Addressing Text Embedding Leakage in Diffusion-based Image Editing

Sunung Mun[1*] Jinhwan Nam[1*] Sunghyun Cho[1,2] Jungseul Ok[1,2†]

Graduate School of AI, POSTECH[1], Dept. of CSE, POSTECH[2]
{mtablo, njh18, s.cho, jungseul}@postech.ac.kr

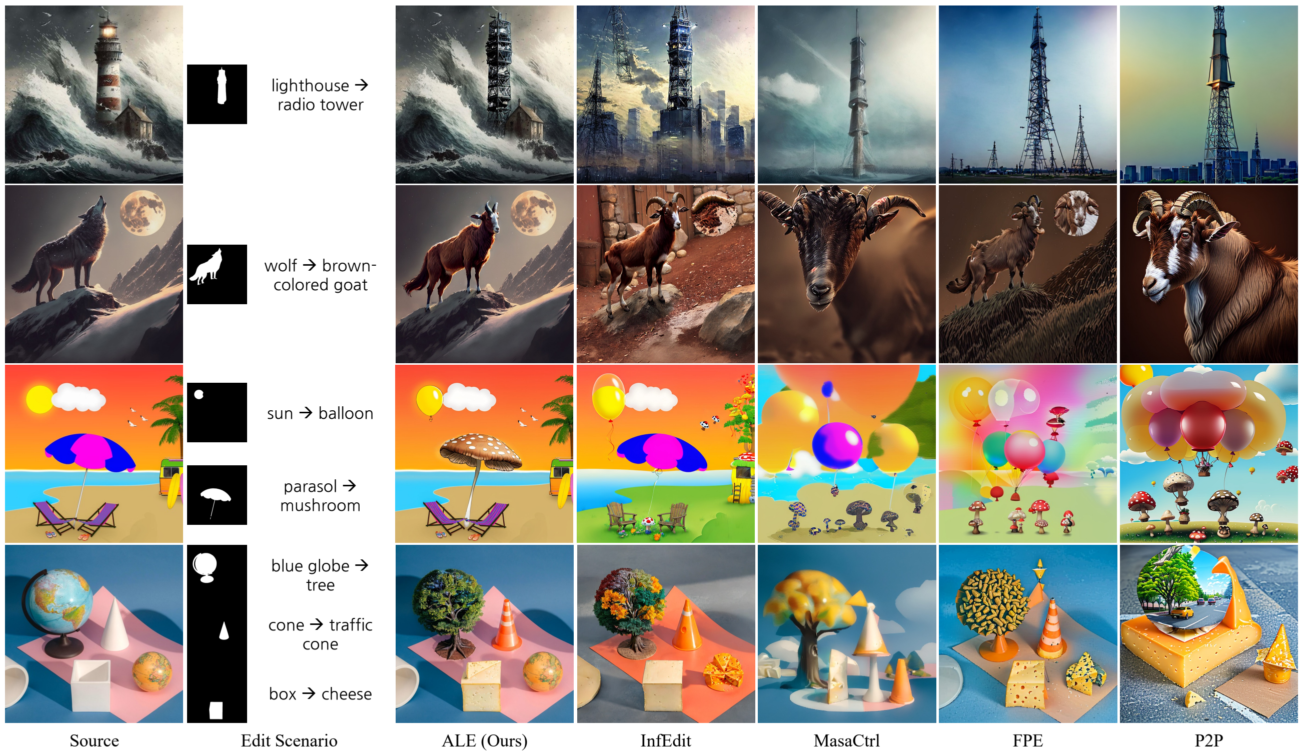


Figure 1. Qualitative comparison of tuning-free image-editing methods. Left to right: (1) source image, (2) visualization-only binary masks of the source objects and corresponding edit prompts (source → target), and (3–7) results from our ALE and four baselines. The masks are not provided to any method; they are shown solely to indicate which region is supposed to change. Rows 1–2 illustrate single-object edits where baselines spill changes outside the intended area or distort geometry, while ALE keeps the background intact. Rows 3–4 demonstrate multi-object edits: baselines often entangle attributes, whereas ALE preserves each attribute in its designated region, yielding leakage-free results.

Abstract

Text-based image editing, powered by generative diffusion models, lets users modify images through natural-language prompts and has dramatically simplified traditional workflows. Despite these advances, current methods still suffer from a critical problem: attribute leakage, *where edits meant for specific objects unintentionally affect unrelated regions or other target objects. Our analysis reveals the root cause as the semantic entanglement inherent in End-of-Sequence (EOS) embeddings generated by autoregressive text encoders, which indiscriminately aggregate attributes across prompts. To address this issue, we introduce* Attribute-Leakage-free Editing *(ALE), a framework that tackles attribute leakage at its source. ALE combines* Object-Restricted Embeddings *(ORE) to disentangle text embeddings,* Region-Guided Blending for Cross-Attention Masking *(RGB-CAM) for spatially precise attention, and* Background Blending *(BB) to preserve non-edited content. To quantitatively evaluate attribute leakage across various*

* Equal contribution; † Correspondence to jungseul@postech.ac.kr
Project page: https://mtablo.github.io/ALE_Edit_page/

editing methods, we propose the Attribute-Leakage Evaluation Benchmark *(ALE-Bench), featuring comprehensive editing scenarios and new metrics. Extensive experiments show that ALE reduces attribute leakage by large margins, thereby enabling accurate, multi-object, text-driven image editing while faithfully preserving non-target content.*

1. Introduction

Text-based image editing, where users modify existing images via natural language prompts, has emerged as a powerful alternative to traditional manual editing. Conventional editing workflows typically demand significant manual effort and domain expertise [1, 16, 25, 33]. In contrast, recent advancements leveraging generative diffusion models have substantially simplified the editing process [6, 12, 15, 20]. These models enable users to perform high-quality edits through intuitive textual prompts, making image editing more accessible and flexible.

Despite these advances, existing text-based editing methods frequently suffer from a critical limitation: *attribute leakage*, where edits intended for specific objects inadvertently affect unrelated regions within the image. Attribute leakage can be categorized into two distinct types: *Target-External Leakage (TEL)*, where attributes of a target object unintentionally affect non-target regions, and *Target-Internal Leakage (TIL)*, where attributes intended for one target object inadvertently influence another target object within the same editing prompt. To mitigate this, recent studies attempt to spatially constrain editing effects by manipulating cross-attention maps [3, 29]. However, as illustrated in Figure 1, even state-of-the-art methods [3, 13, 29, 34] continue to exhibit significant TEL in single-object editing scenarios, and both TEL and substantial TIL in more complex, *multi-object* editing scenarios.

Our analysis highlights that attribute leakage fundamentally stems from overlooked issues of the entanglement of text embeddings, specifically those associated with the *End-of-Sequence (EOS)* tokens, building upon recent study [14]. Most text-based editing pipelines employ autoregressive text encoders such as CLIP [22], which append EOS tokens to text prompts until reaching a fixed length (e.g., 77 tokens) to generate embeddings. Consequently, EOS embeddings inherently aggregate information from *all tokens* within the prompt and attend indiscriminately across image regions via cross-attention layers, exacerbating both TEL and TIL. While recent studies [14] attempt to alleviate leakage stemming from EOS embedding entanglement, our analysis demonstrates that these approaches are insufficient to fully eliminate attribute leakage (see Section 2).

To address these limitations, we propose a novel framework, *Attribute-Leakage-Free Editing (ALE)*, primarily consisting of three complementary components: Object-Restricted Embeddings (ORE), Region-Guided Blending for Cross-Attention Masking (RGB-CAM), and Background Blending (BB). ORE assigns distinct, semantically isolated embeddings to each object in the prompt, explicitly avoiding embedding entanglement. RGB-CAM enhances spatial precision in cross-attention maps by leveraging segmentation masks, restricting attention solely to intended regions. BB preserves the structural integrity of non-edited regions by integrating latents from the source image.

Furthermore, to systematically quantify attribute leakage, we introduce the *Attribute-Leakage Evaluation Benchmark (ALE-Bench)*, a specialized benchmark designed explicitly for leakage evaluation in multi-object editing. Existing benchmarks predominantly focus on single-object scenarios, lacking metrics for evaluating attribute leakage comprehensively [17, 27, 31]. ALE-Bench covers a diverse range of editing scenarios, including multi-object editing in various edit types. We also propose two novel evaluation metrics: *Target-External Leakage Score (TELS)* and *Target-Internal Leakage Score (TILS)*, explicitly quantifying TEL and TIL, respectively.

In summary, our contributions are:

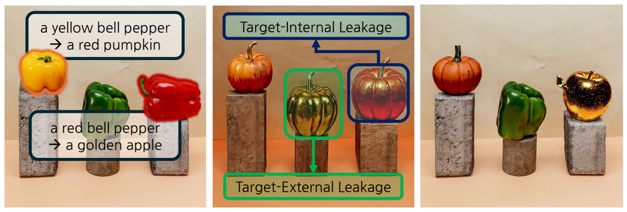
1. Identifying the previously overlooked role of EOS embeddings as a fundamental cause of attribute leakage in text-based image editing (Section 2).
2. Proposing the novel *Attribute-Leakage-Free Editing (ALE)* framework, which specifically addresses leakage induced by EOS embeddings (Section 3).
3. Introducing a comprehensive benchmark, *ALE-Bench*, along with novel metrics (*TELS*, *TILS*) designed explicitly to quantify attribute leakage in multi-object editing scenarios (Section 4).

2. Attribute Leakage Problem and Analysis

In this section, we discuss the *attribute leakage* problem in multi-object text-based image editing and highlight limitations of existing methods. Specifically, Section 2.1 formally introduces multi-object text-based image editing and briefly describes dual-branch frameworks commonly used for such tasks. Section 2.2 defines attribute leakage and categorizes it into two types: TEL and TIL. Finally, Section 2.3 analyzes embedding entanglement, particularly from EOS tokens, as the primary cause of attribute leakage and explains why existing methods fail to adequately address this issue.

2.1. Multi-Object Text-based Image Editing

Text-based image editing modifies specific regions of a source image according to textual prompts describing desired changes. Formally, given a source image x^{src} and a textual prompt pair $(y^{\text{src}}, y^{\text{tgt}})$—where y^{src} specifies the source objects to be edited (i.e., objects to be replaced or modified), and y^{tgt} specifies the target objects (i.e., new or modified objects to appear)—the goal is to generate an edited


a yellow bell pepper → a red pumpkin
a red bell pepper → a golden apple
Target-Internal Leakage
Target-External Leakage

(a) Source Image (b) Leakage Example (c) Reference Result

Figure 2. Illustration of attribute leakage in image editing. (a) The source image and editing prompts. (b) An editing result exhibiting attribute leakage. Regions highlighted in green indicate *target-external leakage*, where the editing spills into non-target (e.g., the green bell pepper turns into a golden pumpkin). Regions in blue show *target-internal leakage*, where the attributes of "a red pumpkin" undesirably influence the appearance of the "golden apple" region. (c) A reference image showing the desired editing result without attribute leakage.

image x^{tgt}. Ideally, x^{tgt} reflects modifications described by y^{tgt} exclusively within regions indicated by y^{src}, leaving all other areas unchanged. When edits involve multiple objects simultaneously, the task is classified as *multi-object editing*.

Practically, multi-object editing prompts can be decomposed into individual object-level prompt pairs $[(y_i^{\text{src}}, y_i^{\text{tgt}})]_{i=1}^K$, typically via language models or noun-chunk parsers [2, 11]. Our research specifically focuses on scenarios involving up to $K = 3$ objects. For example, consider the editing scenario in Figure 2a. Given the prompt pair $(y^{\text{src}}, y^{\text{tgt}})$ = ("a yellow bell pepper and a red bell pepper", "a red pumpkin and a golden apple"), it can be decomposed into object-level prompts $[(y_i^{\text{src}}, y_i^{\text{tgt}})]_{i=1}^2$ = [("a yellow bell pepper", "a red pumpkin"), "a red bell pepper", "a golden apple")]. The resulting image x^{tgt} should contain a red pumpkin and a golden apple, precisely aligned to corresponding regions.

To perform such precise edits, diffusion-based editing methods encode textual prompts into embeddings that guide the editing process through cross-attention layers. These layers spatially align text-described attributes with corresponding regions in the image. Achieving accurate alignment, however, requires effectively preserving the original spatial structure while synthesizing novel visual attributes. To this end, recent methods utilize *dual-branch editing frameworks*, which have become popular due to their ability to simultaneously retain the original image layout and introduce new content [3, 19, 29].

Dual-branch frameworks operate through two parallel pathways—a *source branch* and a *target branch*—using the same pretrained text-to-image diffusion model. Specifically, the *source branch* reconstructs the original image x^{src} guided by the source textual prompt y^{src}, thereby capturing structural and spatial information inherent in x^{src}. The *target branch*, in parallel, synthesizes new visual attributes guided by the target textual prompt y^{tgt}. Structural consistency is maintained by injecting intermediate self-attention layer components—such as queries and keys—computed from the source branch into the corresponding self-attention layers of the target branch. Despite their strengths, dual-branch frameworks face fundamental challenges, particularly *attribute leakage*, caused by embedding entanglement involving EOS tokens, as analyzed in subsequent sections.

2.2. Attribute Leakage

Attribute leakage is a critical challenge in multi-object text-based image editing, characterized by unintended propagation of attributes from target objects to unrelated regions or other target objects. Formally, given a source image x^{src} and object-level prompt pairs $[(y_i^{\text{src}}, y_i^{\text{tgt}})]_{i=1}^K$, attribute leakage occurs when modifying an object from its original description y_i^{src} to a new target description y_i^{tgt} unintentionally impacts regions or objects not specified by the editing prompt. Attribute leakage can be categorized into two distinct types:

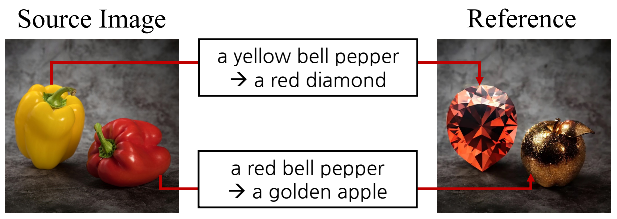
- *Target-External Leakage (TEL)*: This occurs when editing an object specified by $(y_i^{\text{src}}, y_i^{\text{tgt}})$ unintentionally affects regions not described by the prompt pair (i.e., non-target regions). For example, as highlighted by the green region in Figure 2b, editing y_2^{src} = "a red bell pepper" to y_2^{tgt} = "a golden apple" inadvertently transforms an unrelated "green bell pepper" into a golden object.
- *Target-Internal Leakage (TIL)*: This occurs when editing an object specified by $(y_i^{\text{src}}, y_i^{\text{tgt}})$ unintentionally affects another target object specified by a different prompt pair $(y_j^{\text{src}}, y_j^{\text{tgt}})$, where $i \neq j$. For instance, as illustrated by the blue region in Figure 2b, editing y_1^{src} = "a yellow bell pepper" to y_1^{tgt} = "a red pumpkin" inadvertently impacts another target object described by y_2^{tgt} = "a golden apple", causing it to appear as a mixture of red and golden pumpkin-like attributes.

Effectively mitigating both TEL and TIL is essential to achieving precise, user-intended edits, as exemplified by the desired reference result in Figure 2c.

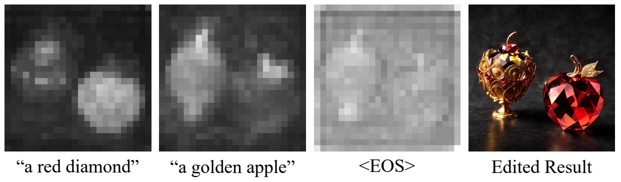
2.3. Causes of Attribute Leakage

Attribute leakage primarily arises from *embedding entanglement*, which occurs during prompt encoding. Widely used text encoders, such as CLIP, encode tokens autoregressively, causing embeddings of later tokens to unintentionally accumulate mixed semantics from preceding attributes or objects. To mitigate this entanglement, some approaches introduce object-wise embeddings by parsing prompts into distinct noun-phrase spans and encoding each segment independently [10]. However, these methods only address entanglement among original tokens in the prompt, failing to adequately resolve entanglement involving EOS tokens.

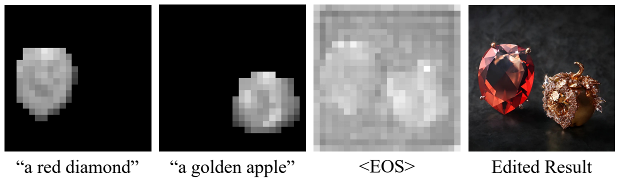
Since CLIP pads prompts to a fixed length using EOS tokens, the EOS embeddings inevitably aggregate semantic information from multiple attributes and objects. For


Source Image
Reference
a yellow bell pepper → a red diamond
a red bell pepper → a golden apple

(a) Source image, edit prompt, and reference result image



(b) Cross-attention maps and an edited result of ETS



(c) Cross-attention maps and an edited result of ETS + Masking

Figure 3. Illustration of attribute leakage from EOS embeddings and misaligned cross-attention. (b) ETS with object-wise embeddings shows spatial misalignment: embeddings of one object (e.g., "a red diamond") can influence unrelated regions. (c) Adding cross-attention masking correctly localizes object-wise embeddings but fails to suppress leakage from EOS embeddings (e.g., diamond-like decoration on the apple). Cross-attention maps are averaged across timesteps and summed over tokens (or padded EOS tokens).

instance, when encoding the prompt "a red diamond and a golden apple", the EOS embeddings inherently encapsulate combined semantics from all attributes and objects (e.g., "red", "diamond", "golden", and "apple"). To address this issue, End-Token-Substitution (ETS) [14] replaces attribute-rich EOS embeddings with attribute-free embeddings obtained from prompts without descriptors (e.g., "a diamond and an apple"). Nevertheless, ETS remains insufficient even when combined with object-wise embeddings, as the simplified EOS embeddings still aggregate semantic information across multiple objects, thereby continuing to propagate attribute leakage (see Figure 3).

Embedding entanglement further exacerbates spatial inaccuracies in dual-branch text-based image editing frameworks. Since dual-branch methods inject structural information from the source image into the target editing branch, entangled embeddings can produce *misaligned cross-attention maps*, causing embeddings to incorrectly attend to visually similar but semantically incorrect regions during editing (Figure 3b). This visual-semantic confusion significantly exacerbates spatial inaccuracies, intensifying

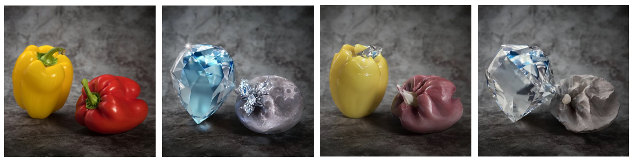


Figure 4. Comparison of different EOS embedding modification strategies in a single editing scenario, where "yellow bell pepper" is edited to "diamond" and "red bell pepper" to "moon". (b) Uses the original EOS embeddings. (c) Replaces EOS embeddings with zero vectors. (d) Replaces EOS embeddings with those obtained from an empty prompt **""**.

attribute leakage. Although existing methods attempt to mitigate this issue through refined cross-attention alignment or explicit masking [3, 29], these strategies fail to address the fundamental problem: EOS embeddings inherently lack spatial specificity, as they integrate semantic content from the entire prompt. Therefore, restricting the spatial attention of EOS embeddings to specific regions is inherently ineffective, further compounding attribute leakage even when employing combined strategies (Figure 3c).

One naive alternative could involve removing semantic content entirely from EOS embeddings by substituting them with zero vectors or embeddings derived from empty prompts. However, as demonstrated empirically in Figure 4 and detailed in Appendix E, this simplistic solution severely degrades visual quality and editing accuracy. These observations imply that diffusion models intrinsically depend on EOS embeddings containing semantics to achieve high-quality image editing outcomes. Therefore, resolving attribute leakage effectively requires a dedicated strategy that carefully mitigates the unintended influence of entangled EOS embeddings without completely eliminating or overly simplifying their semantic content. We propose such a targeted approach in the next section.

3. Attribute-Leakage-Free Editing (ALE)

Given a source image x^{src} and a list of K object–level prompt pairs $\big[(y_i^{\text{src}}, y_i^{\text{tgt}})\big]_{i=1}^{K}$, our goal is to generate an edited image x^{tgt} that (1) replaces every y_i^{src} with y_i^{tgt} inside its designated region, (2) preserves all non-target content, and (3) avoids both TIL and TEL.

Our proposed method, ALE, is built on the dual-branch editing framework with the Denoising Diffusion Consistent Model (DDCM) virtual inversion scheme: a *source branch* reconstructs x^{src} from latent $\{z_\tau^{\text{src}}\}_{\tau=T}^{0}$ under the prompt $y_{\text{base}}^{\text{src}} = $ " and ".join$\big(\big[y_i^{\text{src}}\big]\big)$, while a *target branch* denoises latent $\{z_\tau^{\text{tgt}}\}_{\tau=T}^{0}$ toward the edited image guided by $y_{\text{base}}^{\text{tgt}} = $ " and ".join$\big(\big[y_i^{\text{tgt}}\big]\big)$. On top of this backbone we introduce three key components: ORE, RGB-CAM, and BB. Figure 5 visualizes an overall pipeline, and Algorithm 1 lists the complete procedure.

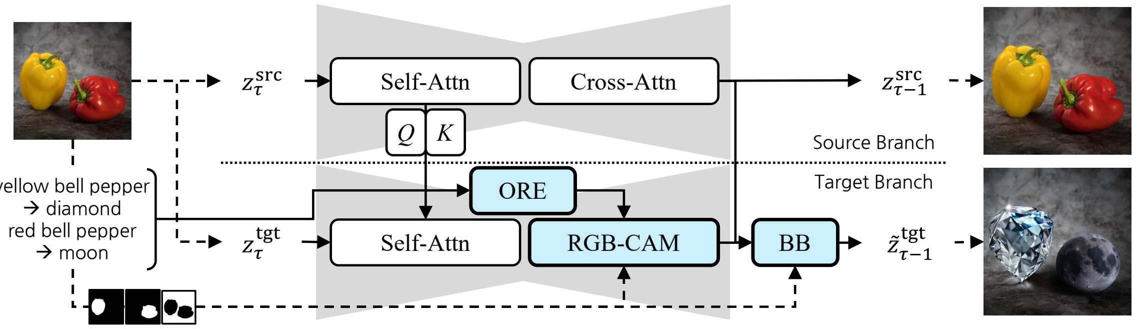


Figure 5. Overview of ALE. The framework consists of two branches: the upper branch (source branch) processes the source latent z_τ^{src}, and the lower branch (target branch) processes the target latent z_τ^{tgt} at each timestep τ. ORE produces multiple semantically-isolated embeddings for each target object to reduce interference between unrelated objects. RGB-CAM refines cross-attention activations using segmentation masks, aligning the attention of each embedding to spatial regions corresponding to its target object. BB merges the source latent for background regions and the target latent for edited regions. Dashed lines indicate omitted components (e.g., the encoder, the decoder, and the segmentation model) for simplicity.

3.1. DDCM and Dual-Branch Editing Framework

Virtual inversion via DDCM We adopt the Denoising Diffusion Consistent Model (DDCM) [29], which chooses the special variance schedule so that any noisy latent z_τ retains a closed-form link to its clean latent z_0 at every timestep $\tau \in [T, \dots, 0]$. This property enables a *virtual inversion* that avoids the costly DDIM / null-text inversion. Because this form matches the multi-step consistency sampler of Latent Consistency Models [8], we can edit images in as few as 4–20 steps without explicit inversion steps.

Dual-branch framework Starting from an initial noise, the *source branch* follows DDCM updates with $y_{\text{base}}^{\text{src}}$, while the *target branch* starts with $z_\tau^{\text{tgt}} = z_\tau^{\text{src}}$ and calculates the next latent $z_{\tau-1}^{\text{tgt}}$ at each step using the U-Net noise $\varepsilon_\theta(z_\tau^{\text{tgt}}, \tau, y_{\text{base}}^{\text{tgt}})$ and the consistency correction term. To preserve structure of x^{src}, we copy the query–key tensors $(Q, K)_{\ell,\tau}^{\text{src}}$ from every self-attention layer ℓ of the source branch into the corresponding layer of the target branch according to self-attention injection schedule $\mathcal{S} = \{\tau \mid T_S \le \tau \le T\}$, where $T_S \in [0, T]$ controls the length of the self-attention injection schedule. Thus, a shorter schedule ($T_s \approx T$) touches only early denoising steps and enables stronger edits, whereas a longer schedule ($T_s \approx 0$) enforces stricter structural preservation.

3.2. Object-Restricted Embeddings (ORE)

To address leakages in prompt embedding level, ORE encodes each object prompt y_i^{tgt} in isolation, yielding a set of token-embedding matrices

$$E_i' = [e_{\text{BOS}},\ \underbrace{e_{\text{token}_1},\ \cdots}_{\text{from tokens in } y_i^{\text{tgt}}},\ \underbrace{e_{\text{EOS}},\ \cdots}_{\text{from padded EOS tokens}}\ ] \in \mathbb{R}^{L\times d}$$

where L is the padded prompt length and d the embedding dimension. For example, in Figure 3 case, E_1' is obtained as $[e_{\text{BOS}},\ e_{\text{a}},\ e_{\text{red}},\ e_{\text{diamond}},\ e_{\text{EOS}},\ \dots]$. And we construct a base embedding E_{base}', by encoding $y_{\text{base}}^{\text{tgt}}$ and splicing $E_i'[y_i^{\text{tgt}}]$ back into their original spans, to calculate base value tensor in RGB-CAM. Because no token embedding in $\{E_i'\}$ can influence another object's span and EOS embeddings in E_i' only contain semantics of y_i^{tgt}, subsequent cross-attention receives *semantically disentangled* embeddings, thereby preventing leakage at its source.

3.3. Region-Guided Blending for Cross-Attention Masking (RGB-CAM)

Standard cross-attention layers in diffusion U-Net accept a *single* value tensor V and thus cannot exploit multiple OREs. RGB-CAM replaces the vanilla cross-attention output with a spatially blended tensor

$$A = \sum_{i=1}^{K} (M \odot m_i)\ V_i \ +\ (M \odot m_{\text{back}})\ V_{\text{base}},$$

where $M = \text{attention\_map}(Q, K)$ is the base cross attention map, $V_i = W_v(E_i')$, $K = W_k(\text{Encoder}_{\text{text}}(y_{\text{base}}^{\text{tgt}}))$, $V_{\text{base}} = W_v(E_{\text{base}}')$, and $\{m_i\}$, m_{back} are object and background segmentation masks from Grounded-SAM [24]. Since masks are not pixel-perfect, we apply a slight dilation. The masked tensors $(M \odot m_i)V_i$ *localize* each ORE to its designated region, eliminating target-internal leakage, while the background term preserves areas outside all masks. Note that only when ORE and RGB-CAM operate *in tandem* does ALE produce leakage-free results.

3.4. Background Blending (BB)

Even with perfect cross-attention, backgrounds remain weakly constrained because $\{y_i^{\text{tgt}}\}$ mention only target objects. At every timestep τ we blend the source latent by the background mask as a final step:

$$\tilde{z}_\tau^{\text{tgt}} = m_{\text{back}} \odot z_\tau^{\text{src}} + (1 - m_{\text{back}}) \odot z_\tau^{\text{tgt}}.$$

```
Algorithm 1: Attribute-Leakage-Free Editing
  Input: Source image $x^{\text{src}}$, object-level prompt pairs
         $\left[(y_i^{\text{src}}, y_i^{\text{tgt}})\right]_{i=1}^{K}$, self-attention injection
         schedule $\mathcal{S} = \{\tau \mid 0 \le \tau \le T_S\}$
  Output: Edited image $x^{\text{tgt}}$

  Pre-processing
      Form concatenated base prompts $y_{\text{base}}^{\text{src}}, \ y_{\text{base}}^{\text{tgt}}$;
      Encode object-restricted embeddings $\{E_i'\}_{i=1}^{K}$
        and the base embedding $E_{\text{base}}'$ (ORE);
      Obtain object masks $\{m_i\}_{i=1}^{K}$ and background
        mask $m_{\text{back}}$ with Grounded-SAM;
  Initialization
      Sample initial noise $z_T^{\text{src}} \sim \mathcal{N}(0, I)$;
      Set $z_T^{\text{tgt}} \leftarrow z_T^{\text{src}}$;
  for $\tau = T$ to $1$ do
      // Source branch
      Predict noise $\hat{\varepsilon}_\tau^{\text{src}} \leftarrow \varepsilon_\theta\left(z_\tau^{\text{src}}, \tau, y_{\text{base}}^{\text{src}}\right)$;
      Update $z_{\tau-1}^{\text{src}}$ with DDCM sampling;
      // Target branch
      if $\tau \in \mathcal{S}$ then copy self-attention $Q, K$ tensors
        from the source branch;
      Predict noise $\hat{\varepsilon}_\tau^{\text{tgt}} \leftarrow$
        $\varepsilon_\theta\left(z_\tau^{\text{tgt}}, \tau, y_{\text{base}}^{\text{tgt}}; \ \text{RGB-CAM}[\{E_i', m_i\}, E_{\text{base}}', m_{\text{back}}]\right)$;

      Update $z_{\tau-1}^{\text{tgt}}$ with DDCM sampling;
      // Background blending (BB)
      $z_{\tau-1}^{\text{tgt}} \leftarrow m_{\text{back}} \odot z_{\tau-1}^{\text{src}} \ + \ \left(1 - m_{\text{back}}\right) \odot z_{\tau-1}^{\text{tgt}}$;
  end
  $x^{\text{tgt}} \leftarrow \text{Decoder}\left(z_0^{\text{tgt}}\right)$;
  return $x^{tgt}$
```

BB guarantees preservation of non-edited regions, suppressing TEL without expensive threshold tuning required by prior local-blending heuristics [13].

4. Experiments

4.1. Experiment Setup

ALE-Bench construction Prior benchmarks for text-guided image editing [7, 17] focus on the visual quality of the edited result itself and neglect *attribute leakage*. Although they measure background preservation metrics that are similar to TELS, they overlook TIL. This makes it difficult to analyze how well a method achieves precise editing. To fill this gap, we introduce Attribute-Leakage-Evaluation Benchmark (ALE-Bench), a dedicated testbed for attribute leakage evaluation. ALE-Bench systematically varies the number of objects to be edited and the editing type, grouped into five categories: (1) *color*, (2) *object*, (3) *material*, (4) *color* + *object*, and (5) *object* + *material*. Each source image is paired with multiple prompts for each edit type, enabling fine-grained analysis across diverse scenarios. Figure 6 shows examples of ALE-Bench. Full construction details and dataset statistics are provided in Appendix B.

Evaluation Metrics We evaluate image-editing performance using the following metrics:

- *Structure Distance* [26] quantifies how well the edited image x^{tgt} preserves the spatial layout of x^{src}. A lower score indicates better structural consistency.
- *Editing Performance* is measured by the cosine similarity between the CLIP embeddings of x^{tgt} and prompt y^{tgt}. Higher similarity reflects more faithful edits.
- *Background Preservation* is assessed on the non-edited regions using PSNR, SSIM [28], LPIPS [32], and MSE between x^{tgt} and x^{src}. Higher PSNR/SSIM and lower LPIPS/MSE signify better preservation.

Furthermore, we introduce new metrics:

- *Target-Internal Leakage Score (TILS)* measures unintended modifications inside other target-object regions as follows:

$$\text{TILS} = \frac{1}{K(K-1)} \sum_{i \neq j}^{K} \text{CLIP}\left(x_{\text{tgt}} \odot m_j, y_i^{\text{tgt}}\right),$$

where CLIP represents the CLIP similarity score, K is the number of objects to be edited, x^{tgt} is the edited image, m_j is the j-th object mask, and y_i^{tgt} is the target prompt for i-th object. A lower TILS implies that, as the user intended, the target objects did not affect each other.

- *Target-External Leakage Score (TELS)* measures unintended changes in the background (non-edited regions) as follows:

$$\text{TELS} = \frac{1}{K} \sum_{i=1}^{K} \text{CLIP}\left(x^{\text{tgt}} \odot \left(\mathbf{1} - \bigcup_{j=1}^{K} m_j\right), y_i^{\text{tgt}}\right).$$

The mean CLIP scores between the background and each target prompt are computed for multiple object edits. A lower TELS indicates minimal TEL, which ensures that the outside of the targets remains unchanged.

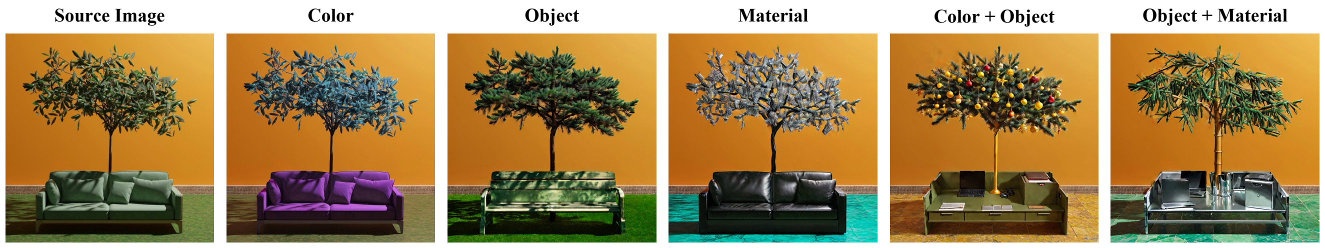
Baselines For comparison, we selected tuning-free image editing methods including Prompt-to-Prompt (P2P) [13], MasaCtrl [3], Free-Prompt-Editing (FPE) [34], and InfEdit [29] (see Appendix A, C for further details).

4.2. Main Results

ALE outperforms existing methods in both mitigating attribute leakage and producing high-quality edits, as seen in Table 1 and Figure 7. In particular, ALE achieves the lowest TELS and TILS, reflecting its ability to precisely apply attributes solely to the designated target regions. Across different numbers of editing objects (Table 2) and different

Source Image	Object Masks	Color	Object	Material	Color + Object	Object + Material
		yoga mat → pink-colored yoga mat	man → alien	footwear → footwear made of amber	footwear → orange-colored sandals	man → woman made of bronze
		cup → white-colored cup, steam → green-colored steam	book → cushion, cup → tumbler	book → book made of chrome, cup → cup made of paper	steam → gray-colored smoke, cup → blue-colored jar	book → journal made of gold, cup → mug made of amethyst
		wolf → pink-colored wolf, moon → purple-colored moon, mountain → gray-colored mountain	wolf → eagle, moon → soccer ball, mountain → glacier	wolf → wolf made of velvet, moon → moon made of steel, mountain → mountain made of leather	wolf → crimson-colored dragon, moon → navy-colored soccer ball, mountain → brown-colored garden	wolf → dog made of linen, moon → balloon made of paper, mountain → glacier made of amber

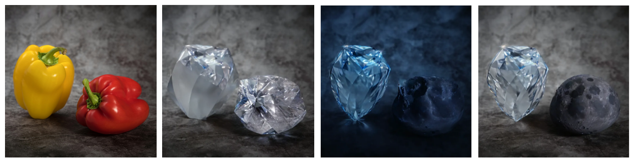
Figure 6. Examples of ALE-Bench. Source images are shown with object binary masks. Edit types are color, object, material modifications, and their combinations. Rows depict 1-, 2-, and 3-object edits, respectively.



Source Image		Color	Object	Material	Color + Object	Object + Material
tree	→	blue-colored tree	pine tree	tree made of diamond	yellow-colored Christmas tree	pine tree made of bamboo
couch	→	purple-colored couch	bench	couch made of leather	khaki-colored desk	desk made of chrome
floor	→	khaki-colored floor	lawn	floor made of turquoise	yellow-colored tile	tile made of jade

Figure 7. Qualitative examples of ALE applied to three source objects (*tree*, *couch*, and *floor*) under different editing types. The bottom rows show the corresponding prompts describing each transformation.

editing types (Table 3), ALE demonstrates its robust performance. Figure 7 illustrates qualitative examples of ALE for various editing types. These results indicate that our method can effectively address both TEL and TIL. Detailed results are in Appendix D. Furthermore, the quantitative and qualitative results on PIE-Bench are in Appendix E.



(a) Source image (b) BB (c) ORE + RGB (d) ALE

Figure 8. Qualitative ablation results for the editing: yellow bell pepper → diamond, red bell pepper → moon. (b) BB resolves TEL but not TIL. (c) ORE + RGB-CAM reduces TIL but not TEL. (d) ALE (BB + ORE + RGB-CAM) shows no TIL and TEL.

4.3. Ablation Studies

We present ablation results on BB, ORE, and RGB-CAM in Figure 8. Each module plays a complementary role in preventing attribute leakage. Using only BB (Figure 8b) helps preserve background regions and suppresses TEL, but fails to prevent TIL. On the other hand, ORE + RGB-CAM (Figure 8c) reduces TIL by disentangling text embeddings and aligned cross-attention. However, without BB, it cannot preserve the original background, and TEL remains. Only when all three modules—ORE, RGB-CAM, and BB—are used together (Figure 8d), ALE achieves precise and leakage-free editing. Qualitative results and more examples are provided in Appendix E.

5. Limitations

While ALE-Bench provides a focused framework for evaluating attribute leakage, and ALE achieves strong performance on *rigid* attribute edits, both are currently lim-

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	21.52	17.26	0.1514	20.67	11.15	0.4495	0.0879	0.5589
MasaCtrl	20.18	16.74	0.0929	20.01	14.99	0.2930	0.0418	0.7346
FPE	21.07	17.38	0.1164	21.89	12.82	0.3903	0.0656	0.6052
InfEdit	19.59	16.69	0.0484	21.78	16.74	0.2034	0.0340	0.7709
ALE	**16.03**	**15.28**	**0.0167**	**22.20**	**30.04**	**0.0361**	**0.0014**	**0.9228**

Table 1. Comparison of editing performances on ALE-Bench. ALE demonstrates the lowest attribute leakage, highest structure preservation, and superior editing performance, indicating a more precise and controlled editing.

# of Editing Objects	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
1	16.41	-	0.00876	22.62	30.01	0.0405	0.00167	0.9049
2	16.00	15.42	0.01648	22.06	30.06	0.0360	0.00146	0.9235
3	15.89	15.36	0.02460	22.19	30.01	0.0323	0.00154	0.9426

Table 2. Performance of ALE on ALE-Bench based on the number of objects edited. Our method maintains low attribute leakage and strong background preservation even as the number of editing objects increases.

Editing Type	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
Color	17.63	16.21	0.00890	23.12	32.97	0.0288	0.00079	0.9309
Material	17.15	15.96	0.01179	22.94	30.63	0.0339	0.00120	0.9248
Object	15.86	16.25	0.01974	21.82	29.03	0.0386	0.00182	0.9218
Color+Object	15.30	14.01	0.02306	22.15	28.60	0.0407	0.00205	0.9206
Object+Material	14.55	14.51	0.01956	21.42	28.88	0.0393	0.00191	0.9201

Table 3. Performance of ALE on ALE-Bench across various editing types (color, object, material, and combinations). The results show consistent low attribute leakage and high editing performance.

ited to local and relatively simple transformations—such as changes in color, object identity, or material. They do not support or evaluate non-rigid transformations like style transfer, pose changes, or adding/deleting objects, where defining and detecting attribute leakage becomes ambiguous. This focus reflects our aim to establish a clear and measurable foundation before addressing more complex editing scenarios. Although the benchmark offers $3,000$ detailed editing scenarios across 20 carefully curated images, the small image set may limit how well results generalize to models trained on larger or more diverse datasets. Future work could expand both the editing model and benchmark to support richer transformations and broader datasets. More limitations are discussed in Appendix F.

6. Conclusion

In this paper, we addressed the issue of attribute leakage in diffusion-based image editing, focusing on two types of leakage: Target-External Leakage (TEL)—unintended edits in non-target regions—and Target-Internal Leakage (TIL)—interference among attributes of different targets. To mitigate these problems, we introduced Attribute-Leakage-Free Editing (ALE), a tuning-free framework that combines three key components: Object-Restricted Embeddings (ORE), which localize attribute semantics to each target object embedding; Region-Guided Blending for Cross-Attention Masking (RGB-CAM), which constrains cross-attention with segmentation masks to avoid unintended inter-object attribute mixing; and Background Blending (BB), which preserves the source image in backgrounds.

We also presented ALE-Bench, a dedicated benchmark for rigorously evaluating attribute leakage across diverse multi-object editing scenarios. ALE-Bench introduces new quantitative metrics—TELS and TILS—that effectively quantify unintended modifications, providing comprehensive measures to assess editing fidelity and consistency.

Our extensive experimental validation demonstrated that ALE significantly outperforms existing tuning-free editing methods, achieving state-of-the-art performance by effectively minimizing attribute leakage while maintaining high editing quality and structural consistency. By effectively addressing attribute leakage with minimal computational overhead, ALE enhances the reliability and precision of multi-object image editing tasks.

Acknowledgements

This work was partly supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) and IITP-ITRC (Information Technology Research Center) grant funded by the Korea government (MSIT) (No.RS-2019-II191906, Artificial Intelligence Graduate School Program (POSTECH); No.RS-2021-II212068, Artificial Intelligence Innovation Hub; IITP-2025-00437866; RS-2024-00509258, Global AI Frontier Lab) and partly supported by Seoul R&BD Program through the Seoul Business Agency (SBA) funded by The Seoul Metropolitan Government (SP240008).

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 2

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 3

[3] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22560–22570, 2023. 2, 3, 4, 6, 11

[4] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM transactions on Graphics (TOG)*, 42(4):1–10, 2023. 11

[5] Wenkai Dong, Song Xue, Xiaoyue Duan, and Shumin Han. Prompt tuning inversion for text-driven image editing using diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7430–7440, 2023. 11

[6] Adham Elarabawy, Harish Kamath, and Samuel Denton. Direct inversion: Optimization-free text-driven real image editing with diffusion models. *arXiv preprint arXiv:2211.07825*, 2022. 2

[7] Chakrabarty et al. Lomoe: Localized multi-object editing via multi-diffusion. In *Proc. ACM Multimedia*, 2024. 6, 12

[8] Luo et al. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv*, 2023. 5

[9] Yu et al. Zero-shot referring image segmentation with global-local context features. In *Proc. CVPR*, 2023. 11

[10] Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. Training-free structured diffusion guidance for compositional text-to-image synthesis. *arXiv preprint arXiv:2212.05032*, 2022. 3, 11

[11] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 3

[12] Qin Guo and Tianwei Lin. Focus on your instruction: Fine-grained and multi-instruction image editing by attention modulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6986–6996, 2024. 2

[13] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022. 2, 6, 11

[14] Taihang Hu, Linxuan Li, Joost van de Weijer, Hongcheng Gao, Fahad Shahbaz Khan, Jian Yang, Ming-Ming Cheng, Kai Wang, and Yaxing Wang. Token merging for training-free semantic binding in text-to-image synthesis. *Advances in Neural Information Processing Systems*, 37:137646–137672, 2024. 2, 4, 11, 13, 14

[15] Mingzhen Huang, Jialing Cai, Shan Jia, Vishnu Suresh Lokhande, and Siwei Lyu. Paralleledits: Efficient multi-aspect text-driven image editing with attention grouping. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[16] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[17] Xuan Ju, Ailing Zeng, Yuxuan Bian, Shaoteng Liu, and Qiang Xu. Direct inversion: Boosting diffusion-based editing with 3 lines of code. *arXiv preprint arXiv:2310.01506*, 2023. 2, 6, 13

[18] Senmao Li, Joost Van De Weijer, Taihang Hu, Fahad Shahbaz Khan, Qibin Hou, Yaxing Wang, and Jian Yang. Stylediffusion: Prompt-embedding inversion for text-based editing. *arXiv preprint arXiv:2303.15649*, 2023. 11

[19] Bingyan Liu, Chengyu Wang, Tingfeng Cao, Kui Jia, and Jun Huang. Towards understanding cross and self-attention in stable diffusion for text-guided image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7817–7826, 2024. 3, 11

[20] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 2

[21] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023. 11

[22] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 2

[23] Royi Rassin, Eran Hirsch, Daniel Glickman, Shauli Ravfogel, Yoav Goldberg, and Gal Chechik. Linguistic binding in diffusion models: Enhancing attribute correspondence

through attention map alignment. *Advances in Neural Information Processing Systems*, 36:3536–3559, 2023. 11

[24] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024. 5

[25] Patsorn Sangkloy, Jingwan Lu, Chen Fang, Fisher Yu, and James Hays. Scribbler: Controlling deep image synthesis with sketch and color. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2017. 2

[26] Narek Tumanyan, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Splicing vit features for semantic appearance transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10748–10757, 2022. 6

[27] Su Wang, Chitwan Saharia, Ceslee Montgomery, Jordi Pont-Tuset, Shai Noy, Stefano Pellegrini, Yasumasa Onoe, Sarah Laszlo, David J Fleet, Radu Soricut, et al. Imagen editor and editbench: Advancing and evaluating text-guided image inpainting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18359–18369, 2023. 2

[28] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004. 6

[29] Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with natural language. *arXiv preprint arXiv:2312.04965*, 2023. 2, 3, 4, 5, 6, 11

[30] Fei Yang, Shiqi Yang, Muhammad Atif Butt, Joost van de Weijer, et al. Dynamic prompt learning: Addressing cross-attention leakage for text-based image editing. *Advances in Neural Information Processing Systems*, 36:26291–26303, 2023. 11

[31] Kai Zhang, Lingbo Mo, Wenhu Chen, Huan Sun, and Yu Su. Magicbrush: A manually annotated dataset for instruction-guided image editing. In *Advances in Neural Information Processing Systems*, 2023. 2

[32] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[33] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *Computer vision–ECCV 2016: 14th European conference, amsterdam, the netherlands, October 11-14, 2016, proceedings, part v 14*, pages 597–613. Springer, 2016. 2

[34] Siyu Zou, Jiji Tang, Yiyi Zhou, Jing He, Chaoyi Zhao, Rongsheng Zhang, Zhipeng Hu, and Xiaoshuai Sun. Towards efficient diffusion-based image editing with instant attention masks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7864–7872, 2024. 2, 6

A. Related Work

A.1. Diffusion-based Image Editing

Diffusion-based image editing strives to (1) preserve the visual content of a given source image while (2) modifying specific regions as instructed by text prompts. Prompt-to-Prompt (P2P) [13] pioneered this line of research by introducing *cross-attention scheduling*, which injects the cross-attention maps obtained during the reconstruction of the source image back into the editing process. Subsequent works further refined attention manipulation: MasaCtrl [3] imposes *mutual self-attention control* to maintain spatial consistency, whereas Free-Prompt-Editing (FPE) [19] decomposes cross-attention scheduling layer by layer for finer control.

A complementary thread focuses on *inversion*, operating under the intuition that an accurate inversion of the source image yields higher-quality edits. Several methods optimize text embeddings during inversion [5, 18, 21, 29, 30]; among them, InfEdit [29] proposes a training-free *Virtual Inversion* technique that achieves state-of-the-art results on multiple benchmarks.

Despite these advances, *multi-object* image editing remains under-explored. ZRIS [9] handles multi-object cases, but segments objects for *referring image segmentation*, not editing. Editing multiple target objects sequentially is straightforward but computationally expensive, as each object requires a separate diffusion pass. Our work addresses this gap by proposing a training-free framework that simultaneously handles multiple editing prompts and aligns user intent with the attention mechanism, enabling efficient multi-object edits without sacrificing quality.

A.2. Mitigating Attribute Leakage

Attribute leakage occurs when a diffusion model assigns an attribute to an unintended object. Early work addressed the problem by injecting explicit linguistic structure: StructureDiffusion Guidance [10] constrains generation with a constituency tree or scene graph, while Attend-and-Excite [4] and SynGen [23] refine cross-attention so that each word attends to a single spatial region. These methods focus on the *attention maps* themselves, yet leakage can also stem from *text embeddings*: even a perfect attention map fails if the prompt embedding is semantically entangled.

ToMe [14] tackles embedding-level entanglement via *End Token Substitution* (ETS). It replaces the EOS embedding of the full prompt (e.g., "a yellow cat and a white dog") with the EOS embedding of a stripped prompt that omits attributes (e.g., "a cat and a dog"), thereby suppressing color–attribute leakage ("yellow dog", "white cat"). However, ETS does not address noun-to-noun confusion ("cat" versus "dog") and, being designed for pure image generation, offers no guarantee of consistency with a given source image—an essential requirement for editing. DPL [30] reduces leakage by iteratively optimizing token embeddings at inference time to align cross-attention maps with the prompt, yet this costly optimization still leaves leakage when EOS embeddings remain entangled (see Figure 3).

In summary, existing approaches either leave EOS embeddings untouched or require high-cost optimization. Our method instead offers a lightweight, *optimization-free* pipeline that simultaneously disentangles embeddings and aligns attention, achieving lower attribute leakage while preserving faithfulness to the source image.

B. Benchmark Construction Details

Benchmark overview Our benchmark is designed to evaluate attribute leakage in image editing tasks using diffusion models. Unlike existing benchmarks that focus on image quality and background preservation, our benchmark emphasizes preventing unintended changes in both target-external and target-internal regions. It consists of 20 diverse images, semi-automated object masks, and succinct prompt pairs for various editing types. To comprehensively evaluate models, we generate 10 random edit prompts for each combination of 5 edit types and 1–3 edited objects per image, resulting in a total of 3,000 diverse editing scenarios. By covering diverse editing scenarios and offering precise evaluation metrics, our benchmark provides a robust framework for improving the precision of image editing methods. Figure 6 illustrates examples, showing the source images, object masks, and associated editing prompts.

Image selection We curated a dataset of 20 images, evenly split between natural and artificial scenes, to provide diverse and challenging editing scenarios. All images were drawn from both free image repositories and the PIE-bench dataset [29]. To ensure complexity, we included only images containing at least three distinct objects.

Prompt construction ALE-Bench provides five editing types. The prompt templates for different editing types are as follows:

1. Color change: "{color}-colored {object}" (e.g., "car" $\rightarrow$ "red-colored car").
2. Object change: "{new object}" (e.g., "car" $\rightarrow$ "bus").
3. Material change: "{object} made of {material}" (e.g., "car" $\rightarrow$ "car made of gold").
4. Color and object change: "{color}-colored {new object}" (e.g., "car" $\rightarrow$ "blue-colored bus").
5. Object and material change: "{new object} made of {material}" (e.g., "car" $\rightarrow$ "bus made of gold").

We intentionally excluded combinations like "color and material" and "color, object and material" because such cases often lead to unrealistic or ambiguous prompts, such as "**silver**-colored car made of **gold**". These kinds of descriptions are inherently challenging to interpret or generate, even for a human, making them impractical editing scenarios.

For each image, we generated 10 unique and random edit prompt instances for every combination of edit type and number of objects to edit. These prompts were created using attribute dictionaries containing target instances for colors, objects, and materials, with the assistance of ChatGPT to ensure diversity and consistency. This approach results in a systematic exploration of the attribute space across 20 images, 5 edit types, and varying numbers of objects, covering a total of 3,000 unique editing scenarios. Additionally, we emphasize the importance of user convenience by designing minimal prompt pairs that specify only the intended modification, avoiding the verbosity commonly seen in previous benchmarks.

Evaluation metrics In addition to standard metrics from PIE-bench—such as structural distance, background preservation (PSNR, SSIM, LPIPS, MSE), and editing performance (CLIP similarity)—we propose two novel metrics specifically designed to evaluate attribute leakage. The Target-External-Leakage Score (TELS) metric quantifies unintended changes to background regions during editing. This is calculated by measuring the CLIP scores between the background regions of the edited image and the target prompt. Lower TELS indicate minimal impact on the background, ensuring that non-target regions remain unaffected. The Target-Internal-Leakage Score (TILS) metric captures unintended cross-influence between multiple edited objects. For each edited object, we compute the CLIP scores between its edited region and the prompts intended for other objects, then take the mean scores across all object pairs. Lower TILS indicate that edits are confined to their respective objects without unintended interactions or overlaps.

Comparison with LoMOE-Bench LoMOE-Bench [7] evaluates overall fidelity in multi-object editing using approximately 1k edits across 64 images. In contrast, ALE-Bench focuses on probing *attribute leakage*, generating *3k edits* from just 20 carefully selected images. Rather than scaling the dataset broadly, ALE-Bench emphasizes depth by designing diverse, leakage-prone scenarios for each image. Since each additional image requires new object masks and source–target prompt pairs, annotation costs grow linearly. As a result, the two benchmarks serve complementary purposes: LoMOE-Bench measures broad editing fidelity, while ALE-Bench targets leakage robustness.

C. Experiments Details

Prompt construction For methods such as MasaCtrl and FPE that require only a single target prompt, the target prompt was constructed by concatenating all target object prompts with "and" to form a prompt. For methods like P2P and InfEdit that require both a source and a target prompt, the source prompt was similarly created by concatenating the source object prompts, while the target prompt was constructed by concatenating the target object prompts.

Method	P2P	MasaCtrl	FPE	InfEdit	ALE (Ours)
Runtime (sec) $\downarrow$	61.2	63.6	50.9	5.41	**4.31**

Table 4. Average runtime per edit on an RTX 6000 Ada Gen.

Hyperparameters For our method, we set the inference steps to 15 and the mask dilation ratio to 0.01, corresponding to a dilation of seven pixels. The self-attention control schedule was adjusted according to the type of edit: 1.0 for colors, 0.5 for objects, color+object, and material+object, and 0.6 for material. The same self-attention control schedule was applied to InfEdit and P2P, as this hyperparameter is shared. For all other hyperparameters of the baseline methods (MasaCtrl, FPE, P2P, InfEdit), we used the default settings provided in their official implementations.

D. Additional Results

Runtime comparison As shown in Table 4, ALE and InfEdit achieve significantly faster runtimes compared to other baselines, requiring only a few seconds per edit. This efficiency comes from leveraging virtual inversion via DDCM. In contrast, methods like P2P, MasaCtrl, and FPE rely on more expensive DDIM or null-text inversion processes, resulting in runtimes of nearly one minute per edit.

By object count Tables 5, 6, and 7 present the quantitative evaluation of our method and baselines on ALE-Bench across different numbers of editing objects. For the baseline methods, TELS and TILS decrease as the number of edited objects increases, as editing more objects provides a more detailed description of the image, reducing ambiguity. This trend highlights the baselines' dependence on long and detailed prompts. However, their editing performance decreases with an increasing number of edited objects, revealing their limitations in handling complex edits. In contrast, our method demonstrates robust performance across all object counts, consistently achieving the lowest leakage values, preserving structure and background, and maintaining competitive or superior editing performance.

By edit type We compare our methods with baselines across different edit types in Tables 8, 9, 10, 11, and 12. Across all edit types, our method consistently outperforms baselines by achieving lower leakage, better structural and background preservation, and strong editing performance. We provide more qualitative examples on ALE-Bench for each edit type in two objects editing in Figure 14.

E. Ablation Study Results

Ablation on EOS embedding methods To evaluate the effect of EOS embeddings, we studied several methods of modifying EOS embeddings: (1) Naive: No modification, using the original EOS embeddings; (2) Zeros: Replacing EOS embeddings with zero-valued vectors; (3) BOS: Substituting EOS embeddings with BOS (beginning-of-sequence) embeddings; (4) Empty String: Using EOS embeddings derived from an empty string. In Figure 4, our method demonstrates robust results across various scenarios, while the other methods often produce images that fail to follow the edit prompt or exhibit attribute leakage. A detailed quantitative comparison is provided in Table 13.

Another EOS modification method is proposed in [14], named End Token Substitution (ETS). ETS substitutes an embedding of EOS in a full prompt into an embedding of EOS in a rephrased prompt, which deletes all attribute expressions, e.g. “a yellow cat and a white dog” into “a cat and a dog”. In Figure 10, TI leakage in ETS is observed, e.g. cats are generated instead of a cat and a dog, and jar is generated in a region where ghost should be. RGB-CAM is applied for both methods, therefore the cross-attention mappings are aligned with the prompt.

Our method consistently achieves the best editing performance while maintaining competitive structure and background preservation metrics. In contrast, the other methods reveal a trade-off between reducing leakage and maintaining high editing performance, highlighting the effectiveness of our approach in balancing these objectives.

Ablation on RGB-CAM and BB The results in Table 14 demonstrate the complementary strengths of RGB-CAM and BB in our method. While RGB-CAM effectively reduces TI leakage by confining edits to the targeted objects, its impact on TE leakage and background preservation is limited. Conversely, BB significantly lowers TE leakage by preserving non-target regions, improving background quality but slightly reducing editing performance. Combining all components (Ours) achieves the best overall balance, minimizing leakage while preserving structure and background, and maintaining strong editing performance, highlighting the synergy of these components.

Evaluation on PIE-Bench We also evaluated our method on the existing PIE-Bench [17] in addition to ALE-Bench. Since our method does not support all edit types in PIE-Bench, we conducted experiments on the four edit types that are compatible: *object change*, *content change*, *color change*, and *material change*.

PIE-Bench only considers scenarios with a single object editing, so we excluded the TI Leakage metric. When running our method, we used the blend word provided by PIE-Bench as the SAM prompt for mask generation. In cases where mask segmentation failed, we edited the image without cross-attention masking and background blending.

In Table 15, the results show that our method demonstrated the lowest attribute leakage and high editing performance among all methods, even on PIE-Bench. These findings further validate the robustness and versatility of our approach across different benchmarks. We also provide qualitative examples for each edit type from the PIE-Bench experiments in Figure 15.

Ablation on self-attention injection schedule The degree to which the structure of a source image needs to be preserved varies depending on the edit type. For edits like color changes, maintaining the original structure is crucial, while object changes may require more deviation from the source. Figure 8 shows the effect of the self-attention schedule across various scenarios. Adjusting the schedule from 0.0 to 1.0 shows that higher values preserve more structure, while lower values allow greater flexibility. Thus, selecting the appropriate self-attention schedule depends on the specific goals of the task. The hyperparameters we used were chosen based on these experimental findings.

F. Limitations

ALE-Bench While our benchmark provides a robust framework for evaluating attribute leakage in image editing, it has certain limitations. First, the range of editing tasks is currently limited to basic and mixed edits such as color, object, and material changes. More complex editing types, such as style transfer or pose modifications, are not covered in ALE-Bench. However, defining attribute leakage in edits like style transfer is inherently ambiguous, as such edits often involve holistic changes to the image, making it unclear which regions should remain unaffected. Addressing these challenges would require redefining attribute leakage for these contexts and designing new evaluation metrics tailored to these specific tasks. Second, the dataset size (20 images) may limit the evaluation of models trained on larger or more diverse datasets. Future updates of ALE-Bench could expand its scope by incorporating additional images, and more diverse editing types to overcome these limitations.

Failure cases Our framework leverages two backbone models, a pre-trained diffusion model and a segmentation model, Grounded-SAM. Consequently, it may fail when the task exceeds the capabilities of these backbone models. For instance, overly rare or complex prompts that the pre-trained diffusion model cannot handle (Figure 12), objects that are difficult for the segmentation model to recognize, or incomplete segmentation masks generated by the model (Figure 13) can lead to unsatisfactory results. However, since our method operates in parallel with advancements in these backbone models, we anticipate that such failure cases will decrease as these models continue to improve.

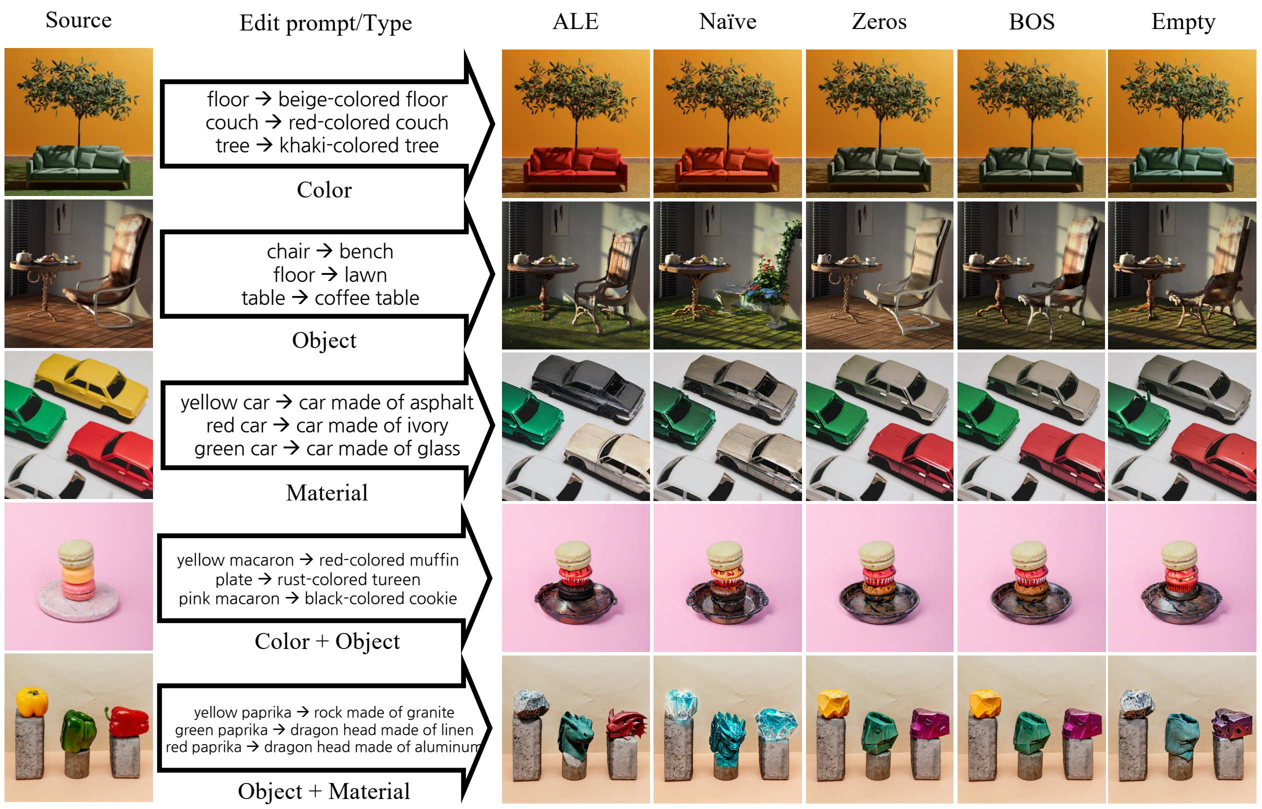

Source
Edit prompt/Type
ALE
Naïve
Zeros
BOS
Empty
floor → beige-colored floor
couch → red-colored couch
tree → khaki-colored tree
Color
chair → bench
floor → lawn
table → coffee table
Object
yellow car → car made of asphalt
red car → car made of ivory
green car → car made of glass
Material
yellow macaron → red-colored muffin
plate → rust-colored tureen
pink macaron → black-colored cookie
Color + Object
yellow paprika → rock made of granite
green paprika → dragon head made of linen
red paprika → dragon head made of aluminum
Object + Material

Figure 9. Qualitative examples from the EOS ablation study. While our method produces convincing results, other methods fail to generate the target object or exhibit attribute leakage. For instance, using the naive EOS to edit an object generates plants in place of the chair. This occurs due to attribute leakage from the word lawn to bench, resulting in chair-shaped flowers.

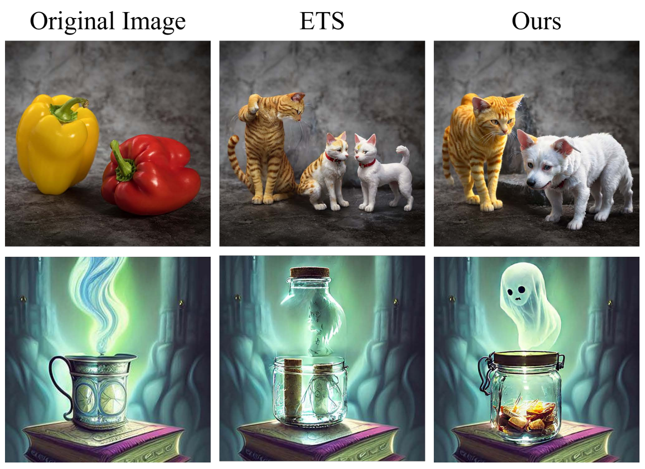


Figure 10. ETS, which is proposed in [14], fails to generate intended results. Edit prompt: (up) yellow paprika → yellow cat, red paprika → white dog. (down) cup → jar, steam → ghost.

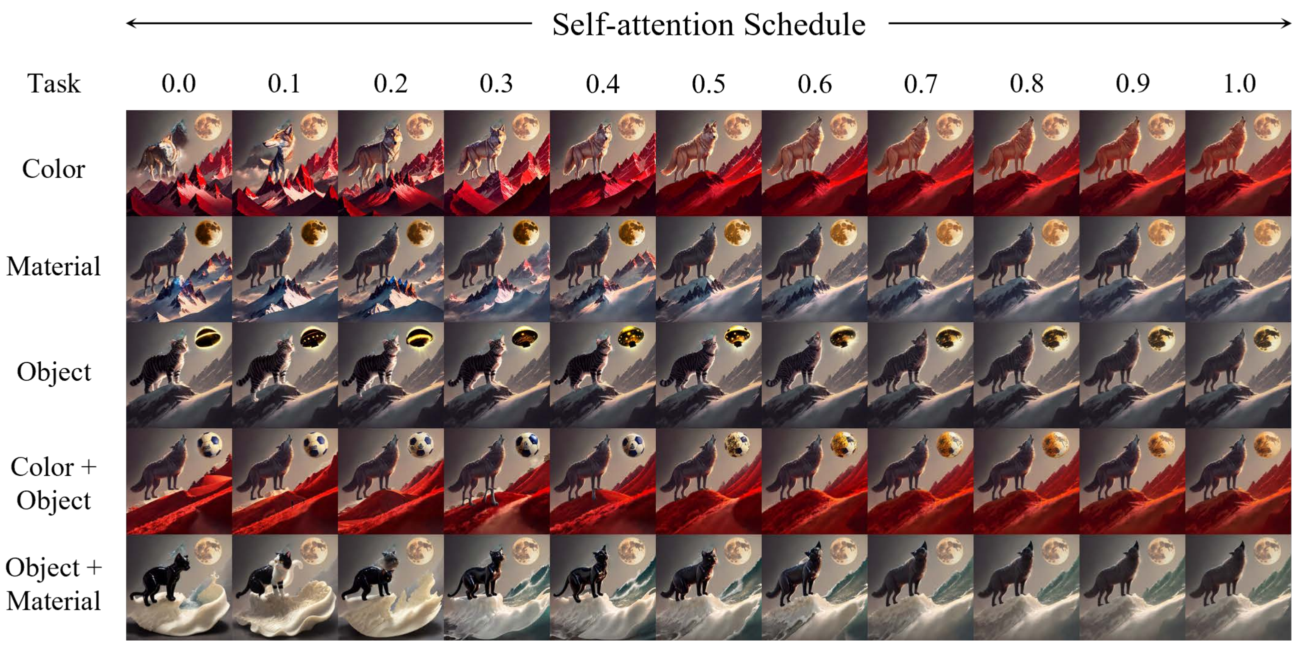


Figure 11. Ablation study on self-attention injection schedule. A schedule value specifies the fraction of early denoising steps during which self-attention maps from the source image are injected (e.g., 0.3 → first 30 % of steps). Larger values preserve more of the source structure and content, whereas smaller values grant greater freedom to satisfy the edit. The optimal schedule therefore varies by edit type. Prompts for each editing type are: (1) color: wolf → cream-colored wolf, mountain → crimson-colored mountain, (2) material: mountain → mountain made of crystal, moon → moon made of gold, (3) object: wolf → cat, moon → UFO, (4) color + object: moon → navy-colored soccer ball, mountain → crimson-colored hill, (5) object + material: cat → wolf made of rubber, mountain → wave made of ivory.

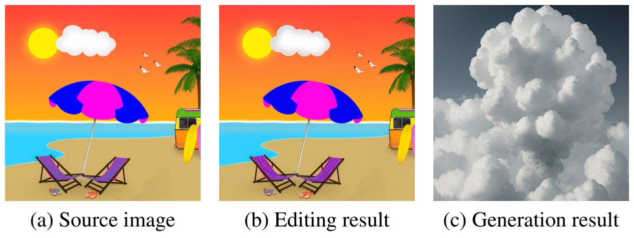


Figure 12. Failure case due to the base model's inability. Editing prompt: cloud → cloud made of chrome. Figure 12c illustrates the generation result when given the prompt "cloud made of chrome".

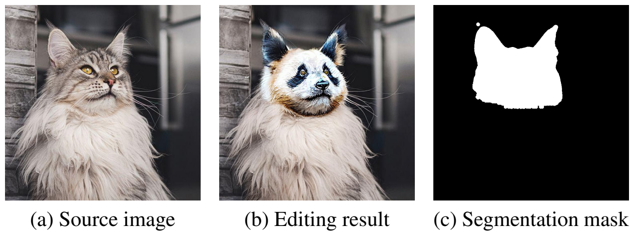


Figure 13. Failure case due to SAM segmentation fail. Editing prompt: . . . cat . . . → . . . panda . . .. Figure 13c shows the unsuccessful segmentation of SAM.

Method	TELS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
				PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	24.91	0.1513	21.53	10.29	0.5306	0.10342	0.4737
MasaCtrl	23.36	0.1012	20.58	13.74	0.3671	0.05250	0.6645
FPE	24.42	0.1172	**22.94**	11.66	0.4677	0.08009	0.5194
InfEdit	21.98	0.0504	22.71	15.34	0.2495	0.04359	0.7057
ALE	**16.41**	**0.0088**	22.62	**30.01**	**0.0405**	**0.00167**	**0.9049**

Table 5. Quantitative evaluation of **editing one object** for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	20.87	17.52	0.1499	20.41	11.11	0.4506	0.08699	0.5560
MasaCtrl	19.58	16.90	0.0911	19.92	14.99	0.2886	0.04058	0.7357
FPE	20.44	17.68	0.1141	21.72	12.81	0.3880	0.06439	0.6050
InfEdit	19.16	16.86	0.0485	21.52	16.69	0.2026	0.03288	0.7719
ALE	**16.00**	**15.42**	**0.0165**	**22.06**	**30.06**	**0.0360**	**0.00146**	**0.9235**

Table 6. Quantitative evaluation of **editing two objects** for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	18.80	17.00	0.1531	20.05	12.05	0.3674	0.07318	0.6469
MasaCtrl	17.60	16.58	0.0866	19.53	16.26	0.2231	0.03245	0.8037
FPE	18.36	17.09	0.1180	20.99	14.00	0.3153	0.05247	0.6911
InfEdit	17.62	16.51	0.0463	21.10	18.18	0.1580	0.02540	0.8350
ALE	**15.89**	**15.36**	**0.0246**	**22.19**	**30.01**	**0.0323**	**0.00154**	**0.9426**

Table 7. Quantitative evaluation of **editing three objects** for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	23.20	18.02	0.1467	21.94	11.03	0.4529	0.0898	0.5753
MasaCtrl	21.70	17.35	0.0964	21.64	14.59	0.3113	0.04557	0.7283
FPE	22.33	17.94	0.1065	**23.23**	12.66	0.3926	0.06901	0.6266
InfEdit	19.68	17.31	0.0343	23.16	18.54	0.1401	0.02695	0.8347
ALE	**17.63**	**16.21**	**0.0089**	23.12	**32.97**	**0.0288**	**0.00079**	**0.9309**

Table 8. Quantitative evaluation of the **color change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	20.65	17.78	0.1535	20.38	11.23	0.4457	0.08750	0.5485
MasaCtrl	19.52	17.33	0.0901	19.72	15.48	0.2744	0.03801	0.7417
FPE	19.76	17.58	0.1221	21.30	13.12	0.3732	0.06236	0.6103
InfEdit	18.67	17.12	0.0504	21.10	16.59	0.2114	0.03381	0.7607
ALE	**15.86**	**16.25**	**0.0197**	**21.82**	**29.03**	**0.0386**	**0.00182**	**0.9218**

Table 9. Quantitative evaluation of the **object change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	21.40	17.07	0.1549	20.75	11.07	0.4417	0.08674	0.5429
MasaCtrl	20.82	17.00	0.0866	20.93	15.39	0.2755	0.03763	0.7418
FPE	21.91	17.68	0.1151	22.64	13.14	0.3781	0.05908	0.5943
InfEdit	20.33	16.87	0.0387	22.59	17.71	0.1753	0.02559	0.7862
ALE	**17.15**	**15.96**	**0.0118**	**22.94**	**30.63**	**0.0339**	**0.00120**	**0.9248**

Table 10. Quantitative evaluation of the **material change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	21.62	17.10	0.1483	20.71	11.12	0.4586	0.0896	0.5786
MasaCtrl	19.59	16.15	0.0986	19.18	14.43	0.3142	0.04634	0.7281
FPE	20.82	17.11	0.1159	21.40	12.42	0.4079	0.07175	0.6081
InfEdit	19.92	16.11	0.0619	21.23	15.04	0.2543	0.04503	0.7333
ALE	**15.30**	**14.01**	**0.0231**	**22.15**	**28.60**	**0.0407**	**0.00205**	**0.9206**

Table 11. Quantitative evaluation of the **color and object change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	20.76	16.31	0.1538	19.54	11.29	0.4488	0.0857	0.5490
MasaCtrl	19.28	15.86	0.0929	18.57	15.07	0.2893	0.04166	0.7332
FPE	20.55	16.62	0.1224	20.86	12.77	0.3998	0.06604	0.5865
InfEdit	19.34	16.03	0.0567	20.80	15.81	0.2356	0.03842	0.7394
ALE	**14.55**	**14.51**	**0.0196**	**21.42**	**28.88**	**0.0393**	**0.0019**	**0.9201**

Table 12. Quantitative evaluation of the **object and material change** edit type for ALE and baselines on ALE-Bench.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
Naive	16.02	15.81	0.0156	21.86	30.14	0.0359	0.0014	0.9232
Zeros	**15.74**	**15.23**	**0.0107**	20.78	**31.22**	**0.0327**	**0.0011**	**0.9254**
BOS	15.76	15.27	0.0115	20.87	31.09	0.0334	0.0011	0.9241
Empty String	15.86	15.33	0.0139	21.25	30.61	0.0342	0.0013	0.9248
ALE	16.03	15.28	0.0167	**22.20**	30.04	0.0361	0.0014	0.9228

Table 13. Ablation study on different strategies for handling EOS embeddings in the prompt. While ALE shows slightly higher leakages compared to others, it achieves the best editing performance. All experiments were conducted with both RGB-CAM and BB applied.

Method	TELS ↓	TILS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
					PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
ORE	20.05	16.87	0.0521	21.81	16.16	0.2182	0.0380	0.7591
ORE+RGB	18.99	15.46	0.0436	**22.42**	17.48	0.1805	0.0291	0.7887
ORE+BB	16.12	16.58	**0.0164**	21.56	29.88	0.0368	0.0015	0.9219
ALE	**16.03**	**15.28**	0.0167	22.20	**30.04**	**0.0361**	**0.0014**	**0.9228**

Table 14. Ablation study comparing the components of our method: object-restricted embeddings (ORE), region-guided blending cross-attention masking (RGB), and background blending (BB). RGB markedly reduces TILS, whereas BB substantially lowers TELS. When ORE is used *without* RGB, it relies solely on the base embedding E'_{base} (i.e., the ORE and ORE + BB cases). Integrating all three components (ALE) yields the best overall performance across nearly every metric, underscoring their complementary strengths.

Method	TELS ↓	Structure Distance ↓	Editing Performance ↑	Background Preservation			
				PSNR ↑	LPIPS ↓	MSE ↓	SSIM ↑
P2P	26.20	0.1571	23.74	11.11	0.4270	0.0919	0.4600
MasaCtrl	24.48	0.0856	22.16	15.81	0.2540	0.0334	0.6803
FPE	25.64	0.1265	**23.89**	13.35	0.3499	0.0581	0.5346
InfEdit	24.51	0.0446	22.92	19.41	0.1519	0.0168	0.7581
ALE	**22.94**	**0.0238**	22.87	**28.77**	**0.0580**	**0.0046**	**0.8865**

Table 15. Evaluation results on PIE-Bench for compatible edit types (object change, content change, color change, and material change). Our method achieves the lowest TELS and demonstrates the best structure and background preservation while maintaining competitive editing performance.

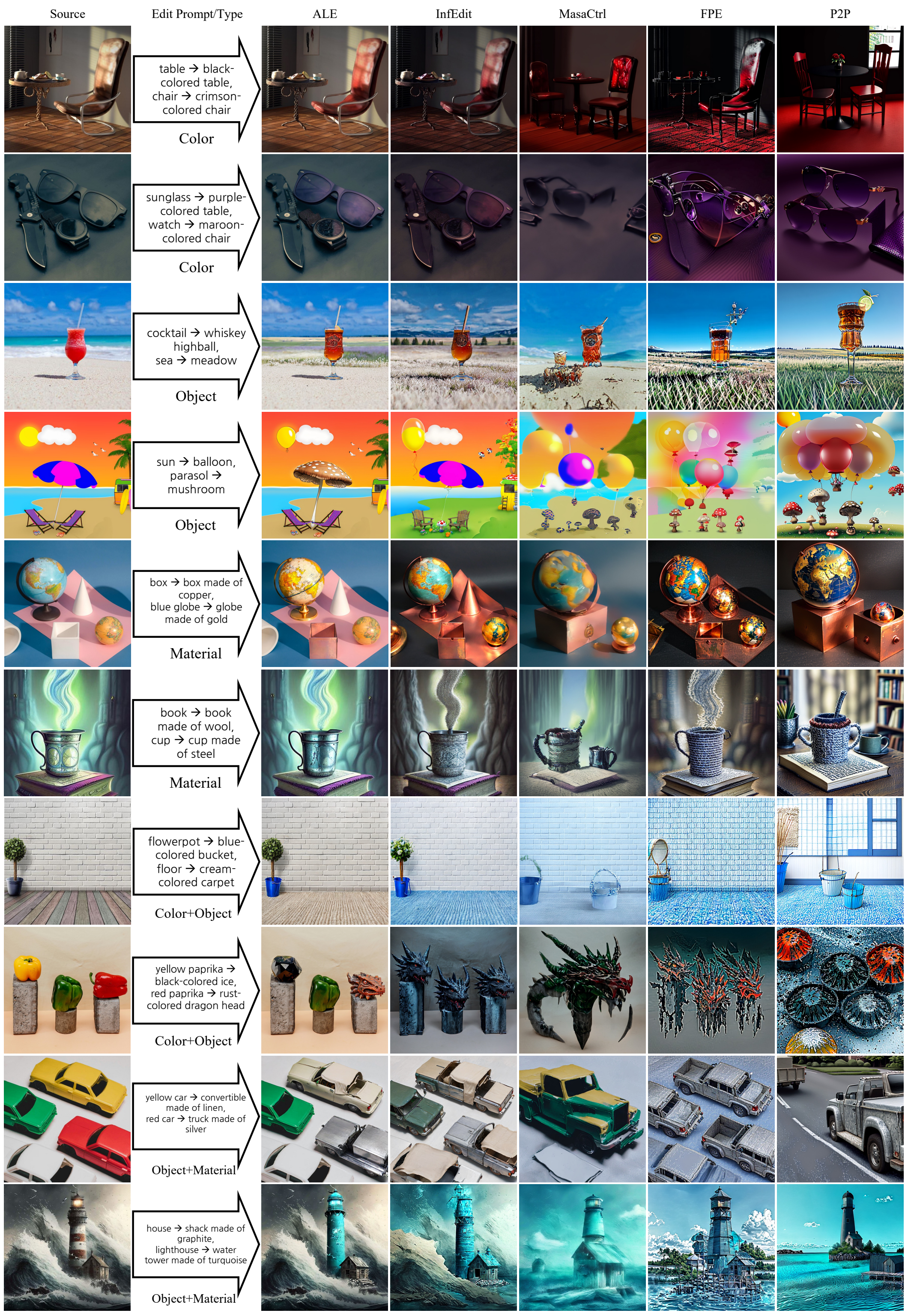

Source
Edit Prompt/Type
ALE
InfEdit
MasaCtrl
FPE
P2P
table → black-colored table, chair → crimson-colored chair
Color
sunglass → purple-colored table, watch → maroon-colored chair
Color
cocktail → whiskey highball, sea → meadow
Object
sun → balloon, parasol → mushroom
Object
box → box made of copper, blue globe → globe made of gold
Material
book → book made of wool, cup → cup made of steel
Material
flowerpot → blue-colored bucket, floor → cream-colored carpet
Color+Object
yellow paprika → black-colored ice, red paprika → rust-colored dragon head
Color+Object
yellow car → convertible made of linen, red car → truck made of silver
Object+Material
house → shack made of graphite, lighthouse → water tower made of turquoise
Object+Material

Figure 14. Qualitative examples of editing results for each edit type on ALE-Bench. Two examples are provided for each edit type. The left side of $\rightarrow$ represents the source prompt, and the right side represents the target prompt.

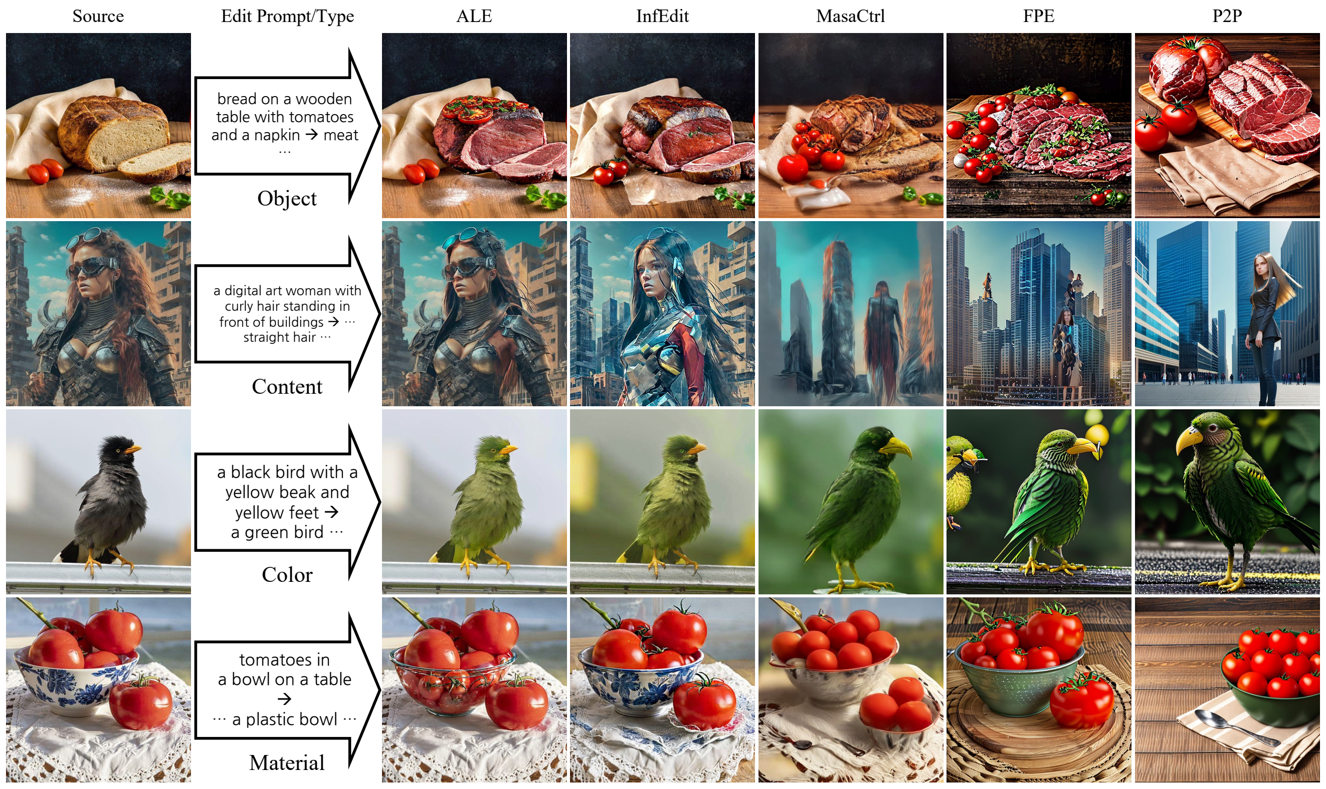

Source
Edit Prompt/Type
ALE
InfEdit
MasaCtrl
FPE
P2P
bread on a wooden table with tomatoes and a napkin → meat ···
Object
a digital art woman with curly hair standing in front of buildings → ··· straight hair ···
Content
a black bird with a yellow beak and yellow feet → a green bird ···
Color
tomatoes in a bowl on a table → ··· a plastic bowl ···
Material

Figure 15. Qualitative examples of editing results for the four compatible edit types on PIE-Bench: object change, content change, color change, and material change. In edit prompt column, the left side of the arrow → represents the source prompt, and the right side represents the target prompt, with unchanged parts omitted as "..." for brevity. Baseline methods exhibit attribute leakage or fail to preserve the source image structure, while our method achieves more precise edits with minimal leakage.