Attribution Analysis Meets Model Editing: Advancing Knowledge Correction in Vision Language Models with *VisEdit*

Qizhou Chen[1], Taolin Zhang[2], Chengyu Wang[3*], Xiaofeng He[1*], Dakan Wang[4], Tingting Liu[2]

[1] East China Normal University, Shanghai, China
[2] Alibaba Group, Hangzhou, China
[3] Alibaba Cloud Computing, Hangzhou, China
[4] Exacity Inc., Shanghai, China
52265901009@stu.ecnu.edu.cn, chengyu.wcy@alibaba-inc.com, hexf@cs.ecnu.edu.cn

Abstract

Model editing aims to correct outdated or erroneous knowledge in large models without costly retraining. Recent research discovered that the mid-layer representation of the subject's final token in a prompt has a strong influence on factual predictions, and developed Large Language Model (LLM) editing techniques based on this observation. However, for Vision-LLMs (VLLMs), how visual representations impact the predictions from a decoder-only language model remains largely unexplored. To the best of our knowledge, model editing for VLLMs has not been extensively studied in the literature. In this work, we employ the contribution allocation and noise perturbation methods to measure the contributions of visual representations for token predictions. Our attribution analysis shows that visual representations in mid-to-later layers that are highly relevant to the prompt contribute significantly to predictions. Based on these insights, we propose *VisEdit*, a novel model editor for VLLMs that effectively corrects knowledge by editing intermediate visual representations in regions important to the edit prompt. We evaluated *VisEdit* using multiple VLLM backbones and public VLLM editing benchmark datasets. The results show the superiority of *VisEdit* over the strong baselines adapted from existing state-of-the-art editors for LLMs.

1 Introduction

With increasing number of LLM applications (Touvron et al. 2023; Roumeliotis et al. 2023; Zeng et al. 2023), there's a rising demand for updating the static knowledge inside the LLM using model editing techniques (Cao et al. 2021; Mitchell et al. 2022a; Yao et al. 2023). It aims to efficiently correct knowledge within LLMs without the necessity for retraining. This technology plays a key role in eliminating illusions (Mishra et al. 2024; Yin et al. 2023), reducing bias (Limisiewicz et al. 2023; Akyürek et al. 2023), and protecting privacy (Ishibashi et al. 2023; Wu et al. 2023) for LLMs.

Recent model editing works primarily focus on the text modality only. They can be broadly categorized into three types: (1) Methods which directly modify parameters (Mitchell et al. 2022a; Meng et al. 2022, 2023; Hu et al.

2024) typically locate and modify the MLP weights important for token prediction. (2) Methods which add extra modules (Huang et al. 2023; Hartvigsen et al. 2024; Yu et al. 2024) make response correction by constructing additional bypass modules for the model. (3) Methods which use prefix editing (Zheng et al. 2023; Jiang et al. 2024; Chen et al. 2024) aim to make the model follow editing instructions inserted before the input text.

Compared to single-modality cases, editing Vision-LLMs (VLLMs) that incorporate both visual and text inputs poses unique challenges, and have not been widely explored. Cheng et al. (2023) attempted to adapt LLM editors to VLLMs and established quantitative metrics for VLLM editing. Their experiments suggest that LLM editors are not very suitable for VLLMs. They hypothesize that the cause of response errors may not only stem from the weights of the LLM decoder but also from the interactions between the two modalities. In exploring the connections between text and visual modalities, some efforts (Schwettmann et al. 2023; Pan et al. 2023) focus on identifying visual neurons within the LLM decoder. Basu et al. (2024) reveal that a few final output representations of the visual encoder significantly contribute to visual constraints in the prompt at the LLM's first layer. Nonetheless, it is still unknown how visual representations affect the final predictions within the whole computation graph of VLLMs. Understanding this can better elucidate the response generation process of VLLMs and benefit VLLM editing.

To address these challenges, we first analyze the impact of visual representations on token predictions in VLLMs based on contribution allocation and noise perturbation methods. Based on the analysis results, we introduce a novel VLLM editor named *VisEdit*. Our works are summarized below.

Visual Representation Attribution: The middle part of Figure 1 illustrates the dependencies between hidden states in the computation graph of a typical VLLM. Firstly, we measure the contributions of each layer's MLP and attention modules to the key token (e.g., "bear") prediction by mapping each module's output to the predicted probability of the key token. Subsequently, we employ a noise perturbation-based attribution method (Ribeiro et al. 2016; Fong et al. 2017; Meng et al. 2022) to assess each visual representation's contribution to the subsequent layer's attention mod-
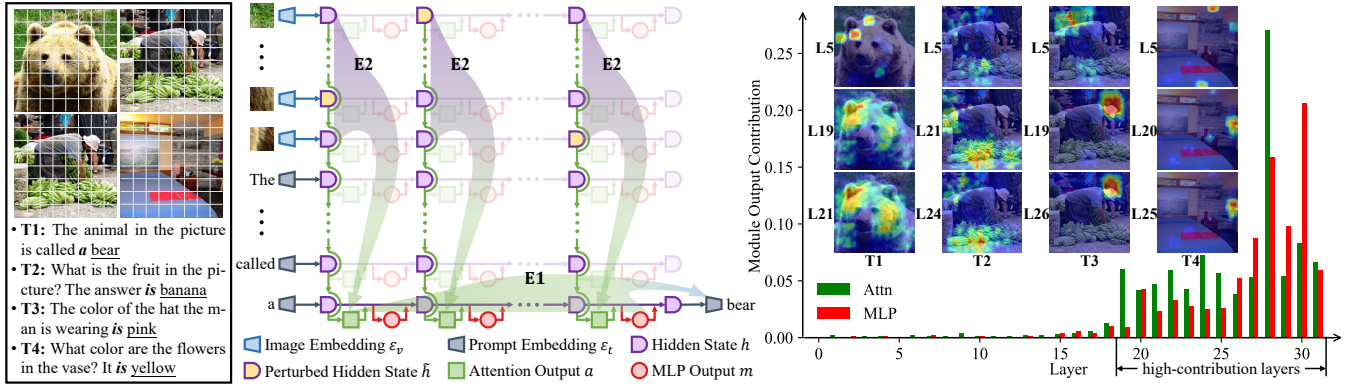

Figure 1: Attribution analysis for LLaVA-V1.5 (Liu et al. 2023). **E1:** Measuring contributions of the attention and MLP outputs at each layer to the prediction of a key token. Average results on E-VQA (Cheng et al. 2023) dataset are displayed in the bar chart. **E2:** Measuring the contributions of visual representations to the attention module outputs. The results for four samples are visualized in the heatmaps, where red indicates higher contributions and blue indicates lower. **T*** and **L*** respectively indicate the test sample index and the layer index selected by the visual representations attribution analysis. In each sample, the italicized bold text and the underlined text respectively represent the last token used for prediction and the key token to be predicted.

ule output. For one specific layer, we perturb one visual hidden state while fixing the others, and then compute a new attention output. A smaller similarity between the new attention output and the original one indicates the change in the selected visual representation affects the attention module output more significantly, or makes a greater contribution in simple terms. The brief results of these two steps are shown on the right side of Figure 1, which indicate:

- Outputs of deep model layers contribute more significantly to the key token compared with shallow layers.
- For high-contribution layers, the attention outputs are mostly influenced by the visual region of the objects mentioned in the prompts.

Based on the observations, we hypothesize that: *In VLLMs, the early layers tend to aggregate the information queried in the prompt at the last token. This setup allows deeper layers to extract information from visual representations of key regions, facilitating the generation of responses.*

The Proposed VLLM Editor *VisEdit*: According to the above results, we design an effective VLLM editor. Based on the first attribution experiment, we place a trainable visual representation adaptor before the high-contribution layer. The adaptor applies cross-attention to infuse information of the edit sample into the visual representations of a given input sample. To ensure that the adaptation of visual representations is applied to the most important regions, based on the second experiment, we introduce an influence mapper module to identify the key visual regions most relevant to the edit prompt. In this way, the editor will focus on the visual representations that are crucial for modifying key responses, thereby enhancing the efficacy of editing while leaving irrelevant visual representations mostly untouched. We conduct editing experiments on three typical VLLMs, including BLIP2-OPT (2.7B) (Li et al. 2023), MiniGPT-4 (7B) (Zhu et al. 2023), and LLaVA-V1.5 (7B) (Liu et al. 2023), with two VLLM editing datasets, E-VQA and E-IC (Cheng et al. 2023). The experimental results demonstrate

that our method excels in reliability, text/modal generality, and text/modal locality metrics, which follow standard evaluation protocols of the benchmark (Cheng et al. 2023). Furthermore, comprehensive exploratory experiments have validated the effectiveness of our module designs. Source codes are available at https://github.com/qizhou000/VisEdit.

2 Related Works

2.1 Vision Language Models

Currently, VLLMs benefited from Vision Transformer (ViT) (Dosovitskiy et al. 2021) and vision-language pre-training (Radford et al. 2021) can be generally categorized into two types (Wadekar et al. 2024), Modal Deep Fusion (MDF) and Modal Early Fusion (MEF). MDF fuses visual and language modalities within the internal layers of LLMs using cross-modal attention (Lu et al. 2019; Alayrac et al. 2022; Dong et al. 2024). MEF integrates visual modalities before feeding them into the LLM by training specific encoding modules (Ye et al. 2023; Bai et al. 2023; Li et al. 2023; Zhu et al. 2023; Liu et al. 2023). For example, BLIP2 (Li et al. 2023) and MiniGPT-4 (Zhu et al. 2023) compress visual representations through a Q-Former, while LLaVA (Liu et al. 2023) directly trains an MLP layer after the visual encoder. Due to its modular architecture, MEF offers greater extensibility and has garnered more attention compared to MDF (Wadekar et al. 2024). Therefore, this paper primarily focuses on model editing for MEF-based VLLMs.

2.2 Attribution Analysis

Attribution analysis enhances the interpretability of deep learning models (Guidotti et al. 2019), with this section focusing on perturbation-based methods. For instance, Ribeiro et al. (2016); Fong et al. (2017) use random masking or perturbation to assess contributions of image parts. Combining feature perturbation with causal mediation analysis (Pearl 2001; Vig et al. 2020), Meng et al. (2022) identify LLM

layers that significantly influence responses. For visual attribution in VLLMs, methods like integrated gradients (Sundararajan et al. 2017) are used to locate neurons recognizing specific visual concepts (Schwettmann et al. 2023; Pan et al. 2023). Basu et al. (2024) demonstrate that the last few visual representations significantly influence the visual constraints of the prompt at the first model layer. However, these works do not consider how the spatial semantics of visual representations influence VLLM prediction.

2.3 Model Editing

Model Editing for LLMs: Model editing methods can be categorized into three types: modifying parameters, adding extra modules, and editing prefix. In modifying parameter-based methods, KE (Cao et al. 2021) and MEND (Mitchell et al. 2022a) generate model weight offsets according to edit signals by training an auxiliary network. ROME (Meng et al. 2022) and MEMIT (Meng et al. 2023) utilize causal mediation analysis (Pearl 2001; Vig et al. 2020) to locate FFN weights that have a significant causal effect on the response of LLMs, and then edit them. In adding extra modules-based methods, SERAC (Mitchell et al. 2022b) trains a counterfactual model to respond to redirected queries related to the edit sample. TP (Huang et al. 2023) trains a piece of knowledge to be edited as an additional neuron. GRACE (Hartvigsen et al. 2024) maps the intermediate layer input of the query to the output in editing space if the distance between their representations is below a threshold. MELO (Yu et al. 2024) adjusts LLM weights by adding query-related editing matrices, retrieved through a mechanism similar to GRACE. In editing prefix-based methods, IKE (Zheng et al. 2023) uses in-context learning to modify LLMs' responses. LTE (Jiang et al. 2024) fine-tunes LLMs to follow editing instructions. RECIPE (Chen et al. 2024) trains a continuous prompt generator to find the shortest editing prefix. Among them, KE and IKE explored the feasibility of related technologies in single editing. SERAC, GRACE, LTE, and RECIPE, extended single editing to multiple by incorporating retrieval mechanisms. To avoid redundant discussions on editing retrieval, we primarily explore the contributing mechanisms of visual representations in VLLM to response generation to inspire the design of VLLM single editing.

Model Editing for VLLMs: To date, editing methods specific to VLLMs have not been widely studied. Cheng et al. (2023) first attempts to migrate LLM editors to edit VLLM and constructs VLLM editing datasets along with corresponding evaluation metrics. Li et al. (2024) and Zhang et al. (2024) expand the VLLM editing datasets related to entity knowledge and modality consistency, which are not yet open-sourced. To address the gap in VLLM editing research, we analyze how spatial semantics of visual representations influence VLLM responses and propose a VLLM-specific editor to encourage further research on the generation and correction of VLLMs.

3 Attribution Analysis

We evaluate the impact of the visual representations on token prediction in two steps. First, we compute the contribu-

tion of each layer's module output to the predicted key token. Then, for a specific layer, we employ a noise perturbation method to evaluate how changes in a visual hidden state (synonymous with representation in the context) affect the attention module output. Figure 1 displays the data flow of VLLM response generation and the results of two attribution experiments. For more attribution analysis results, please refer to Appendix C.

Given a VLLM $f_\theta : \mathcal{X}_v \times \mathcal{X}_t \mapsto \mathcal{O}$ that maps an image-prompt pair (x_v, x_t) into a text response $o = f_\theta(x_v, x_t)$, we set $\hat{f}_\theta : \mathcal{E}_v \times \mathcal{E}_t \mapsto \mathcal{Y}$ as the transformer in f_θ that maps an embedding sequence $\varepsilon = \varepsilon_v \oplus \varepsilon_t \in \mathcal{R}^{N \times d_h}$ to a probability distribution $y \in \mathcal{Y} \subset \mathcal{R}^{|\mathcal{V}|}$ predicting the next token based on vocabulary $\mathcal{V}$, where $\varepsilon_v \in \mathcal{E}_v \subset \mathcal{R}^{N_v \times d_h}$ is the image embedding and $\varepsilon_t \in \mathcal{E}_t \subset \mathcal{R}^{N_t \times d_h}$ is the prompt embedding. N_v, N_t, N are the length of visual, text, and complete embeddings, respectively. d_h is the middle dimension of $\hat{f}_\theta$ and $\oplus$ denotes concatenation. In our attribution experiments, we represent each sample as a tuple $(\varepsilon_v, \varepsilon_t, o^*)$, containing the image embedding ε_v, the prompt embedding ε_t, and the key token o^* to be predicted. The symbols and notations used in this paper are provided in Appendix A.

3.1 Module Output Attribution

In a transformer $\hat{f}_\theta$, each hidden representation $h_n^l \in \mathcal{R}^{d_h}$ of n-th token at l-th layer can be obtained from (Vaswani et al. 2017):

$$h_n^l = h_n^{l-1} + a_n^l + m_n^l, l \in \{1, ..., L\}, n \in \{1, ..., N\} \quad (1)$$

where h^0 equals ε, and $a_n^l = \text{Attn}_l(h_1^{l-1}, ..., h_n^{l-1})$, $m_n^l = \text{MLP}_l(h_n^{l-1} + a_n^l)$ are respectively the outputs of attention and MLP modules. L is the layer count of $\hat{f}_\theta$. Given an input $(\varepsilon_v, \varepsilon_t)$, the predicted probability distribution of the next token is $p = \delta(h_N^L W_\mathcal{V})$, where δ is the softmax function, and $W_\mathcal{V} \in \mathcal{R}^{d_h \times |\mathcal{V}|}$ is the matrix mapping hidden states into logits in vocabulary space $\mathcal{V}$. Unrolling Equation 1, we have $h_N^L = h_N^0 + \sum_{l=1}^L (a_N^l + m_N^l)$. Multiplying both sides by $W_\mathcal{V}$ gives:

$$h_N^L W_\mathcal{V} = h_N^0 W_\mathcal{V} + \sum_{l=1}^L (a_N^l W_\mathcal{V} + m_N^l W_\mathcal{V}) \quad (2)$$

It shows that the token prediction distribution depends on the sum of the outputs from all attention and MLP modules across layers. Therefore, we quantify the contribution of a module output r, which can be either a_N^l or m_N^l, to the prediction of the key token o^*, as below:

$$C_{o^*}(r) = \sqrt{C_{o^*}^p(r) \cdot C_{o^*}^v(r)} \quad (3)$$

$$C_{o^*}^p(r) = \delta(r W_\mathcal{V})_{o^*} \quad (4)$$

$$C_{o^*}^v(r) = \frac{(r W_\mathcal{V})_{o^*}}{\max\limits_{l=1}^{L} \max\left(|(a_N^l W_\mathcal{V})_{o^*}|, |(m_N^l W_\mathcal{V})_{o^*}|\right)} \quad (5)$$

The definition takes into account both the mapped probability $C_{o^*}^p(r)$ and the normalized logit value $C_{o^*}^v(r)$. The rationale for this definition is that, a large mapping probability

does not necessarily lead to a significant contribution to the final prediction when the mapping logit value is small.

The bar chart in Figure 1 illustrates the average results for LLaVA-V1.5 (7B) (Liu et al. 2023) on the E-VQA dataset (Cheng et al. 2023), showing that the outputs from deeper layers have a more substantial impact on the key token than those from shallower layers.

3.2 Visual Representation Attribution

Noting that the contribution trend of the attention and MLP modules across layers are similar, we primarily focus on the contributions of visual representations to the next layer attention output. Given an attention output $a_N^l = \text{Attn}_l(h_1^{l-1}, ..., h_{N_v}^{l-1}, ..., h_N^{l-1})$, we perturb each visual representation by set:

$$\tilde{h}_n^{l-1} = h_n^{l-1} + \epsilon, n \in \{1, ..., N_v\} \quad (6)$$

where $\epsilon \in \mathcal{R}^{d_h}$ follows $\mathcal{N}\left(0; (3\sigma)^2\right)$ (Meng et al. 2022) and σ is the standard deviation of elements in $h_{1:N_g}^{l-1} \in \mathcal{R}^{N_v \times d_h}$. Building on the insight that perturbing a significant module input will noticeably alter its output (Ribeiro et al. 2016; Fong et al. 2017), we quantify the contribution of each visual representation h_n^{l-1} to the attention output as follows:

$$C_{a_N^l}(h_n^{l-1}) = \frac{1}{2}\left(1 - \frac{\tilde{a}_N^l \cdot a_N^l}{\|\tilde{a}_N^l\| \cdot \|a_N^l\|}\right) \in [0, 1] \quad (7)$$

$$\tilde{a}_N^l = \text{Attn}_l(h_1^{l-1}, ..., h_{n-1}^{l-1}, \tilde{h}_n^{l-1}, h_{n+1}^{l-1}, ..., h_N^{l-1}) \quad (8)$$

The heatmaps in Figure 1 display the visualization results (For more results, please refer to Appendix C.2). It can be observed that the model focuses on areas highly relevant to the prompt at deep layers.

Combining the above two attribution experiments, we hypothesize that the VLLM initially aggregates the semantics of a given prompt into the last token representation at shallow layers, and then extracts key information from visual representations at deep layers to generate responses.

4 The Proposed Editor

Inspired by the results of attribution analysis, we devise a VLLM editor named *VisEdit*. In this section, we first introduce the preliminaries of VLLM editing (Cheng et al. 2023). Then, we elaborate on the basic structure of *VisEdit*. Specifically, we introduce an Influence Mapper to help the editor focus on key visual regions based on the prompt, thereby reducing the negative impact on irrelevant visual representations and enhancing editing efficacy. Finally, we describe the training process of *VisEdit*.

4.1 Preliminaries

For a VLLM $f_\theta \in \mathcal{F}$, given an edit sample (x_v^e, x_t^e, o^e) such that $f_\theta(x_v^e, x_t^e) \neq o^e$, an VLLM editor ME : $\mathcal{F} \times \mathcal{X}_v \times \mathcal{X}_t \times \mathcal{O} \mapsto \mathcal{F}$ outputs a post-edit VLLM $f_{\theta_e} = \text{ME}(f_\theta, x_v^e, x_t^e, o^e)$. A good ME should meet the following three criteria (Cheng et al. 2023):

Reliability assesses the response accuracy of the post-edit model on the edited samples:

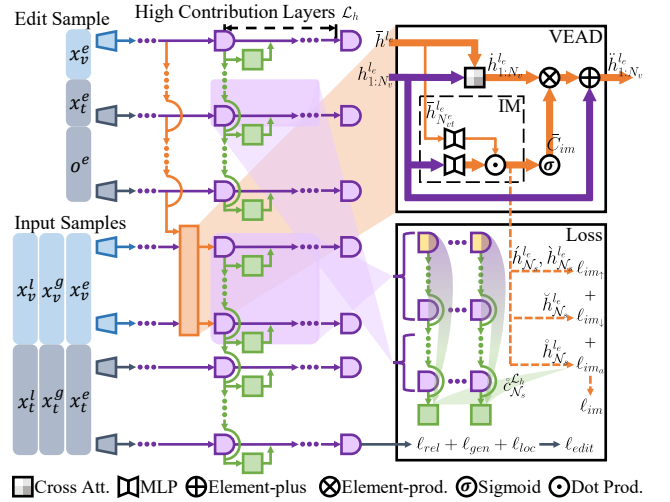$$\mathbb{E}_{(x_v^e, x_t^e, o^e) \sim \mathcal{D}_e} \mathbb{I}\{f_{\theta_e}(x_v^e, x_t^e) = o^e\}$$

Figure 2: Architecture and training loss of *VisEdit*.

where $\mathcal{D}_e$ is edit sample set and $\mathbb{I}$ is the indicator function.

Generality requires the revised model to also make corresponding adjustments to the relevant neighborhoods (e.g., rephrased sentences) of the edited samples, including modal generality and text generality:

$$\mathbb{E}_{(x_v^e, x_t^e, o^e) \sim \mathcal{D}_e} \mathbb{E}_{x_v^{mg} \sim \mathcal{D}_{mg}(x_v^e)} \mathbb{I}\{f_{\theta_e}(x_v^{mg}, x_t^e) = o^e\}$$
$$\mathbb{E}_{(x_v^e, x_t^e, o^e) \sim \mathcal{D}_e} \mathbb{E}_{x_t^{tg} \sim \mathcal{D}_{tg}(x_t^e)} \mathbb{I}\{f_{\theta_e}(x_v^e, x_t^{tg}) = o^e\}$$

where $\mathcal{D}_{mg}(x_v^e), \mathcal{D}_{tg}(x_t^e)$ respectively represent the neighborhoods of the image x_v^e and prompt x_t^e.

Locality requires the revised model to make response consistent with the original model for samples unrelated to the edited samples, including modal locality and text locality:

$$\mathbb{E}_{(x_v^e, x_t^e, o^e) \sim \mathcal{D}_e} \mathbb{E}_{(x_v^{ml}, x_t^{ml}, o^{ml}) \sim \mathcal{D}_{ml}(x_v^e, x_t^e)}$$
$$\mathbb{I}\{f_{\theta_e}(x_v^{ml}, x_t^{ml}) = f_\theta(x_v^{ml}, x_t^{ml})\}$$

$$\mathbb{E}_{(x_v^e, x_t^e, o^e) \sim \mathcal{D}_e} \mathbb{E}_{(x_t^{tl}, o^{tl}) \sim \mathcal{D}_{tl}(x_t^e)}$$
$$\mathbb{I}\{f_{\theta_e}(\emptyset, x_t^{tl}) = f_\theta(\emptyset, x_t^{tl})\}$$

where $\mathcal{D}_{ml}(x_v^e, x_t^e), \mathcal{D}_{tl}(x_t^e)$ respectively represent the multi-modal and text samples irrelevant to the edit sample.

4.2 Architecture of *VisEdit*

Based on attribution analysis, we observe that that the VLLM aggregates the semantics of the prompt into the last token in shallow layers; while in deep layers, it extracts information from visual representations related to the prompt to generate responses. Inspired by this, we devise a VLLM editor, namely *VisEdit*. Specifically, as shown in Figure 2, it inserts a Visual Edit ADapter (VEAD) before the high-contribution layers. The adapter modifies the visual representations that the edit sample attends to. Below, we assume that VEAD is inserted at the l_e-th layer of the VLLM and specify two steps in detail: (1) Compute edit signal from an edit sample. (2) Use the edit signal to adapt the model for future inputs.

Compute Edit Signals: Given an edit sample (x_v^e, x_t^e, o^e), VEAD feeds $(x_v^e, x_t^e \oplus o^e)$ into f_θ and takes the l_e-th layer output $\bar{h}^{l_e} \in \mathcal{R}^{(N_v + N_t^e + N_o^e) \times d_h}$ as the edit signal. Here, $\oplus$ denotes string concatenation, and N_v, N_t^e, N_o^e are the dimensions of representations of x_v^e, x_t^e, o^e respectively.

Adapt Hidden State with Influence Mapper: With the edit signal $\bar{h}^{l_e}$ defined, we now demonstrate how to edit the original visual representation $h_{1:N_v}^{l_e} \in \mathcal{R}^{N_v \times d_h}$ for a given (x_v, x_t). Specifically, a cross-attention operation is applied to integrate the edit signal into $h_{1:N_v}^{l_e}$, i.e.,

$$\acute{h}_{1:N_v}^{l_e} = \delta \left(h_{1:N_v}^{l_e} W_1 \left(\bar{h}^{l_e} W_2 \right)^T \right) \bar{h}^{l_e} W_3 \qquad (9)$$

where $W_1 \in \mathcal{R}^{d_h \times d_a}, W_2 \in \mathcal{R}^{d_h \times d_a}, W_3 \in \mathcal{R}^{d_h \times d_h}$ are the projection matrices, with related biases omitted here. δ is the softmax function, and d_a is the dimension of modules in VEAD. To make sure that the edit signal is not applied to locality samples or visual regions irrelevant to the edit prompt within relevant samples, we further incorporate an Influence Mapper (IM) module f_{im} to control the edit intensity of the adaptation. Based on the previous attribution analysis, we design IM to use the last token of the edit prompt to generate the edit intensity $\bar{C}_{im} \in \mathcal{R}^{N_v \times 1}$ for the visual regions, defined as follows:

$$\begin{aligned}
\bar{C}_{im} &= \sigma \left(f_{im} \left(h_{1:N_v}^{l_e}, \bar{h}_{N_{vt}^e}^{l_e} \right) \right) \\
&= \sigma \left(f_{\mu_1} \left(h_{1:N_v}^{l_e} \right) \cdot f_{\mu_2} \left(\bar{h}_{N_{vt}^e}^{l_e} \right)^\top \right)
\end{aligned} \qquad (10)$$

where $N_{vt}^e = N_v + N_t^e$, and thus $\bar{h}_{N_{vt}^e}^{l_e} \in \mathcal{R}^{d_h}$ is the representation according to the last token embedding of x_t^e in $\bar{h}^{l_e}$. f_{μ_1}, f_{μ_2} are two-layers MLPs, mapping dimension to d_a. σ is the sigmoid function. Finally, the adapted visual representations $\acute{h}_{1:N_v}^{l_e}$ is formulated as:

$$\ddot{h}_{1:N_v}^{l_e} = h_{1:N_v}^{l_e} + \acute{h}_{1:N_v}^{l_e} \times \bar{C}_{im} \qquad (11)$$

where $\times$ indicates element-wise product with broadcast.

4.3 Training Process of *VisEdit*

The training loss of *VisEdit* includes the editing loss and the IM loss, designed to meet the editing requirements and enable IM to learn the edit intensity, respectively.

Editing Loss: Given an edit sample (x_v^e, x_t^e, o^e), corresponding generality samples x_v^{mg}, x_t^{tg}, and locality samples $(x_v^{ml}, x_t^{ml}, o^{ml}), (x_t^{tl}, o^{tl})$, the editing loss is the sum of the reliability loss ℓ_{rel}, the generality loss ℓ_{gen}, and the locality loss ℓ_{loc}, defined as follows:

$$\ell_{edit} = \ell_{rel} + \ell_{gen} + \ell_{loc} \qquad (12)$$

where

$$\ell_{rel} = -\log f_{\theta_e} \left(o^e | x_v^e, x_t^e \right) \qquad (13)$$

$$\ell_{gen} = -\log f_{\theta_e} \left(o^e | x_v^{mg}, x_t^e \right) - \log f_{\theta_e} \left(o^e | x_v^e, x_t^{tg} \right) \quad (14)$$

$$\begin{aligned}
\ell_{loc} = &\mathrm{KL} \left(f_\theta \left(o^{ml} | x_v^{ml}, x_t^{ml} \right) || f_{\theta_e} \left(o^{ml} | x_v^{ml}, x_t^{ml} \right) \right) \\
&+ \mathrm{KL} \left(f_\theta \left(o^{tl} | \emptyset, x_t^{tl} \right) || f_{\theta_e} \left(o^{tl} | \emptyset, x_t^{tl} \right) \right)
\end{aligned} \qquad (15)$$

Here KL denotes the Kullback-Leibler divergence.

IM Loss: As shown in Eq. 11, IM controls the edit intensity. For reliability and generality samples, the intensity should be large, whereas for locality samples, it should be small. Therefore, in each training iteration, we randomly sample a portion of visual representations to compute the corresponding loss:

$$\begin{aligned}
\ell_{im_\uparrow} = \frac{1}{|\mathcal{N}_s|} \sum_{n \in \mathcal{N}_s} &\left(-\log \sigma \left(f_{im} \left(\acute{h}_n^{l_e}, \bar{h}_{N_{vt}^e}^{l_e} \right) \right) \right. \\
&\left. -\log \sigma \left(f_{im} \left(\grave{h}_n^{l_e}, \bar{h}_{N_{vt}^e}^{l_e} \right) \right) \right)
\end{aligned} \qquad (16)$$

$$\ell_{im_\downarrow} = \frac{1}{|\mathcal{N}_s|} \sum_{n \in \mathcal{N}_s} -\log \left(1 - \sigma \left(f_{im} \left(\breve{h}_n^{l_e}, \bar{h}_{N_{vt}^e}^{l_e} \right) \right) \right) \quad (17)$$

where $\mathcal{N}_s \subset \{1, ..., N_v\}$ is the sample index, and $\acute{h}, \grave{h}, \breve{h}$ are the representations of reliability, generality, and locality samples, respectively. To guide IM to put attention on visual regions highly relevant to the edit prompt, we regularize IM to approximate the VLLM-focused visual regions, through visual representation attribution on the sampled visual representations based on the edit prompt. Assuming the set of high-contribution layers is $\mathcal{L}_h \subset \{1, ..., L\}$, we first replace the visual representations of the edit sample, i.e., $\bar{h}$, from layers in $\mathcal{L}_h$ with the counterpart of $\mathring{h}$ randomly selected from $\{\acute{h}, \grave{h}\}$. Then, following Equation 7, we compute the contribution of visual representations in $\mathring{h}$ to the edit prompt as:

$$\mathring{c}_n^l = C_{\bar{a}_{N_{vt}^e}^l} \left(\mathring{h}_n^l \right) \in [0, 1], l \in \mathcal{L}_h, n \in \mathcal{N}_s \qquad (18)$$

where $\bar{a}_{N_{vt}^e}^l$ represents the l-th layer attention output at the last token of the edit prompt. Then, we minimize the discrepancy between IM's mapping results and the attribution results averaged across layers using cross-entropy loss:

$$\ell_{im_a} = \sum_{\substack{n \in \mathcal{N}_s \\ l \in \mathcal{L}_h}} \frac{-\mathring{c}_n^l}{|\mathcal{L}_h| \sum_{j \in \mathcal{N}_s} \mathring{c}_j^l} \log \delta \left(f_{im} \left(\mathring{h}_{\mathcal{N}_s}^{l_e}, \bar{h}_{N_{vt}^e}^{l_e} \right) \right)_n \quad (19)$$

where $\mathring{h}_{\mathcal{N}_s}^{l_e} \in \mathcal{R}^{|\mathcal{N}_s| \times d_h}$ indicates the sampled representations from l_e-th layer, and $\delta(\cdot)_n$ represents the n-th element of the vector outputted from softmax. Thus, the IM loss is formulated as $\ell_{im} = \ell_{im_\uparrow} + \ell_{im_\downarrow} + \ell_{im_a}$.

The total loss of *VisEdit* is: $\ell_{total} = \ell_{edit} + \ell_{im}$. During training, we freeze the parameters of the VLLM and only the parameters in VEAD are updated.

5 Experiments

5.1 Experimental Settings

Datasets: Following Cheng et al. (2023), we employ E-VQA (Editing Visual Question Answering) and E-IC (Editing Image Caption) as evaluation datasets.

VLLM Backbones: To ensure a comprehensive evaluation, we consider both the parameter size and model architecture when selecting VLLM backbones for editing, including BLIP2-OPT (2.7B) (Li et al. 2023), LLaVA-V1.5 (7B) (Liu et al. 2023), and MiniGPT-4 (7B) (Zhu et al. 2023).

Backbone	Editor	E-VQA						E-IC					
		Rel.	T-Gen.	M-Gen.	T-Loc.	M-Loc.	Average	Rel.	T-Gen.	M-Gen.	T-Loc.	M-Loc.	Average
BLIP2-OPT (2.7B)	FT-V	24.01	16.00	20.22	100.00	88.65	49.78$_{(\pm0.47)}$	42.11	40.74	36.43	100.00	89.73	61.80$_{(\pm0.40)}$
	FT-L	24.86	16.39	20.57	98.92	89.61	50.07$_{(\pm0.60)}$	41.61	40.31	37.41	99.35	88.70	61.48$_{(\pm0.77)}$
	KE	67.80	63.00	66.17	97.32	45.89	68.04$_{(\pm0.00)}$	69.00	62.80	61.22	96.21	45.55	66.96$_{(\pm0.00)}$
	IKE	**99.95**	91.59	92.33	13.16	1.88	59.78$_{(\pm0.00)}$	96.70	78.20	83.15	13.36	2.17	54.72$_{(\pm0.00)}$
	SERAC	91.20	91.40	89.81	100.00	0.33	74.55$_{(\pm0.00)}$	94.40	96.00	91.49	100.00	0.47	76.47$_{(\pm0.00)}$
	MEND	92.60	90.80	91.94	96.07	65.15	87.31$_{(\pm0.00)}$	65.00	38.00	36.19	92.67	55.72	57.52$_{(\pm0.00)}$
	TP	68.31	60.88	56.35	98.49	85.27	73.86$_{(\pm0.99)}$	49.71	49.03	45.46	93.88	80.88	63.79$_{(\pm1.11)}$
	LTE	97.74	97.21	96.35	94.34	84.99	94.13$_{(\pm0.97)}$	96.69	95.26	94.06	95.25	87.68	93.79$_{(\pm1.05)}$
	VisEdit	98.83	**98.63**	**97.90**	100.00	92.30	**97.53**$_{(\pm0.70)}$	97.06	96.83	94.85	100.00	91.74	96.10$_{(\pm0.94)}$
LLaVA-V1.5 (7B)	FT-V	31.68	29.96	26.68	100.00	91.23	55.91$_{(\pm0.71)}$	52.85	51.57	48.63	100.00	92.55	69.12$_{(\pm0.29)}$
	FT-L	31.78	30.02	26.91	99.94	92.03	56.14$_{(\pm2.13)}$	53.00	51.02	49.29	98.91	94.89	69.42$_{(\pm1.71)}$
	KE	85.86	84.00	82.23	93.57	73.06	83.74$_{(\pm1.25)}$	83.54	82.15	81.12	92.46	73.83	82.62$_{(\pm0.88)}$
	IKE	91.35	90.84	91.08	60.18	51.08	76.91$_{(\pm1.42)}$	93.72	88.37	76.99	76.60	64.90	80.12$_{(\pm1.18)}$
	SERAC	82.51	81.60	80.05	100.00	57.48	80.33$_{(\pm1.58)}$	43.08	42.37	42.85	100.00	7.63	47.19$_{(\pm0.83)}$
	MEND	92.30	92.16	92.10	90.30	81.13	89.60$_{(\pm2.36)}$	93.76	93.46	92.14	91.60	87.59	91.71$_{(\pm1.42)}$
	TP	38.68	36.27	31.26	95.31	91.41	58.59$_{(\pm1.32)}$	59.07	57.01	55.51	64.79	89.26	65.13$_{(\pm1.85)}$
	LTE	94.16	93.54	93.06	83.76	81.65	89.23$_{(\pm1.90)}$	93.60	92.38	91.18	85.54	88.49	90.24$_{(\pm1.90)}$
	VisEdit	**95.99**	**95.78**	**94.71**	100.00	**94.12**	**96.12**$_{(\pm0.97)}$	95.27	94.64	93.57	100.00	96.20	**95.94**$_{(\pm0.90)}$

Table 1: Editing performance of BLIP2-OPT and LLaVA-V1.5 evaluated on E-VQA and E-IC datasets. "Rel.", "T/M-Gen." and "T/M-Loc." stand for reliability, text/modal generality, and text/modal locality, respectively. Results with a gray background are taken from Cheng et al. (2023). The t-tests demonstrate our improvements are statistically significant with $p < 0.05$ level.

Baseline Editors: To the best of our knowledge, there are currently no editors specifically designed for VLLMs. Therefore, most existing works (Cheng et al. 2023; Li et al. 2024) applied LLM editors in the VLLM case, including FT-V (Fine-tunes visual encoder), TF-L (Fine-tunes last layer of the language model), KE (Cao et al. 2021), IKE (Zheng et al. 2023), SERAC (Mitchell et al. 2022b), MEND (Mitchell et al. 2022a), TP (Huang et al. 2023), and LTE (Jiang et al. 2024). For experimental setup details, as well as the model hyperparameters and training specifics, please refer to Appendix B.

Using the above experimental settings, we comprehensively evaluate the editing performance and conduct in-depth quantitative analyses of VEAD internals to validate that it effectively incorporates insights from the attribution analysis.

5.2 Analysis of Editing Performance

The overall editing performance is exhibited in Table 1 and Appendix D.1. Here we analyze Table 1 from different perspectives.

From the perspective of editors, *VisEdit* demonstrates the best performance. LTE also excels because it fine-tunes the entire LLM. However, unlike pure language models, the combination of visual and language representations makes it harder for the model to learn the mode following editing instructions.

From the perspective of backbones, VEAD performs slightly better on BLIP2. We ascribe this to VEAD intervening only in a local part of the VLLM's data pathway, where the effect of this intervention would be relatively more significant in a smaller model.

From the perspective of datasets, most editors perform better on E-VQA than on E-IC. This is due to E-VQA involving the correction of a few key tokens, while E-IC requires editing captions that convey complete image infor-

Layer	Rel.	T-Gen.	M-Gen.	T-Loc.	M-Loc.	Average
0	95.10	93.70	94.43	100.00	84.51	93.55
5	95.16	94.63	94.49	100.00	86.51	94.16
10	96.36	95.79	95.60	100.00	85.83	94.72
15	97.69	97.48	97.08	100.00	92.20	96.89
19*	**98.83**	**98.63**	**97.90**	**100.00**	**92.30**	**97.53**
25	97.54	96.97	95.87	100.00	88.63	95.80
30	86.22	84.18	83.62	100.00	85.98	88.00

Table 2: Editing results on E-VQA when VEAD is attached to different layers of BLIP2-OPT (2.7B), where the layer marked with an asterisk represents our selection based on the first attribution experiment.

mation, making it more challenging.
From the perspective of metrics, editors with lower average scores often show that high reliability and generality cannot coexist with locality. Additionally, because FT-V and *VisEdit* only edit the visual component, and SERAC uses classifiers to distinguish pure text input, they effectively avoid interference from the editing process on the text locality samples.

5.3 Analysis of VEAD Internals

We conduct three sets of experiments to demonstrate that VEAD fully incorporates insights from the attribution analysis. First, we compare VEAD's editing performances across different layers. Next, we visualize the outputs of IM to demonstrate that it attends to the visual regions important for the edit prompt. Finally, we verify whether VEAD successfully adapts the visual representations critical for response generation.
Editing Performance in Different Layers: Table 2 displays the editing performance when VEAD is applied to various layers of BLIP2. We can observe that the editing performance first gets better as we go deeper until layer 19, and then gets worse for later layers. We hypothesize that the edit signal in too shallow layers is prone to lose before reaching
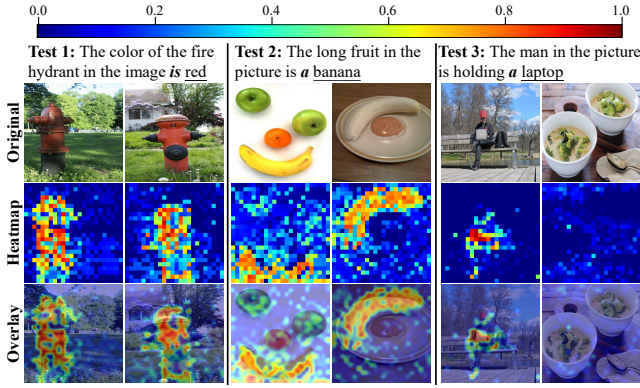
Figure 3: The visualization of IM module in VEAD integrated into LLaVA-V1.5. In each test, VEAD is first edited using the left image along with the prompt. Then, the outputs of IM are visualized for both images.
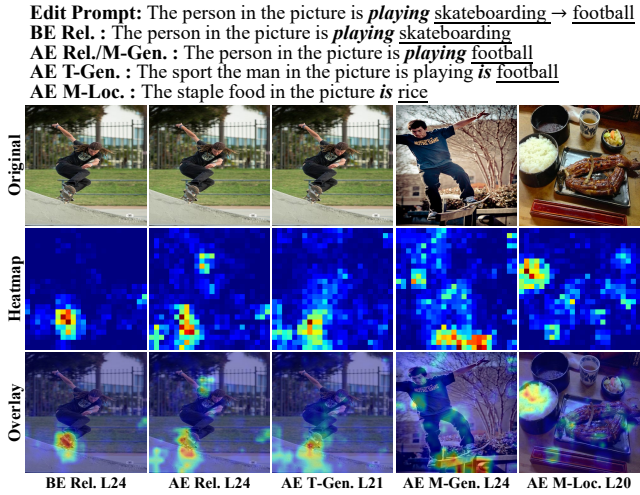
Edit Prompt: The person in the picture is ***playing*** skateboarding → football
BE Rel. : The person in the picture is ***playing*** skateboarding
AE Rel./M-Gen. : The person in the picture is ***playing*** football
AE T-Gen. : The sport the man in the picture is playing ***is*** football
AE M-Loc. : The staple food in the picture ***is*** rice

Figure 4: Visualization of visual representation attribution after *VisEdit* edits a counterfactual sample. "**BE**" and "**AE**" indicate Before Editing and After Editing respectively. **L*** indicates the layer index.

the deep layers that highly contribute to prediction, while for cases in too deep layers, the signal cannot have a substantial enough impact on the final prediction.

Visualization of IM Module: We visualize the outputs of IM in Figure 3 (For more results and analysis, please refer to Appendix D.2). Specifically, Test 1 and Test 2 indicate that IM indeed picks the relevant visual region utilizing the overall semantics of the edit prompt, even when the visual objects in the edited samples are not entirely consistent with those in the input samples, e.g., the peeled and unpeeled bananas in test 2. Test 3 demonstrates that the visual areas of samples unrelated to the edit sample will receive low edit intensities from IM, because of ℓ_{im_a} and $\ell_{im_\downarrow}$.

Instance Analysis for VEAD: Figure 4 illustrates a counterfactual edit example where VEAD forces LLaVA-V1.5 to follow the knowledge even if it is incorrect (For more results and analysis, please refer to Appendix D.3). We first compare the visualization results of three cases: reliability sam-

Settings	Rel.	T-Gen.	M-Gen.	T-Loc.	M-Loc.	Average
VEAD	**95.99**	**95.78**	**94.71**	**100.00**	94.12	**96.12**
- $\ell_{im_\downarrow}$	95.50	94.38	93.87	100.00	86.72	94.09
- $\ell_{im_\uparrow}$	94.13	93.82	93.58	100.00	93.71	95.05
- ℓ_{im_a}	93.70	93.05	92.51	100.00	90.18	93.89
- IM	93.13	92.58	91.89	100.00	83.84	92.29
- CA	33.79	31.98	28.94	100.00	**98.98**	58.74

Table 3: Ablation study of VEAD.

ples before editing (BE Rel.), reliability samples after editing (AE Rel.), and generality samples after editing (AE T/M-Gen.). We observe that the editing process does not significantly change the visual regions VLLM focuses on for similar prompts, even if the "skateboarding" is counterfactually edited to "football". The above observation indicates that VEAD adapts the visual representations in the skateboard region to new visual representations for football, and furthermore alters the final prediction to football, thereby validating the design objective of our model. Additionally, the visualization of modal locality samples shows that VEAD maintains VLLM's original attention to visual objects unrelated to the edit.

5.4 Ablation Study

Table 3 shows the results of the ablation study for VEAD editing LLaVA-V1.5 on E-VQA. Overall, the removal of losses resulted in some degradation in reliability, generality, and locality. Specifically, removing $\ell_{im_\downarrow}$ caused significant damage to modal locality, as it makes IM output a smaller edit intensity for samples unrelated to the edit sample. $\ell_{im_\uparrow}$ and ℓ_{im_a} encourage IM to apply higher edit intensities to visual objects that are consistent with the semantics of the edit prompt; their removal leads to a relatively significant degradation in both reliability and generality scores. Additionally, removing the entire IM module fixes the edit intensity to 1, preventing VEAD from focusing on specific visual regions for modifications and increasing erroneous interventions in unrelated samples. Since the CA (Cross-Attention) module is the key component for integrating the edit signal, its removal essentially caused VEAD to lose its editing capability. These results show the effectiveness of the modules and loss functions designed in VEAD.

6 Conclusion

This paper proposes a two-step attribution method combining contribution allocation and noise perturbation to measure the impact of visual representations on token prediction in VLLM. Results reveal that mid-to-late layers of VLLM focus on visual regions closely related to prompt semantics. Leveraging these insights, we introduce *VisEdit*, a novel VLLM editor with a visual editing adapter that integrates editing information into high-contribution layers. The editor incorporates an IM module to enhance relevant edit signals while ignoring irrelevant ones. *VisEdit* achieves outstanding performance in VLLM editing tasks. Future work will explore deeper insights into VLLM attributions and extend VEAD to broader editing scenarios.

Acknowledgments

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. arXiv:2303.08774.

Akyürek, A. F.; Pan, E.; Kuwanto, G.; and Wijaya, D. 2023. DUnE: Dataset for Unified Editing. In *EMNLP*, 1847–1861.

Alayrac, J.; Donahue, J.; Luc, P.; Miech, A.; Barr, I.; Hasson, Y.; Lenc, K.; Mensch, A.; Millican, K.; Reynolds, M.; Ring, R.; Rutherford, E.; Cabi, S.; Han, T.; Gong, Z.; Samangooei, S.; Monteiro, M.; Menick, J. L.; Borgeaud, S.; Brock, A.; Nematzadeh, A.; Sharifzadeh, S.; Binkowski, M.; Barreira, R.; Vinyals, O.; Zisserman, A.; and Simonyan, K. 2022. Flamingo: a Visual Language Model for Few-Shot Learning. In *NeurIPS*.

Bai, J.; Bai, S.; Yang, S.; Wang, S.; Tan, S.; Wang, P.; Lin, J.; Zhou, C.; and Zhou, J. 2023. Qwen-VL: A Frontier Large Vision-Language Model with Versatile Abilities. arXiv:2308.12966.

Basu, S.; Grayson, M.; Morrison, C.; Nushi, B.; Feizi, S.; and Massiceti, D. 2024. Understanding Information Storage and Transfer in Multi-modal Large Language Models. arXiv:2406.04236.

Cao, N. D.; Aziz, W.; Titov, I.; and . 2021. Editing Factual Knowledge in Language Models. In *EMNLP*, 6491–6506.

Chen, Q.; Zhang, T.; He, X.; Li, D.; Wang, C.; Huang, L.; and Xue, H. 2024. Lifelong Knowledge Editing for LLMs with Retrieval-Augmented Continuous Prompt Learning. arXiv:2405.03279.

Chen, X.; Fang, H.; Lin, T.; Vedantam, R.; Gupta, S.; Dollár, P.; and Zitnick, C. L. 2015. Microsoft COCO Captions: Data Collection and Evaluation Server. arXiv:1504.00325.

Cheng, S.; Tian, B.; Liu, Q.; Chen, X.; Wang, Y.; Chen, H.; and Zhang, N. 2023. Can We Edit Multimodal Large Language Models? In *EMNLP*, 13877–13888.

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

Dong, X.; Zhang, P.; Zang, Y.; Cao, Y.; Wang, B.; Ouyang, L.; Wei, X.; Zhang, S.; Duan, H.; Cao, M.; Zhang, W.; Li, Y.; Yan, H.; Gao, Y.; Zhang, X.; Li, W.; Li, J.; Chen, K.; He, C.; Zhang, X.; Qiao, Y.; Lin, D.; and Wang, J. 2024. InternLM-XComposer2: Mastering Free-form Text-Image Composition and Comprehension in Vision-Language Large Model. arXiv:2401.16420.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*.

Du, Z.; Qian, Y.; Liu, X.; Ding, M.; Qiu, J.; Yang, Z.; and Tang, J. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *ACL*, 320–335.

Fong, R. C.; Vedaldi, A.; .; and . 2017. Interpretable Explanations of Black Boxes by Meaningful Perturbation. In *ICCV*, 3449–3457.

Goyal, Y.; Khot, T.; Summers-Stay, D.; Batra, D.; and Parikh, D. 2017. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. In *CVPR*, 6325–6334.

Guidotti, R.; Monreale, A.; Ruggieri, S.; Turini, F.; Giannotti, F.; and Pedreschi, D. 2019. A Survey of Methods for Explaining Black Box Models. *ACM Comput. Surv.*, 51(5): 93:1–93:42.

Hartvigsen, T.; Sankaranarayanan, S.; Palangi, H.; Kim, Y.; and Ghassemi, M. 2024. Aging with grace: Lifelong model editing with discrete key-value adaptors. In *NeurIPS*, volume 36.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8): 1735–1780.

Hu, C.; Cao, P.; Chen, Y.; Liu, K.; and Zhao, J. 2024. WilKE: Wise-Layer Knowledge Editor for Lifelong Knowledge Editing. arXiv:2402.10987.

Huang, Z.; Shen, Y.; Zhang, X.; Zhou, J.; Rong, W.; and Xiong, Z. 2023. Transformer-Patcher: One Mistake Worth One Neuron. In *ICLR*.

Ishibashi, Y.; Shimodaira, H.; .; and . 2023. Knowledge Sanitization of Large Language Models. arXiv:2309.11852.

Jiang, Y.; Wang, Y.; Wu, C.; Zhong, W.; Zeng, X.; Gao, J.; Li, L.; Jiang, X.; Shang, L.; Tang, R.; Liu, Q.; and Wang, W. 2024. Learning to Edit: Aligning LLMs with Knowledge Editing. arXiv:2402.11905.

Li, J.; Du, M.; Zhang, C.; Chen, Y.; Hu, N.; Qi, G.; Jiang, H.; Cheng, S.; and Tian, B. 2024. MIKE: A New Benchmark for Fine-grained Multimodal Entity Knowledge Editing. arXiv:2402.14835.

Li, J.; Li, D.; Savarese, S.; and Hoi, S. C. H. 2023. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. In *ICML*, volume 202, 19730–19742.

Limisiewicz, T.; Marecek, D.; Musil, T.; and . 2023. Debiasing Algorithm through Model Adaptation. arXiv:2310.18913.

Liu, H.; Li, C.; Wu, Q.; and Lee, Y. J. 2023. Visual Instruction Tuning. In *NeurIPS*.

Lu, J.; Batra, D.; Parikh, D.; and Lee, S. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *NeurIPS*, 13–23.

Marino, K.; Rastegari, M.; Farhadi, A.; and Mottaghi, R. 2019. OK-VQA: A Visual Question Answering Benchmark Requiring External Knowledge. In *CVPR*, 3195–3204.

Meng, K.; Bau, D.; Andonian, A.; and Belinkov, Y. 2022. Locating and Editing Factual Associations in GPT. In *NeurIPS*.

Meng, K.; Sharma, A. S.; Andonian, A. J.; Belinkov, Y.; and Bau, D. 2023. Mass-Editing Memory in a Transformer. In *ICLR*.

Mishra, A.; Asai, A.; Balachandran, V.; Wang, Y.; Neubig, G.; Tsvetkov, Y.; and Hajishirzi, H. 2024. Fine-grained Hallucination Detection and Editing for Language Models. arXiv:2401.06855.

Mitchell, E.; Lin, C.; Bosselut, A.; Finn, C.; and Manning, C. D. 2022a. Fast Model Editing at Scale. In *ICLR*.

Mitchell, E.; Lin, C.; Bosselut, A.; Manning, C. D.; and Finn, C. 2022b. Memory-Based Model Editing at Scale. In *ICML*, 15817–15831.

Pan, H.; Cao, Y.; Wang, X.; and Yang, X. 2023. Finding and Editing Multi-Modal Neurons in Pre-Trained Transformer. arXiv:2311.07470.

Pearl, J. 2001. Direct and Indirect Effects. In *UAI*, 411–420.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; Krueger, G.; and Sutskever, I. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, volume 139, 8748–8763.

Ribeiro, M. T.; Singh, S.; Guestrin, C.; and . 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *SIGKDD*, 1135–1144.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *CVPR*, 10674–10685.

Roumeliotis, K. I.; Tselikas, N. D.; .; and . 2023. ChatGPT and Open-AI Models: A Preliminary Review. *Future Internet*, 15(6): 192.

Schwettmann, S.; Chowdhury, N.; Klein, S.; Bau, D.; and Torralba, A. 2023. Multimodal Neurons in Pretrained Text-Only Transformers. In *ICCV*, 2854–2859.

Sundararajan, M.; Taly, A.; Yan, Q.; and . 2017. Axiomatic Attribution for Deep Networks. In *ICML*, volume 70, 3319–3328.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; Rodriguez, A.; Joulin, A.; Grave, E.; and Lample, G. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NeurIPS*, 5998–6008.

Vig, J.; Gehrmann, S.; Belinkov, Y.; Qian, S.; Nevo, D.; Singer, Y.; and Shieber, S. M. 2020. Investigating Gender Bias in Language Models Using Causal Mediation Analysis. In *NeurIPS*.

Wadekar, S. N.; Chaurasia, A.; Chadha, A.; and Culurciello, E. 2024. The Evolution of Multimodal Model Architectures. arXiv:2405.17927.

Wu, X.; Li, J.; Xu, M.; Dong, W.; Wu, S.; Bian, C.; and Xiong, D. 2023. DEPN: Detecting and Editing Privacy Neurons in Pretrained Language Models. In *EMNLP*, 2875–2886.

Yao, Y.; Wang, P.; Tian, B.; Cheng, S.; Li, Z.; Deng, S.; Chen, H.; and Zhang, N. 2023. Editing Large Language Models: Problems, Methods, and Opportunities. In *EMNLP*, 10222–10240.

Ye, Q.; Xu, H.; Ye, J.; Yan, M.; Hu, A.; Liu, H.; Qian, Q.; Zhang, J.; Huang, F.; and Zhou, J. 2023. mPLUG-Owl2: Revolutionizing Multi-modal Large Language Model with Modality Collaboration. arXiv:2311.04257.

Yin, S.; Fu, C.; Zhao, S.; Xu, T.; Wang, H.; Sui, D.; Shen, Y.; Li, K.; Sun, X.; and Chen, E. 2023. Woodpecker: Hallucination Correction for Multimodal Large Language Models. arXiv:2310.16045.

Yu, L.; Chen, Q.; Zhou, J.; and He, L. 2024. MELO: Enhancing Model Editing with Neuron-Indexed Dynamic LoRA. In *AAAI*, 19449–19457.

Zeng, A.; Liu, X.; Du, Z.; Wang, Z.; Lai, H.; Ding, M.; Yang, Z.; Xu, Y.; Zheng, W.; Xia, X.; Tam, W. L.; Ma, Z.; Xue, Y.; Zhai, J.; Chen, W.; Liu, Z.; Zhang, P.; Dong, Y.; and Tang, J. 2023. GLM-130B: An Open Bilingual Pre-trained Model. In *ICLR*.

Zhang, J.; Zhang, H.; Yin, X.; Huang, B.; Zhang, X.; Hu, X.; and Wan, X. 2024. MC-MKE: A Fine-Grained Multimodal Knowledge Editing Benchmark Emphasizing Modality Consistency. arXiv:2406.13219.

Zheng, C.; Li, L.; Dong, Q.; Fan, Y.; Wu, Z.; Xu, J.; and Chang, B. 2023. Can We Edit Factual Knowledge by In-Context Learning? arXiv:2305.12740.

Zhu, D.; Chen, J.; Shen, X.; Li, X.; and Elhoseiny, M. 2023. MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models. arXiv:2304.10592.