UIP2P: Unsupervised Instruction-based Image Editing via Edit Reversibility Constraint

Supplementary Material

Table of Contents

1. Ethics Statement

Advancements in localized image editing technology offer substantial opportunities to enhance creative expression and improve accessibility within digital media and virtual reality environments. Nonetheless, these developments also bring forth important ethical challenges, particularly concerning the misuse of such technology to create misleading content, such as deepfakes [12], and its potential effect on employment in the image editing industry. Moreover, as also highlighted by [10], it requires a thorough and careful discussion about their ethical use to avoid possible misuse. We believe that our method could help reduce some of the biases present in previous datasets, though it will still be affected by biases inherent in models such as CLIP. Ethical frameworks should prioritize encouraging responsible usage, developing clear guidelines to prevent misuse, and promoting fairness and transparency, particularly in sensitive contexts like journalism. Effectively addressing these concerns is crucial to amplifying the positive benefits of the technology while minimizing associated risks. In addition, our user study follows strict anonymity rules to protect the privacy of participants.

2. Runtime Analysis

Our method modifies the training objectives of IP2P by incorporating Edit Reversibility Constraint (ERC) and additional loss functions. However, these changes do not affect the overall runtime. Inference time remains comparable to the original IP2P framework, as we retain the same architecture and model structure. Consequently, our approach introduces no additional complexity or overhead in terms of processing time or resource consumption. This gives UIP2P an advantage over methods like MGIE [5] and SmartEdit [8], which rely on large language models (LLMs) during inference in terms of runtime and resource consumption.

3. Elo Rating System

The Elo rating system is a widely used method for ranking competitors in pairwise comparisons, originally designed for chess and later adopted in various domains, including generative model evaluation. It assigns each candidate a score that dynamically updates based on comparative performance. A higher Elo score indicates stronger performance relative to other candidates.

3.1. Implementation Details

Our implementation follows the standard Elo rating mechanism with the following key components:

- **Initial Rating:** Each model starts with a rating of **1500**.
- **K-Factor** ($K = 32$)**:** Governs the magnitude of rating updates.
- **Expected Score Calculation:** The probability of a model winning against another is computed as:

$$E_A = \frac{1}{1 + 10^{(R_B - R_A)/400}} \tag{1}$$

  where R_A and R_B are the Elo ratings of the two competing models.
- **Score Update Rule:** After each comparison, the winner's rating increases, and the loser's rating decreases:

$$R'_A = R_A + K(S_A - E_A) \tag{2}$$

  where $S_A = 1$ for a win, 0 for a loss, and 0.5 for a draw.

3.2. Interpreting Elo Scores

Elo scores provide an intuitive understanding of model performance:

- **Higher Scores:** Indicate models that consistently outperform others.
- **Score Differences:** A gap of 400 points implies the higher-rated model is **10 times more likely** to win.
- **Stability of Rankings:** As the number of votes increases, the Elo system converges, producing a reliable performance ranking.

By leveraging the Elo rating system, we ensure a **robust, adaptive, and comparative** evaluation framework for image-editing models. This approach provides a dynamic ranking that accounts for dataset variations and human annotation biases, aligning with best practices in benchmarking generative models.

4. Additional Details on Ablation Studies

4.1. Loss Functions

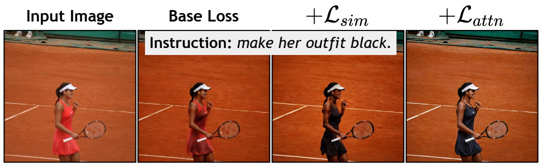

Figure 1. **Visual effects of loss components.** This figure demonstrates how different loss components affect the final editing results. The comparison shows the impact of various loss terms on edit quality and localization.

We focused our ablation studies on $\mathcal{L}_{\text{sim}}$ and $\mathcal{L}_{\text{attn}}$ because these losses are additional components beyond the core $\mathcal{L}_{\text{CLIP}}$ and $\mathcal{L}_{\text{recon}}$. The core losses are essential for ensuring semantic alignment and reversibility in Edit Reversibility Constraint (ERC), forming the foundation of our method. Without $\mathcal{L}_{\text{CLIP}}$ and $\mathcal{L}_{\text{recon}}$, the model risks diverging, losing its ability to preserve both the input's structure and its semantic coherence during edits.

Adding $\mathcal{L}_{\text{sim}}$ enables the model to perform edits more freely by encouraging alignment between image and textual embeddings, thereby expanding its capacity for complex and diverse transformations. On the other hand, $\mathcal{L}_{\text{attn}}$ refines the model's ability to focus on relevant regions during edits, improving localization and reducing unintended changes in non-targeted areas.

$\mathcal{L}_{\text{CLIP}}$ is applied between the input image and the edited image to ensure semantic alignment with the edit instruction. The reconstructed image is already constrained by $\mathcal{L}_{\text{recon}}$, which enforces structural and semantic consistency with the input. Adding $\mathcal{L}_{\text{CLIP}}$ to the reconstructed image would be redundant and could interfere with the reversibility objective. Our design does not apply $\mathcal{L}_{\text{CLIP}}$ to the reconstructed image to preserve the focus on reversibility and prevent conflicting optimization objectives.

4.2. Attention Across Noise Steps in Training

At training time, we sample two different noise steps for the forward and backward processes, which are conditioned on the input image and edit instruction. Attention consistency is enforced between these different noise steps to ensure that the model attends to the same regions during both forward and reverse edits. This is supported by the observation that cross-attention scores in instruction-based editing methods tend to be more consistent across timesteps, as the edit instruction remains fixed and the model's focus shifts only to the regions being edited (see Fig. 2).

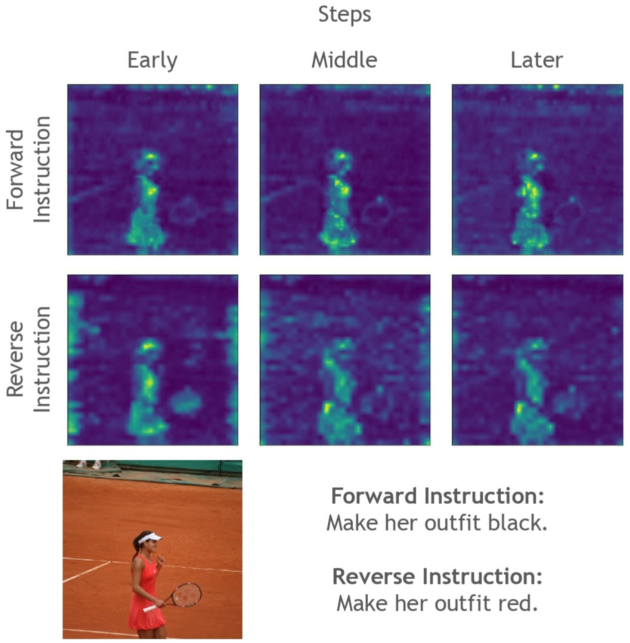

Figure 2. **Attention maps for diffusion steps.** Cross-attention maps for forward (top) and reverse (middle) instructions across early, middle, and later noise steps. The model enforces attention consistency, focusing on relevant regions for both edits.

Recent works, such as those by Guo et al. [6] and Simsar et al. [19], demonstrate that regularizing attention space with a mask during inference enables localized edits in IP2P. Our method builds on these ideas by incorporating attention consistency into the training phase, making it possible to focus on relevant regions from the start and avoiding the need for additional inference-time modifications.

5. Additional Qualitative Results

To further demonstrate the capabilities of our approach, we present additional qualitative comparisons in Fig. 3. These

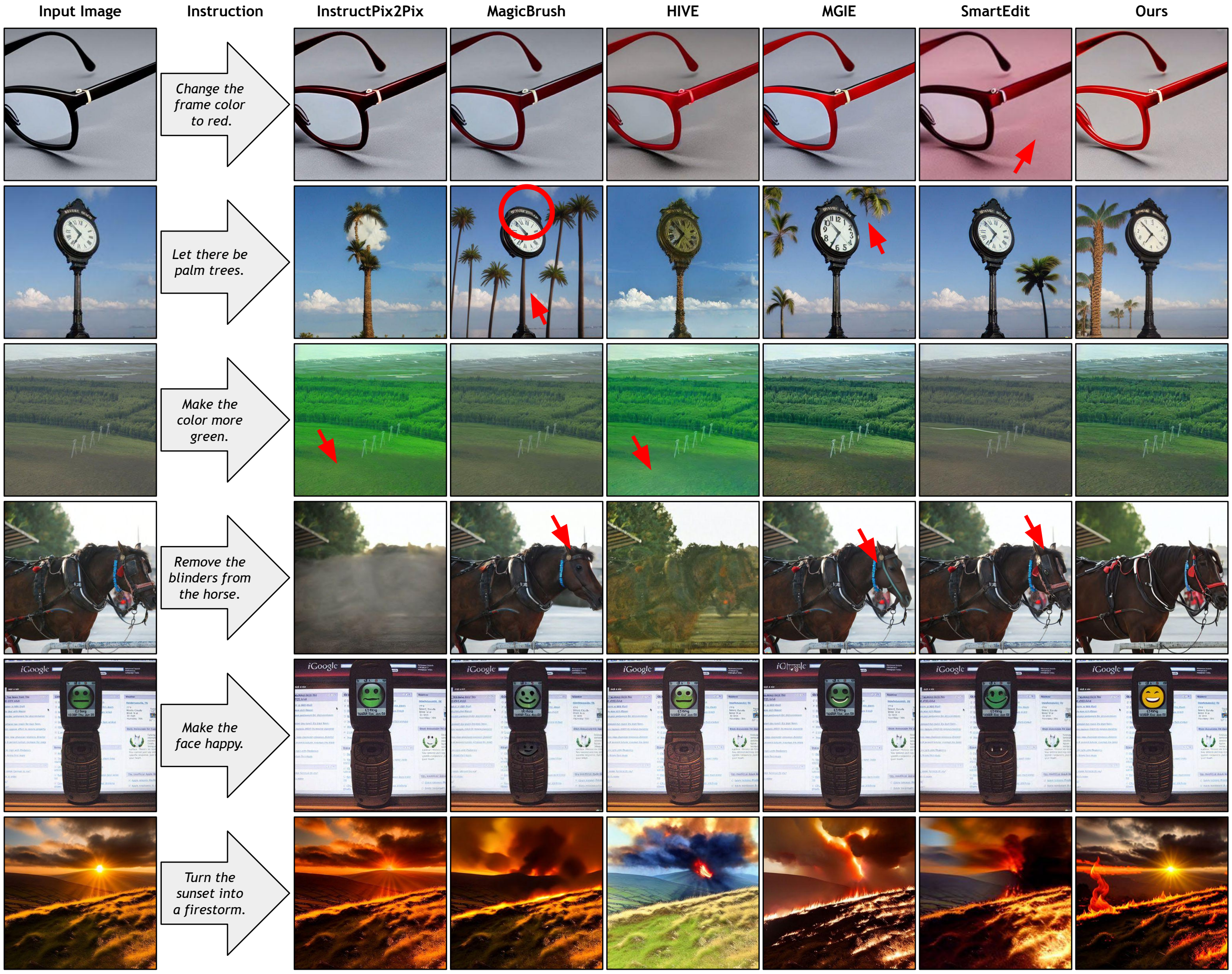
Input Image
Instruction
InstructPix2Pix
MagicBrush
HIVE
MGIE
SmartEdit
Ours
Change the frame color to red.
Let there be palm trees.
Make the color more green.
Remove the blinders from the horse.
Make the face happy.
Turn the sunset into a firestorm.

Figure 3. **Qualitative comparison of our method with baseline models for various editing instructions.** From left to right: Input image, edit instruction, and results from InstructPix2Pix, MagicBrush, HIVE, MGIE, SmartEdit, and our method. Our approach demonstrates superior fidelity and alignment with the provided instructions across diverse tasks, such as expression changes, color adjustments, object transformations, and creative edits. Red circles and arrows indicate drastic problems during the image editing.

results showcase the performance of our method against several baseline models, including InstructPix2Pix, MagicBrush, HIVE, MGIE, and SmartEdit, across a diverse set of editing instructions. These tasks range from simple edits, such as color adjustments and expression changes, to more challenging transformations, including object removal, style changes, and complex scene edits.

The comparison highlights that our method consistently achieves higher fidelity and better alignment with the provided instructions. For instance, when instructed to modify facial expressions, such as "make the face happy," our method produces more natural and expressive results. Similarly, for color adjustments, such as "make the color more green," our approach ensures vibrant and accurate edits that surpass the performance of baseline models. In more challenging scenarios, like "turn the sunset into a firestorm," our method maintains the structural integrity of the original image while executing the desired transformations. Furthermore, in creative edits, such as "remove the blinders from the horse," our model demonstrates exceptional precision and attention to detail.

6. Additional Quantitative Results

6.1. Evaluation on PIE-Bench

We apply our method to the PIE benchmark [9] to evaluate its performance on diverse editing tasks and compare it to IP2P, a representative feed-forward instruction-based editing method and a supervised alternative to our approach. Table 1 summarizes the results. The results show that our method outperforms IP2P across most metrics, including better preservation of structure (PSNR and SSIM), lower perceptual differences (LPIPS), and reduced mean squared error (MSE). These improvements demonstrate the scalability and versatility of our approach on a broader benchmark. This analysis is included in the revised manuscript to provide a comprehensive evaluation of our method.

Table 1. **Performance comparison on the PIE benchmark.** Lower values for Distance, LPIPS, and MSE indicate better performance, while higher values for PSNR, SSIM, Whole, and Edit indicate improved quality and structural preservation.

Methods	**Distance ↓**	**PSNR ↑**	**LPIPS ↓**	**MSE ↓**	**SSIM ↑**	**Whole ↑**	**Edit ↑**
InstructDiffusion	75.44	20.28	155.66	349.66	75.53	23.26	21.34
IP2P	57.91	20.82	158.63	227.78	76.26	23.61	21.64
Ours	**27.05**	**26.85**	**60.57**	**40.07**	**83.69**	**24.78**	**21.89**

6.2. Evaluation on Emu Edit

We apply our method to the Emu Edit benchmark [18] to evaluate its performance on diverse editing tasks and compare it to other baselines. Table 2 presents a quantitative comparison across multiple metrics, assessing the quality of the generated edits. Our approach achieves the best results across most metrics, including higher CLIP scores, which indicate better alignment with textual descriptions, and improved perceptual quality as measured by DINO. Notably, our method surpasses all baselines in CLIP$_{dir}$ and CLIP$_{im}$, demonstrating stronger instruction adherence and output realism. The improvements in these objective measures further support the effectiveness of our method in complex editing tasks.

Table 2. **Performance comparison on the Emu Edit benchmark.**Comparison with image-editing baselines evaluated on Emu Edit test set. For each method, we report CLIP, L1, and DINO metrics.

Method	CLIP$_{dir}$↑	CLIP$_{im}$↑	CLIP$_{out}$↑	L1↓	DINO↑
InstructPix2Pix [2]	0.078	0.834	0.219	0.121	0.762
MagicBrush [21]	0.090	0.838	0.222	0.100	0.776
PnP [20]	0.028	0.521	0.089	0.304	0.153
Null-Text Inv. [15]	0.101	0.761	0.236	**0.075**	0.678
Emu Edit [18]	0.109	0.859	0.231	0.094	0.819
Ours	**0.115**	**0.867**	**0.244**	0.083	**0.834**

6.3. Evaluation on MagicBrush Test

In this section, we present the full quantitative analysis on the MagicBrush test set, including results from both global description-guided and instruction-guided models, as shown in Tab. 3. While our method, UIP2P, is not fine-tuned on human-annotated datasets like MagicBrush, it still achieves highly competitive results compared to models specifically fine-tuned for the task. In particular, UIP2P demonstrates either the best or second-best performance in key metrics such as L1, L2, and CLIP-I, even outperforming fine-tuned models in several cases. This highlights the robustness and generalization capabilities of UIP2P, showing that it can effectively handle complex edits without the need for specialized training on real datasets. These results further validate that UIP2P delivers high-quality edits in a variety of contexts, maintaining competitive performance against fine-tuned models on the MagicBrush dataset, which is human-annotated.

Table 3. **Quantitative comparison on MagicBrush [21] test set.** In the multi-turn setting, target images are iteratively edited from the initial source images. Best results are in **bold**.

Settings	**Methods**	L1↓	L2↓	CLIP-I↑	DINO↑	CLIP-T↑
Single-turn	***Global Description-guided***					
	Open-Edit [13]	0.1430	0.0431	0.8381	0.7632	0.2610
	VQGAN-CLIP [4]	0.2200	0.0833	0.6751	0.4946	**0.3879**
	SD-SDEdit [14]	0.1014	0.0278	0.8526	0.7726	0.2777
	Text2LIVE [1]	0.0636	**0.0169**	0.9244	0.8807	0.2424
	Null Text Inversion [15]	0.0749	0.0197	0.8827	0.8206	0.2737
	Instruction-guided					
	HIVE [22]	0.1092	0.0341	0.8519	0.7500	0.2752
	w/ MagicBrush [21]	0.0658	0.0224	0.9189	0.8655	0.2812
	InstructPix2Pix [2]	0.1122	0.0371	0.8524	0.7428	0.2764
	w/ MagicBrush [21]	0.0625	0.0203	**0.9332**	0.8987	0.2781
	UIP2P w/ IP2P Dataset	0.0722	0.0193	0.9243	0.8876	0.2944
	UIP2P w/ CC3M Dataset	0.0680	0.0183	0.9262	0.8924	0.2966
	UIP2P w/ CC12M Dataset	**0.0619**	0.0174	0.9318	**0.9039**	0.2964
Multi-turn	***Global Description-guided***					
	Open-Edit [13]	0.1655	0.0550	0.8038	0.6835	0.2527
	VQGAN-CLIP [4]	0.2471	0.1025	0.6606	0.4592	**0.3845**
	SD-SDEdit [14]	0.1616	0.0602	0.7933	0.6212	0.2694
	Text2LIVE [1]	0.0989	**0.0284**	0.8795	0.7926	0.2716
	Null Text Inversion [15]	0.1057	0.0335	0.8468	0.7529	0.2710
	Instruction-guided					
	HIVE [22]	0.1521	0.0557	0.8004	0.6463	0.2673
	w/ MagicBrush [21]	0.0966	0.0365	0.8785	0.7891	0.2796
	InstructPix2Pix [2]	0.1584	0.0598	0.7924	0.6177	0.2726
	w/ MagicBrush [21]	**0.0964**	0.0353	**0.8924**	**0.8273**	0.2754
	UIP2P w/ IP2P Dataset	0.1104	0.0358	0.8779	0.8041	0.2892
	UIP2P w/ CC3M Dataset	0.1040	0.0337	0.8816	0.8130	0.2909
	UIP2P w/ CC12M Dataset	0.0976	0.0323	0.8857	0.8235	0.2901

7. Edit Reversibility Constraint Example

We demonstrate ERC with a visual example during inference. In the forward pass, the model transforms the input image based on the instruction (*e.g.*, “turn the forest path into a beach”). In the reverse pass, the corresponding reverse instruction (*e.g.*, “turn the beach back into a forest”) is applied, reconstructing the original image. This showcases the model’s ability to maintain consistency and accuracy across complex edits, ensuring that both the forward and reverse transformations align coherently. Additional examples, such as adding and removing objects, further emphasize UIP2P’s adaptability in diverse editing tasks. Figure 4 illustrates how our method ensures precise, reversible edits while maintaining the integrity of the original content.

Table 4. **Examples of Four Possible Edits for Two Different Input Captions.** Our dataset generation process showcases the flexibility of the reverse instruction dataset by demonstrating multiple transformations for the same caption.

Input Caption	Edit Instruction	Edited Caption	Reverse Instruction
A dog sitting on a couch	change the dog's color to brown	A brown dog sitting on a couch	change the dog's color back to white
	add a ball next to the dog	A dog sitting on a couch with a ball	remove the ball
	remove the dog	An empty couch	add the dog back
	move the dog to the floor	A dog sitting on the floor	move the dog back to the couch
A car parked on the street	change the car color to red	A red car parked on the street	change the car color back to black
	add a bicycle next to the car	A car parked on the street with a bicycle	remove the bicycle
	remove the car	An empty street	add the car back
	move the car to the garage	A car parked in the garage	move the car back to the street

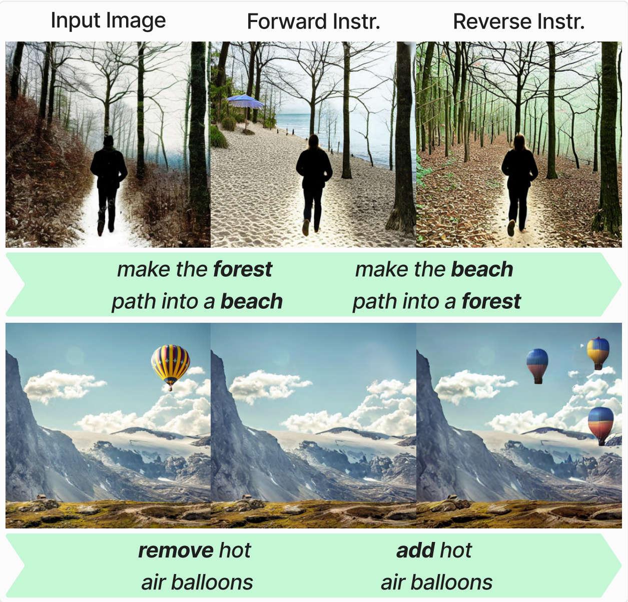

Figure 4. Forward and reverse edits are applied sequentially.

8. Dataset Details

8.1. LLM Prompts for Data Generation

To ensure reproducibility, we provide the exact prompts used with LLMs for generating reverse instructions and expanding datasets. These prompts were crucial for creating the training data without requiring ground-truth edited images.

8.1.1. LLM Prompt for Reverse Instruction Generation (IP2P Dataset)

```
Below is an instruction that describes a
task, paired with an input that provides
further context. Write a response that
appropriately completes the request.

### Instruction:
You are an expert in image editing
instructions. Given an input caption, an
edit instruction, and the resulting edited
caption, generate a reverse instruction
that would undo the edit and return to
the original state.

### Input:
Input Caption: "{input_caption}"
Edit Instruction: "{edit_instruction}"
Edited Caption: "{edited_caption}"

Examples:
- Input: "A man wearing a denim jacket"
  -> "make the jacket a rain coat"
  -> "A man wearing a rain coat"
  Reverse: "make the coat a denim jacket"
- Input: "A sofa in the living room"
  -> "add pillows" -> "A sofa in the
  living room with pillows"
  Reverse: "remove the pillows"

### Response:
{reverse_instruction}
```

8.1.2. LLM Prompt for Multi-Edit Generation (CC3M/CC12M)

```
Below is an instruction that describes
a task, paired with an input that
provides further context. Write a
response that appropriately completes
the request.

### Instruction:
Given an input caption, generate 4
different edit instructions along with
their corresponding edited captions and
reverse instructions. Focus on diverse
edit types: color changes, object
addition/removal, and positional
changes. Ensure edits are realistic
and reversible.

### Input:
Input Caption: "{input_caption}"

### Response:
1. Edit Instruction: [instruction]
   Edited Caption: [result]
   Reverse Instruction: [undo instruction]

2. Edit Instruction: [instruction]
   Edited Caption: [result]
   Reverse Instruction: [undo instruction]

3. Edit Instruction: [instruction]
   Edited Caption: [result]
   Reverse Instruction: [undo instruction]

4. Edit Instruction: [instruction]
   Edited Caption: [result]
   Reverse Instruction: [undo instruction]
```

8.2. Dataset Filtering

We apply CLIP [16] to both the CC3M [17] and CC12M [3] datasets to calculate the similarity between captions and images, ensuring that the text descriptions accurately reflect the content of the corresponding images. Following the methodology used in InstructPix2Pix (IP2P) [2], we adopt a CLIP-based filtering strategy with a similarity threshold set at 0.2. This threshold filters out image-caption pairs that do not have sufficient semantic alignment, allowing us to curate a dataset with higher-quality text-image pairs. For the filtering process, we utilize the CLIP ViT-L/14 model, which provides a robust and well-established framework for capturing semantic similarity across text and images.

By applying this filtering process, we ensure that only relevant and coherent pairs remain in the dataset, improving the quality of training data and helping the model better generalize to real-world editing tasks. As a result, the filtered CC3M dataset contains 2.5 million image-caption pairs, while the filtered CC12M dataset contains 8.5 million pairs. This careful curation of the dataset enhances the reliability of the training process without relying on human annotations, making it scalable for broader real-image datasets without the cost and limitations of human-annotated ground-truth datasets [2, 21].

8.3. More Examples from Reverse Instructions Dataset

To demonstrate the versatility of our reverse instruction dataset, we provide examples with multiple variations of edits for two different input captions. Each caption has four distinct edits, such as color changes, object additions, object removals, and positional adjustments. This variety helps the model generalize across a wide range of tasks and scenarios, as discussed in Sec. 4.2, main paper. The use of large language models (LLMs) to generate reverse instructions further enhances the flexibility of our dataset.

These examples, along with others in Tab. 1, illustrate the diversity of edit types our model learns, enabling it to perform a wide range of tasks across different real-image datasets. The reverse instruction mechanism ensures that the edits are reversible, maintaining consistency and coherence in both the forward and reverse transformations.

9. Additional Implementation Details

9.1. Details of Competitor Methods

Our method offers significant advantages over competitors in both training and inference. Unlike supervised methods that rely on paired triplets of input images, edited images, and instructions, our approach eliminates the need for such datasets, reducing biases and improving scalability. For example, MagicBrush is fine-tuned on a human-annotated dataset, while HIVE leverages Prompt-to-Prompt editing with human annotators, introducing dependency on labor-intensive processes. Furthermore, MGIE and SmartEdit rely on LLMs during inference, which significantly increases computational overhead. These distinctions highlight the efficiency and practicality of our approach, as it avoids the need for expensive human annotations and additional inference-time complexities. Like other editing methods, our approach can produce small variations for different random seeds but consistently applies the specified edit, eliminating the need for manual selection. To the best of our knowledge, the compared methods, *e.g.*, MagicBrush, InstructPix2Pix or other methods, also do not involve manual selection.

InstructPix2Pix [2] is a diffusion-based model that performs instruction-based image editing by training on triplets of input image, instruction, and edited image[1]. The model is

[1] https://github.com/timothybrooks/instruct-pix2pix

fine-tuned on a synthetic dataset of edited images generated by combining large language models (LLMs) and Prompt-to-Prompt [7]. This approach relies on paired datasets, which can introduce biases and limit generalization. InstructPix2Pix serves as one of the key baselines for our comparison, given its supervised training methodology.

HIVE [22] is an instruction-based editing model that fine-tunes InstructPix2Pix based on human feedback[2]. Specifically, HIVE learns from user preferences about which edited images are preferred, incorporating this feedback into the model training. While this approach allows HIVE to better align with human expectations, it still builds on top of InstructPix2Pix and does not start training from scratch. This limits its flexibility compared to methods like UIP2P, which are trained from the ground up.

MagicBrush [21] fine-tunes the pre-trained weights of InstructPix2Pix on a human-annotated dataset to improve real-image editing performance[3]. While this fine-tuning approach makes MagicBrush highly effective for specific tasks with ground-truth labels, it limits its generalizability compared to methods like UIP2P, which are trained from scratch. Moreover, MagicBrush's reliance on human-annotated data introduces significant scalability challenges, as obtaining such annotations is both costly and labor-intensive. This dependency makes it less suited for broader datasets where large-scale annotations may not be feasible.

MGIE [5] introduces a large multimodal language model to generate more precise instructions for image editing[4]. Like InstructPix2Pix, MGIE requires a paired dataset for training but uses the language model to improve the quality of the instructions during inference. However, this reliance on LLMs during inference adds computational overhead. In contrast, UIP2P operates without LLMs at inference time, reducing overhead while maintaining flexibility.

SmartEdit [8] is based on InstructDiffusion, a model already trained for instruction-based image editing tasks[5]. It introduces a bidirectional interaction module to improve text-image alignment, but its reliance on the pre-trained InstructDiffusion limits flexibility, as SmartEdit does not start training from scratch. Additionally, SmartEdit depends on large language models (LLMs) during inference, increasing computational overhead. This makes SmartEdit less efficient than UIP2P in scenarios where real-time or large-scale processing is required.

DiffusionCLIP [11] leverages pre-trained diffusion models for text-driven image manipulation by fine-tuning the reverse diffusion process with a CLIP-based loss[6]. Unlike UIP2P, which enforces edit reversibility constraints for unsupervised training, DiffusionCLIP relies on fine-tuning for each new target attribute, making it less scalable for large-scale instruction-based editing. Additionally, its approach requires per-attribute model tuning and inversion of the input image before the editing process, leading to increased training and inference overhead compared to UIP2P, which generalizes across a diverse set of edits without explicit supervision and additional inversion process.

During evaluation, we use the publicly available implementations and demo pages of the baseline methods. Each baseline provides a different approach to instruction-based image editing, and together they offer a comprehensive set of methods for comparing the performance, flexibility, and efficiency of the proposed method, UIP2P.

9.2. Code Implementation Overview

Our UIP2P implementation with ERC builds on existing frameworks for reproducibility:

- **Base Framework:** The code is based on InstructPix2Pix[7], which provides the foundation for instruction-based image editing.
- **Adopted CLIP Losses:** We adopted and modified CLIP-based loss functions from StyleGAN-NADA[8] to fit ERC, improving image-text alignment for our specific tasks.

9.3. Algorithm Overview

In this section, we explain the proposed method, UIP2P, which introduces unsupervised learning for instruction-based image editing. The core of our approach is the Edit Reversibility Constraint (ERC), which ensures that edits are coherent and reversible when applied sequentially in both forward and reverse instructions.

The algorithm consists of two key processes:

- **Forward Process:** Starting with an input image and a forward edit instruction, noise is first added to the image. The model then predicts the noise, which is applied to reverse the noise process and recover the edited image (*see Algorithm 1, lines 2-4*).
- **Reverse Process:** Given the forward-edited image and a reverse edit instruction, noise is applied again. The model predicts the reverse noise, which is used to undo the edits and reconstruct the original image. This ensures that the reverse edits are consistent with the original input image (*see Algorithm 1, lines 6-8*).

ERC is applied between the original input image, the forward-edited image, and the reconstructed image, along with their respective attention maps and captions (*see Algorithm 1, line 10*). The $\mathcal{L}_{ERC}$ function guides the model's learning through backpropagation (*see Algorithm 1, lines 12-13*).

[2] https://github.com/salesforce/HIVE

[3] https://github.com/OSU-NLP-Group/MagicBrush

[4] https://ml-mgie.com/playground.html

[5] https://github.com/TencentARC/SmartEdit

[6] https://github.com/gwang-kim/DiffusionCLIP.git

[7] https://github.com/timothybrooks/instruct-pix2pix

[8] https://github.com/rinongal/StyleGAN-nada

Algorithm 1 Unsupervised Instruction-Based Image Editing (UIP2P) with ERC

```
Require: Image $I_i$ (input image), Forward edit instruction $F$, Reverse edit instruction $R$, Noise levels $t$ (forward), $\hat{t}$ (reverse), Model $M$, Loss function $L_{ERC}$, Noise function $N$, Input caption $T_i$, Edited caption $T_e$
Ensure: Edited image $I_e$, Reconstructed image $I_r$

Forward Process:
    $z_t \leftarrow N(I_i, t)$    ▷ Add noise $t$ to the input image $I_i$
    $\hat{\epsilon}_F, A_f \leftarrow M(z_t | I_i, F)$    ▷ Model $M$ predicts forward noise $\hat{\epsilon}_F$ and extracts attention map $A_f$
    $I_e \leftarrow$ Apply$(\hat{\epsilon}_F, z_t, t)$    ▷ Apply predicted noise $\hat{\epsilon}_F$ to reverse the process of obtaining $z_t$ and recover $I_e$

Reverse Process:
    $z_{\hat{t}} \leftarrow N(I_e, \hat{t})$    ▷ Add noise $\hat{t}$ to the forward-edited image $I_e$
    $\hat{\epsilon}_R, A_r \leftarrow M(z_{\hat{t}} | I_e, R)$    ▷ Model $M$ predicts reverse noise $\hat{\epsilon}_R$ and extracts attention map $A_r$
    $I_r \leftarrow$ Apply$(\hat{\epsilon}_R, z_{\hat{t}}, \hat{t})$    ▷ Apply predicted noise $\hat{\epsilon}_R$ to reverse the process of obtaining $z_{\hat{t}}$ and recover $I_r$

Edit Reversibility Constraint Loss:
    $L_{ERC} \leftarrow L(I_i, I_e, I_r, A_f, A_r, T_i, T_e)$    ▷ Compute ERC loss using $I_i$, $I_e$, $I_r$, attention maps $A_f$, $A_r$, input text $T_i$, and edited text $T_e$

Update Model:
    Backpropagate the loss $L_{ERC}$ and update the model $M$
    Repeat until convergence
```

10. Limitations and Failure Cases

While our method demonstrates strong performance across various editing tasks, we acknowledge several limitations. Our reliance on CLIP for semantic alignment introduces challenges in fine-grained spatial reasoning, object counting, and complex compositional understanding. The method may struggle with instructions requiring precise spatial relationships (*e.g.*, “add three apples to the left of the table”) or complex multi-object scenarios with occlusions. Additionally, like many diffusion-based methods, our approach has difficulties with text rendering, extreme lighting changes, and cases requiring simultaneous style transfer with significant structural modifications.

Our method’s performance is also constrained by training data quality and the reverse instruction generation process using LLMs. While more efficient than LLM-based approaches during inference, the training process requires additional computational overhead for generating reverse instructions and computing attention consistency losses. Future work could address these limitations by integrating stronger multimodal models for better spatial understanding, incorporating perceptual losses for improved photorealism, and developing more sophisticated evaluation frameworks beyond CLIP-based metrics.

References

[1] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XV*, pages 707–723. Springer, 2022. 4

[2] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 4, 6

[3] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pretraining to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3558–3568, 2021. 6

[4] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. VQGAN-CLIP: open domain image generation and editing with natural language guidance. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXVII*, pages 88–105. Springer, 2022. 4

[5] Tsu-Jui Fu, Wenze Hu, Xianzhi Du, William Yang Wang, Yinfei Yang, and Zhe Gan. Guiding instruction-based image editing via multimodal large language models. *arXiv preprint arXiv:2309.17102*, 2023. 1, 7

[6] 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

[7] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. *CoRR*, abs/2208.01626, 2022. 7

[8] Yuzhou Huang, Liangbin Xie, Xintao Wang, Ziyang Yuan, Xiaodong Cun, Yixiao Ge, Jiantao Zhou, Chao Dong, Rui Huang, Ruimao Zhang, et al. Smartedit: Exploring complex instruction-based image editing with multimodal large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8362–8371, 2024. 1, 7

[9] 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. 3

[10] Krishnaram Kenthapadi, Himabindu Lakkaraju, and Nazneen Rajani. Generative ai meets responsible ai: Practical challenges and opportunities. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5805–5806, 2023. 1

[11] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2426–2435, 2022. 7

[12] Pavel Korshunov and Sébastien Marcel. Deepfakes: a new threat to face recognition? assessment and detection. *arXiv preprint arXiv:1812.08685*, 2018. 1

[13] Xihui Liu, Zhe Lin, Jianming Zhang, Handong Zhao, Quan Tran, Xiaogang Wang, and Hongsheng Li. Open-edit: Open-domain image manipulation with open-vocabulary instructions. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XI*, pages 89–106. Springer, 2020. 4

[14] 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. In *International Conference on Learning Representations*, 2022. 4

[15] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. *CoRR*, abs/2211.09794, 2022. 4

[16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning, 2021*, pages 8748–8763. PMLR, 2021. 6

[17] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018. 6

[18] Shelly Sheynin, Adam Polyak, Uriel Singer, Yuval Kirstain, Amit Zohar, Oron Ashual, Devi Parikh, and Yaniv Taigman. Emu edit: Precise image editing via recognition and generation tasks. *arXiv preprint arXiv:2311.10089*, 2023. 4

[19] Enis Simsar, Alessio Tonioni, Yongqin Xian, Thomas Hofmann, and Federico Tombari. Lime: Localized image editing via attention regularization in diffusion models. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, pages 222–231, 2025. 2

[20] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2023. 4

[21] 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. 4, 6, 7

[22] Shu Zhang, Xinyi Yang, Yihao Feng, Can Qin, Chia-Chih Chen, Ning Yu, Zeyuan Chen, Huan Wang, Silvio Savarese, Stefano Ermon, Caiming Xiong, and Ran Xu. HIVE: harnessing human feedback for instructional visual editing. *CoRR*, abs/2303.09618, 2023. 4, 7