompt key size to be 3, with dimension  $1 \times 768$  for each. We set the length of prompt to be 10, hence each prompt has dimension  $10 \times 768$ . All the choice of hyper-parameters are made by selecting the best-performed ones from multiple analytic experiments.

*Metrics for comparison.* We use the average accuracy and the average forgetting rate on all tasks to evaluate the performance of our method and its ability to tackle catastrophic forgetting. Different from dataset such as VQAv2 [12], where each question is paired with different ground truth answers, questions in CLOVE dataset only contains exactly one correct answer for each question. Thus, the accuracy is simply calculated by  $y == \hat{y}$  for every training and testing data instance. On the other hand, the forgetting rate is calculated as:

$$F = \frac{A_i - A_{ij}}{A_i} \quad (9)$$

where  $A_i$  is the accuracy of task  $i$ , and  $A_{ij}$  is the accuracy of task  $i$  after the model is trained on task  $j$ .

*Hardware Setup and Hyper-parameter.* All experiments were conducted using a single Nvidia A40 GPU. We employed the AdamW optimizer across all experiments, utilizing a cosine learning rate scheduler, and set the initial learning rate to  $lr = 3 \times 10^{-4}$ . The models were trained for 5 epochs with a batch size of 16. The hyper-parameter settings for each of the methods are listed below:

**EWC:** we set the fisher sample percentage to be 0.1 and ewc loss weight equals to 0.1.

**Experiment Replay:** we store 1% of data from each tasks into the memory buffer. During the training of the current task, we randomly select a batch of data from the memory buffer to train the model for every 100 batches of current data training.

**GEM:** we also store 1% of data to memory buffer, which is randomly picked from each tasks.

**CluMo:** we configured the visual prompt key size ( $S_v$ ) and the text prompt key size ( $S_t$ ) both to 3, with the prompt length ( $L_p$ ) set to 10.

**L2P:** we set the prompt pool size equals to 20 and prompt length to be 5.

**DualPrompt:** the G-Prompt was inserted into layers 0 and 1 of the visual encoder, while the E-Prompt was integrated into layers 2, 3, and 4.

**S-Prompt:** we set the prompt length to be 10 and prompt pool size to be 30.

Furthermore, for prompt-based techniques such as L2P, DualPrompt, and SPrompt, we opted to freeze the entire backbone model, allowing only the final classifier layer to remain trainable. Conversely, for all other baseline methods, no parameters were frozen, ensuring the entire network was fine-tuned during training.

## 5.2 Comparative Results

We conduct the comparison experiments on both the **CLOVE-scene** and **CLOVE-function** task sets with a randomly selected task order. In table 1, the task order *abcdef* represents the CL tasks: *ShopAndDining*, *WorkPlace*, *HomeOrHotel*, *Transportation*, *SportAndLeisure*, *Outdoors* in sequence. The *oarls* in **CLOVE-function** represents tasks: *ObjectRecognition*, *AttributeRecognition*, *RelationReasoning*, *LogicReasoning*, *KnowledgeReasoning* and *SceneTextRecognition*.

We observe in Table 1 that our method outperforms all the baselines across all task order sets in terms of both average accuracy and average forgetting rates. We also observe that the performance of different method within the same group tend to be similar. The regularization-based methods, **EWC** and **LwF**, obtain the sub-optimal accuracy and forgetting rate besides. The reason is that the domain for each task in the dataset is significantly different from the rest of tasks and hence regularization methods fail to capture the common space of the parameter distribution. This challenge makes it difficult to maintain the accuracy of the current task and previous tasks at the same time using regularization. The replay methods, **ER**, **GEM**, **MAFED** and **VQACL**, achieve better performance than regularization-based methods. This can be explained by the fact that replaying the data from previous task is an efficient way to remind the model and adjust its parameter distribution not too diverse from previous ones. However, because we need to rely on a memory buffer to store samples for replay, these methods are memory-consuming and thus not space-efficient. Moreover, replay-based methods are still limited by the upper-bound of joint training, as they generally can only reduce catastrophic forgetting without boosting the accuracy of individual tasks. On the other hand, the prompt-based methods, namely **L2P**, **DualPrompt**, and **SPrompt**, achieve superior performances compared to more traditional CL methods. Rather than tuning the whole model with regularization, prompt-based methods store the prior knowledge in trainable prompts, which are smaller and more efficient than memory buffer, and keep the main body of backbone model frozen. With the combination of generalization capacity of pre-trained model and specific previous knowledge stored in prompt, prompt-based method can outperform other methods. Among all the methods, our method achieve the best performance.

Table 2. Comparison experiments on the merged-based VQA dataset consists of **C**:COCOQA, **O**:OKVQA, **V**:VQAv2, **T**:TEXTQA and **G**:GQA with different task orders.

| Method       | COCOQA-OKVQA-VQAv2-TEXTQA-GQA |             |              |              |              |              |
|--------------|-------------------------------|-------------|--------------|--------------|--------------|--------------|
|              | COVTG                         |             | VGOCT        |              | OGCTV        |              |
|              | A ↑                           | F ↓         | A ↑          | F ↓          | A ↑          | F ↓          |
| Finetune     | 28.11                         | 38.41       | 34.84        | 34.27        | 31.02        | 19.97        |
| EWC          | 29.94                         | 33.02       | 36.00        | 29.10        | 34.41        | 16.72        |
| LwF          | 30.72                         | 31.93       | 35.03        | 32.84        | 33.65        | 18.08        |
| ER           | 32.33                         | 24.68       | 36.25        | 27.12        | 35.87        | 15.31        |
| GEM          | 34.07                         | 20.19       | 37.62        | 19.28        | 36.52        | 14.61        |
| MAFED        | 34.65                         | 18.68       | 38.22        | 17.41        | 36.23        | 15.15        |
| VQACL        | 35.89                         | 16.50       | 39.83        | 15.31        | 37.08        | 13.22        |
| L2P          | 34.90                         | 21.56       | 38.70        | 17.11        | 35.90        | 14.69        |
| DualPrompt   | 35.21                         | 22.07       | 39.23        | 16.36        | 37.15        | 12.88        |
| S-Prompt     | 36.08                         | 19.41       | 40.17        | 13.85        | 37.95        | <b>11.97</b> |
| <b>CluMo</b> | <b>39.90</b>                  | <b>8.62</b> | <b>40.83</b> | <b>12.60</b> | <b>38.29</b> | 13.63        |

Table 3. Ablative Experiments

| Methods              | Accuracy | Forgetting |
|----------------------|----------|------------|
| Full Method          | 48.23    | 9.69       |
| Ablative KD          | 46.67    | 12.97      |
| Ablative Clustering  | 47.36    | 10.09      |
| Ablative Textual Key | 47.45    | 10.05      |
| Ablative Visual Key  | 47.23    | 10.54      |

Compared with the baseline prompt-based methods which only consider visual modality for prompt selecting and updating, **CluMo** takes care of both the visual and textual modalities, as well as the fusion of the two for selecting the prompt which deploys the given information more comprehensively to process the prompt. Our design thus fits better in multimodal learning scenario than other existing continual learning methods.

The comparison result for the Merged-Based VQA continual learning dataset is presented in table 2. From the table, we observe that task order has a relatively significant effect on the performance, which, based on our hypothesis, is due to the difference similarity between each task. For example, COCOQA and VQAv2 has similar benchmark design, while TextQA specifically focuses on understanding the text in within the image. When learning similar tasks, the forward transfer is more likely to happen and the forgetting rate will drop. On the other hand, learning dis-similar tasks in sequence will cause forgetting rate to increase. **CLUMO** still beats other baseline methods with different task orders.

### 5.3 Ablation Experiments

To offer a better insight about our method, we perform an ablation study for each component of **CluMo** to study the positive contribution of each component. We study the effect of the following:

- **Visual Prompt Key**, key to separate the inner-task image features by their semantic property.

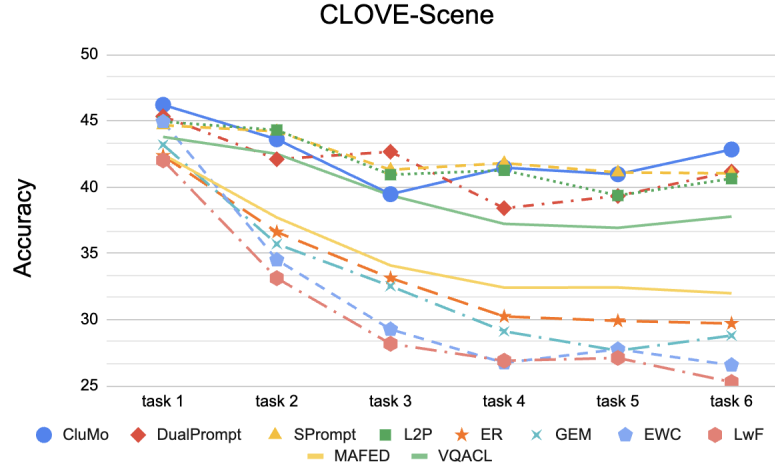


Fig. 4. **Accuracy** of the first task after the model trained on the following tasks. The leading performance of **CluMo** indicates its leading capacity of prevent catastrophic forgetting.

- **Textual Prompt Key**, key to separate the inner-task text features by their semantic property.
- **Minibatch  $k$ -means Clustering** which train the prompt keys as centers of clustering algorithm to better fit the semantic meaning.
- **Knowledge Distillation**, to prevent the drastic parameter shift of unfrozen classifier.

We conduct ablation experiment on **CLOVE-scene** dataset with the task order *abcdef*. We set the size for both the visual prompt key and the textual prompt key to be three. For ablative text experiments, we remove the text prompt keys and change the size of visual prompt keys to 9 to achieve the same total prompt size. We also removed the visual prompt key which is the same for ablative visual experiments.

Results for this experiment is presented in Table 3. We observe that despite having the same number of prompts, the performance values of Ablative Textual Key and Ablative Visual Key are lower than our full pipeline. This result verifies our hypothesis that both modalities should be used to guide the prompt selection and the missing of any will cause information lost and lead to sub-optimal performance. In other words, current approaches for unimodal settings do not use all the information we have in multimodal scenarios. We also observe that without the clustering algorithm, the performance of ablative clustering is the lowest among all the settings which indicate the significance of doing cluster training for learning the prompt keys.

#### 5.4 Analytic Experiments

*Effect of clustering.* To show the effect of clustering algorithm, we empirically show the correlation between the clustering error and the downstream accuracy. As we apply Euclidean distance as metric to learn the clusters, we record the average distance between each point to its assigned cluster center for every task, and take the average for all the tasks:

$$\mathcal{E} = Avg\left(\sum_{i=1}^N Avg\left(\sum_{j=1}^M \|x_j - c_k\|_2\right)\right) \quad (10)$$

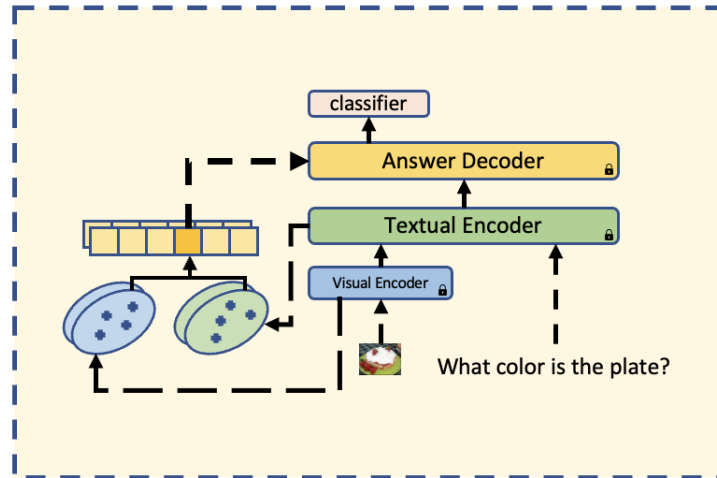


Fig. 5. CluMo-BLIP architecture

Table 4. Comparison experiment on CLOVE-scene dataset with **BLIP** as backbone.

| Method       | CLOVE-scene  |             |              |             |              |             |              |             |
|--------------|--------------|-------------|--------------|-------------|--------------|-------------|--------------|-------------|
|              | abcdef       |             | dbafec       |             | acbefd       |             | caefdb       |             |
|              | A ↑          | F ↓         | A ↑          | F ↓         | A ↑          | F ↓         | A ↑          | F ↓         |
| Finetune     | 40.78        | 22.23       | 41.59        | 23.28       | 38.32        | 25.54       | 40.80        | 20.21       |
| EWC          | 42.08        | 17.23       | 42.44        | 20.7        | 40.86        | 18.69       | 41.02        | 19.6        |
| LwF          | 41.32        | 19.8        | 42.07        | 19.25       | 42.35        | 17.88       | 41.67        | 18.20       |
| ER           | 44.43        | 7.90        | 44.72        | 12.84       | 42.82        | 13.15       | 43.87        | 11.83       |
| GEM          | 45.34        | 9.78        | 46.01        | 9.39        | 43.11        | 11.49       | 44.23        | 10.04       |
| L2P          | 44.37        | 10.67       | 43.15        | 11.69       | 44.64        | 9.58        | 44.31        | 11.04       |
| DualPrompt   | 45.91        | 5.27        | 45.72        | 8.08        | 46.00        | 6.92        | 46.35        | 7.88        |
| S-Prompt     | 46.94        | <b>4.19</b> | 47.36        | <b>4.08</b> | 46.12        | <b>5.34</b> | 47.17        | <b>3.52</b> |
| <b>CluMo</b> | <b>47.60</b> | 12.32       | <b>48.62</b> | 10.98       | <b>47.86</b> | 12.67       | <b>48.85</b> | 9.13        |

where  $i$  represent the number of tasks,  $j$  represent the training data from task  $i$  and  $k$  is the  $k^{th}$  cluster center. We consider both the visual prompt key training and the textual prompt key training in this experiment. Table 5 presents the results. Same as our expectation, we observe a negative correlation between the clustering error and the performance accuracy, in another word, lower  $\mathcal{E}$  for image and text prompt keys leads to a higher accuracy. Without the clustering component, we observe  $\mathcal{E}$  to be as high as 42.38 and 42.8 for image prompt key and text prompt key, respectively. After applying clustering algorithm,  $\mathcal{E}$  drops below 20, which can be considered significant, for both modalities and the accuracy improves 2.97%.

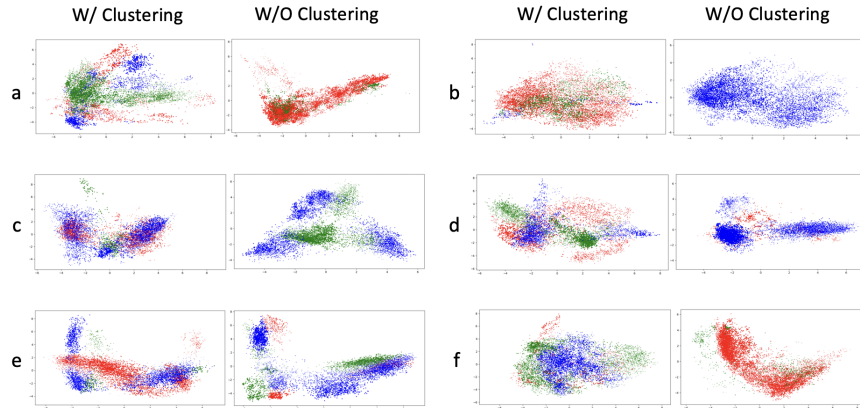


Fig. 6. Cluster distribution on all training image data of **CLOVE-Scene**'s six sub-tasks before and after applying mini-batch  $k$ -means clustering algorithm with visual key size of 3 using PCA. **Color of more diversity indicates more even distribution of key selection.**

*Additional Backbone.* In comparative experiments, we adopt **ALBEF** [21] as our back bone, leveraging its architecture for a consistent foundation across our experiments. While only testing the baselines with a single backbone transformer might be biased and to further show **CluMo**'s generalizability to other VLMs, we evaluate **CluMo**'s performance on **BLIP**, a different transformer architecture than **ALBEF**, [20] to demonstrate its generalizability of more architectures. Without multimodal encoder that takes in both visual and textual feature, architecture such as **BLIP** send visual feature into text encoder and fuse it with cross-attention block. As the architecture is different, we present the adoption of **CluMo** to **BLIP** in Figure 5. Similar to the comparative experiments in previous sections, we choose **CLOVE-scene** dataset as benchmark and randomly select 4 different task orders to perform experiments. The results are presented in Table 4. As **BLIP** is more recent and SOTA model than **ALBEF**, we observe that the overall performance of **BLIP**'s experiments get higher accuracy than those of **ALBEF**, except for prompt-based methods. During experiments, we found that with **ALBEF**, attaching prompts to input features can naturally improve the accuracy of given tasks. In the case of **BLIP**, attaching the same prompt will harm the accuracy, which makes the performance of **BLIP**'s prompt-based methods similar to **ALBEF**'s results. However, even though we observe a deficiency, in some extent, of prompt-based to **BLIP**, **CluMo** still obtains the best accuracy and forgetting rate compared with all the baselines.

*Cluster Visualization.* To show the effect of clustering on prompt key more intuitively, we visualize the visual prompt key selection distribution and textual prompt key selection distribution on the visual and textual portion of the training data for **CLOVE-Scene** in Figure 6 and Figure 7. Since we use three visual prompt keys for each task, the vector features of visual data are split into three groups, which are the green, blue and red points in Figure 6. We observe that without using clustering, visual data are more likely to overlap on the same cluster center which means they would lead to select the same visual prompt key. After performing clustering, we observe that the distribution becomes more evenly, and every cluster of data is diverse and separated from the others which means that the visual data can be separated explicitly. The diversity of prompt key selection indicates each input image can find the "correct" prompt which is semantically closer to it, which contains more specific knowledge about the sub-domain the given image belongs to. The visualization of textual prompt keys indicates similar observation.

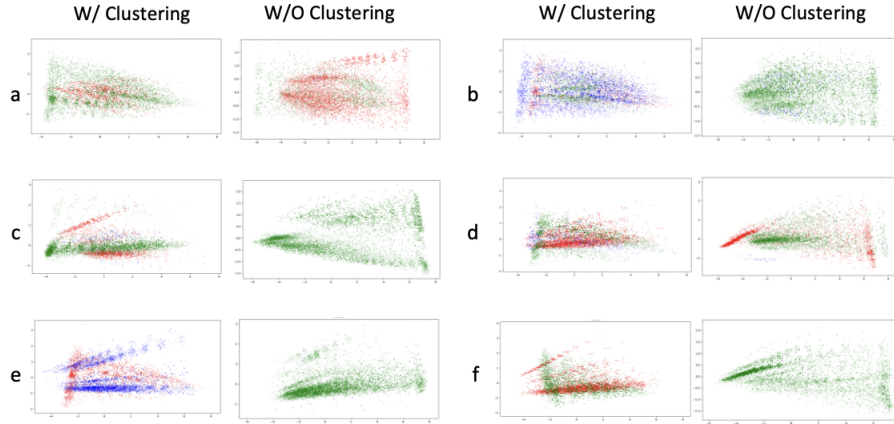


Fig. 7. Cluster distribution on all training text data of **CLOVE-Scene**'s six sub-tasks before and after applying mini-batch  $k$ -means clustering algorithm with textual key size of 3 using PCA.

Table 5. Acc. with different clustering error

| $\mathcal{E}$ . Image | $\mathcal{E}$ . Text | Accuracy |
|-----------------------|----------------------|----------|
| 15.40                 | 10.72                | 48.23    |
| 15.74                 | 12.22                | 47.89    |
| 17.21                 | 12.53                | 47.21    |
| 42.38                 | 42.8                 | 47.06    |

Table 7. Acc. with length/number trade-off

| $S_o \times S_t \times L_p$ | Accuracy |
|-----------------------------|----------|
| $2 \times 2 \times 22$      | 47.82    |
| $3 \times 3 \times 10$      | 48.23    |
| $4 \times 4 \times 6$       | 47.45    |

Table 6. Acc. with different prompt key size

| $S_{img} \times S_{txt}$ | Accuracy |
|--------------------------|----------|
| $1 \times 1$             | 46.01    |
| $2 \times 2$             | 47.49    |
| $3 \times 3$             | 48.23    |
| $4 \times 4$             | 47.71    |
| $5 \times 5$             | 47.45    |
| $10 \times 10$           | 48.12    |

Table 8. Acc. with different freezing layer

| Freeze Num. | Accuracy |
|-------------|----------|
| 6           | 48.23    |
| 5           | 44.66    |
| 3           | 47.46    |
| 0           | 37.48    |

*Tracking the Accuracy for the First Task.* To take a closer look into the effect on preventing catastrophic forgetting and increasing the accuracy in CL, we track the accuracy of the first task while learning the task sequence. The result is shown in Figure 4. We see that the accuracy drops until task 4, and then slightly increases until task 6. Although it is not our main focus, this behavior shows a trend of forward transfer between the tasks. Among all the baseline methods, we notice that prompt-based methods, **SPrompt**, **DualPrompt** and **L2P**, significantly outperform other methods which verifies the SOTA status of prompt learning in CL and its success in preventing catastrophic forgetting. Our method **CluMo**, on the other hand, still outperform all prompt-based

baseline methods. We observe that using the cluster-based prompts, the accuracy on the first task is superior compared to the other methods at the very beginning. Similar to other prompt-based method, our method's accuracy slightly drops until task 4 and improves subsequently. As the accuracy of our proposed method is higher than others at all time steps, our method has the leading performance in terms of both accuracy and backward transfer.

*Effect of Frozen Decoder Layers.* Although previous prompt-based methods keep all the transformer layers frozen during the training, we still conduct experiment on freezing different number of transformer layers to check effect of it. The results are presented in Table 8. We observe keeping whole transformer layers frozen achieve the best performance, and unfreeze all gets the worst. However, freezing 5 layers get worse result than freezing 3 layers. Our hypothesis is that freezing 5 layers makes the model vulnerable to parameter shift yet not flexible enough to learn new knowledge through the trainable transformer layer and only freezing 3 layers makes the model capable for adopting new knowledge with three unfrozen layers. Overall, freezing all the transformer layers achieves the best accuracy and the result matches the choice of freezing model for **L2P**, **DualPrompt** and **SPrompt**.

*Effect of Prompt Key Size.* We also conduct an experiment to study the effect of prompt pool size to show the stability of our method with respect to this hyper-parameter. In Table 6, we choose different visual prompt key and textual prompt key sizes  $1 \times 1$ ,  $2 \times 2$ ,  $3 \times 3$ ,  $4 \times 4$ ,  $5 \times 5$ ,  $10 \times 10$ , corresponding to 1, 4, 9, 16, 25, and 100 prompt pool sizes. When we have prompt key size of 1, which indicates every task has no choice but only one prompt, the accuracy drops significantly. Other than prompt size 1, we find that choosing prompt size 3 has the best result, 47.23. We also observe that having larger prompt size than 3 results in decreasing of performance. This observation possibly indicate that dividing a single task into four sub-domains is specific enough for prompts to learn fine-grained sub-domain information, as the key size continues increasing, the more-detailed information the prompts learn won't boost the performance significantly.

*Prompt Length and Prompt Size.* In the experiment setting, we utilize 3 visual keys and 3 textual keys, and each prompt has length of 10. In sum, we have  $3 \times 3 \times 10 = 90$  total prompt length for each task. We conduct experiments to show the trade-off of prompt length and prompt numbers given the fixed total prompt length. To make a fair comparison, we keep the visual key and textual key the same size, and choose the closest integer prompt length to let the multiplication of the three be around 90. Within table 7, among the three different combination, we find that the current setting, which is the most balanced one, obtains the best result. Regarding the rest of the two, the higher prompt length has the better performance.

*Inference Latency.* Our proposed approach utilizes multi-modal prompts during both training and testing stage for efficient, which introduces extra computation, by matching the text prompt keys and image prompt keys, during inference. On a single Nvidia-A40 GPU, we find that the inference time of **CluMo** for a single test sample is 256 milliseconds (ms), while that for non-prompt-based method, such as fine-tuning, is 228 milliseconds. Since there's only additional 28 milliseconds for selecting prompts during inference, we consider our method applicable in real-time scenario.

## 6 Conclusion

We introduced a novel prompt-based continual learning method for learning multimodal tasks. While most existing methods apply simple prompts on a single modality, our method proposes modal-specific visual prompt keys and textual prompt keys, and trains them to capture the semantic properties of the training dataset using the  $K$ -means clustering algorithm. We use the combination of both the visual prompt key and the textual prompt key to select prompts, which enables the prompt to better boost the performance. Our experiments show that our

method achieves state-of-the-art performance in continual VQA tasks across different domains, outperforming regularization-based, rehearsal-based, and existing prompt-based continual learning methods. Furthermore, we note that in highly sensitive applications, such as medical diagnostics, autonomous driving, or legal reasoning, forgetting previously acquired knowledge or introducing prediction errors can lead to serious consequences. Therefore, improving the stability and reliability of multimodal continual learning systems is not only a matter of performance but also one of safety and trustworthiness. Our approach contributes to the forgetting in the general domain, we also call for the exploration of preventing catastrophic forgetting on application-specific domains to tackle realistic trustworthy issues.

## References

- [1] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Batra, and D. Parikh. 2016. Vqa: visual question answering. (2016). arXiv: [1505.00468](https://arxiv.org/abs/1505.00468) [cs.CL].
- [2] J. Ainslie, J. Lee-Thorp, M. de Jong, Y. Zemlyanskiy, F. Lebrón, and S. Sanghai. 2023. Gqa: training generalized multi-query transformer models from multi-head checkpoints. (2023). <https://arxiv.org/abs/2305.13245> arXiv: 2305.13245 [cs.CL].
- [3] Y. Cai and M. Rostami. 2024. Dynamic transformer architecture for continual learning of multimodal tasks. (2024).
- [4] Y. Cai, J. Thomason, and M. Rostami. 2023. Task-attentive transformer architecture for continual learning of vision-and-language tasks using knowledge distillation. (2023). arXiv: [2303.14423](https://arxiv.org/abs/2303.14423) [cs.LG].
- [5] R. Cohn and E. Holm. 2021. Unsupervised machine learning via transfer learning and k-means clustering to classify materials image data. *Integrating Materials and Manufacturing Innovation*, 10, 2, 231–244.
- [6] W. Cong, Y. Cong, G. Sun, Y. Liu, and J. Dong. 2023. Self-paced weight consolidation for continual learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 34, 4, 2209–2222.
- [7] A. Douillard, A. Ramé, G. Couairon, and M. Cord. 2022. Dytox: transformers for continual learning with dynamic token expansion. (2022). arXiv: [2111.11326](https://arxiv.org/abs/2111.11326) [cs.CV].
- [8] B. Ermiš, G. Zappella, M. Wistuba, A. Rawal, and C. Archambeau. 2022. Memory efficient continual learning with transformers. *Advances in Neural Information Processing Systems*, 35, 10629–10642.
- [9] R. M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3, 4, 128–135.
- [10] H. Gao, Z. Zhang, Y. Wei, L. Zhao, G. Li, Y. Li, L. Kong, and W. Huang. [n. d.] Enhanced continual learning of vision-language models with model fusion. In *First Workshop on Scalable Optimization for Efficient and Adaptive Foundation Models*.
- [11] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. 2017. Making the V in VQA matter: elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [12] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. 2017. Making the v in vqa matter: elevating the role of image understanding in visual question answering. (2017). arXiv: [1612.00837](https://arxiv.org/abs/1612.00837) [cs.CV].
- [13] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. 2021. Lora: low-rank adaptation of large language models. (2021). arXiv: [2106.09685](https://arxiv.org/abs/2106.09685) [cs.CL].
- [14] W. Kim, B. Son, and I. Kim. 2021. Vilt: vision-and-language transformer without convolution or region supervision. (2021). arXiv: [2102.03334](https://arxiv.org/abs/2102.03334) [stat.ML].
- [15] J. Kirkpatrick et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114, 13, (Mar. 2017), 3521–3526. doi: [10.1073/pnas.1611835114](https://doi.org/10.1073/pnas.1611835114).
- [16] C. Lamers, R. Vidal, N. Belbachir, N. van Stein, T. Bäeck, and P. Giampouras. 2023. Clustering-based domain-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3384–3392.
- [17] M. Lao, N. Pu, Y. Liu, Z. Zhong, E. M. Bakker, N. Sebe, and M. S. Lew. 2023. Multi-domain lifelong visual question answering via self-critical distillation. In *Proceedings of the 31st ACM International Conference on Multimedia*, 4747–4758.
- [18] S. W. Lei, D. Gao, J. Z. Wu, Y. Wang, W. Liu, M. Zhang, and M. Z. Shou. 2022. Symbolic replay: scene graph as prompt for continual learning on vqa task. (2022). arXiv: [2208.12037](https://arxiv.org/abs/2208.12037) [cs.CV].
- [19] B. Lester, R. Al-Rfou, and N. Constant. 2021. The power of scale for parameter-efficient prompt tuning. (2021). arXiv: [2104.08691](https://arxiv.org/abs/2104.08691) [cs.CL].
- [20] J. Li, D. Li, C. Xiong, and S. Hoi. 2022. Blip: bootstrapping language-image pre-training for unified vision-language understanding and generation. (2022). <https://arxiv.org/abs/2201.12086> arXiv: 2201.12086 [cs.CV].
- [21] J. Li, R. R. Selvaraju, A. D. Gotmare, S. Joty, C. Xiong, and S. Hoi. 2021. Align before fuse: vision and language representation learning with momentum distillation. (2021). arXiv: [2107.07651](https://arxiv.org/abs/2107.07651) [cs.CV].
- [22] L. Li, M. Pagnucco, and Y. Song. 2022. Graph-based spatial transformer with memory replay for multi-future pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2231–2241.

- [23] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang. 2019. Visualbert: a simple and performant baseline for vision and language. (2019). arXiv: 1908.03557 [cs.CV].
- [24] X. L. Li and P. Liang. 2021. Prefix-tuning: optimizing continuous prompts for generation. (2021). arXiv: 2101.00190 [cs.CL].
- [25] Z. Li and D. Hoiem. 2017. Learning without forgetting. (2017). arXiv: 1606.09282 [cs.CV].
- [26] H. Liu, C. Li, Q. Wu, and Y. J. Lee. 2023. Visual instruction tuning. (2023). <https://arxiv.org/abs/2304.08485> arXiv: 2304.08485 [cs.CV].
- [27] R. Liu and B. Mozafari. 2022. Transformer with memory replay. In *Proceedings of the AAAI Conference on Artificial Intelligence* number 7. Vol. 36, 7567–7575.
- [28] D. Lopez-Paz and M. Ranzato. 2022. Gradient episodic memory for continual learning. (2022). arXiv: 1706.08840 [cs.LG].
- [29] C. Lu, P. Ball, Y. W. Teh, and J. Parker-Holder. 2023. Synthetic experience replay. *Advances in Neural Information Processing Systems*, 36, 46323–46344.
- [30] J. Lu, D. Batra, D. Parikh, and S. Lee. 2019. Vilbert: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. (2019). arXiv: 1908.02265 [cs.CV].
- [31] J. Lu, D. Batra, D. Parikh, and S. Lee. 2019. Vilbert: pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. (2019). <https://arxiv.org/abs/1908.02265> arXiv: 1908.02265 [cs.CV].
- [32] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi. 2019. Ok-vqa: a visual question answering benchmark requiring external knowledge. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [33] M. Mirtaheeri, M. Rostami, and A. Galstyan. 2023. History repeats: overcoming catastrophic forgetting for event-centric temporal knowledge graph completion. In *Findings of the Association for Computational Linguistics: ACL 2023*, 7740–7755.
- [34] M. Nikandrou, G. Pantazopoulos, I. Konstas, and A. Suglia. 2024. Enhancing continual learning in visual question answering with modality-aware feature distillation. In *Proceedings of the 3rd Workshop on Advances in Language and Vision Research (ALVR)*, 73–85.
- [35] M. Nikandrou, G. Pantazopoulos, I. Konstas, and A. Suglia. 2024. Enhancing continual learning in visual question answering with modality-aware feature distillation. (2024). <https://arxiv.org/abs/2406.19297> arXiv: 2406.19297 [cs.CV].
- [36] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel. 2019. Language models as knowledge bases? (2019). <https://arxiv.org/abs/1909.01066> arXiv: 1909.01066 [cs.CL].
- [37] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych. 2021. Adapterfusion: non-destructive task composition for transfer learning. (2021). arXiv: 2005.00247 [cs.CL].
- [38] Z. Qian, X. Wang, X. Duan, P. Qin, Y. Li, and W. Zhu. 2023. Decouple before interact: multi-modal prompt learning for continual visual question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. (Oct. 2023), 2953–2962.
- [39] Z. Qian, X. Wang, X. Duan, P. Qin, Y. Li, and W. Zhu. 2023. Decouple before interact: multi-modal prompt learning for continual visual question answering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2953–2962.
- [40] A. Radford et al. 2021. Learning transferable visual models from natural language supervision. (2021). arXiv: 2103.00020 [cs.CV].
- [41] A. Radford et al. 2021. Learning transferable visual models from natural language supervision. (2021). <https://arxiv.org/abs/2103.00020> arXiv: 2103.00020 [cs.CV].
- [42] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. 2017. Icarl: incremental classifier and representation learning. (2017). arXiv: 1611.07725 [cs.CV].
- [43] M. Ren, R. Kiros, and R. Zemel. 2015. Exploring models and data for image question answering. (2015). <https://arxiv.org/abs/1505.02074> arXiv: 1505.02074 [cs.LG].
- [44] D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap, and G. Wayne. 2019. Experience replay for continual learning. (2019). arXiv: 1811.11682 [cs.LG].
- [45] M. Rostami. 2021. Lifelong domain adaptation via consolidated internal distribution. *Advances in neural information processing systems*, 34, 11172–11183.
- [46] M. Rostami, S. Kolouri, P. Pilly, and J. McClelland. 2020. Generative continual concept learning. In *Proceedings of the AAAI conference on artificial intelligence* number 04. Vol. 34, 5545–5552.
- [47] M. Rostami, S. Kolouri, and P. K. Pilly. 2019. Complementary learning for overcoming catastrophic forgetting using experience replay. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 3339–3345.
- [48] Z. Shanguan, D. Seita, and M. Rostami. 2025. Cross-domain multi-modal few-shot object detection via rich text. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 6570–6580.
- [49] H. Shin, J. K. Lee, J. Kim, and J. Kim. 2017. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30.
- [50] A. Singh, V. Natarajan, M. Shah, Y. Jiang, X. Chen, D. Batra, D. Parikh, and M. Rohrbach. 2019. Towards vqa models that can read. (2019). <https://arxiv.org/abs/1904.08920> arXiv: 1904.08920 [cs.CL].
- [51] J. S. Smith, L. Valkov, S. Halbe, V. Gutta, R. Feris, Z. Kira, and L. Karlinsky. 2024. Adaptive memory replay for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3605–3615.
- [52] T. Srinivasan, T.-Y. Chang, L. Pinto Alva, G. Chochlakis, M. Rostami, and J. Thomason. 2022. Climb: a continual learning benchmark for vision-and-language tasks. *Advances in Neural Information Processing Systems*, 35, 29440–29453.

- [53] I. Stoianov, D. Maisto, and G. Pezzulo. 2022. The hippocampal formation as a hierarchical generative model supporting generative replay and continual learning. *Progress in Neurobiology*, 217, 102329.
- [54] L. Su et al. 2021. Gem: a general evaluation benchmark for multimodal tasks. (2021). arXiv: 2106.09889 [cs.CL].
- [55] Y. Wang, Z. Huang, and X. Hong. 2023. S-prompts learning with pre-trained transformers: an occam's razor for domain incremental learning. (2023). arXiv: 2207.12819 [cs.CV].
- [56] Z. Wang, L. Liu, Y. Duan, Y. Kong, and D. Tao. 2022. Continual learning with lifelong vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 171–181.
- [57] Z. Wang et al. 2022. Dualprompt: complementary prompting for rehearsal-free continual learning. (2022). arXiv: 2204.04799 [cs.LG].
- [58] Z. Wang et al. 2022. Learning to prompt for continual learning. (2022). arXiv: 2112.08654 [cs.LG].
- [59] J. Wei, M. Bosma, V. Y. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. 2021. Finetuned language models are zero-shot learners. (2021).
- [60] S. Yan, L. Hong, H. Xu, J. Han, T. Tuytelaars, Z. Li, and X. He. 2022. Generative negative text replay for continual vision-language pretraining. In *European Conference on Computer Vision*. Springer, 22–38.
- [61] X. Zhang, F. Zhang, and C. Xu. 2023. Vqacl: a novel visual question answering continual learning setting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 19102–19112.
- [62] Z. Zheng, M. Ma, K. Wang, Z. Qin, X. Yue, and Y. You. 2023. Preventing zero-shot transfer degradation in continual learning of vision-language models. In *Proceedings of the IEEE/CVF international conference on computer vision*, 19125–19136.
- [63] K. Zhou, J. Yang, C. C. Loy, and Z. Liu. 2022. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130, 9, (July 2022), 2337–2348. doi: 10.1007/s11263-022-01653-1.

Table 9. size of each task in CLOVE

| task                 | training size | testing size | image source |
|----------------------|---------------|--------------|--------------|
| ShopAndDining        | 20K           | 3K           | MS-COCO      |
| WorkPlace            | 20K           | 3K           | MS-COCO      |
| HomeOrHotel          | 20K           | 3K           | MS-COCO      |
| Transportation       | 20K           | 3K           | MS-COCO      |
| SportAndLeisure      | 20K           | 3K           | MS-COCO      |
| Outdoors             | 20K           | 3K           | MS-COCO      |
| ObjectRecognition    | 20K           | 3K           | MS-COCO      |
| AttributeRecognition | 20K           | 3K           | MS-COCO      |
| RelationReasoning    | 20K           | 3K           | MS-COCO      |
| LogicReasoning       | 20K           | 3K           | MS-COCO      |
| KnowledgeReasoning   | 20K           | 3K           | MS-COCO      |
| SceneTextRecognition | 16.8K         | 2.4K         | VG           |

## A Descriptions of Baseline Methods

We provide a detailed description of the functionality and underlying principles of each baseline method used in our study. This thorough explanation aims to offer readers a more comprehensive understanding of how each method operates, the rationale behind their design, and their specific roles in the comparison experiments. By doing so, we enable a clearer and more informed comparison of the experimental results, highlighting each approach in the context of our research objectives.

*EWC.* works by slowing down the changes to the important parameters for previous tasks when training on new tasks. The importance of each parameter is calculated through **Fisher Information Matrix**, the larger the fisher value, the more critical the parameter is. It achieves this by adding a regularization term to the loss function that penalizes large updates to these critical weights.

*LwF.* works by preserving knowledge from previous tasks using a knowledge distillation approach, where “soft targets” from the previous task act as a regularizer when learning new tasks. While training on new task, LwF freezes a copy of model trained on previous tasks, serving as **teacher model**, and apply a “soft prediction” using teacher model as an additional regularizer to prevent forgetting.

*ER.* works by maintaining a memory buffer to store a limited number of samples or intermediate features from previously learned tasks. These stored samples are then reused when learning new tasks with specific frequency. The buffer usually has a fixed size, and new experiences replace older ones once the buffer is full.

*GEM.* works by using a memory buffer, named “episodic memory” to store a subset of examples from previously learned tasks. During training, it computes gradients for both the current task data and the stored examples and ensures that the gradient update for the current task does not increase the loss on the examples from the previous tasks to prevent the catastrophic forgetting.

*L2P.* works by maintaining a shared uni-modal pool of prompts across all the tasks that are used to condition the model when learning new tasks. These prompts act as a kind of “memory” that encodes relevant information from previous tasks and domains. Each task is assigned one or more prompts from the pool, and these prompts are selected based on their relevance to the new task at hand.

### Shopping and Dining



Q: Is there a sandwich on the tray?  
A: No.



Q: Is the bottle on the couch?  
A: Yes.

### Workplace

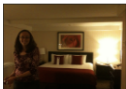


Q: Which room is it?  
A: Office.



Q: Is a laptop to the left of her?  
A: Yes.

### Hotel



Q: Does the blanket look red?  
A: Yes.



Q: What toy on the top of sink?  
A: Rubber Duck.

### Transportation



Q: What is the plane behind the man?  
A: Runway.



Q: What are the letters on?  
A: Stairs.

### SportAndLeisure



Q: Is the man on the right of frisbee wearing a hat?  
A: Yes.



Q: What is the man playing?  
A: Frisbee.

### Outdoors



Q: Does the zebra nose have the white color?  
A: No.



Q: What is flying in the sky?  
A: Kite.

Fig. 8. CLOVE-scene dataset samples

*DualPrompt*. works by maintaining two kind of uni-modal prompts, **Expert Prompts** (task-specific prompt), designed to handle knowledge that is unique to specific tasks, and **General Prompts**, designed to handle general knowledge that can be transferred across multiple tasks. During training, the model dynamically selects **Expert Prompts** and **General Prompts** to help process new tasks while maintaining performance on previously learned tasks to balance both task performance and overall generalization.

*S-Prompt*. works by maintaining an uni-modal prompt pool across different domain of tasks. Within the prompt pool, the prompts from same domain are aligned closer in the feature space with K-mean clustering algorithm. While new task comes in, new set of prompts are added into the prompt and trained to be separated from other

**Attribute Recognition**



Q: What color is the helmet in the middle of the image?  
A: Blue.



Q: Is it indoor or outdoor?  
A: Yes.

**Knowledge Reasoning**



Q: Which red object behind the black piano can keep light out of the room?  
A: Curtain.



Q: What food is next to the object that I can use to for making toast?  
A: Bread.

**Logic Reasoning**



Q: What do the marker and the post have in common?  
A: Color.



Q: Is the color of pillow different than that of counter?  
A: Yes.

**Object Recognition**



Q: What place is it?  
A: Harbor.



Q: What is this, a couch or a table?  
A: Table.

**Relation Reasoning**



Q: What color is the jersey the boy is wearing?  
A: Black.



Q: Who is wearing goggles?  
A: Woman.

**Scene Text Recognition**



Q: What is the brand of this camera?  
A: Dakota.



Q: What is the brand of the phone?  
A: Nokia.

Fig. 9. CLOVE-function dataset samples

prompts via K-mean clustering. During inference stage, the prompts with best alignment with input visual feature is dynamically selected.

**B CLOVE Dataset Detail Description**

In the **CLOVE-Scene** and **CLOVE-Function** datasets, all tasks have a uniform distribution of training and testing data, with the exception of the *SceneTextRecognition* task, which comprises 16.8K training samples and 2.4K testing samples. The remaining tasks within these datasets contain 20K training samples and 3K testing samples each. It is reflected in the Table 7.

To provide a deeper understanding of the **CLOVE** dataset, we offer additional details here. We have included visualizations in Figure 8 and Figure 9 to showcase two sample images from each task within the datasets,



Fig. 10. Examples of images from CLOVE-scene Workplace subtask assigned to each of three visual prompt keys.

emphasizing the distinctiveness of each domain. From these samples, it is evident that the images in the **CLOVE-Scene** dataset vary significantly across tasks, even though the questions associated with them are similar in structure, differing primarily based on the content depicted in the images.

On the other hand, the **CLOVE-Function** dataset presents a different scenario. While the images across various tasks may appear to belong to similar or overlapping domains, making it challenging to distinguish one task from another based solely on visual content, the diversity becomes apparent when considering the questions. Each task within the **CLOVE-Function** dataset involves distinct types of reasoning, as reflected in the varied nature of the questions posed, which are tailored to serve different reasoning purposes.

### C Visual&Textual Prompt Key Cluster Visualization

As introduced in Sec 4.3, we first fine-tune the visual prompt keys and the textual prompt keys via K-means clustering algorithm such that prompt keys are equivalent to clustering centers. Although the clusters produced by K-means are not necessarily aligned with human-interpretable semantic categories, we attempt to provide a human-readable interpretation of the clustering results based on observable patterns. We visualize several example images, in Figure 10, and questions, in Table 10, from **CLOVE-scene Workplace** sub-task that assigned to the same prompt keys, visual prompt keys and textual prompt keys, which the prompt key size for each modalities is 3.

From Figure 10, we observe that the images from visual prompt key 2 are generally human-absent images, which is their main difference with the others. However, the gap between the images from visual prompt key 0 and 1 are much subtler. Even though the similarity, we still find that the human activities in images from prompt key 0 are relatively more casual and diverse than those from prompt key 1, where images are more focusing on formal works with laptops. In Table 10, we first observe that the questions aligned to textual prompt key 2 are relatively shorter than those aligned with prompt key 0 and 1, making them unique among all the questions. On the other hand, the questions aligned to textual prompt key 1 are more related to keyword “*what*” and “*device*”, while the questions of textual prompt key 0 are more diverse and random.

Table 10. Examples of questions from CLOVE-scene WorkPlace subtask assigned to each of three textual prompt keys.

|                      |   |
|----------------------|---|
| Textual Prompt Key 0 | <p>Is the computer monitor to the left of a cd?<br/>         Which kind of device is to the right of the comforter?<br/>         Which part of the image is the white cup in the top or the bottom?<br/>         Which kind of device is to the left of the computer mouse?<br/>         What kind of furniture is to the right of the glasses?<br/>         Which kind of device is to the right of the laptop?<br/>         What kind of appliance is to the right of the picture?<br/>         What device is to the left of the headphones?<br/>         What is the food that is to the right of the envelope?<br/>         What item of furniture is to the left of the television?</p>   |
| Textual Prompt Key 1 | <p>What is the device to the left of the lamp made of metal?<br/>         What kind of device is to the left of the pot?<br/>         What device is behind the girl that is sitting at the table?<br/>         What kind of device is to the right of the man who is wearing a ring?<br/>         What devices are to the left of the monitor that is on the desk?<br/>         What is the device to the right of the device on the left side?<br/>         What is the name of the appliance to the right of the box that looks white?<br/>         What type of room is it a market or an office?<br/>         What kind of device has the same color as the cup that is to the left of the woman?<br/>         What room is it a hotel or maybe an office?</p> |
| Textual Prompt Key 2 | <p>Which side is the black device on?<br/>         What animal is on the bed?<br/>         What is the device that is on the desk?<br/>         Which device is gray?<br/>         Which side is the black phone on?<br/>         What is in the jar?<br/>         Is that a bed or a desk?<br/>         Is the screen on the right?<br/>         What is the girl using?<br/>         That guy is inside what?</p>   |

Received 23 December 2024; accepted 06 June 2025