Region-aware Presentation Image Editing with Diffusion Models

Group24 R13922198 CHU WEI-KE, R14922146 Wu ShanKai,
R14921056 XU YEN-CHIAO, P13942A03 Chang,Yi Hsuan
NTU MAI 2025 Fall Final Project

1 Introduction

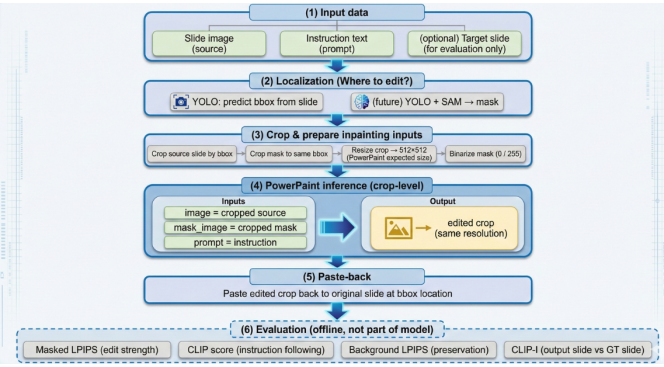

Figure 1: purpose pipeline of this works.

Modern presentation slides, posters, and educational materials frequently incorporate rich visual elements alongside textual content. However, when these artifacts are shared or archived as rasterized images, their structured layout and embedded text become inseparable from the visual regions, rendering direct edits challenging. Users often need to perform localized modifications—such as replacing an icon, updating an illustration, or harmonizing visual styles—while strictly preserving surrounding text and overall layout.

Existing diffusion-based image editing methods, despite their impressive generative power, typically operate on a global semantic level and treat images as fully editable canvases. Applying such approaches naively to presentation images often results in unintended distortion of text regions or layout shifts, severely limiting their practical utility in real-world content refinement scenarios.

To overcome this limitation, we propose a region-aware image editing framework specifically tailored for structured visual artifacts like presentation slides. Our key insight is that explicit spatial localization provides a strong structural prior, enabling precise control over editable regions. We leverage object detection to identify visual elements while protecting textual areas, and employ a parameter-efficiently adapted diffusion model to regenerate content within designated regions according to user prompts.

The proposed pipeline integrates region localization with prompt-guided diffusion editing in a unified manner. Leveraging this region-aware design, our framework establishes a robust baseline for localized edits. This enables users to rapidly update presentation visuals, enforce design consistency across slides, and modify content without access to original editable files. Our main contributions are:

- Identification of structured presentation images as a key practical challenge for current diffusion-based editing methods.
- A unified region-aware pipeline combining object detection for spatial localization and parameter-efficient diffusion adaptation for controllable regeneration.
- Demonstration of effective localized editing on presentation-style images, highlighting real-world applicability.

2 Dataset Design and Construction

To train a model that can *locate and edit images inside slide layouts*, we constructed a custom dataset by compositing an existing image-editing corpus into manually curated lecture slides. The design goal was to jointly supervise (1) *localization* of picture regions on a slide and (2) *prompt-based image editing* within those regions.

2.1 Base Slide Collection and Pre-processing

We started from the lecture slides used in the course and manually selected **75** slides as base layouts. The selection criteria were:

- slides containing only text (e.g., titles, bullet lists), and
- slides containing both text and graphical elements (icons, charts, figures).

Including slides with pre-existing graphics is important: it forces the model to learn to recognize "image-like" regions under realistic visual clutter, rather than relying on overly clean layouts.

For some slides, we manually removed parts of the original content (either text boxes or small images) to create *empty regions* with diverse shapes and positions. This yields:

- slides where an image should be inserted into a previously blank region, and

- slides where an existing visual element was removed, leaving a semantically meaningful "hole" (e.g., a location where a figure or illustration used to be).

This manual editing step ensures that the background layouts resemble real-world presentation slides, while still providing clearly defined, human-designed locations where images can be placed. All base slides were exported at a fixed resolution (e.g., 1280×720) and stored as PNG images. This resolution is used consistently throughout the dataset construction pipeline.

2.2 Manual Annotation of Editable Regions

For each of the 75 base slides, we manually annotated one or more *rectangular editable regions* where an image is allowed to appear. These regions are represented as axis-aligned bounding boxes in pixel coordinates:

$$\text{bbox} = (x_1, y_1, x_2, y_2),$$

where (x_1, y_1) and (x_2, y_2) denote the top-left and bottom-right corners of the region. The regions are chosen to satisfy two constraints:

1. **Semantic plausibility**: the box corresponds to a position where a picture would naturally be placed in a slide (e.g., to the right of a bullet list, under a title, or in a content area).
2. **Non-overlap with critical text**: the box avoids important titles and main bullet points, so that the model is discouraged from overwriting core textual content.

All bounding boxes for all slides are stored in a centralized JSON file (`bboxes.json`), keyed by slide ID. A single slide may have multiple candidate regions, which later allows us to place different images at different positions on the same base slide.

2.3 Re-using MagicBrush as Image-Editing Source

As the underlying image-editing supervision, we adopt the **MagicBrush**[3] dataset, which provides per example:

- **source image** – the original image before editing,
- **target image** – the edited image,
- **mask** – a binary mask indicating the edited region inside the image, and
- **prompt** – a natural language instruction describing the desired edit.

MagicBrush is designed for localized image editing, making it a suitable source of (source, target, mask, text) quadruples. However, its images exist in isolation, without any slide context. Our goal is to embed these editing examples into realistic slides so that the model learns to perform similar edits when the image is part of a slide rather than a standalone photograph.

We focus on *turn-1* examples in MagicBrush (the first editing turn for each image) to avoid complications from multi-step dialogues and to maintain a simple, single-instruction editing setup.

2.4 Slide-Level Composition Procedure

For each base slide and each selected MagicBrush example, we synthesize a new *slide-level editing instance* by pasting the MagicBrush source and target images into the annotated regions of the slide. The pipeline is as follows.

Step 1: Select a base slide and bounding box. A base slide is chosen from the curated set, and one bounding box is randomly sampled from that slide's annotated regions.

Step 2: Select a MagicBrush example. From the MagicBrush corpus, we pick an example with (source, target, mask, prompt). Only a single turn (turn 1) is used for each image.

Step 3: Scale and place the image within the bounding box. Let the original MagicBrush image have size (W_p, H_p) and the target bounding box size be (W_b, H_b). We compute a base scale factor

$$s_{\text{base}} = \min\left(\frac{W_b}{W_p}, \frac{H_b}{H_p}, 1.0\right),$$

so that the image fits inside the box without upscaling (we never enlarge images beyond their original resolution to avoid quality degradation).

The image is resized by s_{base}, and optionally further downscaled by a random factor within a range, while enforcing a *minimum side length* (e.g., ≥ 200 pixels) to keep the pasted image visually salient. The final patch size (W', H') is then randomly positioned inside the bounding box, yielding a placement $(x_{\text{left}}, y_{\text{top}})$. The placement is constrained to ensure the entire patch remains inside the bounding box.

Step 4: Consistent placement for source, target, and mask. The same geometric transform (scaling and translation) is applied to:

- the MagicBrush source image, which is pasted onto the base slide to form the **composite source slide**;
- the MagicBrush target image, which is pasted to form the **composite target slide**;
- the MagicBrush mask, which is resized and pasted onto an all-zero mask of slide resolution to form the **slide-level binary mask**.

Step 5: Sample generation and duplication handling. For each (slide, MagicBrush ID) pair, we may generate multiple samples with different random placements. To avoid ambiguity and overlaps:

- filenames encode slide ID, MagicBrush ID, and sample index (e.g., `001_9_source_001.png`),
- before writing a new sample, we check for filename collisions at both the filesystem level and the metadata level, and skip any duplicate.

Each resulting training instance is a slide-level quintuple:

```
{
  "source": "001_9_source_001.png",
  "target": "001_9_target_001.png",
  "mask":   "001_9_mask_001.png",
  "bbox":   [x1, y1, x2, y2],
  "prompt": "..."
}
```

Here, `bbox` records the slide-level bounding box that encloses the pasted patch, and `prompt` is directly inherited from the MagicBrush instruction. This representation simultaneously supports:

- *where to edit*: via the bounding box and the binary mask on the slide,
- *what edit to perform*: via the natural language prompt,
- *how the slide should look after editing*: via the composite target slide.

2.5 Dataset Organization and Splits

To support robust evaluation and avoid leakage between training and evaluation sets, we structure the dataset as:

- `dataset/train/` – slide-level PNGs used for training,
- `dataset/validation/` – slide-level PNGs used for hyperparameter tuning,
- `dataset/test/` – slide-level PNGs reserved for final evaluation.

For supervision, we maintain corresponding metadata files:

- `gt/train/meta.json`,
- `gt/validation/meta.json`,
- `gt/test/meta.json`.

Each `meta.json` file contains a `"samples"` list with entries of the form shown above.

The split strategy respects the original MagicBrush partitioning:

- **Training set**: uses the training split of MagicBrush. We select the first subset of MagicBrush IDs (e.g., the first 4000 IDs) and paste them onto the training slides.
- **Validation set**: uses a different subset of MagicBrush training IDs (e.g., the tail part of the training IDs) to avoid overlap with training IDs while remaining in the same distribution.
- **Test set**: uses MagicBrush's development (dev) split only, ensuring that none of the test images or prompts overlap with those used for training or validation.

Each split uses disjoint sets of base slides and disjoint sets of MagicBrush IDs, so that both the slide context and the underlying images seen at test time are unseen during training.

2.6 Discussion

The resulting dataset is specifically tailored to the "edit a small picture inside a slide" task:

- The model must **localize** an appropriate region for editing within a realistic slide layout, which can contain titles, bullet points, and existing graphics.
- The model must then perform **prompt-conditioned image editing** within that localized region, guided by the MagicBrush-style instruction and supervised by the composite target slide.
- The inclusion of noise such as surrounding text, icons, and varying background layouts encourages robustness to non-trivial, "messy" slide contexts.

By combining manually annotated slide layouts with a large-scale, high-quality image-editing corpus (MagicBrush), the dataset provides joint supervision for *layout understanding* and *fine-grained visual editing*. This design is crucial for the downstream objective: enabling a model to reliably edit small pictures embedded in presentation slides, rather than only editing isolated images.

3 YOLO-based Slide Image Localization

3.1 Objective

The primary goal of this module is to address the challenge of "slide image localization." To enable automated editing, we aim to train a high-precision object detection model capable of accepting a presentation slide as input and accurately predicting the "Bounding Box" of the target image region. This step is critical as it provides precise Spatial Guidance for the downstream generative editing pipeline, defining exactly where the edit should occur.

3.2 Methodology

We adopted the YOLOv8-Large[1] architecture for its superior balance between inference speed and detection accuracy.

Dataset Construction We utilized a composite dataset where images and masks from the MagicBrush dataset were randomly scaled and embedded into 75 curated slide layouts. To enhance robustness, we trained on "Source," "Target," and "Mask" views simultaneously, effectively tripling the dataset size.

Annotation The model is supervised to predict pixel-level bounding box coordinates (x_1, y_1, x_2, y_2).

3.3 Experimental Results

3.3.1 Training Convergence

The evolution of loss functions and performance metrics during training is illustrated in Figure 2.

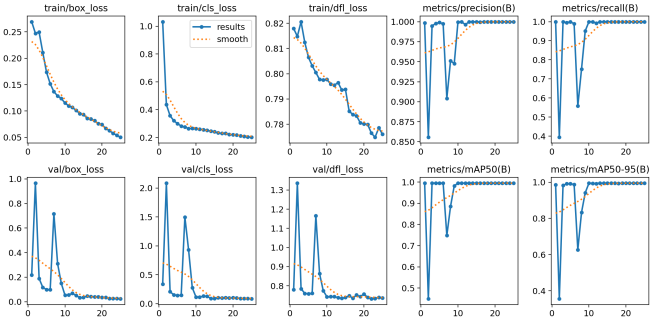

Figure 2: Training Loss and Metrics Curves. The training curves demonstrate a steady decrease in loss and rapid convergence of mAP metrics.

Loss Analysis The training curves demonstrate a steady, smooth decrease in both `train/box_loss` and `train/cls_loss`, indicating effective feature learning. Validation losses stabilized rapidly at near-zero values after initial fluctuations, confirming strong generalization without overfitting.

Performance Metrics

- **mAP50**: The Mean Average Precision at IoU=0.5 quickly reached 1.0, signifying perfect separation between the target object and the background.
- **mAP50-95**: Even under stricter IoU thresholds, the metric approached 1.0, confirming high localization precision.

3.3.2 Quantitative Evaluation: IoU Distribution

To rigorously evaluate localization precision, we computed the Intersection over Union (IoU) scores across the validation set.

Mean IoU (mIoU) The model achieved an exceptional mean IoU of 0.991.

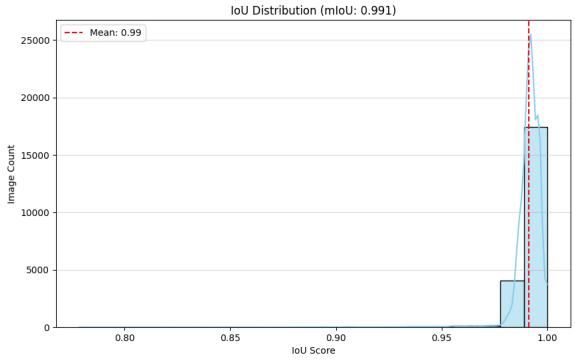

Figure 3: IoU Distribution Histogram. The histogram reveals a strong left-skewed distribution, indicating high precision.

Distribution Analysis The histogram in Figure 3 reveals a strong left-skewed distribution, with the vast majority of samples clustered in the 0.98 to 1.0 range. This indicates that the model rarely makes "passable" predictions; nearly all detections are "perfect."

3.3.3 Qualitative Results

We performed inference on random samples from the test set to visualize real-world performance.

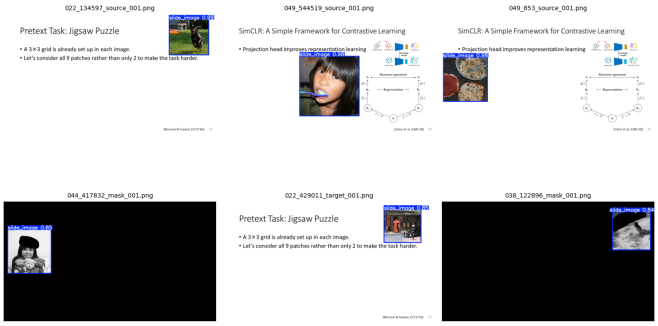

Figure 4: Qualitative Results on Test Samples. The YOLO model accurately localizes target images (blue boxes) even in complex layouts.

Robustness in Complex Layouts As shown in Figure 4, the YOLO model accurately localizes target images (indicated by blue bounding boxes) even within layouts cluttered with text, titles, and other visual elements. Confidence scores consistently exceed 0.85, often reaching 0.99.

Versatility The model performs consistently well across different input types, including full-color slides (Source/Target)

and binary mask images (Mask).

3.4 Conclusion

The quantitative and qualitative results confirm that our trained YOLO model solves the "slide image localization" task with near-perfect accuracy. The mIoU score of 0.991 validates the model's capability to extract relevant features from the composite dataset, ensuring highly precise spatial guidance for subsequent image editing tasks.

4 Evaluation

This section evaluates the effectiveness of the proposed automated editing pipeline. We first detail the experimental setup in Sec. 4.1, followed by the definition of evaluation metrics in Sec. 4.2. Our experiments are designed to compare two scenarios: the **Automated Inference** (YOLO-based) and the **Oracle Inference** (Ground Truth-based), quantifying the impact of detection errors on editing quality.

4.1 Experimental Setup

To quantify the error introduced by automated object detection, we establish an **Oracle Setting** as our baseline. By utilizing the Ground Truth Bounding Boxes directly as input, we determine the performance **Upper Bound** of the system, excluding interferences from detection inaccuracies.

For the PowerPaint inference phase, we implement the following configurations:

- **Task Prompting:** The `P_obj` token is automatically appended to all input prompts to explicitly activate the object editing mode of the model.
- **Model Variants:** We evaluate both the fine-tuned model (w/ LoRA) and the base model (w/o LoRA) to assess the impact of domain adaptation.
- **Resolution:** The inference resolution is standardized to 512×512 due to the architectural constraints of Stable Diffusion v1.5.

Detailed hyperparameters are provided in Table 3.

4.2 Evaluation Metrics

To comprehensively assess generation quality and semantic consistency, we adopt the following three metrics. Note that traditional pixel-wise metrics (e.g., L1/L2, SSIM, PSNR) are excluded, as they fail to account for the inherent stochasticity and diversity of generative editing tasks.

- **LPIPS [4]:** Unlike pixel-wise metrics which are sensitive to minor spatial shifts, LPIPS leverages pre-trained deep network features to compute perceptual distance, aligning better with human visual judgment. We report:
  - **Masked LPIPS:** Evaluates the generation quality specifically within the bounding box (masked region).
  - **Background LPIPS:** Assesses the capability of *Background Preservation* in non-edited regions.
- **CLIP Score (CLIP-S) [2]:** Computes the cosine similarity between the generated image and the target prompt in the CLIP feature space. This serves as a measure of **Instruction Following**.
- **CLIP Image Similarity (CLIP-I):** Calculates the feature similarity between the generated image and the ground truth target image, evaluating the overall **Semantic Consistency**.

4.3 Quantitative Results

Table 1 presents the quantitative comparison across four experimental configurations: combining two inference strategies (Oracle vs. Automated) with two model variants (Base vs. LoRA).

Performance Gap Analysis. As shown in Table 1, our overall scores fall behind state-of-the-art benchmarks on natural image datasets (e.g., MagicBrush). We attribute this gap to three primary factors identified in our analysis:

1. **High-Frequency Information Loss:** A significant bottleneck lies in the resolution constraints. Slide images contain dense text and sharp graphical lines (high-frequency information). The process of resizing crops to 512×512 and encoding them via the VAE (Variational Autoencoder) causes inevitable information loss and artifacts. When decoded and resized back to the original slide, these artifacts degrade the perceptual quality, resulting in higher LPIPS scores compared to natural images.
2. **Limitations of LoRA Fine-tuning:** Comparing the *Base* and *LoRA* rows, we observe that the improvement in Instruction Following (CLIP Score) is marginal. This suggests that the complexity of slide editing tasks—which requires understanding spatial layout and text preservation—may exceed the capacity of low-rank adaptation (Rank=16). While LoRA helps in domain adaptation, it appears insufficient to significantly alter the model's fundamental instruction-following capabilities without full parameter fine-tuning.
3. **Hardware Constraints:** Due to computational resource limitations, we were unable to perform full fine-tuning of the UNet. This restricted our ability to fully optimize the model for the slide domain, leading to the observed performance plateau.

Effectiveness of Automated Detection. Despite the aforementioned challenges, a positive observation can be drawn from

Table 1: **Quantitative Ablation Study.** We evaluate the impact of the detection pipeline (YOLO) and fine-tuning (LoRA). **Oracle**: GT Bounding Box. **Automated**: YOLOv8 Prediction. **Base**: Original PowerPaint. **LoRA**: Fine-tuned model. Comparison shows the trade-offs in different configurations.

Model	Inference Mode	Masked LPIPS ↓	Bg. LPIPS ↓	CLIP Score ↑	CLIP-I ↑
PowerPaint (Base)	Oracle (No-YOLO)	0.130	0.033	0.253	0.951
PowerPaint (LoRA)	Oracle (No-YOLO)	0.129	0.021	0.245	0.098
PowerPaint (Base)	Automated (YOLO)	0.162	0.041	0.256	0.942
PowerPaint (LoRA)	Automated (YOLO)	0.161	0.028	0.251	0.981

the **Background LPIPS**. The gap between *Oracle* and *Automated* settings is negligible. This indicates that our YOLOv8-based detector is highly precise, generating bounding boxes that closely match the Ground Truth. Consequently, the automated pipeline successfully localizes editing regions without introducing additional background noise, validating the feasibility of the proposed detection module.

4.4 Ablations

To verify our hypothesis, we utilized the original MagicBrush dataset to fine-tune PowerPaint via LoRA. We recorded the model outputs every 2,000 training steps. As shown in Figure 5, even after extensive training (up to 10,000 steps), the model fails to execute the requested edit (changing the table to a dog). Instead, it tends to reconstruct the original object, essentially ignoring the text prompt. This indicates that simply fine-tuning on the MagicBrush dataset is ineffective, as the model fails to learn the mapping between textual instructions and visual transformations.

This observation is further supported by the quantitative analysis in Table 2. We observe that fine-tuning leads to a slight decrease in CLIP Score (from 0.251 to 0.243) while maintaining a high CLIP-I score (0.849). This quantitative result aligns with the visual observation in Figure 5: the model prioritizes **preserving the original visual structure** over following the text instructions, suggesting that the model failed to learn the editing capabilities from the dataset.

Sensitivity to Guidance Scale. We further investigated the impact of the classifier-free guidance scale on generation quality. While a higher guidance scale typically encourages stricter adherence to text prompts, we empirically observed a severe trade-off in our fine-tuned model.

At lower scales ($s < 7.5$), the model consistently disregarded the editing instructions, defaulting to background reconstruction. Conversely, increasing the guidance scale ($s > 10$) did not yield correct semantic edits; instead, it introduced significant visual artifacts, such as **color saturation** and **high-frequency noise**, without achieving the desired object transformation. This inability to find an optimal inference configuration further suggests that the current LoRA adaptation failed to capture the robust semantic alignment required for this task.

5 Conclusion

In this paper, we introduced the task of **Region-aware Presentation Image Editing**, aiming to address the limitation of naive text-to-image models that often disrupt the global layout when modifying localized visual elements in slides. To facilitate this, we proposed an automated pipeline for constructing a presentation-image dataset and developed a unified framework integrating **YOLO**-based localization with **LoRA**-adapted diffusion. This design allows the model to focus exclusively on the editable region, theoretically preserving the structural integrity of the presentation.

Our experimental results present a mixed outcome. While the YOLO module successfully achieved high-precision localization, significantly improving the quality of the input fed into the generative model, the subsequent LoRA fine-tuning exhibited signs of "ineffective learning." Despite the domain adaptation, the model's output remained suboptimal, often prioritizing background reconstruction over semantic editing.

Based on these findings, we identify two primary directions for future research:

- **Full Parameter Fine-tuning:** We hypothesize that the limited capacity of LoRA may be insufficient to alter the model's fundamental sensitivity to text tokens in this specific domain. With adequate hardware support, full parameter fine-tuning could enable the model to better learn the mapping between editing instructions and visual transformations.

- **High-Fidelity Reconstruction (Super-Resolution):** We observed that the VAE used in Latent Diffusion Models frequently causes high-frequency distortion (e.g., text blurring) during the encoding-decoding process. To address this, future pipelines should incorporate super-resolution modules or advanced post-processing techniques to ensure that the visual quality of the edited slides meets professional standards.

We hope that our proposed dataset construction pipeline and region-aware framework serve as a foundational baseline for future advancements in structured document editing.

Prompt: ”Change the table for a dog”

(a) Source Image (b) Mask (c) Target (Ground Truth)

(d) Step 500 (e) Step 2,000 (f) Step 6,000 (g) Step 10,000

Figure 5: Visual ablation study on MagicBrush dataset. The first row displays the inputs and ground truth, while the second row shows the output of PowerPaint fine-tuned with LoRA at different training steps. No significant improvement is observed.

Table 2: **Generalization Analysis on MagicBrush Dataset.** We compare our fine-tuned model against state-of-the-art methods to assess domain generalization. **Metrics for InstructPix2Pix are cited from the original paper.*

Method	CLIP-I ↑	CLIP Score ↑
InstructPix2Pix w/ MB* [3]	0.852	0.276
PowerPaint (Base)	0.842	0.251
PowerPaint (LoRA w/ MB)	0.849	0.243

Table 3: **Hyperparameters for Training and Inference.** We fine-tune the UNet of PowerPaint on the slide dataset using the following settings.

Hyperparameter	Value
Base Model	PowerPaint-V1 (SDv1.5)
Training Resolution	512×512
LoRA Rank	16
Learning Rate	1×10^{-4}
Batch Size (Physical)	2
Num Epochs	50
Optimizer	AdamW
Training Hardware	NVIDIA GeForce RTX 4090
Denoise Steps	50
Guidance Scale	10

References

[1] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolo, 2023. Accessed: 2025-12-23. 3

[2] 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. 5

[3] 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*, volume 36, 2023. 2, 7

[4] 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 (CVPR)*, pages 586–595, 2018. 5