Decouple Before Interact: Multi-Modal Prompt Learning for Continual Visual Question Answering

Zi Qian[1,2], Xin Wang[1]*, Xuguang Duan[1], Pengda Qin[2], Yuhong Li[2], Wenwu Zhu[1]*
[1]Department of Computer Science and Technology, BNRist, Tsinghua University
[2]Alibaba Group
qianz9729@gmail.com, xin_wang@tsinghua.edu.cn, duan_xg@outlook.com
pengda.qpd@alibaba-inc.com, daniel.lyh@alibaba-inc.com, wwzhu@tsinghua.edu.cn

Abstract

*In the real world, a desirable Visual Question Answering model is expected to provide correct answers to new questions and images in a continual setting (recognized as CL-VQA). However, existing works formulate CL-VQA from a vision-only or language-only perspective, and straightforwardly apply the uni-modal continual learning (CL) strategies to this multi-modal task, which is improper and suboptimal. On the one hand, such a partial formulation may result in limited evaluations. On the other hand, neglecting the interactions between modalities will lead to poor performance. To tackle these challenging issues, we propose a comprehensive formulation for CL-VQA from the perspective of multi-modal vision-language fusion. Based on our formulation, we further propose Mul**T**i-Modal P**R**ompt Learn**I**ng with Decou**PL**ing b**E**fore In**T**eraction (TRIPLET), a novel approach that builds on a pre-trained vision-language model and consists of decoupled prompts and prompt interaction strategies to capture the complex interactions between modalities. In particular, decoupled prompts contain learnable parameters that are decoupled w.r.t different aspects, and the prompt interaction strategies are in charge of modeling interactions between inputs and prompts. Additionally, we build two CL-VQA benchmarks for a more comprehensive evaluation. Extensive experiments demonstrate that our TRIPLET outperforms state-of-the-art methods in both uni-modal and multi-modal continual settings for CL-VQA.*

1. Introduction

Visual Question Answering (VQA) [2, 11, 37, 26] aims to train a machine learning model capable of answering questions given visual images as accurately as possible.

*Corresponding authors

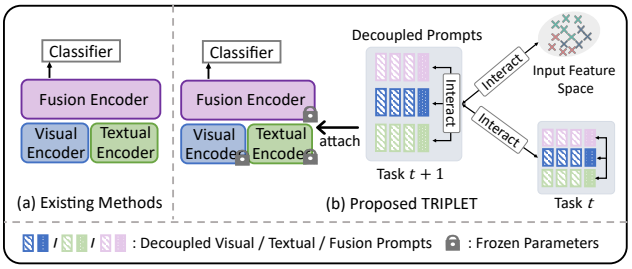

Figure 1: Comparison between (a) existing CL-VQA methods [45, 29] and (b) our proposed TRIPLET model. Existing methods train all the parameters similar to typical uni-modal CL-methods, while our TRIPLET model trains parameters in prompts and classifiers, as well as explicitly model the rich and complex modality-wise interactions.

In real-world dynamic environments [23], an ideal VQA model is expected to generate answers for new questions, new images, as well as new question-image simultaneously, which is recognized as CL-VQA [20], *i.e.*, learn a sequence of VQA tasks with a single model without suffering from catastrophic forgetting [28] on previously observed data.

Existing works [20, 29] formulate CL-VQA as a vision-only or language-only continual learning setting, and straightforwardly apply the uni-model continual learning (CL) methods to this multi-modal task. However, modeling CL-VQA from such a uni-model view is suboptimal, posing two challenging issues. First, the existing partial formulation does not take the multi-modal nature of CL-VQA into account, which leads to a limited view and improper evaluations. Second, by straightforwardly employing the uni-model CL methods, existing CL-VQA methods may neglect the rich and complex interactions between modalities, which leads to deteriorating performance.

To tackle the two challenging issues, we first propose a comprehensive formulation for CL-VQA explicitly covering both multi-modal and uni-modal perspectives, so that more extensive evaluations can be conducted in terms of

input distributions. Specifically, we carefully design three scenarios according to different input distributions, *i.e.*, *Continual Vision Scenario, Continual Language Scenario, and Continual Vision-Language Scenario*, depending on incremental visual images, textual questions, and both.

Secondly, based on our CL-VQA formulation with three scenarios, we propose Mul**T**i-Modal P**R**ompt Learn**I**ng with Decou**PL**ing b**E**fore In**T**eraction (TRIPLET), a multi-modal prompt learning-based continual model for CL-VQA. TRIPLET employs the widely adopted pre-trained vision-language models with state-of-the-art VQA performance as initialization, and consists of decoupled prompts and prompt interaction strategies. To be specific, decoupled prompts contain a set of learnable parameters decoupled in three aspects, *i.e.*, modality aspect, layer aspect, and complementary aspect, which are attached to transformer layers. Then the prompt interaction strategies are designed to model the interactions between the input and prompts, modality-wise prompts, as well as task-wise prompts. Fig. 1 illustrate a comparison between existing CL-VQA methods and our proposed TRIPLET model.

In addition, we build two CL-VQA benchmarks on two datasets (*i.e.*, TDIUC [14] and VQA2.0 [9]), carrying out extensive experiments on three scenarios. Our TRIPLET model is able to consistently outperform baselines and SOTAs[1] significantly across various settings. Besides, we conduct ablation studies to validate the effectiveness of different components in TRIPLET, demonstrating TRIPLET's superiority. In summary, our contributions are as follows:

- We propose a comprehensive formulation for CL-VQA with multi-modal continual setting, enabling the continual evaluations of various approaches in three scenarios based on different input distributions.
- We propose TRIPLET, a novel CL-VQA model containing decoupled prompts and prompt interaction strategies, which is able to accurately generate answers in three continual scenarios without rehearsal buffer. To the best of our knowledge, TRIPLET is the first multi-modal prompt learning-based continual model for CL-VQA.
- We build up two CL-VQA benchmarks (*i.e.*, CL-VQA2.0 and CL-TDIUC) for empirical evaluations of CL-VQA including multi-modal continual setting. Our proposed TRIPLET model achieves significant improvement over state-of-the-art approaches in all three scenarios for both two benchmarks. Extensive ablation studies further demonstrate the effectiveness of different components in TRIPLET.

2. Related Works

Visual Question Answering Visual Question Answering (VQA) aims to answer related questions given an image, which requires multi-modal reasoning ability. Existing VQA methods [2, 11, 37, 26] and datasets [9, 14, 13, 18] are usually designed for a stable environment, while the VQA system being able to cope with dynamic environment (CL-VQA) is rarely studied. In this paper, we focus on the CL-VQA problem and propose the effective TRIPLET method.

Continual Learning Methods There exist numerous continual learning methods which could be categorized into three categories: (1) Regularization-based methods [23, 17, 43, 1] try to reduce catastrophic forgetting by regularizing import parameters for previous tasks. (2) Rehearsal-based methods [31, 33, 3, 42, 4, 35, 8, 41] use a buffer to store representative samples or pseudo samples for previous task to avoid catastrophic forgetting. In particular, [20] generates pseudo scene graphs for replay to mitigate forgetting for CL-VQA. However, scene graphs are not easily available in real-world applications, making it less applicable. (3) Architecture-based methods [15, 47, 25, 22, 44, 34, 42] associate different parameters for different tasks to mitigate forgetting. Recent works [38, 40, 39, 7, 30] adopt prompt tuning technique, trying to assign each task with learnable parameters. However, these methods are designed for uni-modal continual learning, failing to take multi-modal fusion and reasoning characteristic of CL-VQA into account. In particular, S-Prompts [38] is suited for CL image classification and not directly applicable to CL-VQA. S-iPrompts in [38] handles only uni-modal inputs, while S-liPrompts in [38], based on CLIP, calculates scores between all possible labels and images, which is unsuitable for open-ended CL-VQA involving lengthy textual question inputs and thousands of answers in CL-VQA settings.

Continual Learning Benchmarks for VQA There exists a few continual learning benchmarks for VQA. [10, 20, 29] construct CL-VQA benchmarks from the uni-modal perspective. [45] builds CL-CrossVQA from the multi-domain perspective and formulates each domain as a distribution, while fails to characterize different distribution types and corresponding real scenarios. In this paper, we provide a comprehensive formulation from the multi-modal perspective for CL-VQA, and build two benchmarks with three scenarios, respectively.

3. Task Formulation

Continual Learning (CL) aims to capture the ever-changing world and update models on a continuum of sequential coming data and tasks[2] [40], where the data from previous task is not available during training [49]. In this paper, we focus on continual learning for the Visual Question Answering (VQA) task that is to answer questions based on a given image, which is usually formulated as a multi-label classification task involving thousands

[1] We necessarily modify some SOTAs for better adaptation to CL-VQA.

[2] Also known as 'session,' 'phase,' or 'stage.'

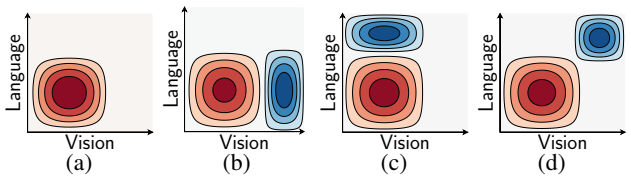

Figure 2: Graphical explanations of: (a) the red part denotes the ideal data distribution of task $1 \sim$ task t, (b) Continual Vision Scenario, (c) Continual Language Scenario, and (d) Continual Vision Language Scenario. The blue part represents the distributions of the task $t+1$.

of classes [2, 11, 37, 26]. As time passed, new images, new questions, and even new answers would appear, and we have to update the VQA model accordingly. Following [20, 29], we namely define this problem as CL-VQA. Besides, we consider the more challenging CL-VQA setting where the task identity is unknown for each sample during inference, *i.e.*, we do not know which task the samples belong to during test time.

We denote the sequential tasks of CL-VQA as $D = \{D_1, D_2, ..., D_T\}$, where $D_t = \{(\boldsymbol{x}_i^t, \mathrm{y}_i^t)\}_{i=1}^{n_t}$ is the available data at t-th training task with n_t instances. Unlike most of the classical CL tasks where the input data $\boldsymbol{x}$ is uni-modal [49], the VQA input data $\boldsymbol{x} = (\boldsymbol{v}, \boldsymbol{q})$, containing a visual scene $\boldsymbol{v}$ and a question $\boldsymbol{q}$, is a multi-modal data. Thus, the input distribution $\Pr(\boldsymbol{x}) = \Pr(\boldsymbol{v}, \boldsymbol{q})$ depends on the marginal distribution $\Pr(\boldsymbol{v})$ and $\Pr(\boldsymbol{q})$, and the interaction between the two modalities. However, most of previous CL-VQA works [20, 29] only focus on partial settings (*i.e.* only $\Pr(\boldsymbol{v})$ and $\Pr(\boldsymbol{q})$) from uni-modal perspective, therefore not providing all-inclusive evaluations for continual methods.

In this paper, we consider continual learning scenarios systematically, explicitly from the uni-modal distribution as well as their joint-distribution, formulating CL-VQA in a more comprehensive way. Namely, we design three scenarios in CL-VQA:

- **Continual Vision Scenario (ConVS)** considers the changes of vision distribution $\Pr(\boldsymbol{v})$, while keeps $\Pr(\boldsymbol{q}|\boldsymbol{v})$ unchanged. ConVS addresses the scenarios when new visual scenes occur while the possible questions remain the same.
- **Continual Language Scenario (ConLS)** considers the changes of question distribution $\Pr(\boldsymbol{q})$, while keeps $\Pr(\boldsymbol{v}|\boldsymbol{q})$ unchanged. ConLS addresses the scenarios when new questions arise on current available visual scene.
- **Continual Vision-Language Scenario (ConVLS)** considers the changes both vision and questions $\Pr(\boldsymbol{v}, \boldsymbol{q})$. ConVLS addresses the free-form changes of both modalities and their interactions, *i.e.*, new visual scene appear, new questions arise, and $\Pr(\boldsymbol{v}|\boldsymbol{q})$ or $\Pr(\boldsymbol{q}|\boldsymbol{v})$ would also change.

We further provide a graphical explanation of these scenarios in Fig. 2. A desirable CL-VQA method is supposed to perform well across all the aforementioned scenarios.

4. The Proposed Methods

To address the aforementioned three scenarios, it is important that we model both vision and language modalities and their interaction at the same time. In this paper, we follow the general Prompt Learning framework [12] and propose the novel Mul**T**i-Modal P**R**ompt Learn**I**ng with Decou**PL**ing b**E**fore In**T**eraction (TRIPLET) method to address the exemplar-free continual VQA problem.

4.1. Preliminary

Transformer-Based VQA Model A modern transformer based VQA model usually contains three encoders, namely visual encoder, textual encoder, and fusion encoder [6, 21, 36]. Formally, the answer of a question $\boldsymbol{q}$ given an image $\boldsymbol{v}$ can be written as follows:

$$\hat{y}(\boldsymbol{v}, \boldsymbol{q}) = \mathcal{F}\Big(\texttt{FT}\big(\big[\texttt{VT}(\boldsymbol{v}); \texttt{TT}(\boldsymbol{q})\big]\big)[0]\Big), \tag{1}$$

where VT and TT are the pretrained visual transformer encoder and textual transformer encoder that encodes $\boldsymbol{v}$ and $\boldsymbol{q}$, respectively. $\texttt{FT}(\cdots)[0]$ fuses the multimodel features together, and output the first fused feature into a classifier $\mathcal{F}(\cdot)$ to predict an answer a. Our proposed TRIPLET is built upon this structure.

Prompt Learning Given an input sequence data $\boldsymbol{x} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_{n_x}]$ and a transformer encoder T, prompt learning aims to find several "call-words" $P = [P_0, P_1, \cdots, P_{n_p}]$ that when P is attached with $\boldsymbol{x}$, the output feature would meet certain requirements. In the following, we use the notation $\texttt{T}([P; \boldsymbol{x}])$ to denote that we add prompts to $\boldsymbol{x}$.

4.2. TRIPLET: Decouple Before Interact

Our proposed method, TRIPLET is illustrated in Fig. 3. Built upon transformer-based VQA models, our goal is to design a set of proper prompts and interaction strategies that could solve CL-VQA problem. We will first introduce our Prompt Decoupling Design separately in Sec. 4.2.1, and then combine them to train together with our Prompt Interaction Strategies in Sec. 4.2.2, finally, overall training and inference are introduced in Sec. 4.2.3.

4.2.1 Prompt Decoupling

Multi-Modal Decoupling Unlike those uni-modal prompts proposed by previous work [39, 40], in this paper, we disentangle prompts into multi-modal format to fully address the modality-related knowledge from both the pre-trained vision-language model and training data.

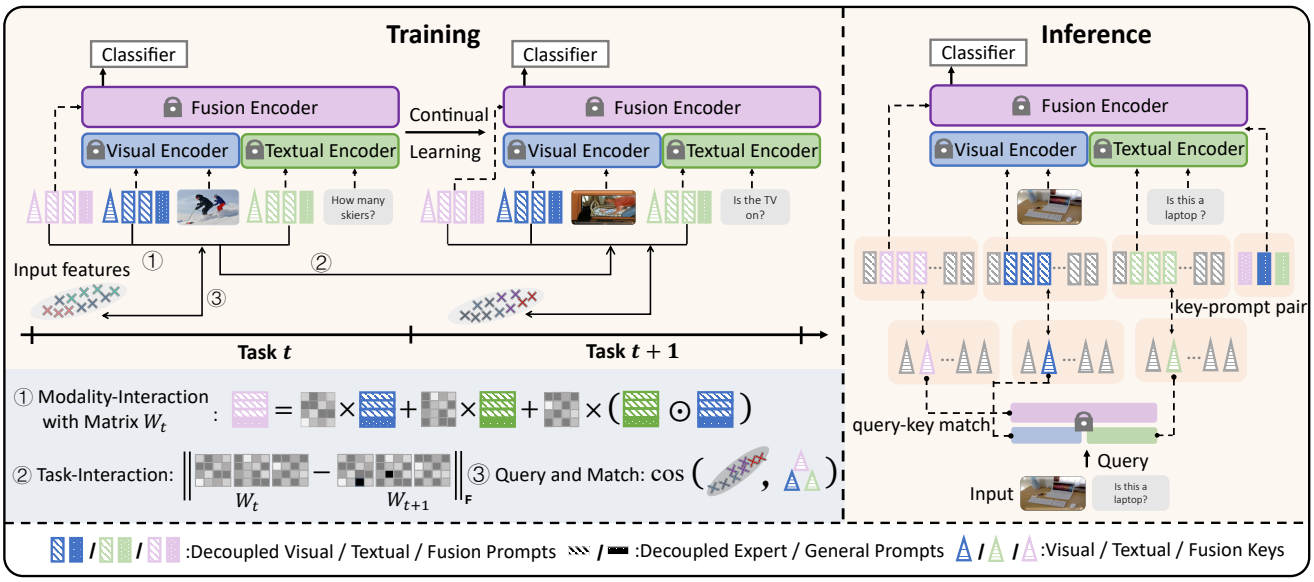

Figure 3: The TRIPLET framework. **Left:** during training, the pre-trained encoders are frozen, and parameters in classifier, decoupled prompts, task-specific keys and interaction matrix are learnable. At task $t+1$, we train the decoupled prompts (including three aspects, *i.e.*, modality-wise, layer-wise and complementary). We further apply three interaction strategies (within the light blue colored rectangle) to model modality-wise prompt interaction, task-wise prompt interaction, and interaction between input features and prompt keys. **Right:** during inference, we first calculate multi-modal representations with the query function, which are used to match the most similar multi-modal keys. Then decoupled E-Prompts paired with matched keys, together with decoupled G-Prompts, are appended to the inputs (or features) for answer generation.

Basically, Eq. (1) would be modified with:

$$\hat{y}(\boldsymbol{v},\boldsymbol{q}) = \mathcal{F}\Big(\big[\mathtt{FT}\big(\big[P^{(f)};\mathtt{VT}([P^{(v)};\boldsymbol{v}]);\mathtt{TT}([P^{(q)},\boldsymbol{q}])\big]\big)[0]\Big), \tag{2}$$

where $P^{(v)}$,$P^{(q)}$, $P^{(f)}$ are the vision, question, and fusion prompt, respectively.

Selective Deep Decoupling We then disentangle prompts in a layer-wise format, and attaching it to selective layers. Rather than keeping attaching prompts to all the selected multi-head attention (MHA) layers [39], in this paper, we add prompts to some MHA layers in a replacing schema, which is more memory-efficient. Given a transformer $\mathtt{T}$ containing K layers, $\mathtt{T}([P;\boldsymbol{x}]) = (\mathtt{L}_K \circ \mathtt{L}_{K-1} \cdots \circ \mathtt{L}_0)([P;\boldsymbol{x}])$ could be decomposed layer-by-layer:

$$\begin{aligned} \bar{\boldsymbol{h}}_k^P &= \alpha_k \cdot \boldsymbol{h}_k^P + (1-\alpha_k)\cdot P_k, \\ [\boldsymbol{h}_{k+1}^{\mathtt{CLS}};\boldsymbol{h}_{k+1}^P;\boldsymbol{h}_{k+1}^{\boldsymbol{x}}] &= \mathtt{L}_k([\boldsymbol{h}_k^{\mathtt{CLS}};\bar{\boldsymbol{h}}_k^P;\boldsymbol{h}_k^{\boldsymbol{x}}]), \end{aligned} \tag{3}$$

where $[\boldsymbol{h}_0^{\mathtt{CLS}};\bar{\boldsymbol{h}}_0^P;\boldsymbol{h}_0^{\boldsymbol{x}}] = [\mathtt{CLS}, P_0, \boldsymbol{x}]$ are the raw inputs, and the output of $\mathtt{L}_K$ is regarded as model output. Moreover, $\alpha_k \in \{0,1\}$ is a predefined switch that controls whether using the output prompt feature $\boldsymbol{h}_k^P$ or the k-th layer-specific prompt P_k as input.

Complementary Decoupling Following the complementary design principle [39], each prompt is further split into two parts: a General Prompt (G-Prompt) to extract task-invariant knowledge, and an Expert Prompt (E-Prompt) to extract task-specific knowledge. For example, the visual prompt $P^{(v)} = \{G^{(v)};\{E^{(v)}\}\}$ is composed of G-prompt $G^{(v)}$ shared for all tasks and E-prompt $E_t^{(v)}$ specialized for the t-th task . When the t-th task comes, we train the prompt $P_t^{(\mathtt{m})} = \{G^{(\mathtt{m})};E_t^{(\mathtt{m})}\}$ where $\mathtt{m} = v,q,f$.

In our implementation, we combine all the three aforementioned decoupling designs. That is, we have three sets of prompts for three modalities, where each set of prompts contains layer-wise deep-prompts and each layer-wise deep prompt contains a G-prompt and a set of E-prompts. In summary, all the learnable prompts include:

$$\begin{aligned} P^{(\mathtt{m})} = \Big\{G_k^{(\mathtt{m})} \in \mathbb{R}^{L_G \times D}\Big\} \bigcup \Big\{E_{t,k}^{(\mathtt{m})} \in \mathbb{R}^{L_E \times D}\Big\}, \\ \mathtt{m} = v,q,f, \end{aligned} \tag{4}$$

with subscripts t for tasks, k for the k-th MHA layers, L_G /L_E for G / E-Prompt's length, D for embedding dimension.

4.2.2 Prompt Interaction

With the proposed decoupled prompts, then we need interaction strategies to train them all together. We first have Query-and-Match Strategy to match between input features and related task-specific prompts. We further introduce Modality-Interaction Strategy and Task-Interaction Strategy to promote interactions between prompts. The former

one would encourage mutual propagation between different modalities of prompts, thus strength the model performance [16]. And the latter one would make prompts less affected by sequential tasks, thus reduces catastrophic forgetting.

Query-and-Match Strategy As our decoupled prompts include task-specific prompts, we need accurate task-specific keys to link input features to these prompts. We extend the "Query-and-Match" strategy in [39, 40]'s scope to the multi-modal domain to train the corresponding task-specific key $\boldsymbol{u}_t^{(\mathtt{m})}$ via a query matching loss $\mathcal{L}_{qm}$, making $\boldsymbol{u}_t^{(\mathtt{m})}$ closer to samples from the task t than others. Firstly, given $(\boldsymbol{v}, \boldsymbol{q})$, the queries are obtained using the frozen transformers (see Eq. (1)) as

$$\begin{aligned} \boldsymbol{h}^{(v)} &= \mathtt{VT}(\boldsymbol{v}), \quad \boldsymbol{h}^{(q)} = \mathtt{TT}(\boldsymbol{q}), \quad \boldsymbol{h}^{(f)} = \mathtt{FT}([\boldsymbol{h}^{(v)}, \boldsymbol{h}^{(q)}]), \\ \mathtt{Q}^{(v)} &= \boldsymbol{h}^{(v)}[0], \quad \mathtt{Q}^{(q)} = \boldsymbol{h}^{(q)}[0], \quad \mathtt{Q}^{(f)} = \boldsymbol{h}^{(f)}[0], \end{aligned}$$

where $\boldsymbol{h}[0]$ means selecting the first element from the vector, *i.e.*, selecting $\boldsymbol{h}^{\mathtt{CLS}}$ as shown in Eq. (3). Using cosine similarity γ, the query matching loss $\mathcal{L}_{qm}$ is:

$$\mathcal{L}_{qm}(D_t) = -\sum_{(\boldsymbol{v},\boldsymbol{q})\in D_t} \sum_{\mathtt{m}\in\{v,q,f\}} \gamma\Big(\boldsymbol{u}_t^{(\mathtt{m})}, \mathtt{Q}^{(\mathtt{m})}\Big). \tag{5}$$

Modality-Interaction Strategy We present the Prompt Modality-Interaction that acts as a bridge between different modalities of prompts. We introduce the following interaction mapping:

$$\hat{P}_{t,k}^{(f)} = \boldsymbol{W}_{t,k}^{(v)} \otimes P_{k,t}^{(v)} + \boldsymbol{W}_{t,k}^{(q)} \otimes P_{t,k}^{(q)} + \boldsymbol{W}_{t,k}^{(v,q)} \otimes \Big(P_{t,k}^{(v)} \odot P_{t,k}^{(q)}\Big), \tag{6}$$

where $\odot$ is the element-wise multiplication, $\otimes$ is the matrix multiplication, and $\boldsymbol{W}^{(\cdot)}$ are the learnable interaction matrixes. In this paper, we constrain the rank of these interaction matrixes with $\boldsymbol{W} = \boldsymbol{U} \otimes \boldsymbol{V}^\top$, where $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{D\times d}$ are two low-rank matrixes. We use the following $\mathcal{L}_{mod}$ to address this modality-interaction:

$$\mathcal{L}_{mod}(D_t) = -\sum_k \gamma\left(\hat{P}_{t,k}^{(f)}, P_{t,k}^{(f)}\right). \tag{7}$$

Task-Interaction Strategy As our prompt learning-based method is built upon the frozen pre-trained model, the representations for different tasks share the same semantic space. Therefore, prompts share the invariant semantic space between different tasks to align with pre-trained model, which leads to invariant prompt modalities-interaction structure between different tasks. To this end, we introduce the task-interaction constraint $\mathcal{L}_{task}$ to regulate the invariant structure as follows:

$$\mathcal{L}_{task}(D_t) = \sum_{\mathtt{m},t,k} \left(\left\| \boldsymbol{W}_{t,k}^{(\mathtt{m})} - \langle \boldsymbol{W}_{t,k}^{(\mathtt{m})} \rangle_{t-1} \right\|_F^2\right), \tag{8}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, and $\langle \boldsymbol{W}_k^{(\mathtt{m})} \rangle_{t-1}$ is the cached copy of $\boldsymbol{W}_k^{(\mathtt{m})}$ when training task $(t-1)$.

4.2.3 Training and Inference

Training When a new task t comes, we instantiate $\mathcal{F}$ as a classifier g_t (a fully connected layer), and allocate the task-specific querying keys $(\boldsymbol{u}_t^{(v)}, \boldsymbol{u}_t^{(q)}, \boldsymbol{u}_t^{(f)})$ and prompts $(E_t^{(v)}, E_t^{(q)}, E_t^{(f)})$. Then, the decoupled prompts, interaction matrix, classifier, querying keys as jointly trained with:

$$\begin{aligned} \mathcal{L}(D_t) = &\sum_{(\boldsymbol{v},\boldsymbol{q},y)\in D_t} \ell_{\text{CE}}(\hat{y}_t(\boldsymbol{v},\boldsymbol{q}), y) \\ &+ \lambda_1 \mathcal{L}_{qm}(D_t) + \lambda_2 \mathcal{L}_{mod}(D_t) + \lambda_3 \mathcal{L}_{task}(D_t), \end{aligned} \tag{9}$$

where $\hat{y}(\boldsymbol{v},\boldsymbol{q})$ is the network prediction (see Eq. (2)), y is the target answer, $\ell_{\text{CE}}(\hat{y}, y)$ is the cross entropy loss, and $\lambda_{(\cdot)}$ are the hyperparameters.

Inference During inference, given an input sample $(\boldsymbol{v},\boldsymbol{q})$, we choose the best matched task index $\arg\max_{t^{(\mathtt{m})}} \gamma\big(\boldsymbol{u}_{t^{(\mathtt{m})}}^{(\mathtt{m})}, \mathtt{Q}^{(\mathtt{m})}\big)$. Then the corresponding prompts $P_{t^{(\mathtt{m})},\cdot}^{(\mathtt{m})}$ are selected, and fed into the corresponding transformer. Finally, the corresponding classifiers $g_{t^{(\cdot)}}$ are selected to predict an answer.

The full picture of TRIPLET at training and inference is described in the Appendix.

5. Experiments

We evaluate our proposed TRIPLET on the three aforementioned scenarios on two well-known VQA datasets, *i.e.*, TDIUC [14] and VQA2.0 [9]. We carefully compare TRIPLET with state-of-the-art (SOTA) methods of different categories under the same experiment settings. Moreover, we conduct extensive ablation studies to provide a better understanding of our proposed TRIPLET method.

5.1. Evaluation Benchmarks

Given the two commonly adopted VQA datasets, TDIUC [14] and VQA2.0 [9], we build continual learning benchmarks (denoted as CL-TDIUC and CL-VQA2.0) by dividing their images and questions into several disjoint hyper-categories, and then construct the benchmarks according to scenarios. For the Continual Vision Scenario (ConVS) and Continual Language Scenario (ConLS) scenarios, we split datasets according to the hyper-categories on images and questions, respectively [24, 5]. For the Continual Vision-Language Scenario (ConVLS), we collect questions of different types from each hyper-category of images to form 5 tasks, such that both image hyper-category and question type are different between tasks.

To note, we follow the original train-validation split while building these two benchmarks to avoid data breach

Table 1: Results for the CL-VQA2.0 and CL-TDIUC built upon ALBEF [21]. **Bold**: best exemplar-free CL-VQA results, Underline: second best exemplar-free CL-VQA results, †: best rehearsal-based CL-VQA results, ‡: rehearsal-based results which outperform the best exemplar-free results, Upper-bound: supervised fine-tuning on the i.i.d. data of each task, ⋄: enhanced methods as discussed in Sec. 5.2, A: average accuracy, F: forgetting.

Method	Buffer Size	CL-VQA2.0						CL-TDIUC					
		ConLS		ConVS		ConVLS		ConLS		ConVS		ConVLS	
		A(↑)	F(↓)	A(↑)	F(↓)	A(↑)	F(↓)	A(↑)	F(↓)	A(↑)	F(↓)	A(↑)	F(↓)
DER [42]	2000	48.56	19.37	51.15	6.48	56.43	5.52	62.83	8.93	74.43	6.98	70.74	14.70
WA [46]	2000	50.09	18.04	54.74	2.57	55.28	6.74	66.02	6.98	75.47	4.88	75.00	8.18
iCaRL [31, 27]	2000	48.71	19.55	53.76	1.12	54.96	7.08	63.56	8.71	74.53	6.37	73.49	10.26
DER [42]	5000	53.39	13.35†	52.34	4.61	58.78†	3.86‡	63.71	7.95	75.18	6.96	71.63	13.42
WA [46]	5000	53.91†	13.51	55.89†	1.95	58.75	3.96‡	69.23†	4.49†	75.84†	4.39†	77.51†	4.68
iCaRL [31, 27]	5000	53.42	14.09	54.72	0.59†	58.58	4.06	67.94	5.46	75.78	4.75	74.98	7.41
LwF [23]	0	37.49	26.08	54.90	2.80	36.87	24.03	39.25	30.50	72.19	5.50	73.71	8.11
EWC [17]	0	37.21	34.13	54.54	3.69	33.78	27.22	14.61	66.37	71.27	8.26	73.65	8.28
L2P⋄ [40]	0	41.38	25.80	41.55	3.86	32.43	27.25	33.95	29.21	75.51	0.60	69.18	15.65
DualPrompt⋄ [39]	0	44.26	24.16	53.56	1.68	41.30	21.37	44.50	14.70	77.38	3.93	81.36	2.31
S-Prompts⋄ [38]	0	45.50	**8.00**	44.18	0.78	46.36	8.65	59.70	7.32	69.89	4.35	72.77	2.25
Ours	0	**56.76**	9.66	**59.41**	**0.12**	**60.53**	**4.08**	**70.80**	**1.64**	**80.47**	**0.15**	**83.06**	**0.54**
Upper-bound	-	64.53	-	59.62	-	64.08	-	74.60	-	80.57	-	83.33	-

when we use pre-trained vision-language models[3]. Detailed analysis for the data splits is provided in the appendix.

5.2. Experimental Details

Backbones We select two public pre-trained models as our backbones, namely ALBEF [21] and FLAVA [36]. These two models differ in fusion encoder, where ALBEF uses cross-attention between two modalities, while FLAVA uses self-attention.

We mainly analyze results on ALBEF in the main paper and provide additional results on FLAVA in the appendix.

Evaluation Metrics Following the common evaluation protocols [40, 39], we use two metrics, namely *Average accuracy* (higher is better) and *Forgetting* (lower is better). We use $S_{t,\tau}$ to represent the accuracy on the τ-th task after training the model on the t-th task. Then, *Average accuracy* is defined as $\sum_{t\leq T}\sum_{\tau\leq t}\alpha_{t,\tau}S_{t,\tau}$ where $\alpha_{t,\tau}$ is a weighted factor to balance the number of testing instances in different tasks, *Forgetting* is defined as $\frac{1}{T-1}\sum_{\tau<T}\max_{t\geq\tau}(S_{t,\tau}-S_{T,\tau})$.

Comparing Methods Based on [29] and our preliminary experiments, vanilla VQA models fail to tackle CL-VQA tasks, we thus focus on those SOTA continual learning approaches from different categories. We compare our TRIPLET with non-prompting rehearsal-based methods: DER [42], WA [46], iCaRL [31]; regularization-based methods: LwF [23], EWC [17]; and the newly proposed prompt-based methods L2P [40], DualPrompt [39] and S-Prompts [38]. Upper-bound is the supervised fine-tuning on the i.i.d. data of each task.

To compare fairly, we use the same backbone for all these approaches, and we train with the backbone for non-prompting methods while freezing the backbone for prompt-based methods. All these approaches and our TRIPLET use the same classifier head. For rehearsal-based methods iCaRL [31], DER [42] and WA [46], we further test two sizes of replay buffer, *i.e.*, 2000 and 5000, which show high performance in [49]. For non-prompting methods, we use the representation of "CLS" token for classification. For prompt-based methods L2P [40], DualPrompt [39] and S-Prompts [38][4], we symmetrically add textual key-prompt pairs to enhance model performance, which we denoted as L2P⋄, DualPrompt⋄ and S-Prompts⋄. Experimental results for original structures of L2P and DualPrompt are in the appendix.

Training Details For those non-prompting methods, we follow the original paper [21, 36] to set up the optimizer. For those prompt-based methods, we follow DualPrompt [39] to set up the optimizer as adamW with cosine scheduler and $4e^{-4}$ start learning rate. For all approaches, we set the training batch size to 16 for CL-VQA2.0 and 64 for CL-TDIUC. For L2P⋄ [40], we use the same hyperparameters as [39] does. For DualPrompt⋄ [39], we add deep-prompts to the [0-2] MHA layers for G-prompts and [2-5] MHA layers for E-prompts, and set $L_G = 5, L_E = 20$ (See Eq. (4)). For TRIPLET, we keep the same hyperparameter with DualPrompt⋄'s for Multi-Modal Prompt. After hyperparameter searching, we set $d = 20, \lambda_1 = 0.1, \lambda_2 =$

[3]These models are usually trained with images from COCO [24] and Visual Genome [18].

[4]We adapted S-iPrompts for CL-VQA.

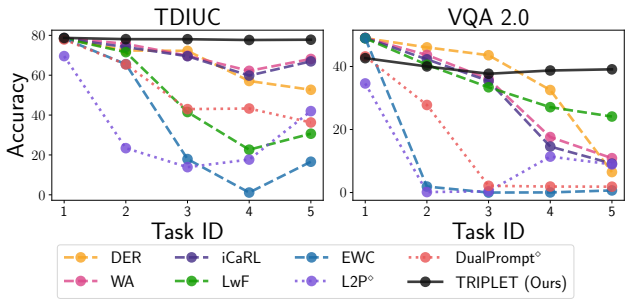

Figure 4: Tracking the accuracy of the first task on Continual Language Scenario (ConLS).

$0.2, \lambda_3 = 0.05$ for all benchmarks with ALBEF, and set $d = 10, \lambda_1 = 0.1, \lambda_2 = 0.1, \lambda_3 = 0.05$ for FLAVA.

Overheads For each task, our proposed TRIPLET method trains a set of additional key-prompt pairs, as well as an interaction constraint matrix, which leads to the 0.55% and 0.44% extra memory cost based on ALBEF [21] and FLAVA [36], respectively. Other SOTA prompt learning-based methods L2P$^\diamond$ and DualPrompt$^\diamond$ take 0.47% and 0.31% extra memory based on ALBEF, and 0.41% and 0.27% based on FLAVA, respectively. We also compare our methods with DualPrompt$^\diamond$ with the same 0.55% extra memory on ALBEF as shown in Sec. 5.4.

5.3. Main Results

We summarize the main results in Tbl. 1 for the continual scenarios on CL-VQA 2.0 and CL-TDIUC.

Overall Performance The results indicate that the proposed TRIPLET significantly outperforms baseline methods across various settings, including those models using extra buffer and the two recently proposed prompt-based methods L2P$^\diamond$ and DualPrompt$^\diamond$, considering average accuracy and forgetting. We also find baseline methods' performances differ across various scenarios, demonstrating the importance of our proposed comprehensive formulation. Methods generally achieve higher average accuracy in CL-TDIUC than CL-VQA2.0, which is consistent with the i.i.d. accuracy in original splits [9, 14]. However, there is no obvious partial order relationship for the forgetting metric on the two splits, as forgetting is also related to the task-wise differences inside each scenario.

We also trace the first task's accuracy during different training stages (denoted as task ID) in Fig. 4, in ConLS settings. We could find that our method shows the best overall performance. Besides, as we formulate CL-VQA *w.r.t.* inputs, there exists some answer overlap between tasks, which would help the model recall previous knowledge and result in the accuracy ascent for all methods after the final task.

Findings Moreover, we observe some interesting findings for pre-trained vision-language model-based continual learning. In Continual Language Scenarios, rehearsal-based methods (DER, WA, and iCaRL) achieve much higher performance than exemplar-free methods (EWC and LwF). However, in Continual Vision Scenarios, they achieve comparable results, and this observation is consistent with the results in [29]. A possible explanation is that with pre-trained knowledge, Continual Language Scenarios, where tasks have significant different answer distributions from each other, is more difficult than Continual Vision Scenarios, where tasks have similar answer distributions. Another phenomenon is that the larger size of the buffer offers little help for performance. This is because VQA datasets usually contain high-dimensional and long-tailed answer labels, and it is difficult to select representative replay examples with the existing strategies.

Table 2: Ablation study for position of prompts on ConLS VQA2.0. E means E-Prompts and G means G-Prompts, the numbers in $[\cdot]$ means layers to attach prompts.

Prompt Position	Avg. Acc (↑)	Forgetting (↓)
E: [2,3,4], G: [0,1,2]	55.75	10.72
E: [2,3,4,5], G: [0,1,2]	**56.32**	**10.37**
E: [0,1,2,3,4,5], G: [0,1,2,3,4,5]	54.59	12.32

Table 3: Ablation Study of Modality (M) Interaction Strategy and Task (T) Interaction Strategy for three scenarios on two benchmarks. **Bold**: best results.

Scenario	M & T Interaction	CL-TDIUC Avg. Acc (↑)	CL-TDIUC FGT (↓)	CL-VQA2.0 Avg. Acc (↑)	CL-VQA2.0 FGT (↓)
ConLS	✗	70.26	2.05	56.32	10.37
	✓	**70.80**	**1.64**	**56.76**	**9.66**
ConVS	✗	80.27	0.40	59.27	1.02
	✓	**80.47**	**0.15**	**59.41**	**0.12**
ConVLS	✗	82.94	0.59	60.05	4.43
	✓	**83.06**	**0.54**	**60.53**	**4.08**

5.4. Ablation Study

We conduct first four ablation studies based on the ALBEF backbone for a more in-depth understanding of the proposed TRIPLET method.

The Effectiveness of Selective Deep Decoupling We learn from [39]'s empirical results that the prompts work better in the first six layers. In ALBEF, the visual encoder has 12 layers, and fusion and textual encoders have 6 layers. Thus, we conduct an ablation study on the ConLS CL-VQA2.0 to search best layers. As shown in Tbl. 2, we find the best performance to add E-Prompt from layers 2 to 5 and G-Prompt from layers 0 to 2. The highest performance in the second row demonstrates the effectiveness of Selective Deep Decoupling.

The Effectiveness of Prompt Interactions As shown in Tbl. 3, our performance stably improves with prompt interaction strategies in all scenarios. We also conduct additional experiments for different components of prompt modality and task interaction strategies in Tbl. 4. Improved perfor-

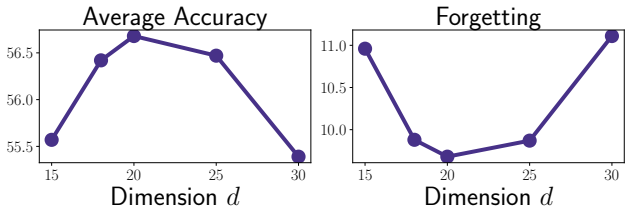

Figure 5: Exploration of Dimension d in Sec. 4.2.2.

Table 4: Ablation Study for Exploration of Modality-wise and Task-wise Prompt Interactions. MI: Modality-Interaction, TI on G/E-Prompt: Task-Interaction (See Eq. (8)) for G/E-Prompt.

MI	TI on G-Prompt	TI on E-Prompt	Avg. Acc (↑)	Forgetting (↓)
			56.32	10.37
✓			56.63	9.87
✓	✓		56.30	10.02
✓		✓	56.53	9.80
✓	✓	✓	**56.76**	**9.66**

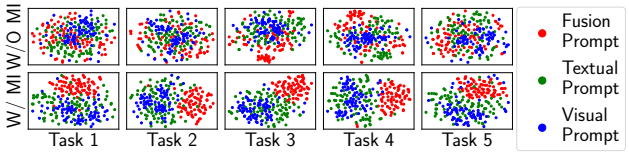

Figure 6: Visualization of decoupled prompts w/o and w/ Modality-Interaction(MI) by t-SNE.

mance between the first two rows shows that mutual propagation between different modalities helps the alignment of decoupled prompts in modality aspect. The results in the last three rows show that it is important to keep the invariant prompt modality-interaction structure between different tasks for both G and E-prompts in selective deep layers.

Exploration of Modality-Interaction Matrix We explore the dimension d on ConVS CL-VQA2.0 to explore the best hyperparameter for the proposed Modality-Interaction Strategy (See Sec. 4.2.2) as shown in Fig. 5. With the increasing dimension d, the performance first increases and then decreases with the peak performance at dimension $d = 20$. Interestingly, the best dimension d remains stable across different scenarios. Compared with dimension d, another two hyperparameters λ_2 and λ_3 have less influence on the model performance. We set $\lambda_2 = 0.2$ and $\lambda_3 = 0.05$ across different scenarios. As shown in Figure 6, we also visualize decoupled prompts for 5 tasks after the final training stage, w/o and w/ Modality-Interaction(MI) by t-SNE, further verifying that prompts within different modalities become more clustered. The above two ablation studies demonstrate the effectiveness of explicitly modeling the complex multi-modal interactions.

Exploration of Extra Memory We set $L_G/L_E = 20/35$ for visual and textual prompts in DualPrompt$^\diamond$ [39] to make a fair comparison with TRIPLET in extra memory. We choose to conduct experiments on ConVLS TDIUC when DualPrompt$^\diamond$ achieves its best time (See Tbl. 1). From

Table 5: Results for exploration for extra memory.

Method	Extra Memory	Avg. Acc (↑)	FGT (↓)
DualPrompt$^\diamond$ [39]	0.31%	81.36	2.31
DualPrompt$^\diamond$ [39]	0.55%	80.41	3.68
Ours	0.55%	**83.06**	**0.54**

Table 6: Results for Continual Language Scenario built upon **FLAVA** [36]. For details about the meaning of the fonts and notations, see Tbl. 1.

Method	Buffer Size	Average Acc CL-VQA2.0	Average Acc CL-TDIUC
DER [42]	5000	41.66†	44.91
WA [46]		33.02	66.27‡
iCaRL [31, 27]		34.14	64.59
L2P$^\diamond$ [40]	0	<u>36.98</u>	<u>27.21</u>
DualPrompt$^\diamond$ [39]		23.65	25.99
Ours		**44.00**	**64.86**
Upper-bound	-	64.14	75.08

Tbl. 5, we find DualPrompt$^\diamond$ performs worse with more extra memory, as the previous chosen hyperparameters have the best performance reported in [39].

Exploration of Different Backbones In order to explore the impact of different backbones and demonstrate the stability of our proposed TRIPLET method, we conduct extensive experiments based on FLAVA [36]. We select the baselines with the top performance across different settings, namely WA, iCaRL, and DER with 5000 buffer size and DualPrompt$^\diamond$. We also select L2P$^\diamond$ as it belongs to the prompt learning-based category as ours. As shown in Tbl. 6, our method consistently outperforms exemplar-free baselines, and achieves comparable results with rehearsal-based baselines. Generally, ALBEF-based results are higher than FLAVA-based results, which may be partially due to different fusion structures, thus being consistent with [45].

6. Conclusions

In this paper, we are the first to propose a comprehensive formulation for CL-VQA to conduct extensive multimodal continual evaluations. Based on our formulation, we further propose TRIPLET, the first multimodal prompt learning-based continual model for CL-VQA, which achieves state-of-the-art results across various settings in the experiments.

7. Acknowledgment

This work was supported in part by the National Key Research and Development Program of China No. 2020AAA0106300, National Natural Science Foundation of China (No. 62222209, 62250008, 62102222), Beijing National Research Center for Information Science and Technology Grant No. BNR2023RC01003, BNR2023TD03006, and Beijing Key Lab of Networked Multimedia.

References

[1] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European conference on computer vision (ECCV)*, pages 139–154, 2018. 2

[2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018. 1, 2, 3

[3] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020. 2

[4] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *Proceedings of the IEEE/CVF International conference on computer vision*, pages 9516–9525, 2021. 2

[5] Riccardo Del Chiaro, Bartłomiej Twardowski, Andrew Bagdanov, and Joost Van De Weijer. Ratt: Recurrent attention to transient tasks for continual image captioning. *Advances in Neural Information Processing Systems*, 33:16736–16748, 2020. 5

[6] Zi-Yi Dou, Yichong Xu, Zhe Gan, Jianfeng Wang, Shuohang Wang, Lijuan Wang, Chenguang Zhu, Pengchuan Zhang, Lu Yuan, Nanyun Peng, et al. An empirical study of training end-to-end vision-and-language transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18166–18176, 2022. 3

[7] Arthur Douillard, Alexandre Ramé, Guillaume Couairon, and Matthieu Cord. Dytox: Transformers for continual learning with dynamic token expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9285–9295, 2022. 2

[8] Yizhao Gao, Nanyi Fei, Haoyu Lu, Zhiwu Lu, Hao Jiang, Yijie Li, and Zhao Cao. Bmu-moco: Bidirectional momentum update for continual video-language modeling. In *Advances in Neural Information Processing Systems*, 2022. 2

[9] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017. 2, 5, 7, 14

[10] Claudio Greco, Barbara Plank, Raquel Fernández, and Raffaella Bernardi. Psycholinguistics meets continual learning: Measuring catastrophic forgetting in visual question answering. *arXiv preprint arXiv:1906.04229*, 2019. 2

[11] Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 804–813, 2017. 1, 2, 3, 11

[12] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIII*, pages 709–727. Springer, 2022. 3

[13] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017. 2

[14] Kushal Kafle and Christopher Kanan. An analysis of visual question answering algorithms. In *Proceedings of the IEEE international conference on computer vision*, pages 1965–1973, 2017. 2, 5, 7, 13

[15] Zixuan Ke, Bing Liu, and Xingchang Huang. Continual learning of a mixed sequence of similar and dissimilar tasks. *Advances in neural information processing systems*, 33:18493–18504, 2020. 2

[16] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. *arXiv preprint arXiv:2210.03117*, 2022. 5

[17] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 2, 6, 11

[18] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123:32–73, 2017. 2, 6, 13

[19] Lillian Lee. On the effectiveness of the skew divergence for statistical language analysis. In *International Workshop on Artificial Intelligence and Statistics*, pages 176–183. PMLR, 2001. 14

[20] Stan Weixian Lei, Difei Gao, Jay Zhangjie Wu, Yuxuan Wang, Wei Liu, Mengmi Zhang, and Mike Zheng Shou. Symbolic replay: Scene graph as prompt for continual learning on vqa task. *arXiv preprint arXiv:2208.12037*, 2022. 1, 2, 3, 11

[21] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in neural information processing systems*, 34:9694–9705, 2021. 3, 6, 7, 11, 13

[22] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, pages 3925–3934. PMLR, 2019. 2

[23] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 1, 2, 6

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 5, 6, 13, 14

[25] Noel Loo, Siddharth Swaroop, and Richard E Turner. Generalized variational continual learning. *arXiv preprint arXiv:2011.12328*, 2020. 2

[26] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. *Advances in neural information processing systems*, 29, 2016. 1, 2, 3

[27] Francesco Marra, Cristiano Saltori, Giulia Boato, and Luisa Verdoliva. Incremental learning for the detection and classification of gan-generated images. In *2019 IEEE international workshop on information forensics and security (WIFS)*, pages 1–6. IEEE, 2019. 6, 8, 12

[28] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 1

[29] Mavina Nikandrou, Lu Yu, Alessandro Suglia, Ioannis Konstas, and Verena Rieser. Task formulation matters when learning continually: A case study in visual question answering. *arXiv preprint arXiv:2210.00044*, 2022. 1, 2, 3, 6, 7, 13, 14

[30] Chengwei Qin and Shafiq Joty. Lfpt5: A unified framework for lifelong few-shot language learning based on prompt tuning of t5. *arXiv preprint arXiv:2110.07298*, 2021. 2

[31] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010, 2017. 2, 6, 8, 11, 12

[32] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019. 14

[33] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[34] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 2

[35] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *Advances in neural information processing systems*, 30, 2017. 2

[36] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15638–15650, 2022. 3, 6, 7, 8, 11, 12

[37] Damien Teney, Lingqiao Liu, and Anton van Den Hengel. Graph-structured representations for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2017. 1, 2, 3

[38] Yabin Wang, Zhiwu Huang, and Xiaopeng Hong. S-prompts learning with pre-trained transformers: An occam's razor for domain incremental learning. *arXiv preprint arXiv:2207.12819*, 2022. 2, 6

[39] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, et al. Dualprompt: Complementary prompting for rehearsal-free continual learning. *arXiv preprint arXiv:2204.04799*, 2022. 2, 3, 4, 5, 6, 7, 8, 11, 12

[40] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149, 2022. 2, 3, 5, 6, 8, 11, 12

[41] Shipeng Yan, Lanqing Hong, Hang Xu, Jianhua Han, Tinne Tuytelaars, Zhenguo Li, and Xuming He. Generative negative text replay for continual vision-language pretraining. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVI*, pages 22–38. Springer, 2022. 2

[42] Shipeng Yan, Jiangwei Xie, and Xuming He. Der: Dynamically expandable representation for class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2021. 2, 6, 8, 11, 12

[43] Yang Yang, Da-Wei Zhou, De-Chuan Zhan, Hui Xiong, Yuan Jiang, and Jian Yang. Cost-effective incremental deep model: Matching model capacity with the least sampling. *IEEE Transactions on Knowledge and Data Engineering*, 2021. 2

[44] Jaehong Yoon, Eunho Yang, Jeongtae Lee, and Sung Ju Hwang. Lifelong learning with dynamically expandable networks. *arXiv preprint arXiv:1708.01547*, 2017. 2

[45] Yao Zhang, Haokun Chen, Ahmed Frikha, Yezi Yang, Denis Krompass, Gengyuan Zhang, Jindong Gu, and Volker Tresp. Cl-crossvqa: A continual learning benchmark for cross-domain visual question answering. *arXiv preprint arXiv:2211.10567*, 2022. 1, 2, 8

[46] Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13208–13217, 2020. 6, 8, 11, 12

[47] Tingting Zhao, Zifeng Wang, Aria Masoomi, and Jennifer Dy. Deep bayesian unsupervised lifelong learning. *Neural Networks*, 149:95–106, 2022. 2

[48] Da-Wei Zhou, Fu-Yun Wang, Han-Jia Ye, and De-Chuan Zhan. Pycil: A python toolbox for class-incremental learning, 2021. 11

[49] Da-Wei Zhou, Qi-Wei Wang, Zhi-Hong Qi, Han-Jia Ye, De-Chuan Zhan, and Ziwei Liu. Deep class-incremental learning: A survey. *arXiv preprint arXiv:2302.03648*, 2023. 2, 3, 6

Appendix of Decouple Before Interact: Multi-Modal Prompt Learning for Continual Visual Question Answering

A. More Experimental Details

A.1. Experimental Results on CLOVE [20]

To demonstrate the effectiveness of our proposed TRIPLET, we conduct experiments on the public benchmark CLOVE [20] and report average accuracy in the table bellow. We select the baselines with top performance and report previous SOTA [20], which needs extra scene graphs for training. Our TRIPLET outperforms previous SOTA, as well as baselines with the same backbone, substantiating the effectiveness of our TRIPLET.

Table 7: DER, WA and iCaRL are rehearsal-based methods with 5000 buffer size.

Methods	DER	WA	iCaRL	L2P$^\diamond$	DualPrompt$^\diamond$	**TRIPLET**	Previous SOTA
CLOVE-scene	35.08	36.48	37.24	32.54	43.18	**47.73**	36.91
CLOVE-function	31.95	38.56	37.22	22.47	49.12	**51.01**	45.97

A.2. Exploration of the Task Orders on Forgetting Issues

We randomly sample 4 task orders from all possible permutations on ConLS CL-VQA2.0 for evaluation. We select the baselines with top performance and report the results (Average±Standard Deviation) over these four task orders in the table bellow. The results illustrate that task orders do impact performance and some orders may lead to larger forgetting, while TRIPLET outperforms the baselines and exhibits the lowest standard deviation, which demonstrates its robustness to variations in task orders.

Table 8: DER, WA and iCaRL are rehearsal-based methods with 5000 buffer size.

Methods	DER	WA	iCaRL	L2P$^\diamond$	DualPrompt$^\diamond$	**TRIPLET**
Average Accuracy (↑)	54.77±2.68	54.56±3.61	55.72±3.80	35.19±7.98	44.31±8.29	**57.37±1.37**
Forgetting (↓)	9.84±3.72	10.11±3.77	9.91±3.91	25.11±6.07	17.11±10.58	**6.82±2.79**

A.3. Implementation of Baseline Methods

We extend Class Incremental Learning (CIL) framework [48]'s scope to the multi-modal domain for the implementation of None-prompt learning baselines (*i.e.*, iCaRL [31], DER [42], WA [46], EWC [17] and LwF [11]). Moreover, we use the same backbone as ours (*i.e.*, ALBEF [21] and FLAVA [36]). For rehearsal-based methods (*i.e.*, iCaRL, DER, and WA), we also enhance the rehearsal selection strategies. To be specific, we select the examples closest to the center using fusion features. When the total number of answer labels is larger than the buffer size, we randomly select as many answer labels as the buffer size to store examples. As for the prompt learning-based method (*i.e.*, L2P [40] and DualPrompt [39]), we symmetrically add textual key-prompt pairs to enhance model performance for CL-VQA, which we denoted as L2P$^\diamond$ and DualPrompt$^\diamond$. Detailed structures of L2P$^\diamond$, DualPrompt$^\diamond$ and our proposed TRIPLET are shown in Fig. 7.[5]

A.4. Exploration of Backbone FLAVA

To explore the impact of different backbones and demonstrate the stability of our proposed TRIPLET methods, we additionally conduct experiments based on FLAVA [36]. As we have introduced in the main paper, we select the baselines with the top performance across different settings, namely WA, iCaRL, and DER with 5000 buffer size and DualPrompt$^\diamond$. We also select L2P$^\diamond$ as it belongs to the prompt learning-based category as ours. We use two evaluation metrics (*i.e.*, Average Accuracy and Forgetting), and regard Average Accuracy as our main evaluation metric as it reflects both the greater learning capacity and less catastrophic forgetting [39].

Performance and Findings As the results are shown in Table 9 and Table 10, our method consistently outperforms exemplar-free baselines and achieves comparable results with rehearsal-based baselines. We also find baselines and our method have

[5]Prompt learning-based methods are built upon the public repository https://github.com/JH-LEE-KR/l2p-pytorch

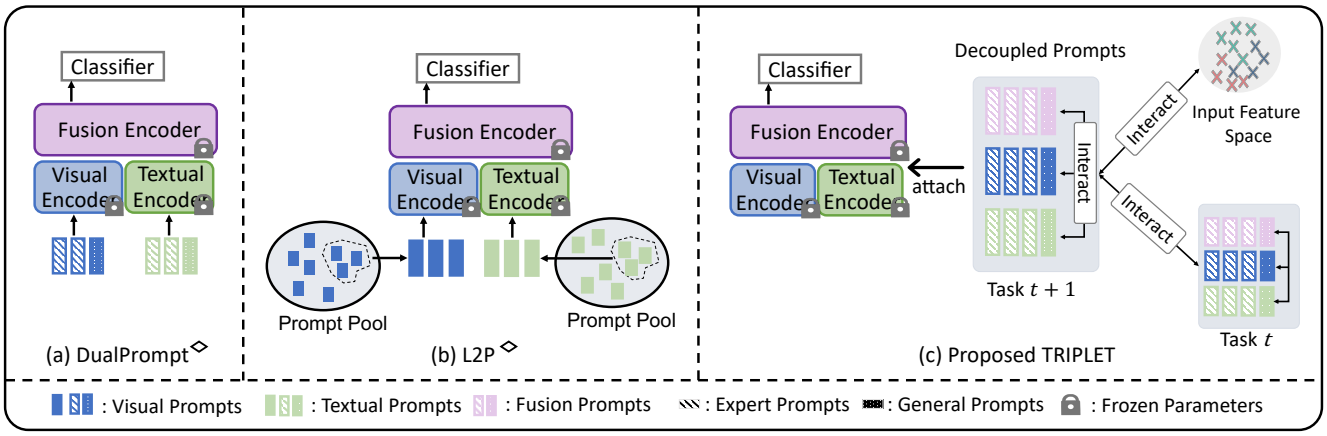

Figure 7: Modified multi-modal DualPrompt◇ (a) and L2P◇ (b). Comparison with our proposed TRIPLET (c).

Table 9: Results for the CL-VQA2.0 built upon **FLAVA** [36]. **Bold**: best exemplar-free CL-VQA results, Underline: second best exemplar-free CL-VQA results, †: best rehearsal-based CL-VQA results, ‡: rehearsal-based results which outperform the best exemplar-free results, Upper-bound: supervised fine-tuning on the i.i.d. data of each task, ◇: enhanced methods as discussed in Sec. 5.2, Avg. Acc: average accuracy, FGT: forgetting., Grey Color: results with backbone **ALBEF**.

Method	Buffer Size	ConLS		ConVS		ConVLS	
		Avg. Acc (↑)	FGT (↓)	Avg. Acc (↑)	FGT (↓)	Avg. Acc (↑)	FGT (↓)
DER [42]	5000	41.66† / 53.39	22.86‡ / 13.35	31.77† / 52.34	6.68 / 4.61	28.66† / 58.78	14.28 / 3.86
WA [46]		33.02 / 53.91	32.77 / 13.51	31.11 / 55.89	5.51 / 1.95	24.96 / 58.75	11.02 / 3.96
iCaRL [31, 27]		34.14 / 53.42	32.65 / 14.09	29.42 / 54.72	4.03† / 0.59	23.11 / 58.58	9.56† / 4.06
L2P◇ [40]	0	36.98 / 41.38	37.02 / 25.80	34.99 / 41.55	2.87 / 3.86	25.84 / 32.43	41.58 / 27.25
DualPrompt◇ [39]		23.65 / 44.26	45.57 / 24.16	37.66 / 56.56	19.86 / 1.68	20.32 / 41.30	43.17 / 21.37
Ours		**44.00** / 56.76	**27.51** / 9.66	**42.42** / 59.41	**0.12** / 0.12	**55.05** / 60.53	**7.02** / 4.08
Upper-bound	-	64.14 / 64.53	-	52.34 / 59.62	-	61.68 / 64.08	-

Table 10: Results for the CL-TDIUC built upon **FLAVA** [36]. **Bold**: best exemplar-free CL-VQA results, Underline: second best exemplar-free CL-VQA results, †: best rehearsal-based CL-VQA results, ‡: rehearsal-based results which outperform the best exemplar-free results, Upper-bound: supervised fine-tuning on the i.i.d. data of each task, ◇: enhanced methods as discussed in Sec. 5.2, Avg. Acc: average accuracy, FGT: forgetting, Grey Color: results with backbone **ALBEF**.

Method	Buffer Size	ConLS		ConVS		ConVLS	
		Avg. Acc (↑)	FGT (↓)	Avg. Acc (↑)	FGT (↓)	Avg. Acc (↑)	FGT (↓)
DER [42]	5000	44.91 / 63.71	19.06 / 7.95	61.20† / 75.18	10.27† / 6.96	49.53 / 71.63	22.96 / 13.42
WA [46]		66.27‡ / 69.23	11.67† / 4.49	56.98 / 75.84	13.55 / 4.39	54.97† / 77.51	20.26† / 4.68
iCaRL [31, 27]		64.59 / 67.94	11.72 / 5.46	56.55 / 75.78	13.52 / 4.75	49.17 / 74.98	23.33 / 7.41
L2P◇ [40]	0	27.21 / 33.95	50.99 / 29.21	52.80 / 75.51	3.29 / 0.60	67.44 / 69.18	12.34 / 16.65
DualPrompt◇ [39]		25.98 / 44.50	47.35 / 14.70	69.95 / 77.38	15.12 / 3.93	65.31 / 81.36	23.50 / 2.31
Ours		**64.86** / 70.80	**10.82** / 1.64	**75.24** / 80.47	**0.12** / 0.15	**70.01** / 83.06	**1.65** / 0.54
Upper-bound	-	75.08 / 74.60	-	80.92 / 80.57	-	82.56 / 83.33	-

different performances with different backbones (*i.e.*, FLAVA and ALBEF). Firstly, all the methods (including our proposed TRIPLET) achieve better performances based on ALBEF than FLAVA. Secondly, the performances of baselines vary across different backbones. For example, based on FLAVA, L2P◇ generally achieves higher performance than DualPrompt◇, while DualPrompt◇ generally achieves higher performance when based on ALBEF. These observations may be partially due to their different fusion structures. However, our proposed TRIPLET consistently outperforms baselines across various backbones, demonstrating our effectiveness.

A.5. Experimental Details

We list our experimental details based on ALBEF in Table 11. We set different learning rates for two types of methods, as none-prompt learning methods train with the backbone, while prompt learning-based methods freeze the backbone. Therefore, we follow [21] to set a smaller learning rate for the former ones. For training epochs, we train all the models until they converge.

Table 11: Training details based on ALBEF for baselines and our proposed TRIPLET.

Method	Batch Size	Learning Rate	Training Epoch	Other Hyper Parameters	Resource
EWC	16 for CL-VQA2.0 64 for CL-TDIUC	8e-5	120 for the first task 80 for the other tasks	$\lambda = 1000$	8×A100 GPUs 80G
LwF				$\lambda = 1$	
iCaRL				-	
DER				-	
WA				-	
L2P$^\diamond$	16 for CL-VQA2.0 64 for CL-TDIUC	4e-4	50	$\{N = 30, top_k = 5\}$	4×A100 GPUs 80G for CL-VQA2.0 8×A100 GPUs 80G for CL-TDIUC
DualPrompt$^\diamond$				$\{L_G = 5, L_E = 20\}$	
Ours				$\{L_G = 5, L_E = 20, d = 20,$ $\lambda_1 = 0.1, \lambda_2 = 0.2, \lambda_3 = 0.05\}$	

B. Details of the Benchmarks

B.1. How to Split Original Datasets?

In this section, we discuss in detail how to generate the continual vision scenario (ConVS), continual language scenario (ConLS), and continual vision-language scenario (ConVLS) in CL-VQA2.0 and CL-TDIUC, respectively, according to the definition in the main paper. We first introduce the common ideas for designing these scenarios, and then introduce designing details for two benchmarks which are personalized according to the various datasets' annotations. For ConVS, we explicitly consider the changes in vision distribution based on visual object categories contained in images [29], and different tasks include images containing different object categories. For ConLS, we explicitly consider the changes in question distribution based on question types, and different tasks include questions belonging to different question types. For ConVLS, we explicitly consider the changes in both vision and question distribution based on visual object categories contained in images and question types. In different tasks, images contain different object categories and question types also change.

CL-TDIUC CL-TDIUC benchmark is built upon TDIUC [14], which contains images from Visual Genome [18] and MSCOCO [24], and questions generated from both image annotations and human annotators. Besides question-answer pairs, annotations include question types (*e.g.*, object presence and scene recognition.), and object categories from Visual Genome [18] and MSCOCO [24].

- **Continual Vision Scenario (ConVS)**. According to the object category annotations provided by MSCOCO and Visual Genome, we divide images into five hyper-categories according to the object categories they contain: indoor activity, outdoor activity, animal, food, and traffic. We list 10 representative object categories in each hyper-category as shown in Table 12, and an example image in each hyper-category as shown in Fig. 8. After splitting images into five tasks, we collect corresponding questions and answers to form sequential VQA tasks.
- **Continual Language Scenario (ConLS)**. As TDIUC has annotations about question types, we select five question types, *i.e.*, Query-Color, Scene-Recognition, Object-Recognition, Counting, and Positional-Reasoning, to make the total number of tasks consistent with that of ConVS. We list 2 representative questions for each question type as shown in Table 13. After splitting questions into five tasks, we collect corresponding images and answers to form sequential VQA tasks.
- **Continual Vision-Language Scenario (ConVLS).** As we have categorized images into five hyper-categories as discussed in ConVS, we then sequentially select corresponding questions of certain question types. For example, we select images in indoor activity, and select corresponding questions of certain question types (*i.e.*, sentiment-understanding, attribute, positional-reasoning, utility-affordance, sport-recognition and object-recognition) to form task 1. Detailed information about selected images and question types for each task is shown in Table 14.

Detailed statistics for CL-TDIUC are listed in Table 15. We then draw the word cloud figures for the top-20 answers based on the answer frequency as shown in Fig. 9, 10, and 11. We can find ConLS and ConVLS have rather different answer distributions across different tasks, while ConVS have much more similar answer distributions across different tasks. This is

Table 12: Representative object categories in each hyper-category for ConVS in CL-TDIUC.

Task	Representative Object Categories
Indoor Activity	backpack, umbrella, laptop, bookshelf, table, couch, computer monitor, shelf, bed, cell phone
Food	kitchen, bottle, fork, knife, bowl, food, sandwich, carrot, hot dog, pizza
Outdoor Activity	sports, skis, snowboard, sports ball, kite, baseball, skateboard, surfboard, tennis, racket
Traffic	traffic light, fire hydrant, stop sign, parking meter, bench, road, sidewalk, street, vehicle, motorcycle
Animal	animal, bird, cat, dog, horse, sheep, cow, elephant, bear, zebra

reasonable as tasks in ConVS usually share the same question types (*e.g.*, counting), though some answers vary according to the objects in the images (*e.g.*, *dining table* in Task 2 and *baseball* in Task 3.)

CL-VQA2.0 CL-VQA2.0 benchmark is built upon VQA2.0 [9], which is the most wildly used VQA dataset. VQA2.0 contains images from MSCOCO [24]. Besides question-answer pairs, annotations include coarse-grained object categories from MSCOCO [24].

- **Continual Vision Scenario (ConVS)** Similar to [29], we organize 80 object categories into five groups as shown in Table 16. Images with objects from multiple groups are discarded to create clean task splits. After splitting images into five tasks, we collect corresponding questions and answers to form sequential VQA tasks.
- **Continual Language Scenario (ConLS)** As there are no clear annotations for question types, we manually divide the question types into five categories, namely Counting, Query-Color, Action, Subcategory, and Query-Scene. In particular, Counting, Query-Color, and Subcategory are derived from [29], and then we cluster the rest question embedding from Sentence-BERT [32], forming the following two question types. We list 2 representative questions for each question type as shown in Table 17. After splitting questions into five tasks, we collect corresponding images and answers to form sequential VQA tasks.
- **Continual Vision-Language Scenario (ConVLS)**. As we have categorized images into five hyper-categories as discussed in ConVS, we then sequentially select corresponding questions of certain question types. For example, we select images from Group 1 and select corresponding questions of certain question types (i.e., query-color) to form task 1.

Detailed statistics for CL-VQA2.0 are listed in Table 18. Also, we draw the word cloud figures for the top-20 answers based on the answer frequency as shown in Fig. 12, 13, and 14.

B.2. What Factors may Matter the Difficulty of Different Scenarios?

From experimental results shown in Table 9 and Table 10, we find methods have different performance across various scenarios. On the one hand, it is because of the effectiveness of the methods, while on the other hand, it may due to different difficulties for different scenarios. In this section, we use divergence and similarity to analyze the difficulty of different scenarios.

As there exist partial none-overlapping answers for different tasks, we use skew divergence [19] between Task i and Task j in the same scenario to measure the answer divergence [29] as shown in Figure 15. We then calculate cosine similarity between tasks using question features, image features, and fusion features, respectively, which are shown in Figure 16. We observe that ConLSs have the highest image feature similarity between tasks, while ConVSs have the highest question feature similarity between tasks, demonstrating the rationality of our splitting benchmarks. CL-VQA aims to maintain past knowledge while learning new knowledge well, thus the more similar tasks and distribution will make the problem easier. Therefore, from the perspective of similarity and divergence, we find ConVS is easier than ConLS and ConVLS, which is consistent with our experimental results in Table 9 and Table 10 that methods perform better in ConVS than in other scenarios.

C. Algorithms for TRIPLET

The training and inference algorithms for TRIPLET are illustrated in Algorithm 1 and 2, respectively.

Algorithm 1 TRIPLET during training

Input: Pre-trained vision-language backbone with vision encoder VT, textual encoder TT and fusion encoder FT, classifiers $\{g^{\phi}\}_{t=1}^{T}$, number of tasks T, training data set $\{D_t = \{(\boldsymbol{v}_i^t, \boldsymbol{q}_i^t, y_i^t)\}_{i=1}^{n_t}\}_{t=1}^{T}$, decoupled prompts $P^{(\mathtt{m})} = \{G_k^{(\mathtt{m})}\} \bigcup \{E_{t,k}^{(\mathtt{m})}\}_{t=1}^{T}$, task keys $U^{(\mathtt{m})} = \{\boldsymbol{u}_t^{(\mathtt{m})}\}_{t=1}^{T}$, G-prompts attaching layer list K_G, E-prompts attaching layer list K_E, query function $f_{\mathtt{Q}}$, modality interaction function f_{mi} with interaction matrix $\boldsymbol{W}^{(\mathtt{m})} = \{\boldsymbol{W}_t^{(\mathtt{m})}\}_{t=1}^{T}$, task interaction function f_{ti}, number of training batches $\{M_t\}$, $\mathtt{m} \in \{v, q, f\}$
Initialize: $\phi, P^{(\mathtt{m})}, U^{(\mathtt{m})}, \boldsymbol{W}^{(\mathtt{m})}, \mathtt{m} \in \{v, q, f\}$
for $t = 1, \ldots, T$ **do**
    Select decoupled prompts $P_t^{(\mathtt{m})}$ and task keys $U_t^{(\mathtt{m})}$
    Attach $P_t^{(\mathtt{m})}$ to MSA layers in K_G and K_E to assemble the prompted architecture
    for $\mathtt{batch} = 1, \ldots, M_t$ **do**
        Draw a mini-batch $B = \{(v_i^t, q_i^t, y_i^t)\}_{i=1}^{l} \subset D_t$
        Calculate the batch loss $\mathcal{L}(B)$ as a stochastic proxy of $\mathcal{L}(D_t)$ via Equ. (9)
        Update $\phi, P^{(\mathtt{m})}, U^{(\mathtt{m})}, \boldsymbol{W}^{(\mathtt{m})}$ by backpropagation
        Store $\boldsymbol{W}_t^{(\mathtt{m})}$ for the next task's loss calculation
    end for
end for

Algorithm 2 TRIPLET during inference

Given components: Frozen pre-trained vision-language backbone with vision encoder VT, textual encoder TT and fusion encoder FT, trained classifiers $\{g^{\phi}\}_{t=1}^{T}$, number of task T, decoupled prompts $P^{(\mathtt{m})} = \{G_k^{(\mathtt{m})}\} \bigcup \{E_{t,k}^{(\mathtt{m})}\}_{t=1}^{T}$, task keys $U^{(\mathtt{m})} = \{\boldsymbol{u}_t^{(\mathtt{m})}\}_{t=1}^{T}$, G-prompts attaching layer list K_G, E-prompts attaching layer list K_E, $\mathtt{m} \in \{v, q, f\}$
Input: test example $(\boldsymbol{v}, \boldsymbol{q})$
Generate query features $\mathtt{Q}^{(\mathtt{m})}$ via Equ. (5)
Matching for the best indexes $t^{(m)} \in \{t^{(v)}, t^{(q)}, t^{(f)}\}$ via $\arg\max_{t^{(\mathtt{m})}} \gamma\left(\boldsymbol{u}_{t^{(\mathtt{m})}}^{(\mathtt{m})}, \mathtt{Q}^{(\mathtt{m})}\right)$
Select the corresponding task-specific prompts $E_{t^{(\mathtt{m})}}^{(\mathtt{m})}$, forming corresponding decoupled prompts $P_{t^{(\mathtt{m})}}^{(\mathtt{m})}$
Attach $P_t^{(\mathtt{m})}$ to MSA layers in K_G and K_E to assemble the prompted architecture
Select corresponding classifiers $g_{t^{(\mathtt{m})}}$ for prediction via Equ. (2)

Figure 8: Example image in each task of ConVS CL-TDIUC.

Figure 9: Word Cloud for top20 answers based on the answer frequency on CL-TDIUC ConLS.

Table 13: Representative questions in each task for ConLS in CL-TDIUC.

Task	Representative Questions
Query-Color	What color is the center of the train? What describing the color of the zebras mane what is it?
Scene-Recognition	Is this indoor or outdoor? What is the weather like at this ski resort?
Object-Recognition	What cutlery piece is in the photo? What vehicle is on the street next to the tree?
Counting	How many planes are in the picture? How many people are sitting down in this picture?
Positional-Reasoning	What is behind the in the picture suitcase? What fruit is to the right of the fruit with the face?

Table 14: Selected image categories and question types for ConVLS in CL-TDIUC.

Task	Selected Image Category	Selected Question Types
Group 1	Indoor Activity	Sentiment-Understanding, Attribute, Positional-Reasoning, Utility-Affordance Sport-Recognition, Object-Recognition
Group 2	Food	Attribute, Positional-Reasoning, Utility-Affordance, Sport-Recognition Object-Recognition, Scene-Recognition, Activity-Recognition
Group 3	Outdoor Activity	Positional-Reasoning, Utility-Affordance, Sport-Recognition, Object-Recognition Object-Presence, Scene-Recognition, Activity-Recognition
Group 4	Traffic	Utility-Affordance, Sport-Recognition, Object-Recognition, Counting Object-Presence, Scene-Recognition, Activity-Recognition
Group 5	Animal	Sport-Recognition, Object-Recognition, Counting, Color Object-Presence, Scene-Recognition, Activity-Recognition

Table 15: Detailed statistics for CL-TDIUC.

Scenario	Task	Train	Test	Classes
ConVS	Indoor Activity	99,276	48,279	671
	Food	127,766	61,319	668
	Outdoor Activity	66,760	31,458	513
	Traffic	116,964	56,143	750
	Animal	119,314	57,502	696
ConLS	Query-Color	133,074	62,490	16
	Scene-Recognition	44,674	22,032	71
	Object-Recognition	62,862	30,693	281
	Counting	111,857	52,905	16
	Positional-Reasoning	26,042	12,284	965
ConVLS	Group1	17,717	8,617	623
	Group2	35,421	17,037	646
	Group3	39,127	18,494	418
	Group4	44,478	21,785	176
	Group5	43,521	21,273	192

Table 16: Object categories in each hyper-category for ConVS in CL-VQA2.0.

Task	Object Categories
Group 1	carrot, fork, microwave, remote, dining table, stop sign, bus, pizza, mouse, suitcase, banana, skis, baseball bat, sink, skateboard, bed
Group 2	airplane, cake, motorcycle, umbrella, couch, tennis racket, oven, refrigerator, train, truck potted plant, car, boat, tv, toaster, zebra
Group 3	donut, cat, person, hair drier, surfboard, laptop, parking meter, bowl, bottle, vase cell phone, cup, snowboard, bird, elephant, traffic light
Group 4	sheep, bench, spoon, tie, backpack, kite, horse, toothbrush, sports ball, chair book, orange, cow, toilet, clock, sandwich
Group 5	frisbee, bear, broccoli, baseball glove, teddy bear, handbag, knife, scissors, apple, giraffe keyboard, fire hydrant, dog, wine glass, bicycle, hot dog

Table 17: Representative questions in each task for ConLS in CL-VQA2.0.

Task	Representative Questions
Counting	How many people are standing up in the boat? How many boats are in the water?
Query-Color	What color is the customer's shirt? What colors make the checkerboard pattern on his shirt?
Action	What does the man have over his head? What are these men wearing on their bodies?
Subcategory	What kind of juice is made with this fruit? What type of filling is in the dish on the bottom left?
Query-Scene	Are the lights on in the building? Is there a handicap sign near the smoker's area?

Table 18: Detailed statistics for CL-VQA2.0.

Scenario	Task	Train	Test	Classes
ConVS	Group1	10,093	4,962	1,548
	Group2	28,031	13,125	2,809
	Group3	38,875	19,914	3,281
	Group4	22,711	11,428	2,296
	Group5	17,846	8,507	1,633
ConLS	Counting	48,506	23,261	123
	Query-Color	43,166	21,559	572
	Subcategory	27,743	13,564	3,749
	Query-Action	41,653	19,712	500
	Query-Scene	62,952	30,310	100
ConVLS	Group1	8,871	4,406	63
	Group2	8,142	4,249	206
	Group3	5,552	2,745	1,004
	Group4	6,138	2,965	288
	Group5	6,687	3,248	57

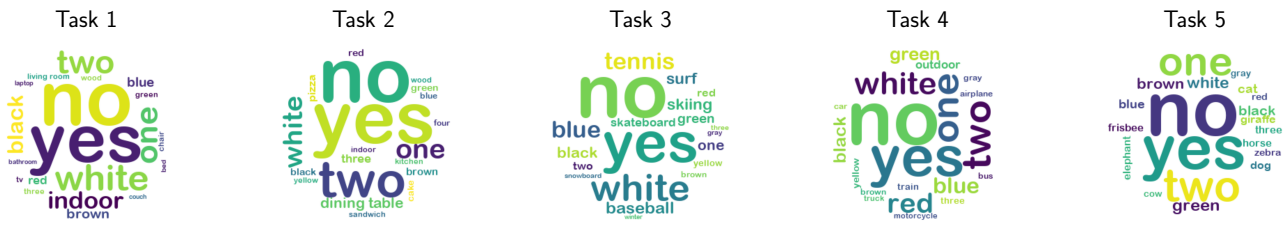

Figure 10: Word Cloud for top20 answers based on the answer frequency on CL-TDIUC ConVS.

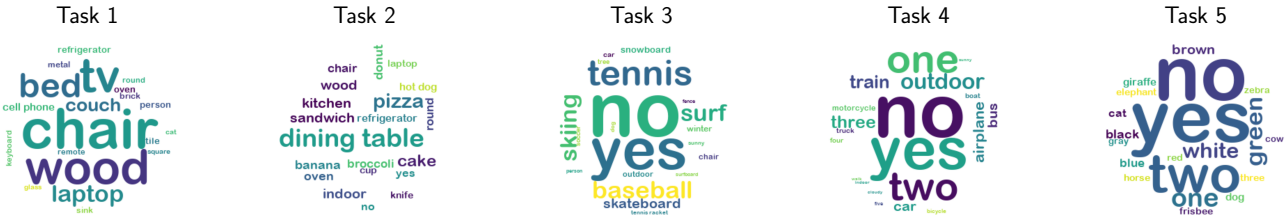

Figure 11: Word Cloud for top20 answers based on the answer frequency on CL-TDIUC ConVLS.

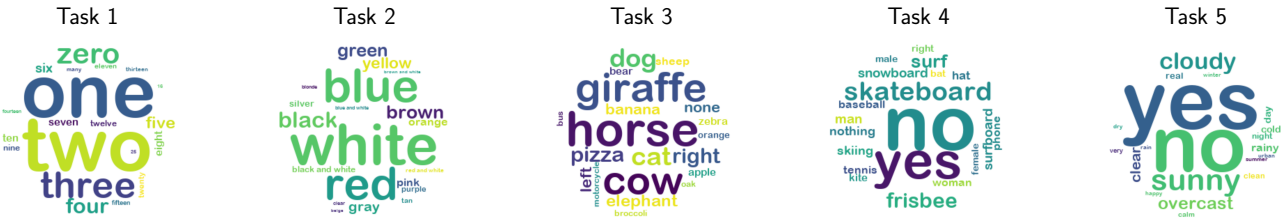

Figure 12: Word Cloud for top20 answers based on the answer frequency on CL-VQA2.0 ConLS.

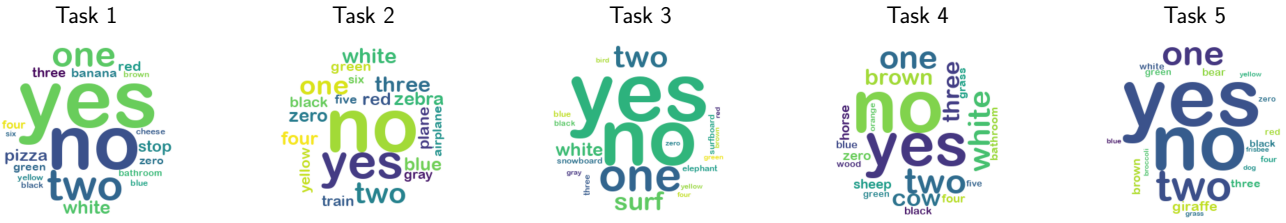

Figure 13: Word Cloud for top20 answers based on the answer frequency on CL-VQA2.0 ConVS.

Figure 14: Word Cloud for top20 answers based on the answer frequency on CL-VQA2.0 ConVLS.

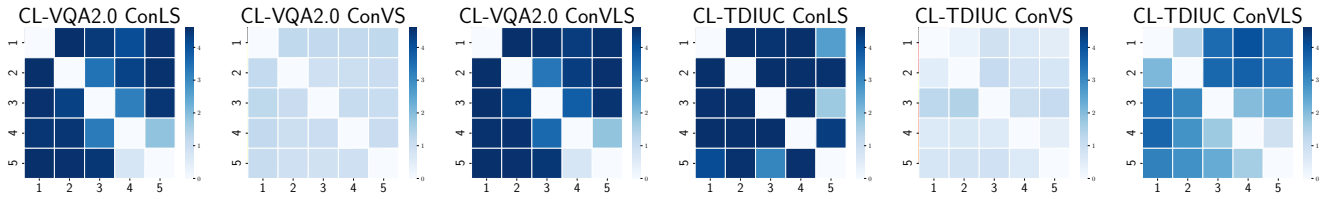

Figure 15: Divergence of answer distributions between different tasks for CL-VQA2.0 and CL-TDIUC. The darker the color, the greater the divergence.

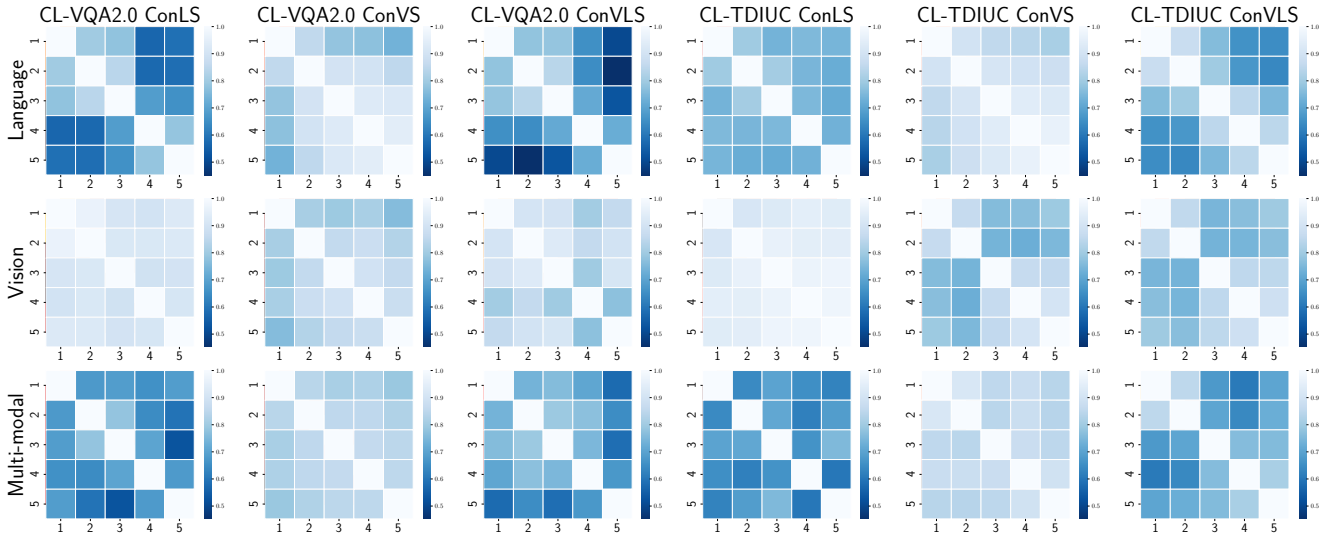

Figure 16: Task similarities for scenarios in CL-VQA2.0 and CL-TDIUC. The darker the color, the lower the similarity. The heatmaps on the top line use language features (question features) to calculate similarity, the heatmaps on the second line use vision features (image features) to calculate similarity, and the heatmaps on the bottom line use multi-modal features (fusion features) to calculate similarity.