Lifelong Knowledge Editing for Vision Language Models with Low-Rank Mixture-of-Experts

Qizhou Chen[1,3*], **Chengyu Wang**[2*], **Dakan Wang**[4], **Taolin Zhang**[5], **Wangyue Li**[1], **Xiaofeng He**[1†]
[1]East China Normal University, Shanghai, China
[2]Alibaba Cloud Computing, Hangzhou, China [3]Alibaba Group, Hangzhou, China
[4]Exacity Inc., Shanghai, China [5]Hefei University of Technology, Hefei, China
52265901009@stu.ecnu.edu.cn, chengyu.wcy@alibaba-inc.com, hexf@cs.ecnu.edu.cn

Abstract

Model editing aims to correct inaccurate knowledge, update outdated information, and incorporate new data into Large Language Models (LLMs) without the need for retraining. This task poses challenges in lifelong scenarios where edits must be continuously applied for real-world applications. While some editors demonstrate strong robustness for lifelong editing in pure LLMs, Vision LLMs (VLLMs), which incorporate an additional vision modality, are not directly adaptable to existing LLM editors. In this paper, we propose LiveEdit, a Lifelong vision language model Edit to bridge the gap between lifelong LLM editing and VLLMs. We begin by training an editing expert generator to independently produce low-rank experts for each editing instance, with the goal of correcting the relevant responses of the VLLM. A hard filtering mechanism is developed to utilize visual semantic knowledge, thereby coarsely eliminating visually irrelevant experts for input queries during the inference stage of the post-edited model. Finally, to integrate visually relevant experts, we introduce a soft routing mechanism based on textual semantic relevance to achieve multi-expert fusion. For evaluation, we establish a benchmark for lifelong VLLM editing. Extensive experiments demonstrate that LiveEdit offers significant advantages in lifelong VLLM editing scenarios. Further experiments validate the rationality and effectiveness of each module design in LiveEdit. [1]

1. Introduction

Large language models (LLMs) have become key techniques for text generation in NLP [1–3]. Benefiting from vision-language pre-training and pure LLMs, Vision-LLMs (VLLMs) are capable of generating text responses based on images and text [4–7]. However, outdated or erroneous built-in knowledge can undermine the value of these models. To avoid the costly retraining of large-scale parameters, model editing aims to adapt models by adjusting a small number of parameters to update specific knowledge. This plays a critical role in areas such as privacy protection [8, 9], detoxification [10, 11], bias reduction [12–14], and hallucination correction [15, 16].

Recent research on model editing techniques has primarily focused on pure LLMs [17–20]. However, the additional visual modality and the interactions between visual and textual modalities make these pure LLM editors less suitable. For example, LLM editors such as those in [17, 18, 21], which are based on locate-then-edit methods, assume that the subject in the query is crucial for model reasoning. These methods perform causal mediation analysis on input text queries containing the subject to identify linear layer weights critical to the LLM's reasoning. However, in vision-dominated tasks such as Visual Question Answering (VQA), where visual inputs often include substantial relevant information, attribution becomes more challenging. As a result, only limited work has explored how visual representations within VLLMs contribute to response generation and has proposed single-shot editing algorithms [22–24].

In most LLM applications, single-shot model editing is insufficient to keep the model updated. Thus, the concept of lifelong editing has emerged to address the continuous need for model updates [19–21, 25]. In lifelong editing scenarios, some LLM editors have demonstrated strong performance. Retrieval-based methods, in particular, avoid directly editing the original model and apply on-the-fly edit retrieval and parameter fusion during inference [19, 20, 25]. This approach offers greater robustness to a growing number of edits compared to editors that rely on permanent parameter modifications. For VLLMs, to the best of our knowledge,

*Q. Chen and C. Wang contributed equally to this work.
†Corresponding Author
[1]The source code is available at https://github.com/qizhou000/LiveEdit.

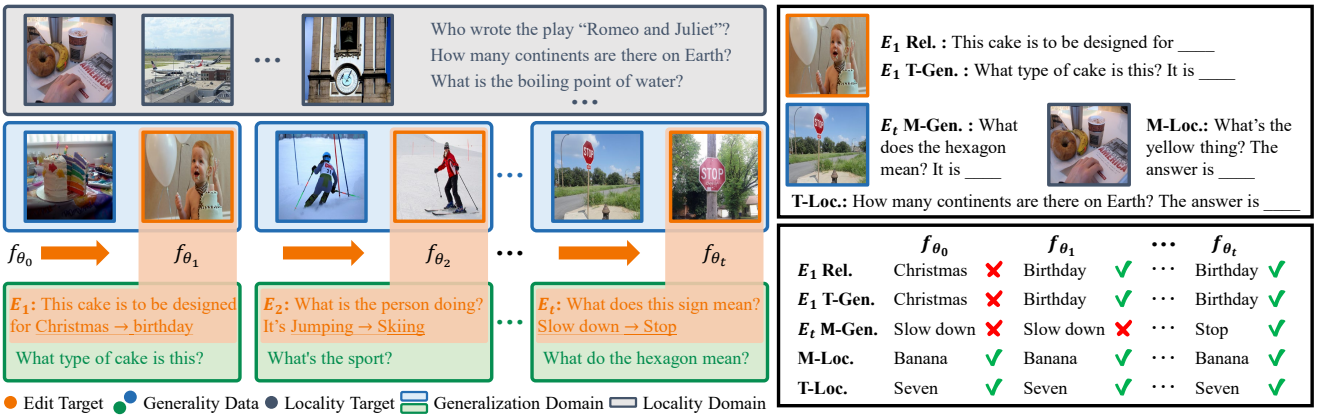

Figure 1. Lifelong VLLM Editing. In this scenario, the edited VLLM is required to correctly respond to queries involving the edited data within the generalization domain, while maintaining consistent responses in locality domains. The top left shows test cases where Rel., T-Gen./M-Gen., T-Loc., and M-Loc. denote reliability, text/modal generality, and text locality, respectively. The bottom right illustrates the responses of an effectively edited VLLM across several editing timesteps.

there is currently no related research on lifelong editing, as illustrated in Figure 1. Visual modality is quite different from text modality since it typically contains more information and is more noisy. Therefore, the approaches working for LLMs cannot be directly applied to VLLMs.

In this paper, we introduce *LiveEdit*, a novel framework designed for Lifelong vision language model Editing, which bridges the gap between lifelong LLM editing and VLLMs. In this framework, we design a generative low-rank mixture-of-experts combined with hard and soft routing as a powerful VLLM editor. The two key techniques are outlined as follows:

Generation of Low-Rank Experts: Mixture-of-Experts (MoE) combines multiple "experts," each specializing in specific data patterns or sub-tasks [26, 27]. In VLLM editing, we treat VLLM's adherence to a single edit sample as a sub-task, where each new edit sample corresponds to a low-rank expert that adjusts the model's response. MoE components are typically trained on sub-tasks, but directly fitting on individual edit samples results in poor generalization and inefficiency. To solve this, we propose an expert generator that creates low-rank experts for each new edit sample. The generator is trained to align VLLM with key edit metrics: reliability, textual/modal generality, and textual/modal locality [22]. The generated experts are stored in an expert repository. For new inputs, LiveEdit will select relevant experts from the repository and combines their adapted responses using hard and soft routing, as detailed below.

Hard and Soft Routing: We propose a two-stage routing strategy for expert utilization during inference. Attribution in [24] shows that VLLM processes prompts in early layers and extracts key visual features in later layers. In the first phase, we perform a text-to-vision interaction to extract key visual features and filter out noise. For each incoming sample, we compare its extracted features with those of edit samples, routing to visually relevant experts while filtering out visually irrelevant ones. Since this hard routing only considers visual semantics, it may select multiple visually matched but text-irrelevant experts. In the second phase, we apply soft routing through multi-expert fusion, incorporating the semantic similarity between the input query and the edit text. By combining absolute and relative weights, relevant edit samples are assigned higher weights and irrelevant ones lower weights. This approach suppresses text-irrelevant experts and avoids redundant interactions.

In the experiments, the proposed LiveEdit framework is tested with 1, 10, 100, and 1,000 edits on the LLaVA-V1.5 (7B) [28], MiniGPT-4 (7B) [29], and BLIP2-OPT (2.7B) [4] backbones across the E-VQA [22], E-IC [22], and VLKEB [30] benchmark datasets. Comparisons with other strong editors demonstrated the superiority of our approach.

2. Related Works

2.1. Vision Large Language Models

Motivated by recent achievements of LLMs [31], researchers have invested substantial effort in merging LLMs with vision models [32]. VLLMs synchronize pre-trained image encoders, usually a Vision Transformer (ViT) [33], with an LLM decoder. Consequently, this configuration produces a model proficient in handling images alongside text inputs [4–7, 34]. The training process for VLLMs typically unfolds in a two-phase approach. Initially, an alignment component, which may be a feed forward network [28] or more sophisticated structures such as a resampler [4, 35], is developed to bridge the image encoder with the LLM. This component is trained using pairs of images and their corresponding captions, effectively mapping image tokens onto the input space of the LLM. Subsequently, the

focus shifts to broad-spectrum inference capabilities. The model is then refined through exposure to a diverse array of tasks, encompassing visual question-answering scenarios [36, 37] and instruction-based interactions in both visual and textual contexts [38, 39], thereby enriching its functional versatility. However, despite the broad application potential mentioned above, VLLMs still rely on meticulous fine-tuning and editing to ensure adaptability and accuracy across diverse scenarios.

2.2. Model Editing

Model Editing for LLMs: We classify LLM editing into four categories. (1) **Locate-Then-Edit** methods identify and modify specific model parameters related to target knowledge [40]. ROME [17] uses causal mediation analysis for localization, while MEMIT [18] and WILKE [21] extend it for multi-editing. (2) **Meta Learning** methods employ a hyper-network to generate updated weights for edits [41–44]. (3) **Additional Parameters** methods introduce trainable parameters dedicated to edits while preserving original weights [45, 46]. Amon them, LEMOE [46] is based on MoE, but its greedy routing harms old experts' influence when integrating new ones. (4) **Adding Extra Modules** methods store and retrieve edits via external memory mechanisms [19, 25, 47–49]. In lifelong editing, the accumulation of shifts in the first two types hinders performance, while the latter two mitigate this by adding extra parameters and decoupling edits. However, in VLLM, extra modality and noise reduce efficacy.

Model Editing for VLLMs: Leveraging multimodal data for knowledge editing on VLLMs better resonates with practical contexts. Previously, the outlined methods were tailored for LLMs, operating solely on single-modal data. Yet, when it comes to knowledge editing on VLLMs, employing multimodal data offers a closer approximation to real-life settings. In the literature, MMEdit [22] and VLKEB [30] contribute novel datasets designed specifically for multimodal knowledge editing tasks. [24] use attribution analysis to explore how VLLMs extract key information from visual representations to generate responses, leading to the design of a single-step VLLM editing technique.

2.3. Mixture-of-Experts (MoE)

The MoE technique [26, 27] decomposes complex tasks into simpler ones, using dedicated models called experts. Recently, MoE layers have been integrated into transformer architectures. For instance, GShard [50] utilized MoE in transformer, achieving significant improvements in machine translation for 100 languages. Switch Transformers [51] further scaled language models with a trillion parameters through efficient MoE designs. However, naive MoE training may cause load imbalance, wherein a few experts are overused while others are underutilized. To combat this, various strategies such as the BASE layer [52], HASH layer [53], and Expert Choice [54] have been developed to optimize MoE models' capacity. Recent efforts focus on training a decoder-only MoE model with a modified UL2 objective [55]. Notably, Mixtral [56] enhances decision-making by employing token-choice routing to select two out of eight experts, improving overall performance. Our work approaches the MoE structure from a different angle, treating each knowledge update as a mini-task and leveraging low-rank experts to store knowledge for VLLM editing.

3. Preliminaries and Task Definition

We formally define the VLLM editing task and its lifelong extension. Next, we introduce the evaluation criteria.

A VLLM $f_\theta : \mathcal{V} \times \mathcal{P} \rightarrow \mathcal{O}$ can be considered a function that maps an image-prompt pair (v, p) to a textual output $o = f_\theta(v, p)$. Given an edit sample (v_e, p_e, o_e), where $f_\theta(v_e, p_e) \neq o_e$, a VLLM editor ME : $\mathcal{F} \times \mathcal{V} \times \mathcal{P} \times \mathcal{O} \rightarrow \mathcal{F}$ produces an updated VLLM $f_{\theta'} = \text{ME}(f_\theta, v_e, p_e, o_e)$. Starting from an initial VLLM f_{θ_0}, ME iteratively applies edits as new editing requirements arise in a lifelong context:

$$f_{\theta_t} = \text{ME}(f_{\theta_{t-1}}, v_{e_t}, p_{e_t}, o_{e_t}), t = 1, 2, 3, ...$$

At any timestep t, an effective ME should ensure that f_{θ_t} satisfies the following three criteria, as outlined in [22]:

Reliability measures the accuracy of the modified model's responses on edited samples:

$$\mathbb{E}_{(v_e, p_e, o_e) \sim \{(v_{e_\tau}, p_{e_\tau}, o_{e_\tau})\}_{\tau=1}^{t}} \mathbb{I}\{f_{\theta_t}(v_e, p_e) = o_e\}$$

where $\mathbb{I}$ is the indicator function that evaluates to 1 when the condition is true.

Generality requires f_{θ_t} can also adapt to relevant variations (e.g., rephrased prompts) in the edited samples, including modal and text generality:

$$\mathbb{E}_{(v_e, p_e, o_e) \sim \{(v_{e_\tau}, p_{e_\tau}, o_{e_\tau})\}_{\tau=1}^{t}} \mathbb{E}_{v_g \sim \mathcal{G}(v_e)} \mathbb{I}\{f_{\theta_t}(v_g, p_e) = o_e\}$$
$$\mathbb{E}_{(v_e, p_e, o_e) \sim \{(v_{e_\tau}, p_{e_\tau}, o_{e_\tau})\}_{\tau=1}^{t}} \mathbb{E}_{p_g \sim \mathcal{G}(p_e)} \mathbb{I}\{f_{\theta_t}(v_e, p_g) = o_e\}$$

where $\mathcal{G}(\cdot)$ represents the relevant neighbors.

Locality requires f_{θ_t} remains consistent with f_{θ_0} for samples unrelated to edits, including modal and text locality:

$$\mathbb{E}_{(v_e, p_e, o_e) \sim \{(v_{e_\tau}, p_{e_\tau}, o_{e_\tau})\}_{\tau=1}^{t}} \mathbb{E}_{(v_l, p_l) \sim \mathcal{L}(v_e, p_e)} \mathbb{I}_l(v_l, p_l)$$
$$\mathbb{E}_{(v_e, p_e, o_e) \sim \{(v_{e_\tau}, p_{e_\tau}, o_{e_\tau})\}_{\tau=1}^{t}} \mathbb{E}_{p_l \sim \mathcal{L}(p_e)} \mathbb{I}_l(\emptyset, p_l)$$
$$\text{s.t. } \mathbb{I}_l(v, p) = \mathbb{I}\{f_{\theta_t}(v, p) = f_{\theta_0}(v, p)\}$$

where $\mathcal{L}(\cdot)$ represents the irrelevant samples.

4. The Proposed LiveEdit Framework

In this section, we formally introduce the LiveEdit framework, with the overall architecture shown in Figure 2. First,

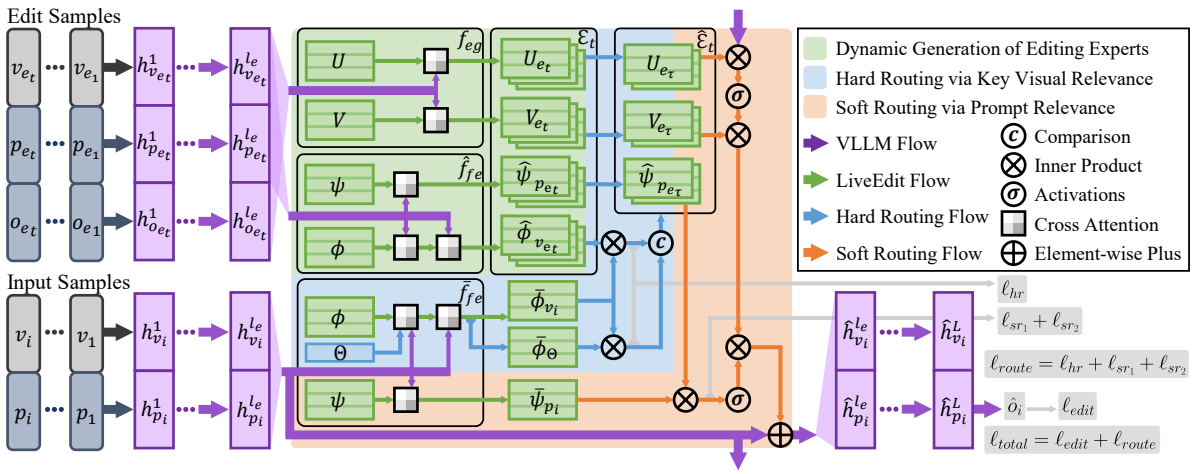

Figure 2. Illustration of the LiveEdit framework. The upper part illustrates the editing process of LiveEdit. At time step t, the representation of an edit sample $(v_{e_t}, p_{e_t}, o_{e_t})$ at layer l_e serves as an editing signal to generate the editing expert (U_{e_t}, V_{e_t}) via f_{eg} and routing features $(\hat{\phi}_{v_{e_t}}, \hat{\psi}_{p_{e_t}})$ via $\hat{f}_{fe}$. Both are then added to the expert repository $\mathcal{E}_t$. The lower part shows the VLLM inference process with LiveEdit, where $\bar{f}_{fe}$ extracts input sample features at layer l_e to route editing experts, which then adapt the representation.

we explain how to generate corresponding experts for edit samples and how to extract the semantic features of the editing prompt. We then describe their use in extracting key visual features from the visual representation, thereby maintaining the expert repository. Next, we describe how LiveEdit routes experts during a single inference of the VLLM to instantly adjust its response. Finally, we elaborate on the overall training process of the LiveEdit model.

4.1. Construction and Update of Expert Repository

Since the LLM transformer is the primary module for semantic understanding and response generation, in this work, we consider inserting the MoE editor between the layers of the transformer for editing. Previous work [24] on representation attribution in VLLM editing has shown that the latter layers in the model leverage the prompt semantics to extract relevant visual information to generate responses. Following their findings, we deploy our editor in a high-contribution layer l_e within the transformer. The expert repository is initially set to $\mathcal{E}_0 = \{\}$ and is updated from $\mathcal{E}_{t-1}$ to $\mathcal{E}_t$ at timestep t as a new editing sample $(v_{e_t}, p_{e_t}, o_{e_t})$ is input into the model.

Specifically, given a VLLM f_θ, the image and text of the edit sample are converted into embeddings, concatenated, and fed into the transformer. Let $h^{l_e} \in \mathbb{R}^{N\times d}$ represent the intermediate output at layer l_e, where N and d correspond to the sequence length and the intermediate dimension, respectively. Let $h^{l_e}_{v_{e_t}}$, $h^{l_e}_{p_{e_t}}$, $h^{l_e}_{o_{e_t}}$ denote the respective representations of v_{e_t}, p_{e_t}, and o_{e_t}. We define $f_{eg}(\cdot)$ to extract the editing signal and generate the expert:

$$\begin{aligned} (U_{e_t}, V_{e_t}) &= f_{eg}(h^{l_e}_{v_{e_t}} \oplus h^{l_e}_{p_{e_t}} \oplus h^{l_e}_{o_{e_t}}), \\ \text{s. t. } \; f_{eg}(h) &= (\mathrm{CA}_U(U,h), \mathrm{CA}_V(V,h)) \end{aligned} \tag{1}$$

where $U \in \mathbb{R}^{r\times d_m}$ and $V \in \mathbb{R}^{r\times d_m}$ are two trainable matrices. $\oplus$ denotes concatenation. r and d_m are hyperparameters representing the number of ranks and the module dimension, respectively. The cross attention $\mathrm{CA}(\cdot)$ is formulated as:

$$\mathrm{CA}(x,y) = \delta\left(xW_q\,(yW_k)^T\right)\cdot yW_v \tag{2}$$

where δ denotes softmax, and W_q, W_k, W_v are matrices that map inputs into query, key, and value spaces, respectively.

To perform both hard and soft routing for experts, we respectively extract key visual feature $\hat{\phi}_{v_{e_t}}$ using prompt semantics and the pure prompt feature $\hat{\psi}_{p_{e_t}}$ through a feature extractor $\hat{f}_{fe}(\cdot)$:

$$\begin{aligned} &(\hat{\phi}_{v_{e_t}}, \hat{\psi}_{p_{e_t}}) = \hat{f}_{fe}(h^{l_e}_{v_{e_t}}, h^{l_e}_{p_{e_t}}) \\ \text{s.t.} \quad &\hat{f}_{fe}(h_v, h_p) = (\mathrm{CA}_{\phi 2}\,(\mathrm{CA}_{\phi 1}\,(\phi, h_p), h_v), \mathrm{CA}_\psi\,(\psi, h_p)) \end{aligned} \tag{3}$$

where $\phi \in \mathbb{R}^{1\times k d_m}$ and $\psi \in \mathbb{R}^{1\times k d_m}$ are trainable feature extraction vectors, and k controls the dimension of the vectors. The extracted features $\hat{\phi}_{v_{e_t}}$ and $\hat{\psi}_{p_{e_t}}$ have the same shape as ϕ and ψ. Finally, the expert repository is updated by inserting the group of experts and the routing features as $\mathcal{E}_t = \mathcal{E}_{t-1} \cup \{(U_{e_t}, V_{e_t}, \hat{\phi}_{v_{e_t}}, \hat{\psi}_{p_{e_t}})\}$.

4.2. Expert Routing and Editing on the Fly

Given an input image-prompt pair (v_i, p_i), let its output at the l_e-th layer be h^{l_e}, where $h^{l_e}_{v_i}$ and $h^{l_e}_{p_i}$ denote the components corresponding to v_i and p_i, respectively. We use an additional feature extraction function $\bar{f}_{fe}$ (defined in consistency within Eq.3, but taking different inputs) to extract routing features from the input:

$$(\bar{\phi}_{v_i}, \bar{\psi}_{p_i}) = \bar{f}_{fe}(h^{l_e}_{v_i}, h^{l_e}_{p_i}). \tag{4}$$

Given $\mathcal{E}_t = \{(U_{e_\tau}, V_{e_\tau}, \hat{\phi}_{v_{e_\tau}}, \hat{\psi}_{p_{e_\tau}})\}_{\tau=1}^{t}$, we filter for experts that are highly relevant to the input sample's visual content by calculating the similarity between the key visual features of the input and the edit samples:

$$\hat{\mathcal{E}} = \left\{(U_{e_\tau}, V_{e_\tau}, \hat{\psi}_{p_{e_\tau}}) \mid \bar{\phi}_{v_i}\hat{\phi}_{v_{e_\tau}}^T > \bar{\phi}_{v_i}\bar{\phi}_{\Theta}^T,\ \tau = 1, ..., t\right\}, \quad \text{s.t. } (\bar{\phi}_{\Theta}, \_) = \bar{f}_{fe}(\Theta, h_{p_i}^{l_e}) \tag{5}$$

where a trainable vision sentinel $\Theta \in \mathbb{R}^{N_v \times d}$ is set to dynamically determine the filtering threshold, following [20], which effectively avoids the bias caused by manually set thresholds. N_v is the vision token count of f_θ. Intuitively, if the input sample is more visually similar to the edit sample than the visual sentinel, then this edit sample should not be selected.

Although the above process effectively selects visually relevant editing experts, some results may still have low prompt semantic relevance. We further use the similarity between prompt features to achieve multi-expert fusion. Thus, the post-edit representation $\hat{h}^{l_e}$ is obtained as follows:

$$\hat{h}^{l_e} = h^{l_e} + \sum_{(U_e, V_e, \hat{\psi}_{p_e}) \in \hat{\mathcal{E}}} f_{sr}(\bar{\psi}_{p_i}, \hat{\psi}_{p_e}, \hat{\mathcal{E}}) \rho(h^{l_e} U_e^T) V_e \tag{6}$$

$$\text{s.t.} \quad f_{sr}(\bar{\psi}, \hat{\psi}, \hat{\mathcal{E}}) = \sigma\left(\bar{\psi}\hat{\psi}^T\right) \frac{\exp(\bar{\psi}\hat{\psi}^T)}{\sum_{(\_,\_,\hat{\psi}_{p_e}) \in \hat{\mathcal{E}}} \exp(\bar{\psi}\hat{\psi}_{p_e}^T)} \tag{7}$$

where $\rho(\cdot)$ and $\sigma(\cdot)$ are ReLU and sigmoid, respectively. The inner products are rescaled by $\sqrt{d_m}$, which is omitted above. The $\hat{h}^{l_e}$ will proceed to complete the subsequent layer inference and generate the modified response. The Soft Routing function $f_{sr}(\cdot)$ multiplies absolute weights from the sigmoid and relative weights from the softmax. The absolute weights control the output strength of each expert based on similarity. The relative weights balance the similarity among the selected experts to constrain the scale of the fused residual output within 1, preventing the combined output from generating an excessively large norm.

4.3. Training of LiveEdit

The training primarily consists of two parts: the edit loss, which ensures that the generated MoEs effectively guide the VLLM to follow the editing instructions, and the routing loss, which ensures hard and soft MoE routing. Given a batch of edit samples $\mathcal{D}_e = \{(v_{e_b}, p_{e_b}, o_{e_b})\}_{b=1}^{B}$, and their corresponding sampled generality and locality samples $\mathcal{D}_g = \{(v_{g_b}, p_{g_b}, o_{g_b})\}_{b=1}^{B}$ and $\mathcal{D}_l = \{(v_{l_b}, p_{l_b}, o_{l_b})\}_{b=1}^{B}$, the losses are formulated as follows.

4.3.1. Edit Loss

We mix the experts for the entire batch of edit samples to simulate the scenario during inference, where hard routing leads to multiple experts. First, we obtain the l_e-th layer outputs $\{(h_{v_{e_b}}^{l_e}, h_{p_{e_b}}^{l_e}, h_{o_{e_b}}^{l_e})\}_{b=1}^{B}$ for each part of an edit sample in $\mathcal{D}_e$. Then, through Eqs. 1 and 3, their corresponding experts and routing features can be obtained as $\{(U_{e_b}, V_{e_b})\}_{b=1}^{B}$ and $\{(\hat{\phi}_{v_{e_b}}, \hat{\psi}_{p_{e_b}})\}_{b=1}^{B}$. We define the expert set for soft routing fusion as $\hat{\mathcal{E}} = \{(U_{e_b}, V_{e_b}, \hat{\psi}_{p_{e_b}})\}_{b=1}^{B}$. Thus, for any input sample, its representation at layer l_e will be modified as in Eq. 6. Defining the VLLM modified in this way as $f_{\theta_{\hat{\mathcal{E}}}}$, the edit loss is defined as follows:

$$\ell_{\text{edit}} = \frac{1}{B}\sum_{b=1}^{B}\left(\ell_{\text{rel}}^{(b)} + \ell_{\text{gen}}^{(b)} + \ell_{\text{loc}}^{(b)}\right) \tag{8}$$

where

$$\ell_{\text{rel}}^{(b)} = -\log f_{\theta_{\hat{\mathcal{E}}}}\left(o_{e_b} \mid v_{e_b}, p_{e_b}\right) \tag{9}$$

$$\ell_{\text{gen}}^{(b)} = -\log f_{\theta_{\hat{\mathcal{E}}}}\left(o_{g_b} \mid v_{g_b}, p_{g_b}\right) \tag{10}$$

$$\ell_{\text{loc}}^{(b)} = \text{KL}\left(f_\theta\left(o_{l_b} \mid v_{l_b}, p_{l_b}\right) \,||\, f_{\theta_{\hat{\mathcal{E}}}}\left(o_{l_b} \mid v_{l_b}, p_{l_b}\right)\right) \tag{11}$$

Here, KL denotes the Kullback-Leibler divergence.

4.3.2. Routing Loss

In the routing part, we maximize the feature similarity between samples within the generality domain, while minimizing the feature similarity between unrelated samples. First, we randomly assign samples within the same generalization domain (i.e., edit samples and their corresponding generality samples) into two new sets, defined as $\hat{\mathcal{D}}_g = \{[\mathcal{D}_e^{(b)}, \mathcal{D}_g^{(b)}]_{\pi_1^{(b)}}\}_{b=1}^{B}$, and $\bar{\mathcal{D}}_g = \{[\mathcal{D}_e^{(b)}, \mathcal{D}_g^{(b)}]_{\pi_2^{(b)}}\}_{b=1}^{B}$. Here, $\pi_1^{(b)}, \pi_2^{(b)} \in [0,1]^B$ are the random integer vectors applied across the batch. This approach equalizes the reliability and generality of samples in feature matching with the edited and input samples, enhancing routing robustness. We use $\hat{f}_{fe}$ to extract the routing features $\{(\hat{\phi}_{g_b}, \hat{\psi}_{g_b})\}_{b=1}^{B}$ from $\hat{\mathcal{D}}_g$ corresponding to edit end, and use $\bar{f}_{fe}$ to extract the routing features $\{(\bar{\phi}_{g_b}, \bar{\psi}_{g_b})\}_{b=1}^{B}$ and $\{(\bar{\phi}_{l_b}, \bar{\psi}_{l_b})\}_{b=1}^{B}$ from $\bar{\mathcal{D}}_g$ and $\mathcal{D}_l$ corresponding to input end, respectively. The routing loss is formulated as follows:

$$\ell_{\text{route}} = \sum_{b=1}^{B}\ell_{hr}^{(b)} + \ell_{sr_1}^{(b)} + \ell_{sr_2}^{(b)} \tag{12}$$

Hard Routing loss $\ell_{hr}^{(b)}$ is defined as follows:

$$\ell_{hr}^{(b)} = f_{nce}(\bar{\phi}_{g_b}, \hat{\phi}_{g_b}, \hat{\Phi} \cup \{\bar{\phi}_{\Theta_{g_b}}\}) + f_{nce}(\bar{\phi}_{l_b}, \bar{\phi}_{\Theta_{l_b}}, \hat{\Phi} \cup \{\bar{\phi}_{\Theta_{l_b}}\}) \tag{13}$$

where $\hat{\Phi} = \{\hat{\phi}_{g_b}\}_{b=1}^{B}$. $\bar{\phi}_{\Theta_{g_b}}$ and $\bar{\phi}_{\Theta_{l_b}}$ represent the features extracted by the corresponding generality and locality data at the input end (as defined in Eq.5) from the vision sentinel. f_{nce} is the InfoNCE loss [57] formulated as:

$$f_{nce}(\alpha, \beta_+, \{\beta_j\}_{j=1}^{n}) = -\log\frac{\exp(\alpha\beta_+^T)}{\sum_{j=1}^{n}\exp(\alpha\beta_j^T)} \tag{14}$$

We set the temperature to 1, which is omitted here. The above loss function brings the key visual features of data within the generalization domain closer—even closer than those extracted using the visual sentinel. Meanwhile, it pushes the locality inputs further from the generalization domain, making them relatively closer to the features extracted from the visual sentinel.

Soft Routing includes absolute loss $\ell_{sr_1}^{(b)}$ and relative loss $\ell_{sr_2}^{(b)}$, defined as follows:

$$\ell_{sr_1}^{(b)} = -\log \sigma(\bar{\psi}_{g_b} \hat{\psi}_{g_b}^T) - \log(1 - \sigma(\bar{\psi}_{g_b} \hat{\psi}_{\setminus g_b}^T)), \quad (15)$$

$$\ell_{sr_2}^{(b)} = f_{nce}\left(\bar{\psi}_{g_b}, \hat{\psi}_{g_b}, \{\hat{\psi}_{g_j}\}_{j=1}^{B} \cup \{\hat{\psi}_{l_j}\}_{j=1}^{B}\right), \quad (16)$$

where $\hat{\psi}_{\setminus g_b}$ represents a feature randomly selected from $\{\hat{\psi}_{g_j}\}_{j=1}^{B} \cup \{\hat{\psi}_{l_j}\}_{j=1}^{B} \setminus \{\hat{\psi}_{g_b}\}$. Thus, the total training loss is: $\ell_{total} = \ell_{edit} + \ell_{route}$. During training, the parameters of the VLLM, f_θ, are frozen. The trainable modules are an experts generation module, f_{eg}, and two feature extraction modules, $\hat{f}_{fe}$ and $\bar{f}_{fe}$.

5. Experiments

5.1. Experimental Settings

Datasets: Following [22], we use E-VQA (Editing Visual Question Answering) and E-IC (Editing Image Caption) as evaluation datasets. Additionally, we incorporate VLKEB [30], which is composed of real images to better represent real-world scenarios.

VLLM Backbones: For comprehensive evaluation, we select VLLM backbones based on both model architecture and parameter scale, including BLIP2-OPT (2.7B) [4], LLaVA-V1.5 (7B) [28], and MiniGPT-4 (7B) [29].

Baseline Editors: To our knowledge, there are currently no editors specifically designed for lifelong VLLM editing. Therefore, following [22], in addition to the basic FT-L and FT-M, which respectively fine-tune the final layer of the LLM and the visual encoder, we adapt LLM-based editing techniques to VLLM. These include MEND [42], TP [45], LTE [19], RECIPE [20], and LEMoE [46].

For details on the experimental setup, model hyper-parameters, and training specifics, please refer to Appendix 7. Building on the experimental settings above, we conduct a comprehensive evaluation of edit performance and perform an in-depth analysis of LiveEdit's internals.

5.2. General Performance of Lifelong Editing

Table 1 shows partial results of lifelong editing experiments. In single-edit scenarios, our method generally achieves optimal performance. Methods such as FT-L, FT-M, and MEND [42] try to modify original model parameters and perform well initially. However, their performance deteriorates with more edits due to overfitting and cumulative

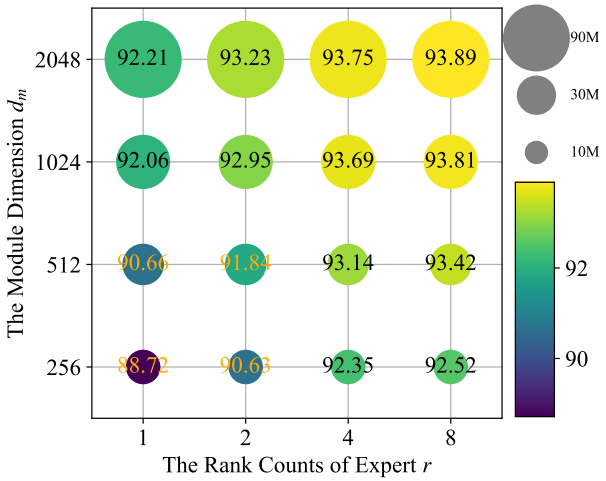

Figure 3. The impact of module dimension d_m and expert rank r on LiveEdit's edit performance. Experiments are conducted on the E-VQA dataset, with 1,000 edits on BLIP2. Circle size represents LiveEdit's training parameters, and color intensity indicates the average edit performance across five metrics.

parameter scaling [21, 58]. TP [45] addresses parameter scaling by introducing additional neurons, but a single neuron can't encapsulate each edit's visual information. Retrieval-based methods (SERAC [47], LTE [19], RECIPE [20]) remain robust in lifelong editing by decoupling edits from model parameters, yet struggle with semantically similar but visually different edit samples. LEMoE [46], a MoE-based editor, excels with few edits but suffers from issues like disruptive greedy routing and limited generalization due to overfitting experts to batch edits. Our method, LiveEdit, surpasses in edit performance. Contrastive learning-based expert routing resolves LEMoE's issues, enhancing both edit speed and generalization. By decoupling edit samples into independent experts and applying a fusion strategy for vision-related experts, we prevent the semantic conflicts LEMoE encounters from its sequential batching. Importantly, our method maintains nearly 100% locality performance even with increasing edits. Three main factors contribute to our performance: First, hard routing filters out visually irrelevant experts. Second, soft routing scales influence by assigning lower scales to textually irrelevant experts. Finally, the locality edit loss further confines experts' influence on response adaptation.

5.3. Hyper-Parameter Search

We conducted a comprehensive hyper-parameters search for LiveEdit to select the most suitable combination. The most important hyper-parameters are discussed below.

Trade-off Between Model Scale and Edit Performance: Figure 3 reports the effects of different combinations of module dimension d_m and expert rank r on edit performance and model scale. The d_m has a significant impact on the parameter count of LiveEdit. In terms of edit per-

Baseline	# Edit	Editors	E-VQA Rel.	E-VQA T-Gen.	E-VQA M-Gen.	E-VQA T-Loc.	E-VQA M-Loc.	E-VQA Average	VLKEB Rel.	VLKEB T-Gen.	VLKEB M-Gen.	VLKEB T-Loc.	VLKEB M-Loc.	VLKEB Average
LLaVA-V1.5 (7B)	1	FT-L	93.88	87.98	80.25	99.61	94.78	91.30 (±0.42)	94.29	87.00	92.22	91.16	91.37	91.21 (±1.09)
		FT-M	87.29	76.11	53.23	100.00	96.95	82.72 (±1.05)	76.31	65.57	59.43	100.00	92.35	78.73 (±0.76)
		MEND	91.23	90.05	91.29	91.02	90.22	90.76 (±0.64)	92.13	91.28	90.22	89.19	90.13	90.59 (±1.24)
		SERAC	89.33	83.72	84.97	82.05	23.78	72.77 (±0.36)	89.77	89.11	87.92	66.68	14.20	69.54 (±0.83)
		TP	35.95	36.12	28.65	93.87	97.61	58.44 (±0.33)	50.77	55.70	51.65	87.93	90.43	67.30 (±0.29)
		LTE	94.16	93.57	**93.59**	94.08	86.26	92.33 (±1.56)	94.42	93.57	93.22	86.84	79.69	89.55 (±1.41)
		RECIPE	91.37	86.51	87.73	94.27	88.88	89.75 (±1.13)	92.67	92.35	91.01	89.67	82.85	89.71 (±0.57)
		LEMoE	93.60	92.77	89.99	99.28	96.98	94.52 (±1.09)	94.85	93.09	91.67	87.03	87.88	90.90 (±0.29)
		LiveEdit	**94.28**	**94.51**	88.01	**100.00**	**100.00**	**95.36** (±0.57)	**96.43**	**95.22**	**93.72**	**100.00**	**100.00**	**97.08** (±0.62)
	10	FT-L	90.57	84.14	73.21	95.56	81.50	85.00 (±1.07)	88.05	85.32	85.23	74.53	85.74	83.77 (±1.22)
		FT-M	84.90	73.53	49.99	100.00	55.98	72.88 (±0.63)	68.63	57.57	56.56	100.00	82.99	73.15 (±0.23)
		MEND	3.58	3.55	3.53	2.10	1.26	2.80 (±0.02)	0.18	0.24	0.05	0.03	0.19	0.14 (±0.00)
		SERAC	88.09	83.40	83.57	64.91	15.50	67.10 (±0.92)	81.55	74.49	80.24	54.71	13.15	60.83 (±0.98)
		TP	32.71	31.23	28.58	75.10	91.17	51.76 (±0.60)	44.56	47.52	45.36	52.21	66.61	51.25 (±0.69)
		LTE	92.83	91.41	**90.82**	86.38	85.52	89.39 (±0.34)	90.06	81.52	88.11	83.40	81.48	84.91 (±0.78)
		RECIPE	90.22	85.92	86.24	90.34	88.11	88.17 (±1.48)	83.92	76.23	82.84	86.33	83.69	82.60 (±0.72)
		LEMoE	91.95	86.54	79.82	85.19	49.81	78.66 (±1.03)	91.55	84.58	81.03	67.19	72.81	79.43 (±0.52)
		LiveEdit	**93.79**	**93.21**	86.42	**100.00**	**100.00**	**94.68** (±1.03)	**95.54**	**94.52**	**91.25**	**100.00**	**100.00**	**96.26** (±0.33)
	1000	FT-L	71.39	59.83	57.41	55.55	48.99	58.63 (±0.17)	68.14	66.38	66.98	65.61	75.35	68.49 (±0.32)
		FT-M	69.57	56.34	44.07	100.00	41.47	62.29 (±0.40)	53.41	48.80	43.16	100.00	57.03	60.48 (±0.50)
		MEND	0.04	0.05	0.05	0.08	0.09	0.06 (±0.00)	0.03	0.05	0.07	0.06	0.08	0.06 (±0.00)
		SERAC	85.57	75.58	82.01	62.46	15.69	64.26 (±0.37)	60.93	56.49	60.06	52.94	15.04	49.09 (±0.36)
		TP	16.56	16.80	15.65	7.28	15.60	14.38 (±0.14)	5.46	4.81	5.51	2.77	7.19	5.15 (±0.07)
		LTE	83.93	82.55	81.34	83.97	73.09	80.98 (±1.36)	64.51	56.26	64.80	80.85	76.52	68.59 (±0.60)
		RECIPE	87.00	76.81	83.09	86.95	87.03	84.18 (±0.80)	62.00	56.84	61.50	85.37	82.07	69.56 (±0.31)
		LEMoE	30.80	25.75	24.32	71.45	46.23	39.71 (±0.23)	67.97	61.07	58.16	48.48	44.06	55.95 (±0.36)
		LiveEdit	**92.93**	**90.16**	**84.30**	**100.00**	**96.43**	**92.76** (±0.20)	**92.22**	**83.97**	**82.75**	**100.00**	**100.00**	**91.79** (±0.55)
BLIP2-OPT (2.7B)	1	FT-L	52.86	48.80	32.94	98.24	94.27	65.42 (±0.69)	54.31	54.27	54.08	98.40	94.37	71.09 (±1.05)
		FT-M	91.70	87.24	33.30	100.00	85.22	79.49 (±0.72)	92.64	80.97	63.62	100.00	83.02	84.05 (±0.70)
		MEND	93.13	92.76	93.07	92.00	75.81	89.35 (±0.93)	94.91	93.81	93.84	94.98	86.54	92.82 (±0.82)
		SERAC	88.39	84.50	84.25	85.82	26.00	73.79 (±1.01)	87.95	84.67	85.20	68.10	17.75	68.73 (±0.97)
		TP	70.14	65.80	53.05	98.11	85.33	74.49 (±0.38)	50.98	49.47	50.88	94.76	78.57	64.93 (±1.02)
		LTE	95.74	93.86	86.90	97.93	87.97	92.48 (±0.70)	94.13	91.93	92.23	93.89	92.27	92.89 (±1.01)
		RECIPE	89.42	86.24	87.53	99.87	89.16	90.45 (±1.46)	92.38	89.74	89.17	97.13	94.46	92.58 (±1.16)
		LEMoE	93.56	92.23	91.40	98.50	85.21	92.18 (±0.73)	94.59	93.14	92.37	94.53	61.53	87.23 (±0.34)
		LiveEdit	**96.67**	**94.20**	**93.82**	**100.00**	**100.00**	**96.94** (±1.32)	**98.77**	**98.08**	**94.89**	**100.00**	**100.00**	**98.35** (±1.58)
	1000	FT-L	45.10	34.62	35.42	48.42	41.24	40.96 (±0.29)	55.39	54.34	53.87	50.80	54.00	53.68 (±0.80)
		FT-M	40.40	31.46	27.85	100.00	27.44	45.43 (±0.68)	47.03	49.68	46.99	100.00	41.41	57.02 (±0.13)
		MEND	15.84	14.35	17.73	91.74	70.17	41.97 (±0.12)	37.22	38.03	37.19	91.49	84.10	57.61 (±0.58)
		SERAC	83.35	70.80	80.32	67.66	13.13	63.05 (±0.87)	53.58	45.78	52.42	56.81	16.90	45.10 (±0.38)
		TP	20.63	15.09	18.41	8.65	8.25	14.21 (±0.18)	24.36	24.21	24.25	16.37	19.96	21.83 (±0.14)
		LTE	89.32	82.82	81.51	94.86	69.83	83.67 (±1.05)	61.67	51.05	61.60	94.78	90.94	72.01 (±0.66)
		RECIPE	84.99	74.20	82.04	96.82	87.73	85.16 (±1.32)	54.64	46.54	54.10	94.60	96.93	69.37 (±1.04)
		LEMoE	19.73	17.34	18.22	72.01	31.06	31.67 (±0.14)	34.74	33.43	32.05	55.55	50.04	41.16 (±0.58)
		LiveEdit	**94.42**	**91.98**	**84.65**	**100.00**	**97.38**	**93.69** (±0.67)	**97.00**	**91.92**	**87.53**	**100.00**	**100.00**	**95.29** (±1.48)

Table 1. Partial results of lifelong edit performance for BLIP2-OPT and LLaVA-V1.5 on the E-VQA and VLKEB datasets. Due to space limitations, please refer to Appendix 8.1 for the complete results, including those for the E-IC dataset and the MiniGPT-4 model. "Rel.", "T/M-Gen." and "T/M-Loc." stand for reliability, text/modal generality, and text/modal locality, respectively. "# Edit" indicates the number of edits. The t-tests demonstrate our improvements are statistically significant with $p < 0.05$ level.

formance, both d_m and r have substantial effects. Increasing one while keeping the other fixed improves edit performance, though the improvement gradually becomes flat. Additionally, combinations along the diagonal show generally consistent performance. Regarding model scale, d_m predominantly influences variation, while r has minimal impact. Based on this analysis, the optimal configuration strategy is to select an appropriate d_m and maximize r as much as possible. However, since r linearly controls the growth rate of the expert repository, specific choices should also consider the memory needed to store the expert repository. Additionally, we expand the dimension control parameter k for feature extraction, as shown in Figure 4. Increasing k enhances the feature extraction capability, but an excessively high k may introduce noise, leading to incorrect matches, such as reduced modal locality.

The Attached Layer of LiveEdit: Figure 5 shows the impact of the layer attached by LiveEdit on edit performance. It can be seen that as the layer depth increases, edit performance improves, reaching a peak at 21 layers. After this point, edit performance slightly declines. We speculate that transformers typically perform semantic understanding in the early layers and response generation in the later layers [17, 24, 59]. Attaching to an earlier layer prevents LiveEdit from leveraging VLLM's semantic understanding capabilities to enhance the feature extraction process. For too deep layers, VLLM has largely stabilized the predictive tendencies of the response, making it more challenging for LiveEdit to adapt. The above results also align with the attribution conclusions of [24].

5.4. Ablation Study

Table 2 presents the ablation results for LiveEdit, where 1000 edits are applied to BLIP2 on the E-VQA dataset.

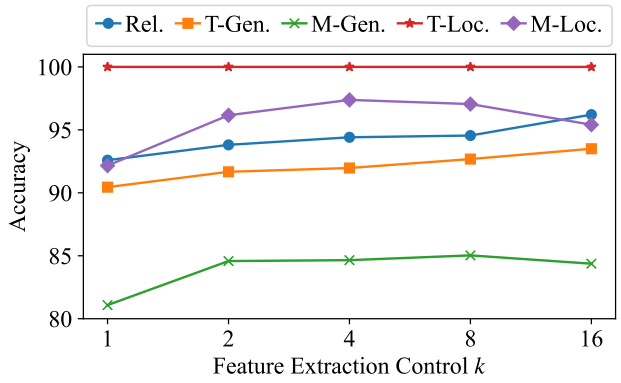

Figure 4. The dimension control parameter k for feature extraction.

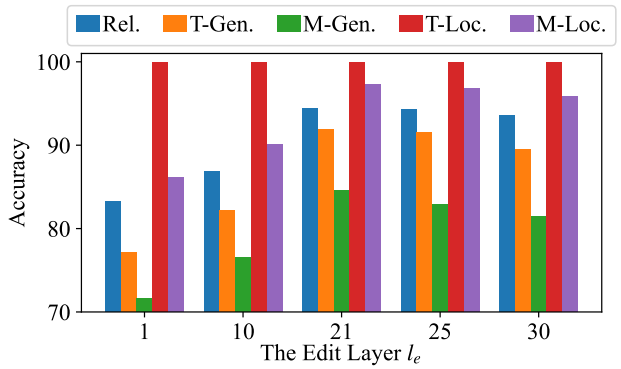

Figure 5. Impact of LiveEdit attached layer index l_e. Results of 1,000 edits for BLIP2 on E-VQA dataset are reported.

Settings	Rel.	T-Gen.	M-Gen.	T-Loc.	M-Loc.	Average
N/A	20.57	19.03	14.17	100.00	100.00	50.75
LiveEdit	**94.42**	**91.98**	**84.65**	**100.00**	**97.38**	**93.69**
- ℓ_{sr1}	87.93	83.77	73.40	100.00	76.93	84.41
- ℓ_{sr2}	89.21	85.46	77.34	100.00	91.69	88.74
- SR	88.92	81.49	70.94	100.00	75.66	83.40
HR*	93.60	88.50	80.37	100.00	84.77	89.45

Table 2. Ablation study of LiveEdit.

The removal of soft routing (- SR, which directly averages experts) results in a significant performance drop for LiveEdit. We can also observe that a large portion of this loss originates from the removal of the absolute soft routing loss ℓ_{sr_1}, particularly impacting modal locality. This is because all hard-routed experts are assigned a weight to adapt the representation, even if they are irrelevant to the input prompt, ensuring that the sum of adaptation strengths equals 1. Similarly, removing the relative soft routing loss ℓ_{sr_2} also leads to performance degradation, as the combined absolute weights of multiple experts may exceed 1 during testing, resulting in excessively high values. HR* modifies the visual extraction that leverages prompt semantics by directly compressing the entire visual representation for hard routing. This introduces additional visual noise, leading to the selection of more irrelevant experts, which in turn reduces the efficacy of subsequent steps.

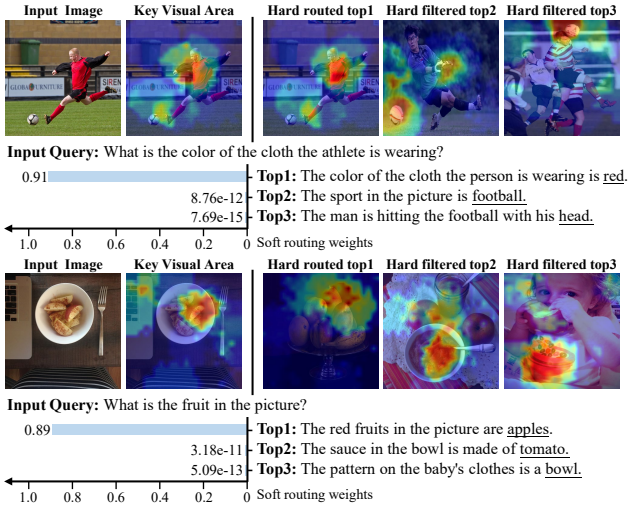

Figure 6. Instance analysis. The left side shows the model input to LLaVa after editing with LiveEdit. The right side displays the top 3 experts based on hard routing. The heatmaps represent the visual regions focused on when extracting key visual features. The bar chart below shows fusion weights from soft routing. Please refer to Appendix 8.2 for more analysis.

5.5. Instance Analysis

We conducted an instance analysis of LiveEdit, as shown in Figure 6. We perform 100 edits on LLaVa, including samples partially related in visual context to the incoming input samples. The figure reports the top 3 hard routing results for two inputs, as well as the fusion weights these experts received in soft routing. Using a perturbation-based attribution tool [24, 60], we visualize the visual regions focused on for key visual feature extraction. It can be observed that the highlighted regions are closely aligned with the prompt semantics, which filters out irrelevant visual noise and benefits hard routing. The bar chart indicates that experts unrelated to the input query receive very low fusion weights, significantly benefiting the locality of edits.

6. Conclusion

In conclusion, we introduce LiveEdit to bridge the gap between LLM and VLLM editing, with an editing expert generator and a combination of hard/soft routers. The framework successfully addresses the limitation of exiting editors in the VLLM lifelong editing scenarios. Our benchmark and extensive experiments confirm its superiority and highlight the effectiveness of each component, which support more accurate and adaptable VLLM editing in real-world applications.

Acknowledgments. This work is supported by the National Key R&D Program of China (2022ZD0120302).

References

[1] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "Llama: Open and efficient foundation language models," *CoRR*, vol. abs/2302.13971, 2023. 1

[2] K. I. Roumeliotis and N. D. Tselikas, "Chatgpt and open-ai models: A preliminary review," *Future Internet*, vol. 15, no. 6, p. 192, 2023.

[3] A. Zeng, X. Liu, Z. Du, Z. Wang, H. Lai, M. Ding, Z. Yang, Y. Xu, W. Zheng, X. Xia, W. L. Tam, Z. Ma, Y. Xue, J. Zhai, W. Chen, Z. Liu, P. Zhang, Y. Dong, and J. Tang, "GLM-130B: an open bilingual pre-trained model," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023. 1

[4] J. Li, D. Li, S. Savarese, and S. C. H. Hoi, "BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models," in *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, vol. 202 of *Proceedings of Machine Learning Research*, pp. 19730–19742, 2023. 1, 2, 6, 3

[5] W. Dai, J. Li, D. Li, A. M. H. Tiong, J. Zhao, W. Wang, B. Li, P. Fung, and S. C. H. Hoi, "Instructblip: Towards general-purpose vision-language models with instruction tuning," in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[6] J. Bai, S. Bai, S. Yang, S. Wang, S. Tan, P. Wang, J. Lin, C. Zhou, and J. Zhou, "Qwen-vl: A frontier large vision-language model with versatile abilities," *CoRR*, vol. abs/2308.12966, 2023.

[7] P. Zhang, X. Dong, B. Wang, Y. Cao, C. Xu, L. Ouyang, Z. Zhao, S. Ding, S. Zhang, H. Duan, W. Zhang, H. Yan, X. Zhang, W. Li, J. Li, K. Chen, C. He, X. Zhang, Y. Qiao, D. Lin, and J. Wang, "Internlm-xcomposer: A vision-language large model for advanced text-image comprehension and composition," *CoRR*, vol. abs/2309.15112, 2023. 1, 2

[8] J. Jang, D. Yoon, S. Yang, S. Cha, M. Lee, L. Logeswaran, and M. Seo, "Knowledge unlearning for mitigating privacy risks in language models," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 14389–14408, 2023. 1

[9] Y. Ishibashi and H. Shimodaira, "Knowledge sanitization of large language models," *CoRR*, vol. abs/2309.11852, 2023. 1

[10] Y. Li, T. Li, K. Chen, J. Zhang, S. Liu, W. Wang, T. Zhang, and Y. Liu, "Badedit: Backdooring large language models by model editing," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024. 1

[11] M. Wang, N. Zhang, Z. Xu, Z. Xi, S. Deng, Y. Yao, Q. Zhang, L. Yang, J. Wang, and H. Chen, "Detoxifying large language models via knowledge editing," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 3093–3118, 2024. 1

[12] C. Yu, S. Jeoung, A. Kasi, P. Yu, and H. Ji, "Unlearning bias in language models by partitioning gradients," in *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pp. 6032–6048, 2023. 1

[13] T. Limisiewicz, D. Marecek, and T. Musil, "Debiasing algorithm through model adaptation," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

[14] R. Chen, Y. Li, J. Yang, J. T. Zhou, and Z. Liu, "Editable fairness: Fine-grained bias mitigation in language models," *CoRR*, vol. abs/2408.11843, 2024. 1

[15] Y. Chuang, Y. Xie, H. Luo, Y. Kim, J. R. Glass, and P. He, "Dola: Decoding by contrasting layers improves factuality in large language models," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024. 1

[16] S. Zhang, T. Yu, and Y. Feng, "Truthx: Alleviating hallucinations by editing large language models in truthful space," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 8908–8949, 2024. 1

[17] K. Meng, D. Bau, A. Andonian, and Y. Belinkov, "Locating and editing factual associations in GPT," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. 1, 3, 7

[18] K. Meng, A. S. Sharma, A. J. Andonian, Y. Belinkov, and D. Bau, "Mass-editing memory in a transformer," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023. 1, 3

[19] Y. Jiang, Y. Wang, C. Wu, W. Zhong, X. Zeng, J. Gao, L. Li, X. Jiang, L. Shang, R. Tang, Q. Liu, and W. Wang, "Learning to edit: Aligning llms with knowledge editing," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 4689–4705, 2024. 1, 3, 6, 2

[20] Q. Chen, T. Zhang, X. He, D. Li, C. Wang, L. Huang, and X. Hui, "Lifelong knowledge editing for LLMs with retrieval-augmented continuous prompt learning," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13565–13580, Nov. 2024. 1, 5, 6, 2

[21] C. Hu, P. Cao, Y. Chen, K. Liu, and J. Zhao, "Wilke: Wise-layer knowledge editor for lifelong knowledge editing," in *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pp. 3476–3503, 2024. 1, 3, 6

[22] S. Cheng, B. Tian, Q. Liu, X. Chen, Y. Wang, H. Chen, and N. Zhang, "Can we edit multimodal large language models?," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023,*

Singapore, December 6-10, 2023, pp. 13877–13888, 2023. 1, 2, 3, 6

[23] S. Basu, M. Grayson, C. Morrison, B. Nushi, S. Feizi, and D. Massiceti, "Understanding information storage and transfer in multi-modal large language models," *CoRR*, vol. abs/2406.04236, 2024.

[24] Q. Chen, T. Zhang, C. Wang, X. He, D. Wang, and T. Liu, "Attribution analysis meets model editing: Advancing knowledge correction in vision language models with visedit," *CoRR*, vol. abs/2408.09916, 2024. 1, 2, 3, 4, 7, 8

[25] T. Hartvigsen, S. Sankaranarayanan, H. Palangi, Y. Kim, and M. Ghassemi, "Aging with GRACE: lifelong model editing with discrete key-value adaptors," in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 1, 3

[26] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Comput.*, vol. 3, no. 1, pp. 79–87, 1991. 2, 3

[27] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. V. Le, G. E. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017. 2, 3

[28] H. Liu, C. Li, Q. Wu, and Y. J. Lee, "Visual instruction tuning," in *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. 2, 6, 1, 3

[29] D. Zhu, J. Chen, X. Shen, X. Li, and M. Elhoseiny, "Minigpt-4: Enhancing vision-language understanding with advanced large language models," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024. 2, 6, 1, 3

[30] H. Huang, H. Zhong, T. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan, "Vlkeb: A large vision-language model knowledge editing benchmark," 2024. 2, 3, 6, 1

[31] Y. Chang, X. Wang, J. Wang, Y. Wu, L. Yang, K. Zhu, H. Chen, X. Yi, C. Wang, Y. Wang, W. Ye, Y. Zhang, Y. Chang, P. S. Yu, Q. Yang, and X. Xie, "A survey on evaluation of large language models," *ACM Trans. Intell. Syst. Technol.*, vol. 15, no. 3, pp. 39:1–39:45, 2024. 2

[32] J. Wu, W. Gan, Z. Chen, S. Wan, and P. S. Yu, "Multimodal large language models: A survey," in *IEEE International Conference on Big Data, BigData 2023, Sorrento, Italy, December 15-18, 2023*, pp. 2247–2256, 2023. 2

[33] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021. 2

[34] H. Laurençon, L. Saulnier, L. Tronchon, S. Bekman, A. Singh, A. Lozhkov, T. Wang, S. Karamcheti, A. M. Rush, D. Kiela, M. Cord, and V. Sanh, "OBELISC: an open web-scale filtered dataset of interleaved image-text documents," *CoRR*, vol. abs/2306.16527, 2023. 2

[35] J. Alayrac, J. Donahue, P. Luc, A. Miech, I. Barr, Y. Hasson, K. Lenc, A. Mensch, K. Millican, M. Reynolds, R. Ring, E. Rutherford, S. Cabi, T. Han, Z. Gong, S. Samangooei, M. Monteiro, J. L. Menick, S. Borgeaud, A. Brock, A. Nematzadeh, S. Sharifzadeh, M. Binkowski, R. Barreira, O. Vinyals, A. Zisserman, and K. Simonyan, "Flamingo: a visual language model for few-shot learning," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. 2

[36] Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh, "Making the V in VQA matter: Elevating the role of image understanding in visual question answering," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 6325–6334, 2017. 3, 1

[37] D. A. Hudson and C. D. Manning, "GQA: A new dataset for real-world visual reasoning and compositional question answering," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 6700–6709, 2019. 3

[38] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, *et al.*, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality." https://vicuna.lmsys.org, 2023. Accessed: 2023-04-14. 3, 1

[39] Y. Liu, H. Duan, Y. Zhang, B. Li, S. Zhang, W. Zhao, Y. Yuan, J. Wang, C. He, Z. Liu, K. Chen, and D. Lin, "Mmbench: Is your multi-modal model an all-around player?," in *Computer Vision - ECCV 2024 - 18th European Conference, Milan, Italy, September 29-October 4, 2024, Proceedings, Part VI*, vol. 15064 of *Lecture Notes in Computer Science*, pp. 216–233, 2024. 3

[40] D. Dai, L. Dong, Y. Hao, Z. Sui, B. Chang, and F. Wei, "Knowledge neurons in pretrained transformers," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 8493–8502, 2022. 3

[41] N. D. Cao, W. Aziz, and I. Titov, "Editing factual knowledge in language models," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pp. 6491–6506, 2021. 3

[42] E. Mitchell, C. Lin, A. Bosselut, C. Finn, and C. D. Manning, "Fast model editing at scale," in *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*, 2022. 6, 1

[43] C. Tan, G. Zhang, and J. Fu, "Massive editing for large language models via meta learning," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024.

[44] T. Zhang, Q. Chen, D. Li, C. Wang, X. He, L. Huang, H. Xue, and J. Huang, "Dafnet: Dynamic auxiliary fusion for sequen-

tial model editing in large language models," in *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pp. 1588–1602, 2024. 3

[45] Z. Huang, Y. Shen, X. Zhang, J. Zhou, W. Rong, and Z. Xiong, "Transformer-patcher: One mistake worth one neuron," in *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023. 3, 6, 2

[46] R. Wang and P. Li, "Lemoe: Advanced mixture of experts adaptor for lifelong model editing of large language models," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, pp. 2551–2575, 2024. 3, 6, 2

[47] E. Mitchell, C. Lin, A. Bosselut, C. D. Manning, and C. Finn, "Memory-based model editing at scale," in *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, vol. 162 of *Proceedings of Machine Learning Research*, pp. 15817–15831, 2022. 3, 6, 1

[48] A. Madaan, N. Tandon, P. Clark, and Y. Yang, "Memory-assisted prompt editing to improve GPT-3 after deployment," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pp. 2833–2861, 2022.

[49] W. Wang, B. Haddow, and A. Birch, "Retrieval-augmented multilingual knowledge editing," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 335–354, 2024. 3

[50] D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen, "Gshard: Scaling giant models with conditional computation and automatic sharding," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, 2021. 3

[51] W. Fedus, B. Zoph, and N. Shazeer, "Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity," *J. Mach. Learn. Res.*, vol. 23, pp. 120:1–120:39, 2022. 3

[52] M. Lewis, S. Bhosale, T. Dettmers, N. Goyal, and L. Zettlemoyer, "BASE layers: Simplifying training of large, sparse models," in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, vol. 139 of *Proceedings of Machine Learning Research*, pp. 6265–6274, 2021. 3

[53] S. Roller, S. Sukhbaatar, A. Szlam, and J. Weston, "Hash layers for large sparse models," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 17555–17566, 2021. 3

[54] Y. Zhou, T. Lei, H. Liu, N. Du, Y. Huang, V. Y. Zhao, A. M. Dai, Z. Chen, Q. V. Le, and J. Laudon, "Mixture-of-experts with expert choice routing," in *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. 3

[55] F. Xue, Z. Zheng, Y. Fu, J. Ni, Z. Zheng, W. Zhou, and Y. You, "Openmoe: An early effort on open mixture-of-experts language models," in *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, 2024. 3

[56] M. A. Team *et al.*, "Mixtral of experts: A high quality sparse mixture-of-experts," *Mistral AI Blog. Accessed: December*, vol. 18, p. 2023, 2023. 3

[57] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018. 5

[58] C. Hu, P. Cao, Y. Chen, K. Liu, and J. Zhao, "Knowledge in superposition: Unveiling the failures of lifelong knowledge editing for large language models," *CoRR*, vol. abs/2408.07413, 2024. 6

[59] G. Jawahar, B. Sagot, and D. Seddah, "What does BERT learn about the structure of language?," in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 3651–3657, 2019. 7

[60] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 3449–3457, 2017. 8

[61] X. Chen, H. Fang, T. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick, "Microsoft COCO captions: Data collection and evaluation server," *CoRR*, vol. abs/1504.00325, 2015. 1

[62] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 10674–10685, 2022. 1

[63] Z. Du, Y. Qian, X. Liu, M. Ding, J. Qiu, Z. Yang, and J. Tang, "GLM: general language model pretraining with autoregressive blank infilling," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 320–335, 2022. 1

[64] K. Marino, M. Rastegari, A. Farhadi, and R. Mottaghi, "OK-VQA: A visual question answering benchmark requiring external knowledge," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 3195–3204, 2019. 1

[65] Y. Liu, H. Li, A. García-Durán, M. Niepert, D. Oñoro-Rubio, and D. S. Rosenblum, "MMKG: multi-modal knowledge graphs," in *The Semantic Web - 16th International Conference, ESWC 2019, Portorož, Slovenia, June 2-6, 2019, Proceedings*, vol. 11503 of *Lecture Notes in Computer Science*, pp. 459–474, 2019. 1

[66] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings*

of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, 2021. 1

[67] OpenAI, "GPT-4 technical report," *CoRR*, vol. abs/2303.08774, 2023. 1

[68] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), pp. 4171–4186, Association for Computational Linguistics, 2019. 2

[69] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. T. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer, "OPT: open pre-trained transformer language models," *CoRR*, vol. abs/2205.01068, 2022. 2

[70] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (K. Inui, J. Jiang, V. Ng, and X. Wan, eds.), pp. 3980–3990, Association for Computational Linguistics, 2019. 2

[71] Z. Zong, B. Ma, D. Shen, G. Song, H. Shao, D. Jiang, H. Li, and Y. Liu, "Mova: Adapting mixture of vision experts to multimodal context," in *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024* (A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, eds.), 2024. 3