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

Anonymous EMNLP submission

Abstract

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 limited because finetuning a VLM on a task normally leads to reducing its generalization power and the capacity of learning new tasks. Enabling using VLMs in multimodal continual learning (CL) settings can help to address such scenarios. Hence, we propose a novel prompt-based CL method for VLMs, namely **Clu**ster-based **Mo**dality Fusion Prompt (**CluMo**). Our approach addresses catastrophic forgetting through constructing modality-specific prompts using k-means clustering for selecting the best semantically matched prompt, which also enables benefiting from past experiences through forward transfer. Experiments on two benchmarks demonstrate that our method achieves SOTA against existing alternatives.

1 Introduction

Visual Question Answering (VQA) is a complicated task, where the goal is to answer questions descibed in text based on a given image. Addressing VQA requires understanding and fusion of information from both the visual and textual domains to generate accurate responses. Recently, significant advancements in addressing VQA tasks have emerged due to the development of pre-trained large vision-langue models (VLMs) (Radford et al., 2021; Kim et al., 2021). Despite these advances, one of the persistent challenges in VQA tasks is the ability to adapt a VLM in CL setting to learn new tasks and continuously improve without forgetting previously learned knowledge, also known as catastrophic forgetting (French, 1999). To address catastrophic forgetting, a group of CL algorithms are deployed. Regularization-baesd methods (Kirkpatrick et al., 2017; Li and Hoiem, 2017) constrain the drastic parameter shift when learning new tasks. Expansion-based methods (Douillard et al., 2022; Cai et al., 2023) expand the model with small portion of additional weights and use the expanded weights to learn the new incoming tasks. Rehearsal-based methods (Rebuffi et al., 2017; Rolnick et al., 2019) store a representative subset of training dataset for each tasks into a small memory buffer and replay them back during the learning of the current task to maintain the encoded knowledge of previous tasks. More recently, prompt-based methods (Wang et al., 2022b,a) aim to use prompts that contains task-specific or semantic-specific information which are attached to the embedded features of the input to prevent catastrophic forgetting.

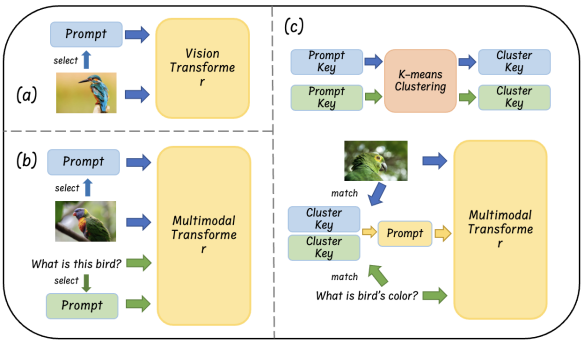

Figure 1: Comparison between existing prompt-based CL methods and our proposed method: **(a)** Uni-modal based methods use image feature to select prompts from a prompt pool. **(b)** Multi-modal based methods use image features to select image prompts and use text features to select text prompts. **(c)** 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.

Most existing CL methods consider unimodal, i.e., vision-only and language-only, settings and hence are inapplicable to address VQA tasks. To tackle this shortcoming, we propose a novel two-stage prompt learning-based CL method, namely cluster-based modality fusion prompt (CluMo). Our method adopts a pre-trained VLM as its back-

bone and benefits from a clustering-based modal-specific key strategy to boost forward transfer and minimize catastrophic forgetting. 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 key and textual key. We then use the combination of two modal-specific keys to find the best-matched prompt. 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.
- We use a two-stage training strategy to train the prompt keys before training the whole model to guarantee the optimal prompt selection.
- We offer extensive experiments to demonstrate that the proposed approach achieves SOTA performance against existing methods.

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. Initially, VQA was formulated as a classification task in which answers are selected from a predefined set of answers (Agrawal et al., 2016) and was solved by using CNNs for image feature extraction and RNNs for text processing. These models were too simple to be used in most practical cases. With the development of transformer and BERT-like models (Lu et al., 2019; Li et al., 2019), performance in VQA tasks has significantly been improved due to the better capacity of capturing the intricate relationship between two modalities. Despite these advances, VQA tasks are mostly studied in static environments (Goyal et al., 2017; Johnson et al., 2016; Marino et al., 2019) which makes existing methods inapplicable in dynamic environments and settings such as continual learning (CL).

Prompt-Based Learning Prompt learning is a powerful technique for leveraging pre-trained language models to frame downstream tasks in NLP. It is more memory-efficient than using Adapters (Pfeiffer et al., 2021) or LoRA (Hu et al., 2021) and has been used successfully to guide responses of VLMs for a particular task. Browon et al. 2020 introduced the concept of prompt for the natural language instruction task to guide the model towards desired outputs. Prompt learning is based on providing a fixed function to condition a model so that it gets extra information token which specializes it to perform the down-stream task. Prompts are mostly considered as trainable parameters, task-specific or domain-specific, to guide the model by obtaining task-specific knowledge (Lester et al., 2021; Li and Liang, 2021).

Prompt Learning for Continual Learning 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. L2P (Wang et al., 2022b) 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 (Wang et al., 2022a) extended the idea of prompt pool in l2p by introducing E-prompt and G-prompt. While E-prompt is task-specific, G-prompt encodes the knowledge used for all tasks to further allow knowledge sharing and transferring while mitigating negative transfer. S-Prompt (Wang et al., 2023) applied clustering to build the prompt pool with domain-specific prompts. These prompt learning methods for CL only consider single-modality, i.e., vision-only or text-only, and hence are sub-optimal for tasks with multi-modal inputs such VQA. Our method benefits from the specific properties of multi-modal data to address VQA in CL settings using prompt learning.

3 Problem Description

Consider a set of VQA tasks, $\{\mathcal{T}_i\}_{i=1}^{T}$, which are encountered sequentially and each of them are from different domain. For each of the tasks, a labeled training dataset $\mathcal{D}^i = \{\langle(\boldsymbol{I}_i^j, \boldsymbol{L}_i^j)^i, y_i^j\rangle_{j=1}^{N_i}\}$ is accessible, N_i denotes the size of dataset, $\boldsymbol{I}_i^j \in \mathbb{R}^{H\times W\times C}$ denotes the input image, $\boldsymbol{L}_i^j \in \mathbb{R}^{L\times|V|}$ 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 train-

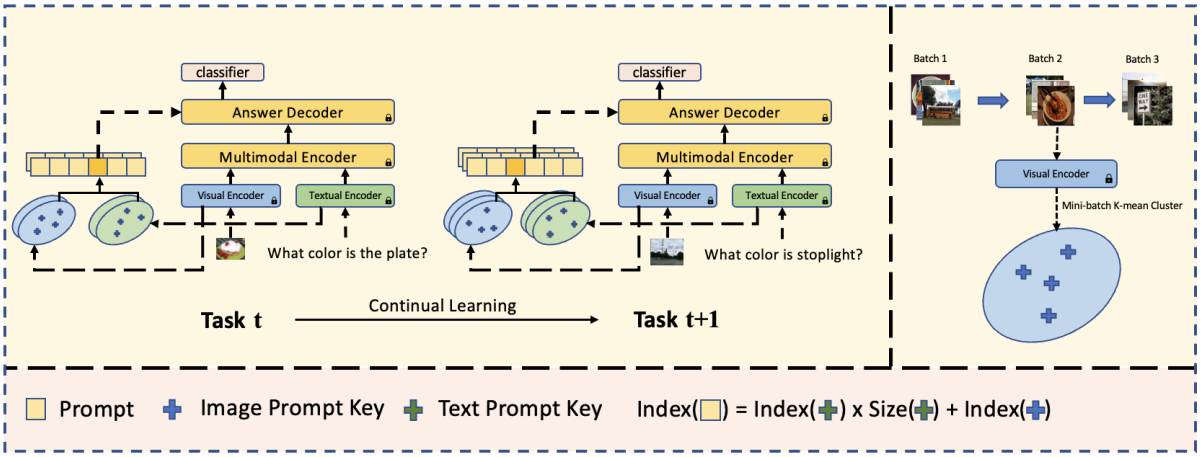

Figure 2: Block diagram of the proposed approach: **Left**: the backbone contains a pre-trained frozen visual encoder, a textual encoder, and a multimodal encoder. The answer decoder shares the same architecture as multimodal encoder. During the training phase, a visual prompt key, a textual prompt key, 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.

ing 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, i.e., by preventing catastrophic forgetting.

We formulate our problem in a **domain-incremental** learning setting (Van de Ven and Tolias, 2019) which assume each tasks is from different domains and the boundaries between them are known during learning time. We consider that each task can be learned individually by adapting a pre-trained large multimodal transformer $f^i_{\theta M}(\cdot,\cdot)$ via minimizing a suitable discrimination loss $\mathcal{L}$, e.g., cross entropy. In our approach, all the model parameters, except the final classifier layer θ_{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 is attached to the model $f^i_{\theta M}(\cdot,\cdot)$ 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 prior approaches (Lopez-Paz and Ranzato, 2022; Rebuffi et al., 2017).

4 Proposed Architecture

Our architecture, named cluster-based modality fusion prompt (**CluMo**), contains two task-specific cluster-based keys for vision and text embeddings and one prompt pool. The combination of the selections from both keys is then used to select the best matched prompt from prompt pool. In this section, we first introduce the 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 *VE*, the textual encoder *TE*, and the multimodal fusion encoder *FE*. Given a visual input $\mathbf{V}$, i.e., a single image, and a textual input $\mathbf{T}$, i.e., a question, the data processing pipeline for the model is:

$$\hat{y}(\mathbf{V},\mathbf{T}) = \mathcal{F}(FE([VE(\mathbf{V});TE(\mathbf{T})])), \quad (1)$$

where $\mathcal{F}(\cdot)$ is the classifier to predict the answer.

Prompt Pool As an adoption of prompt learning in continual learning, a prompt pool is a set of trainable key-value (K-P) pair, in which $K \in R^{1\times D}$ denotes the "prompt key", and $P \in R^{L_p\times D}$ is the

prompt. L_p and D denote the length and dimension of the prompt. Given an input image $\mathbf{V}$, we compute $v_I = VE(\mathbf{V}) \in R^{L_v \times D}$, where L_v is the dimension of the features, after passing the image through the visual encoder. $v_{I_0} = v_I[0]$ is matched with all the keys K within prompt pool via cosine similarity to find the most similar K_i. The corresponding P_i is selected and prepend to $\mathbf{V}$ as $\mathbf{V}' = [P_i; \mathbf{V}]$. Parameters of K and P are updated through back-propagation.

4.2 Modality Fusion Prompt

Previous prompt-based CL methods such as L2P (Wang et al., 2022b) 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. The intuition behind this idea is that feature vectors with small geometric distance in the feature space are semantically similar (Wang et al., 2023).

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 help guiding prompt selection which is more representative of the input 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** K_v, the **textual prompt keys** K_t, and the **prompt pool** P as following:

$$\begin{aligned} K_t &= [K_{t_1}, K_{t_2}, ..., K_{t_{S_t}}], \\ K_v &= [K_{v_1}, K_{v_2}, ..., K_{v_{S_v}}], \\ P &= [P_1, P_2..., P_{S_p}], \\ K_{t_m} &\in R^D, K_{v_n} \in R^D, P_l \in R^{L_p \times D}, \end{aligned} \tag{2}$$

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:

$$(K_{v_m}, K_{t_n}) \rightarrow P_{m*S_v+n}. \tag{3}$$

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.

4.3 Cluster-based Prompt Key

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 semantically cluster center of the sub-domains for both vision and text inputs, we adopt mini-batch K-means clustering algorithm on prompt keys of K_v and 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:

$$v_I = VE(I), v_T = TE(T), \tag{4}$$

where $v_I \in R^{B \times L_I \times D}$ and $v_T \in R^{B \times L_T \times D}$, 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 K_v:

$$similarity(n, m) = ||\hat{v}_{I_n} - K_{v_m}||_2, \tag{5}$$

and the prompt key with highest similarity is assigned to match v_{I_n}. After calculation of the whole batch $\mathcal{B}$, the prompt keys are then updated by calculating the mean of all $\hat{v}_{I_n}$ assigned to the specific image prompt key. The procedure of updating the text prompt key 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 minibatch k-means Cluster on the visual and the textual prompt key K_v and K_t until reaching the

Algorithm 1 Prompt Key Training

Require: Dataset D, Image Prompt Key Pool P_I, Text Prompt Key Pool P_T, Image Prompt Size S_I, Text Prompt Size S_T
  while Not Converge **do**
    Random Select batch of image I, text T from D
    $\hat{v}_I = mean(VE(I), dim = 1)$
    $\hat{v}_T = mean(VT(T), dim = 1)$
    $Cluster_I$ = dictionary()
    $Cluster_T$ = dictionary()
    for *i*, *t* in $v_{I_{mean}}$, $v_{T_{mean}}$ **do**
      Key_{img} = image key with top $similarity(i, P_I)$
      Key_{txt} = text key with top $similarity(t, P_T)$
      $Cluster_I[Key_{img}]$.append(*i*)
      $Cluster_T[Key_{txt}]$.append(*t*)
    end for
    for i in S_I **do**
      P_I[i] = $mean(Cluster_I$[i])
    end for
    for i in S_T **do**
      P_T[i] = $mean(Cluster_T$[i])
    end for
  end while

convergence of the clustering algorithm. During the second stage, the trained K_v and 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 P_k from the prompt pool P. P_k is then attached to the model pipeline:

$$\hat{y}(\mathbf{V}, \mathbf{T}) = \mathcal{F}(FE([P_k; VE(\mathbf{V}); TE(\mathbf{T})])) \quad (6)$$

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

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

The final objective loss function would be:

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

Where $\mathcal{L}_{ce}$ is the same cross entropy loss.

During inference, the model is frozen and we follow a procedure similar to the second stage of training. For every training image-text pair, 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. The combination of prompt keys is deployed to find the corresponding prompt, which is pre-pend to the output of multi-modal encoder.

5 Experiments

5.1 Experiment Setup

Backbone We used the public pre-trained large multimodal transformer, ALBEF (Li et al., 2021) 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, and a answer decoder, which has same architecture as multimodal encoder.

Baselines for comparison We use seven methods for comparison. We include algorithms from major CL approaches. We include two regularization-based methods: **EWC** (Kirkpatrick et al., 2017) and **LwF** (Li and Hoiem, 2017), two rehearsal-based methods: **ER** (Rolnick et al., 2019) and **GEM** (Su et al., 2021). We also include three prompt-based continual learning methods, **L2P** (Wang et al., 2022b), **DualPrompt** (Wang et al., 2022a), and **S-Prompt** (Wang et al., 2023). 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.

Metrics for comparison we use the average accuracy on all tasks and the forgetting rate to evaluate the performance of our method and its ability to tackle catastrophic forgetting.

CL Tasks We evaluate our method on tasks built using the **CLOVE** (Lei et al., 2022) dataset which is a VQA-based continual learning dataset. The benchmark contains both scene-incremental setting benchmark, **CLOVE-scene**, and function-incremental setting benchmark, **CLOVE-function**, and 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.

For details about the optimization and implementation processes, please refer to the Appendix.

5.2 Comparative Results

We conduct the comparison experiments on both the **CLOVE-scene** and **CLOVE-function** task

Method	CLOVE-scene						CLOVE-function					
	abcdef		dbafec		bdcafe		oarlks		skaolr		ksoarl	
	A ↑	F ↓	A ↑	F ↓	A ↑	F ↓	A ↑	F ↓	A ↑	F ↓	A ↑	F ↓
Finetune	34.03	34.28	34.89	34.99	38.83	21.65	24.09	62.79	16.34	74.82	17.50	84.46
EWC	37.49	28.04	37.00	29.10	37.95	27.46	40.74	33.89	37.53	37.22	40.85	32.32
LwF	38.18	26.82	35.03	32.84	37.31	29.11	36.81	41.29	30.49	53.11	29.17	55.84
ER	41.05	19.92	42.09	17.12	42.37	18.09	37.14	33.38	33.41	48.99	38.23	38.01
GEM	41.52	18.33	43.14	14.73	42.89	17.43	39.81	28.77	36.88	39.14	40.26	31.87
L2P	43.01	18.22	45.84	15.03	44.64	17.41	42.54	19.18	40.4	31.92	43.37	24.19
DualPrompt	45.51	15.86	46.58	13.49	45.83	16.48	43.69	15.31	39.32	34.78	45.65	20.54
S-Prompt	45.73	14.11	45.93	14.17	46.17	13.86	42.98	20.20	42.85	25.82	44.09	22.32
CluMo	**48.73**	**10.76**	**48.8**	**10.25**	**48.83**	**9.73**	**45.95**	**9.15**	**45.66**	**19.89**	**46.89**	**17.41**

Table 1: Comparative experimental results: the accuracy and forgetting rate for different task order are reported. For each task sequence, **A** ↑ indicates the accuracy of the method, while **F** ↓ is the forgetting rate of each.

sets with a randomly selected task order. In table 1, the task order *abcedf* represents the CL tasks: *ShopAndDining*, *WorkPlace*, *HomeOrHotel*, *Transportation*, *SportAndLeisure Outdoors* in sequence. The *oarlks* 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 accuracy and 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** and **GEM**, achieve better performance than regularization-based methods. This can be explained by the fact tha 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 tune 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 generalizability of pre-trained model and specific previous knowledge stored in prompt, prompt-based method can outperform the replay and regularization methods. Our method is the best method in this group.

Compared with the baseline prompt-based learning which only considers 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.

Table 2: Ablative Experiments

Methods	Accuracy	Forgetting
Full Method	48.73	10.76
Ablative KD	47.36	11.25
Ablative Clustering	46.08	12.86
Ablative Textual Key	46.16	12.49
Ablative Visual Key	46.53	12.22

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

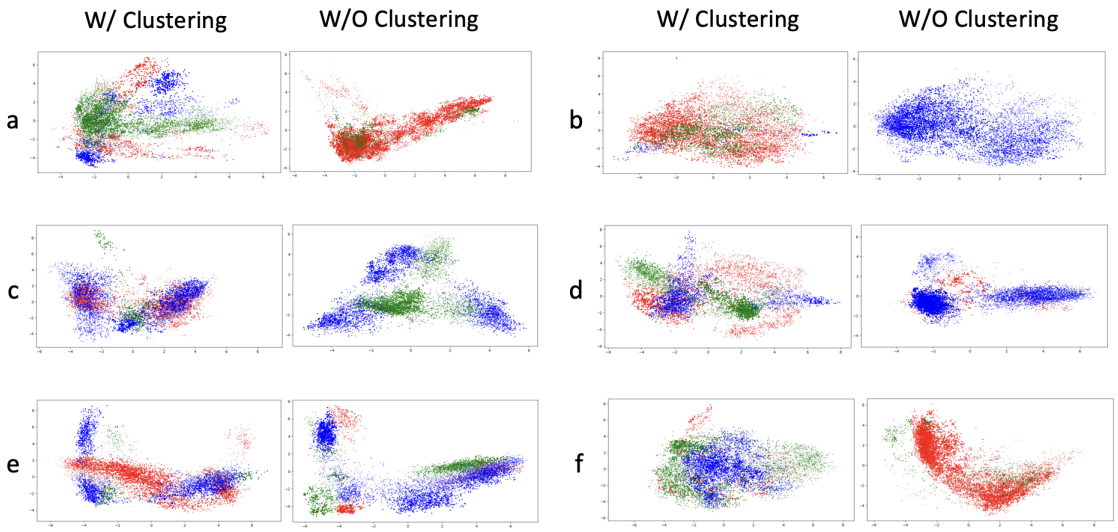

Figure 3: Cluster distribution on all training image data of **CLOVE-Scene**'s six sub-tasks before and after applying mini-batch k-means clustering algorithm with image prompt key size of 3 using PCA.

component. We study the effect of the following:

- **Visual Prompt Key**, key to separate the inner-task image features by their semantic property.
- **Textual Prompt Key**, prompt key to separate the inner-task text features.
- **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 change the size of textual prompt key to 9 to achieve the same prompt size. We also removed the visual prompt key which is the same for ablative image experiments. Results for this experiment is presented in Table 2. 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

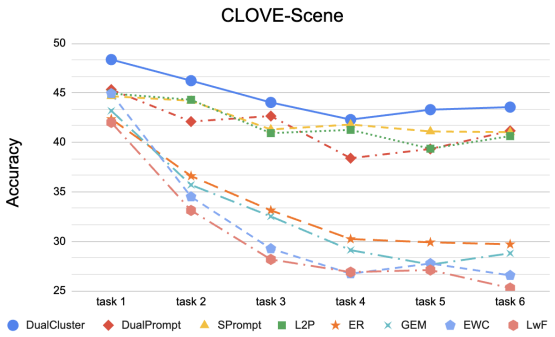

Figure 4: Accuracy on the first task after running task sequence.

Table 3: Accuracy with different clustering error

$\mathcal{E}$. Image	$\mathcal{E}$. Text	Accuracy
15.40	10.72	48.73
15.74	12.22	48.03
17.21	12.53	47.94
42.38	42.8	47.32

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:

$$\mathcal{E} = Avg(\sum_{i=1}^{N} Avg(\sum_{j=1}^{M} ||x_j - c_k||_2)) \quad (9)$$

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 3 presents the results. We observe a negative correlation between the clustering error and the performance accuracy, i.e., 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 for both modalities and the accuracy improves 2.97%.

Cluster Visualization To show the effect of clustering on prompt key more intuitively, we visualize the visual prompt key selection distribution on the visual portion of the training data for **CLOVE-Scene** in Figure 3. Since we use three visual prompt keys for each task, the visual training data are split into three groups, which are the green, blue and red points in Figure 3. 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. Due to space limitations, we include the cluster visualization for text prompt key in Appendix. It indicates similar observation.

Table 4: Accuracy with different prompt pool size

$S_{img} \times S_{txt}$	Accuracy
2×2	48.51
3×3	48.73
4×4	48.32
5×5	48.32
10×10	48.51

Tracking the Accuracy for the First Task To take a closer look in 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. This behavior is an indication 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 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 hyperparameter. In Table 4, we choose different visual prompt key and textual prompt key sizes, $2 \times 2,\ 3 \times 3,\ 4 \times 4,\ 5 \times 5,\ 10 \times 10$, corresponding to 4, 9, 16, 25, 1and 00 prompt pool sizes. We observe minor changes in accuracy in Table 4 when prompt pool size changes, i.e., between 48.32 and 48.73. This observations means that our method is not sensitive to the change of the prompt pool size and hence we don't need to tune it.

6 Conclusion

We introduced a novel prompt-based continual learning method for learning multimodal tasks. While most of existing methods apply single prompts on a single modality, our method proposes modal-specific prompt key pool and train it to capture the semantic properties of the training dataset using a clustering algorithm. We use the combination of both the visual prompt key and the textual prompt key to select prompts, which enable the prompt to better boost the performance. Our experiments show that our method achieves the state-of-the-art performance in continual VQA tasks in different domains compared to other regularization-based, rehearsal-based and prompt-based CL methods.

7 Limitations

Due to limited computational resources, all of our experiments are done using a single GPU and we haven't explored the performance of our method in a distributed system setting. We will embed distributed training in our code to boost the training speed and further analyze the performance within multi-GPU settings. Moreover, in our setting we may have single-modal inputs due to occlusions in one of the modalities. In such cases, our performance may suffer and we may need a new technique to address this challenge. Meanwhile, although **CluMo** is designed for visual question answering, it has the potential to be expanded to other multimodal tasks such as image captioning, speech emotion recognition, cross-modal retrieval, etc. We will further explore such possibility with corresponding experiments. Furthermore, **CluMo** is designed for domain incremental learning where each task is diverse from the others. We haven't tested its performance on other CL experiment settings such as class-incremental or task-incremental, which will be in our exploration plan in the future.

References

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. 2016. Vqa: Visual question answering.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Yuliang Cai, Jesse Thomason, and Mohammad Rostami. 2023. Task-attentive transformer architecture for continual learning of vision-and-language tasks using knowledge distillation.

Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. 2022. Dytox: Transformers for continual learning with dynamic token expansion.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2016. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Stan Weixian Lei, Difei Gao, Jay Zhangjie Wu, Yuxuan Wang, Wei Liu, Mengmi Zhang, and Mike Zheng Shou. 2022. Symbolic replay: Scene graph as prompt for continual learning on vqa task.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning.

Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting.

David Lopez-Paz and Marc'Aurelio Ranzato. 2022. Gradient episodic memory for continual learning.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. 2019. Ok-vqa: A visual question answering benchmark requiring external knowledge.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. 2017. icarl: Incremental classifier and representation learning.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy P. Lillicrap, and Greg Wayne. 2019. Experience replay for continual learning.

Lin Su, Nan Duan, Edward Cui, Lei Ji, Chenfei Wu, Huaishao Luo, Yongfei Liu, Ming Zhong, Taroon Bharti, and Arun Sacheti. 2021. Gem: A general evaluation benchmark for multimodal tasks.

Gido M Van de Ven and Andreas S Tolias. 2019. Three scenarios for continual learning. *arXiv preprint arXiv:1904.07734*.

Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. 2023. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022a. Dualprompt: Complementary prompting for rehearsal-free continual learning.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022b. Learning to prompt for continual learning.

A Appendix

A.1 Hardware Setup and Hyper-parameter

All the experiments are done on single Nvidia A40 GPU. For all the experiments, we use AdamW optimizer with cosine scheduler and we set learning rate lr= 3e-4. We set the training epoch = 5 and training batch size = 16.

For CluMo, we set visual prompt key size S_v = 3, text prompt key size S_t = 3 and prompt length L_p = 10. For DuamPrompt, we insert G-Prompt to [0,1] layers of visual encoder, and insert E-Prompt to [2,3,4] layers of the visual encoder. For all the prompt-based baselines, L_p = 10.

For all the prompt-based methods such as L2P, DualPrompt and SPrompt, we freeze the whole backbone model except the last classifier layer. For the rest of the baseline methods, we don't freeze any parameters.

A.2 More Experiment Result

We present more comparison experiments with different task order for **CLOVE-scene** and **CLOVE-function** here in table 5 and table 6. Similar as what we present in main paper, our method outperforms other baseline methods in all tasks. Among all the continual learning methods, the accuracy of regularization-based methods, **EWC** and **LwF** is significantly lower than other methods, which indicates that the regularization-based method may not be the state-of-the-art continual learning method regarding large multimodal model with incoming tasks from different domain. Prompt-based methods, on the other hand, are the state-of-the-art methods regarding both forgetting rate and accuracy. Without the need of memory buffer and finetune the whole model, prompt-based methods are memory-efficient than replay-based methods and time-efficient than regularization-based methods, which makes them the most preferred choice in current condition. Our method, which is prompt-based method, further improve the accuracy and forgetting rate on the top of exiting prompt-based baselines, while keep the advantage of prompt-based method.

A.3 CLOVE dataset detail description

For all the 12 tasks in **CLOVE-Scene** and **CLOVE-Function**, except $SceneTextRecognition$ which has 16.8K training data and 2.4K testing data, all the other tasks have 20K training data and 3K testing data. We present more detail about **CLOVE** dataset here. To visualize the dataset and explicitly show that each task is from different domain, we present two samples for each dataset in Figure 5 and Figure 6. From the samples, we can see that the image in **CLOVE-scene** are diverse from each other between different tasks, while the questions are similar with each other with the only difference regarding the content of the pictures. However, for **CLOVE-function** dataset, we cannot tell the image from different tasks are from different domain, as they are mixed up. But we can see that the type of questions that each task asks is quite diverse for different purposes of reasoning.

A.4 Text Prompt Key Cluster Visualization

We put the visualization of text prompt key cluster here as the supplementary material of image prompt key cluster visualization in main paper. By observing the figures of "W/O Clustering", we find

Method	CLOVE-scene					
	acbefd		caefdb		bafedc	
	A ↑	F ↓	A ↑	F ↓	A ↑	F ↓
Finetune	34.45	35.14	34.42	34.47	33.95	35.67
EWC	37.99	29.68	37.13	28.63	37.83	27.97
LwF	37.85	29.87	37.94	28.15	38.21	27.48
ER	41.91	20.28	41.11	19.65	42.08	20.52
GEM	42.54	20.13	41.90	20.88	43.11	19.86
L2P	45.63	14.96	44.78	17.99	46.58	14.85
DualPrompt	46.27	15.45	46.21	15.89	47.01	13.16
S-Prompt	46.99	14.38	46.68	14.77	47.53	12.19
Ours	**48.94**	**10.29**	**48.26**	**11.04**	**48.98**	**10.23**

Table 5: More Comparative Experiment with different task sequence order of **CLOVE-scene** dataset.

Method	CLOVE-function					
	soarkl		caefdb		bafedc	
	A ↑	F ↓	A ↑	F ↓	A ↑	F ↓
Finetune	31.55	53.76	37.34	39.64	23.34	57.32
EWC	35.70	47.92	37.82	41.55	38.92	40.48
LwF	37.18	46.86	36.81	44.12	39.21	39.81
ER	42.22	32.97	39.78	38.62	41.22	35.79
GEM	44.58	30.87	41.43	29.46	40.87	32.98
L2P	44.80	16.38	43.39	21.26	43.27	21.97
DualPrompt	45.01	15.90	44.26	17.43	44.66	18.50
S-Prompt	45.45	13.47	45.01	14.76	45.27	14.29
Ours	**46.18**	**10.62**	**45.36**	**11.69**	**46.34**	**10.22**

Table 6: More Comparative Experiment with different task sequence order of **CLOVE-scene** dataset.

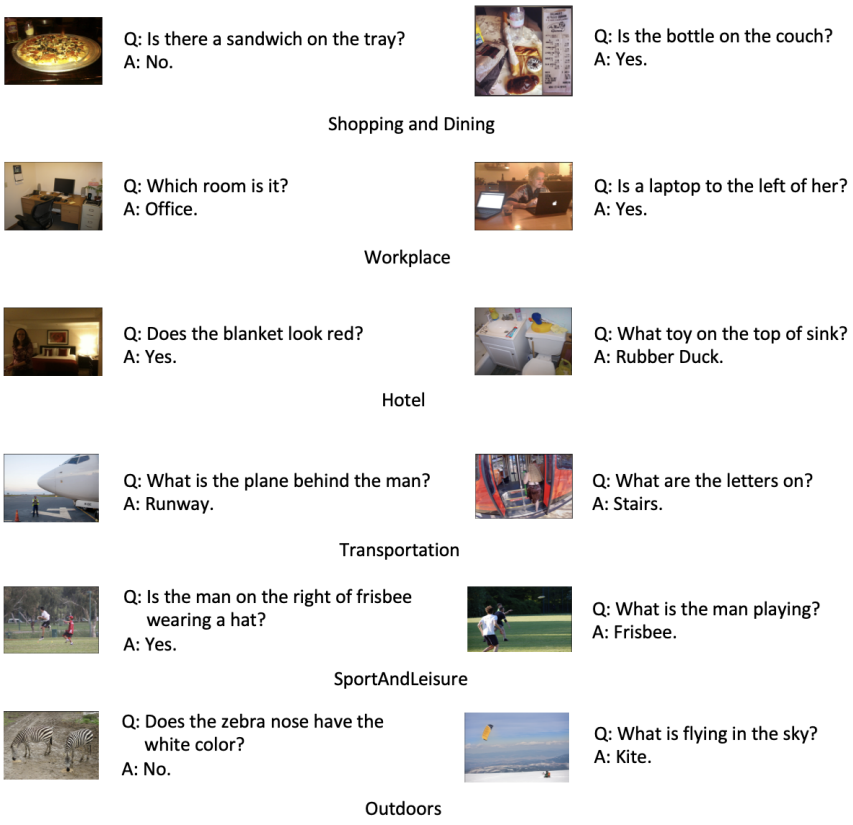

Figure 5: CLOVE-scene dataset sample

that most of the tasks' text input, b, c, e, f, are concentrating to single text prompt key, and the text input of task a and d are distributed into two text prompt keys while the boundary is blurred and not explicit. After applying clustering algorithm, the text inputs are more evenly distributed among three different text prompt keys. However, compared with the clustering of image prompt key, the distribution of text input does not show apparent diversity among different text prompt keys, which indicate that the clustering of text, which is the question in VQA setting, is more difficult than the clustering of images, and thus offer us new field to further explore.

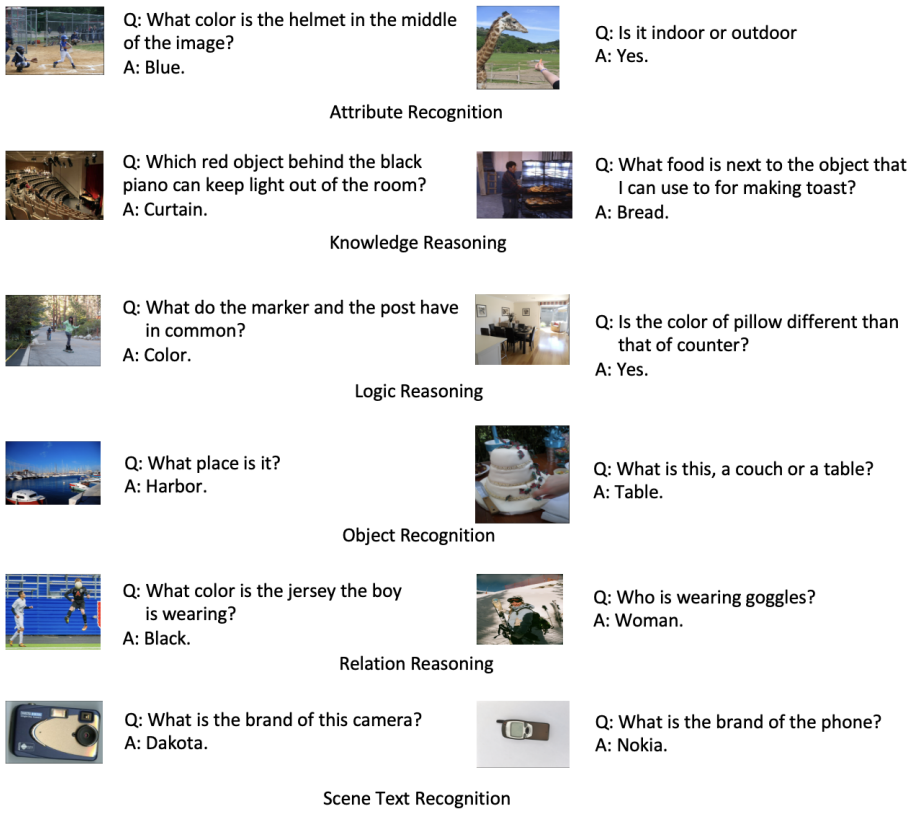

Figure 6: CLOVE-function dataset sample

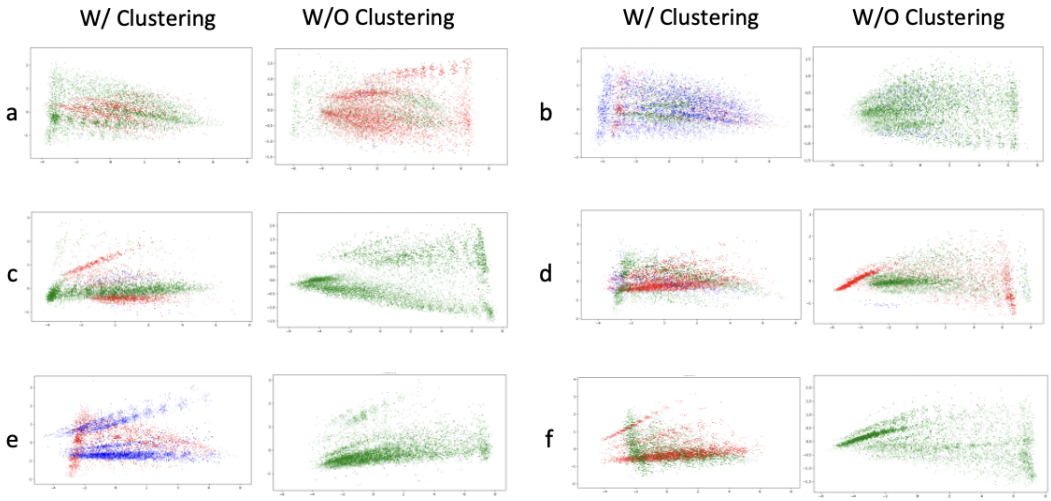

Figure 7: Text prompt key clustering visualization