InstructEdit: Improving Automatic Masks for Diffusion-based Image Editing With User Instructions

Anonymous authors
Paper under double-blind review

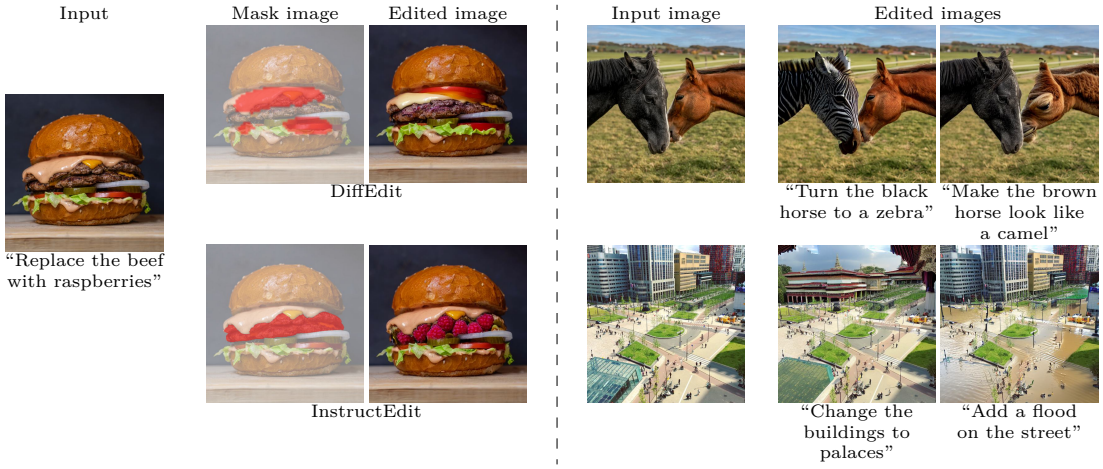

Figure 1: Left: a comparison between DiffEdit and InstructEdit. Right: examples of editing using InstructEdit. Note that InstructEdit only requires user instructions as input, while DiffEdit needs an input caption and an edited caption instead.

Abstract

Recent works have explored text-guided image editing using diffusion models and generated edited images based on text prompts. However, the models struggle to accurately locate the regions to be edited and faithfully perform precise edits. In this work, we propose a framework termed **InstructEdit** that can do fine-grained editing based on user instructions. Our proposed framework has three components: language processor, segmenter, and image editor. The first component, the language processor, processes the user instruction using a large language model. The goal of this processing is to parse the user instruction and output prompts for the segmenter and captions for the image editor. We adopt ChatGPT and optionally BLIP2 for this step. The second component, the segmenter, uses the segmentation prompt provided by the language processor. We employ a state-of-the-art segmentation framework Grounded Segment Anything to automatically generate a high-quality mask based on the segmentation prompt. The third component, the image editor, uses the captions from the language processor and the masks from the segmenter to compute the edited image. We adopt Stable Diffusion and the mask-guided generation from DiffEdit for this purpose. Experiments show that our method outperforms previous editing methods in fine-grained editing applications where the input image contains a complex object or multiple objects. We improve the mask quality over DiffEdit and thus improve the quality of edited images. We also show that our framework can be combined with the NeRF or video editing pipeline to achieve fine-grained scale NeRF or video editing application.

1 Introduction

Generative diffusion models are a versatile tool to generate images Saharia et al. (2022); Ramesh et al. (2022); Balaji et al. (2022); Rombach et al. (2022); Li et al. (2023b), videos Ho et al. (2022b); Esser et al. (2023); Ho et al. (2022a); Blattmann et al. (2023); Yang et al. (2022), and 3D shapes Hui et al. (2022); Zhang et al. (2023); Gupta et al. (2023); Zheng et al. (2023). In addition, the powerful representation learned by generative diffusion models makes them a great basis for downstream editing operations. In this paper, we are particularly interested in image editing operations.

Training-free and tuning-free text-guided image editing methods using diffusion models Hertz et al. (2022); Couairon et al. (2022); Wang et al. (2023b); Liew et al. (2022) usually rely on descriptive text captions for both input image and edited image. However, one line of work focuses on accepting human instructions as input to edit images, as human instructions are more intuitive for a user to provide, and can be free from specific prompting structures. InstructPix2Pix Brooks et al. (2022) constructs an "input caption, edited caption, user instruction" triplet dataset by fine-tuning GPT-3 Brown et al. (2020) and generates pairs of input image and corresponding edited image using Prompt-to-Prompt Hertz et al. (2022). Here we also try to accept human instructions as input. Instead of fine-tuning a large language model, we utilize its in-context learning ability to parse user instructions on the fly.

Using a mask to provide guidance to the models about the specific areas to edit is a logical and intuitive approach. There already exist several editing models Brooks et al. (2022); Hertz et al. (2022); Liew et al. (2022); Wang et al. (2023b) that do not require a mask as input. This approach generally works well when there is a single object in the image or if there is no object of the same type as the object the user wants to edit. However, with multiple objects in the input image, diffusion models struggle to correctly identify which object the user intends to edit. For example, if the user wants to edit a specific object like "A yellow chair" in an image with multiple chairs present or "The cat on the left" in a group of cats. In such cases, an input mask can help the model to correctly locate the object of interest.

While it is possible to ask a user to paint a mask, it requires time-consuming and detailed user interaction. We therefore follow the previous work DiffEdit Couairon et al. (2022) that employs automatically generated masks. While DiffEdit obtains satisfactory masks in many cases, there are several instances where DiffEdit fails to produce high-quality masks: 1) the descriptive captions are not informative enough; 2) the threshold of the mask filtering is not correctly set; 3) there is more than one object of the same or different type; and 4) there are many parts of one object in the image that may cause ambiguity. In our work, we set out to tackle the challenge of computing improved masks in such challenging cases. We adopt a powerful grounding segmentation module called Grounded Segment Anything (Grounded SAM) Liu et al. (2023); Kirillov et al. (2023) to automatically generate masks.

In this work, we propose a framework called **InstructEdit** to use large pre-trained models to edit images following user instructions based on DiffEdit. We automatically extract higher-quality masks compared to DiffEdit, and therefore achieve more preferable and stable editing results in more complex multi-object image editing scenarios. Specifically, our method has three components: a language processor, a segmenter, and an image editor. We first use the language processor to understand the user instruction by identifying which object(s) should be edited and how. This first step yields the segmentation prompt for the segmenter and captions for the image editor. We use ChatGPT to parse the user instruction and optionally adopt BLIP2 Li et al. (2023a) when the user instruction is unclear. The segmenter then accepts the segmentation prompt and generates a mask that outlines the region according to the segmentation prompt. We utilize Grounded Segment Anything as the segmenter which combines both high grounding and segmentation abilities. Finally, the image editor performs image editing using the captions along with the generated mask. We adopt Stable Diffusion Rombach et al. (2022) along with the mask-guided image editing process to do the editing.

We show that our method outperforms previous editing methods in fine-grained scale object-level editing applications. We summarize our contributions as below:

- We propose a diffusion-based text-guided image editing framework that accepts user instruction as input instead of input caption and edited caption.

- We outperform baseline methods in fine-grained object-level editing.
- We improve the mask quality over the original DiffEdit and thus improve the image quality.
- Our framework can be combined with the NeRF or video editing pipeline to achieve fine-grained scale NeRF or video editing application.

2 Related work

2.1 Image editing using diffusion models

Pre-trained diffusion models Ramesh et al. (2022); Saharia et al. (2022); Rombach et al. (2022); Huang et al. (2023a) can be used to do various image editing tasks. Several works Valevski et al. (2022); Kawar et al. (2022); Kwon & Ye (2022) fine-tune the diffusion model weights or optimize a loss function to perform image editing. However, in these works fine-tuning requires a relatively long time to obtain a single edited image, and each editing prompt requires its own fine-tuning process.

Many works Meng et al. (2021); Hertz et al. (2022); Parmar et al. (2023); Wang et al. (2023b) achieve remarkable image editing results using a tuning-free approach. Prompt-to-Prompt Hertz et al. (2022) proposes to edit the cross-attention maps by comparing the input caption and the edited caption. MDP Wang et al. (2023b) proposes to manipulate the diffusion path by analyzing the sampling formula. In this line of work, a mask is not required during the editing process, thereby making it easier to use than systems that rely on masking. These kinds of methods usually perform well when there is a single foreground object in the image, but fail when more fine-grained control is needed.

By providing a manually designed mask as input, a user can explicitly control which region to edit and which region to preserve. Blended Diffusion Avrahami et al. (2022b) and Blended Latent Diffusion Avrahami et al. (2022a) utilize a mask to perform text-guided image editing by operating either in pixel space or in latent space. Shape-guided Diffusion Park et al. (2022) proposes to use a mask with inside-outside attention to preserve the shape of the object to be edited. MasaCtrl Cao et al. (2023) focuses on performing non-rigid editing while using a mask to alleviate the confusion of foreground with background objects.

Although a mask provides additional control in the editing process, generating a manual mask can be a burden for the user if the mask is not automatically estimated. Park et al. (2022); Cao et al. (2023) can also replace the manual mask with an automatic mask generated by cross-attention maps. Another prominent work DiffEdit Couairon et al. (2022) proposes a way to automatically predict a mask by contrasting the predicted noise conditioned on the input caption and the edited caption. In this work, we intend to exploit the benefit of using a mask without bringing the burden of manually labeling a mask to the user. We adopt large pre-trained models to help us automatically infer a mask.

2.2 Foundational models

Large language models. The development of large language models has been a rapidly evolving field in recent years Radford et al. (2018; 2019); Brown et al. (2020); Devlin et al. (2019); Raffel et al. (2020); Clark et al. (2020). One important contribution is the GPT series of models Radford et al. (2018; 2019); Brown et al. (2020). These models are pre-trained on massive amounts of text data and then fine-tuned for specific tasks.

Most notably, ChatGPT OpenAI (2023) is a variant of the GPT series of models that are designed specifically for generating human-like responses in conversational settings. We make use of ChatGPT in our work for the purpose of information extraction from user instruction.

Segmentation model and grounding detector. Segment Anything Kirillov et al. (2023) is a segmentation model that uses a combination of different input prompts and enables zero-shot generalization to unfamiliar objects and images without requiring additional training. Grounding DINO Liu et al. (2023) is an open-set object detector that combines the Transformer-based detector DINO Zhang et al. (2022) with grounded pre-training to detect arbitrary objects with human inputs such as category names or referential expressions. Their system outputs multiple pairs of object boxes and noun phrases give a prompt.

Images and language. Vision language models (VLMs) Radford et al. (2021); Alayrac et al. (2022); Mañas et al. (2022); Huang et al. (2023b); Li et al. (2023a) are a powerful class of models that combine computer vision and natural language processing. VLMs have gained significant attention in recent years due to their ability to bridge the gap between visual and textual information, enabling a range of applications such as image captioning, visual question answering, and image retrieval. Especially, BLIP-2 Li et al. (2023a) unlocks the capability of zero-shot instructed image-to-text generation. Given an input image, BLIP-2 can generate various natural language responses according to the user's instruction.

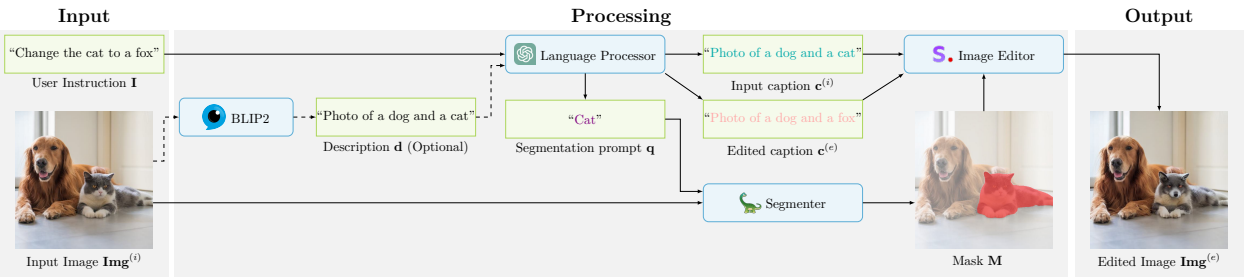

Figure 2: Pipeline: given a user instruction, a language processor first parses the instruction into a segmentation prompt, an input caption, and an edited caption. A segmenter then generates a mask based on the segmentation prompt. The mask along with the input and edited captions are then going to an image editor to produce the final output.

3 Method

In this work, in order to edit images in a fine-grained scale localized way, we propose a framework called **InstructEdit** to use large pre-trained models to edit images following user instructions. We show our overall pipeline in Fig. 2. Specifically, we design three components to perform this task: a language processor, a segmenter, and an image editor. We introduce those components in the following sections.

3.1 Preliminaries

During the training process of diffusion models, we have the objective function

$$\min_{\theta} \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon},t} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}, t) \right\|^2, \tag{1}$$

where $\mathbf{x}_0$ is the input image, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathcal{I})$ is Gaussian noise that is added to the input image and $\boldsymbol{\epsilon}_\theta$ is the noise estimator that is used to predict the added noise. t is the denoising timestep while $\mathbf{c}$ is the condition of the diffusion model. In this work, we only consider $\mathbf{c}$ to be a text prompt of a text-guided diffusion model. After training, the noise estimator $\boldsymbol{\epsilon}_\theta$ can be used to generate new samples. We use DDIM (Denoising Diffusion Implicit Models) Song et al. (2021), which is a deterministic sampler with denoising steps

$$\mathbf{x}_{t-1} = \sqrt{\alpha_{t-1}} \cdot f_\theta(\mathbf{x}_t, \mathbf{c}, t) + \sqrt{1-\alpha_{t-1}} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}, t), \tag{2}$$

where $f_\theta(\mathbf{x}_t, \mathbf{c}, t) = \frac{\mathbf{x}_t - \sqrt{1-\alpha_t} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}, t)}{\sqrt{\alpha_t}}$, and α_t is the noise schedule factor in DDIM. We denote $\boldsymbol{\epsilon}_t = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}, t)$. Given an input image, we can use DDIM inversion to invert it into an initial noise tensor $\mathbf{x}_T$. Each inversion step is calculated as

$$\mathbf{x}_{t+1} = \sqrt{\alpha_{t+1}} \cdot f_\theta(\mathbf{x}_t, \mathbf{c}, t) + \sqrt{1-\alpha_{t+1}} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}, t). \tag{3}$$

We iteratively apply this formula until obtaining $\mathbf{x}_T$. However, if we stop the inversion step at timestep $r \leq T$, we encode $\mathbf{x}_0$ into a less noised version $\mathbf{x}_r$. r is called the encoding ratio as in DiffEdit. A larger value r indicates a stronger editing effect, making the edited image guided more by the edited caption but look less like the input image.

3.2 Language processor

Given a user instruction $\mathbf{I}$ as input, we use a large language model ChatGPT OpenAI (2023) to extract segmentation prompt $\mathbf{q}$ for the segmenter, and an input caption $\mathbf{c}^{(i)}$ and an edited caption $\mathbf{c}^{(e)}$ for the image editor. Here, we utilize the in-context learning ability of large language models Dong et al. (2023) to achieve zero-shot task learning. In-context learning does not require any tuning of the parameters of the language model. Instead, it learns the pattern from the task examples and makes predictions when it sees a new example. In this work, by giving a few examples, ChatGPT is able to learn the task and follow the instructions. We show one task example in the Supplementary Materials. A task example shows how ChatGPT should manipulate a user instruction that is provided as input.

When the user instruction $\mathbf{I}$ or the description of the object to be edited is unclear, it is difficult for ChatGPT to correctly provide the prompts as it has no access to the content of the input image. The vision-language model BLIP2 can process the image and is able to answer questions about its content. BLIP2 can provide a short description of the original image, which can assist ChatGPT in providing prompts for the segmenter and captions for the image editor.

In this work, we optionally query BLIP2 to obtain a description $\mathbf{d}$ of the image. Given an input image, we first ask BLIP2 *"Is this a photo, a painting, or another kind of art?"*. We denote the answer as ρ and reuse it in another query to BLIP2 composed as *"ρ of"* to obtain a completed sentence describing the image as input prompt to ChatGPT. ChatGPT in turn can refine the prompt by identifying which object to edit and provide more details even when the user instruction does not specify the content to be edited or the description is incomplete to unambiguously refer to the intended objects in the image.

3.3 Segmenter

We use Grounded Segment Anything as our segmenter to locate the object(s) to be edited and to compute a corresponding mask. Grounded Segment Anything is a framework which combines Grounding DINO Liu et al. (2023) and Segment Anything Kirillov et al. (2023). Grounding DINO is an open-set object detector, which can accept a given text and output one or multiple detected bounding boxes and a text similarity score per bounding box. Segment Anything (SAM) is a powerful segmentation model. It can accept the bounding box output by Grounding DINO and produce high-quality binary masks for the downstream tasks.

Grounding DINO is first applied to get a bounding box for a given segmentation prompt $\mathbf{q}$ by

$$\text{DINO}(\mathbf{x}_0, \mathbf{q}) = \mathbf{b} = [h, w, \Delta h, \Delta w],$$

where $[h, w]$ is the top-left corner coordinate of the detected bounding box in pixel space, and $[\Delta h, \Delta w]$ is the size of the bounding box. Then, the bounding box is refined to a per-pixel binary mask $\mathbf{M}$ by

$$\text{SAM}(\mathbf{x}_0, \mathbf{b}) = \mathbf{M}.$$

3.4 Image editor

We adopt the mask-guided image editing as in DiffEdit Couairon et al. (2022). Given an input image $\mathbf{Img}^{(i)}$ (which is also denoted as $\mathbf{x}_0$ in Section 3.1), we want to edit it to get the edited image $\mathbf{Img}^{(e)}$. With the automatically generated mask $\mathbf{M}$ and an encoded noise $\mathbf{x}_r$, the mask-guided DDIM denoising step is formulated as

$$\widetilde{\mathbf{y}}_t = \mathbf{M}\mathbf{y}_t + (1 - \mathbf{M})\mathbf{x}_t, \tag{4}$$

where $\mathbf{y}_t = \begin{cases} \mathbf{x}_r & \text{if } t = r, \\ \boldsymbol{\epsilon}_\theta(\mathbf{y}_{t-1}, \mathbf{c}, t) & \text{otherwise.} \end{cases}$ The region within the mask will have the changes guided by the edited caption, while the region outside the mask will be mapped back to the original pixels. We obtain $\mathbf{Img}^{(e)} = \widetilde{\mathbf{y}}_0$ after iteratively applying Eq. (4).

Table 1: Quantitative comparisons between our method and baselines.

	MDP-$\boldsymbol{\epsilon}_t$	InstructPix2Pix	DiffEdit	**InstructEdit**
BRISQUE ↓	**22.658**	25.054	23.607	23.104
LPIPS ↓	0.260	0.269	0.152	**0.132**
CLIP score ↑	25.613	25.681	25.707	**26.584**
CLIP directional similarity ↑	0.095	0.125	0.103	**0.153**

4 Experiments

4.1 Settings

Baselines. We choose three text-guided image editing methods that do not require a manual mask as input as our baselines. MDP-$\boldsymbol{\epsilon}_t$ Wang et al. (2023b) is a mask-free editing method that manipulates the diffusion paths by mixing the predicted noise guided by the input caption and the edited caption according to a defined mixing schedule. InstructPix2Pix Brooks et al. (2022) is also a mask-free editing method. It trains a conditional diffusion model that accepts both an image and a user instruction as input to edit the input image. It highlights the user instruction as input without edited captions by exploiting a large language model. DiffEdit Couairon et al. (2022) is a mask-based editing method that produces an automatically computed mask by subtracting the predicted noises guided by the input caption and the edited caption. InstructEdit adopts the same mask guidance as in DiffEdit to generate the edited image. However, InstructEdit uses a pre-trained segmentation model to automatically compute a new mask.

All the experiments are performed on a single NVIDIA A100. Each edit takes around 20 seconds. The language processor takes around 2 seconds, the segmenter takes around 8 seconds and the image editor takes around 10 seconds. Additionally enabling BLIP2 will cost 0.3 second. Compared to the baseline methods, our computation overhead is mainly caused by the querying time of the language processor and the segmenter. However, our method removes the burden of manual prompting from the user (necessary in DiffEdit and MDP-ϵ_t), which reduces the potential manual labeling time especially when there are many samples to edit.We use Stable Diffusion v1.5 as the backbone of the image editor. We use the model's weights and implementation of Grounded Segment Anything from [1]. More details can be found in the Supplementary Materials.

Evaluation. We provide both qualitative and quantitative results for our method. For quantitative metrics, we use BRISQUE Mittal et al. (2012) as a non-reference metric to measure the image quality, LPIPS Zhang et al. (2018) to measure the similarity between the input image and the edited image, CLIP score Hessel et al. (2022) to measure the instruction-image compatibility and CLIP directional similarity Gal et al. (2021) to see if the change in images is consistent with the change in captions. As quantitative metrics alone usually cannot align with human judgment, we additionally conduct a user study for a better evaluation.

4.2 Baseline comparisons

We provide selected qualitative comparisons between our method and other baselines in Fig. 5. It can be seen that InstructEdit can accurately locate either an object or a part of an object to be edited according to the user instruction in a fine-grained manner. In addition, the edits are faithfully performed within the regions InstructEdit identifies. The three baselines have difficulty to accurately locate the object of interest.

We show the results of the quantitative evaluation in Tab. 1 for the 30 editing examples shown in Fig. 5 and Supplementary Materials. Results show that our method has the best performance in terms of LPIPS, CLIP score and CLIP directional similarity. That means our method has high ability to edit the images according to the instructions while keeping the unrelated part largely unchanged. Although our method does not achieve the best BRISQUE results, the quality of the images edited our method is still reasonable, which can be also observed by the qualitative results. Nevertheless, those metrics do not capture the large improvements due to

[1] https://github.com/IDEA-Research/Grounded-Segment-Anything

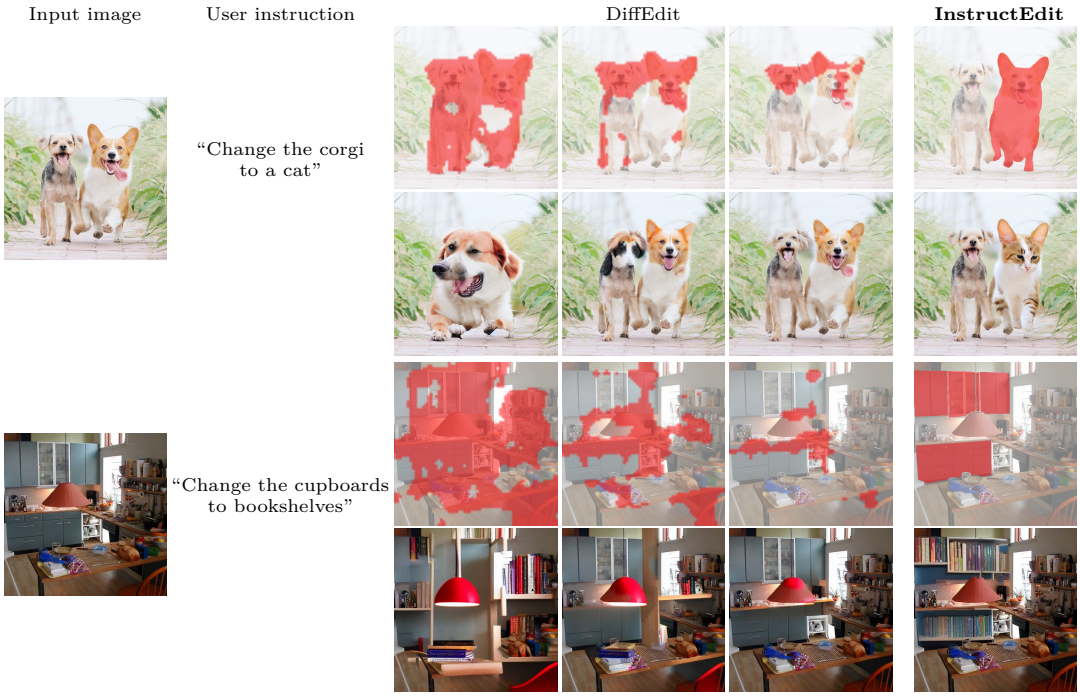

Figure 3: Comparison of the masks (colored in red and blended with the input image) and the corresponding edited image (below each mask) generated by DiffEdit and InstructEdit.

Figure 4: Examples of how BLIP2 improves the quality of edited images by improving the generated prompts. The three generated prompts are segmentation prompt, input caption, and edited caption, respectively. For the first example, the BLIP2 description of the image is "Photo of a dog and a cat". We show three examples for w./wo. BLIP2 with increasing encoding ratio r from left to right. In the second example, the BLIP2 description is "Painting of a girl with a pearl earring by Jan Van Gogh". (BLIP2 gets the name of the painting correct but the name of the artist is wrong.)

our method, mainly because CLIP itself does not capture the fine-grained spatial localization required to judge our complex edits. We therefore perform a user study for these 30 editing examples by comparing our method with the other baseline methods. The results show that InstructEdit was preferred over MDP-$\boldsymbol{\epsilon}_t$, InstructPix2Pix, and DiffEdit in 72.7%, 68.3%, and 66.2% of the cases, respectively.

4.3 Mask improvement

We show examples of how InstructEdit improves the mask quality over the original DiffEdit in Fig. 3. For each example, we use the user instruction as input to InstructEdit and design specific input captions and edited captions for DiffEdit. We also select three different mask thresholds θ for DiffEdit to show how the

Figure 5: Qualitative results of baselines and our method. Here we use the same form of instruction "Change ... to ..." as an example.

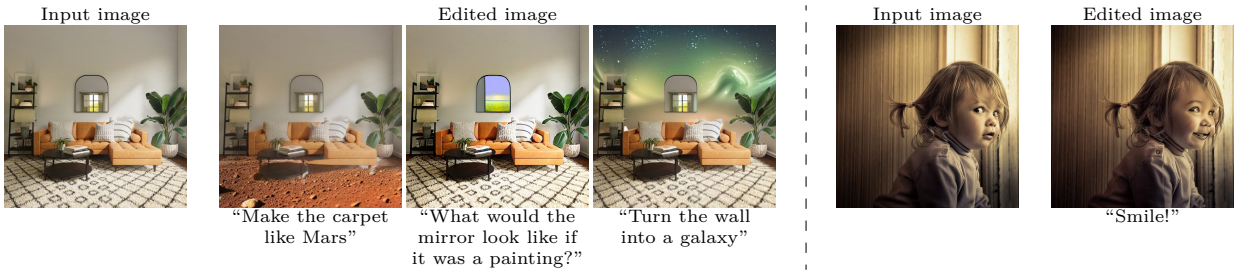

Figure 6: Examples of the variations of the input user instructions. Under each edited image is the input instruction.

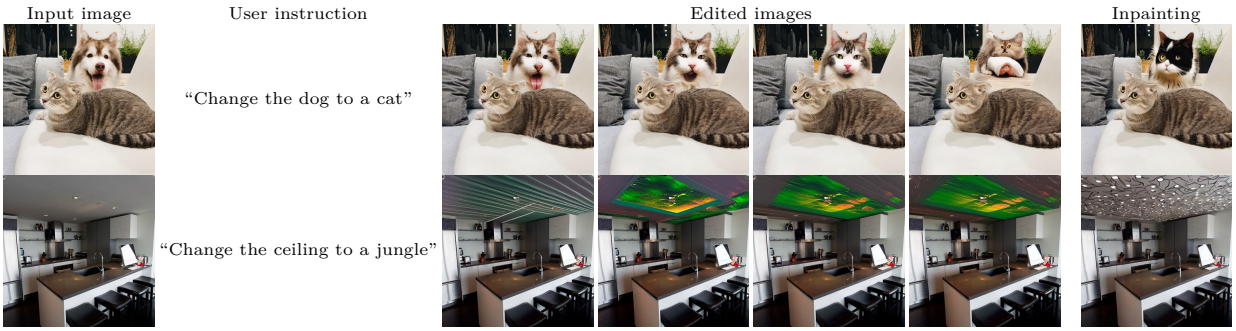

Figure 7: Effect of increasing encoding ratio and comparison with an inpainting baseline. From left to right the edited images are edited by increasing encoding ratios. Note that the inpainting baseline does not accept user instruction as input, but only inpainting prompt.

mask threshold influences the mask quality and therefore the image quality. We show that for DiffEdit the generated mask cannot accurately outline the intended region as specified in the user instruction for all different θ. Therefore, the generated images either have too many or too few changes. On the contrary, for InstructEdit there is no such mask threshold and the generated masks can exactly localize the intended region to be edited. By improving the mask quality, InstructEdit can faithfully do the edit and outperform DiffEdit. We show that without the help of the grounding pre-trained network, it is very hard for the diffusion model itself to accurately outline the region and do fine-grained editing at this scale.

4.4 Instruction processing

In Fig. 4 we show the results of how BLIP2 helps to improve the quality of the edited images. When the instruction does not clearly refer to all the prominent objects in the image, e.g. not mentioning a visible "dog" like in the the first example, the input and edited captions provided by ChatGPT do not contain "dog" as well. This leads to difficulties when the encoding ratio r is increasing. In the second example, where an unclear "it" is used to refer to the object to be edited, without BLIP2 ChatGPT fails to correctly generate a segmentation prompt and the captions. In such a case, BLIP2 provides an extra description of the image such that ChatGPT can understand which object should be edited and provides improved captions.

We also show in Fig. 6 that InstructEdit is very robust and can correctly understand differently phrased instructions as input. This demonstrates the benefit of adopting a large language model to process the instruction rather than hard-coded parsing.

4.5 Encoding ratio

We show the effect of increasing the encoding ratio and compare the results with an inpainting baseline in Fig. 7. The inpainting baseline from [2] is also an image editing framework, which also adopts the Grounded

[2] https://github.com/IDEA-Research/Grounded-Segment-Anything/tree/main

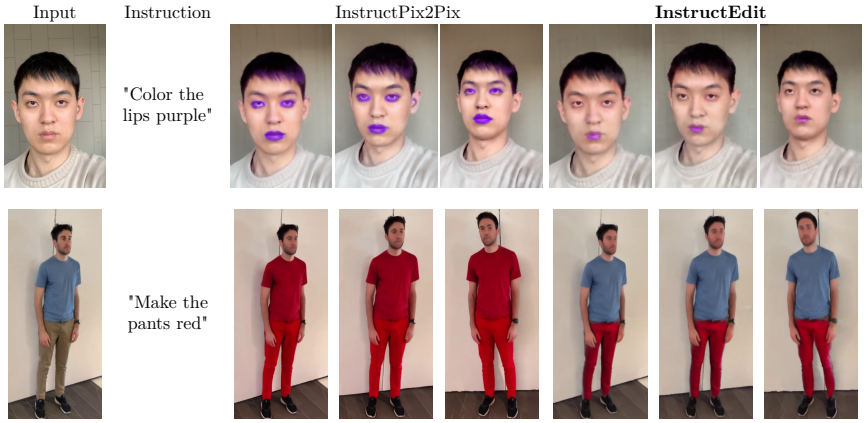

Figure 8: Comparison of the NeRF editing results of our method (InstructEdit) and InstructPix2Pix as backbone using Instruct-NeRF2NeRF as the editing pipeline.

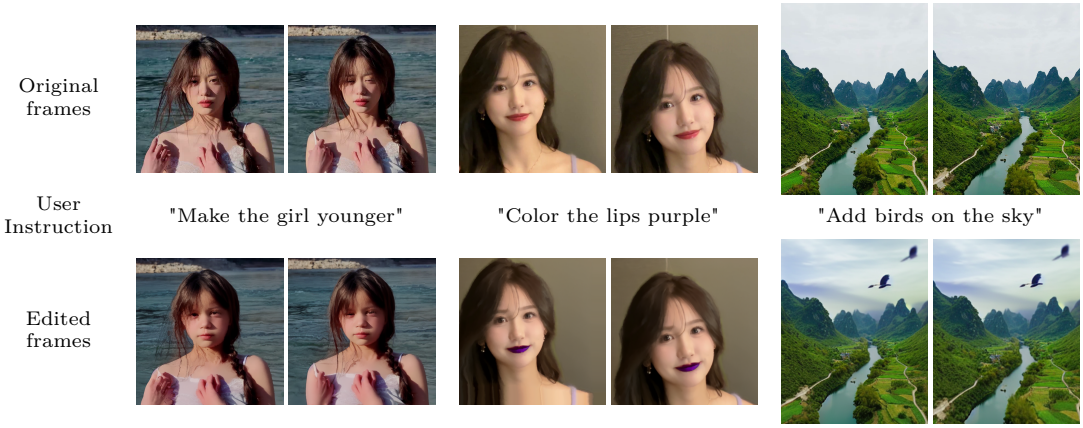

Figure 9: Video editing results. Please refer to the project website for the videos.

Segment Anything to extract a high-quality mask for the input image, but uses the Stable Diffusion Inpainting model [3] as the image editing model.

In general, increasing the encoding ratio can lead to a larger change in the edited image compared to the original image. The edited image will follow the user instruction more and look less like the input image. Compared to using an inpainting model as an image editing model, using the mask-guided generation enables more flexibility in choosing either a larger change or a smaller change. An inpainting model however will completely ignore the original pixel information inside the mask region and only follow the inpainting prompt.

4.6 NeRF editing

We show that our framework can be used as an image editing backbone for a 3D scenes editing pipeline. Here we choose NeRF as the 3D representation. We adopt Instruct-NeRF2NeRF Haque et al. (2023), which is a NeRF editing method based on user instructions. Instruct-NeRF2NeRF iteratively updates images from the training dataset using InstructPix2Pix while optimizing the NeRF scene. Instead of using InstructPix2Pix as the image editor, we replace it with our method and follow the same editing pipeline as Instruct-NeRF2NeRF. We show the comparison in Fig. 8 and video in the project webpage. Our method can do the edit within the desired region, while InstructPix2Pix tends to overedit the image.

[3] https://huggingface.co/runwayml/stable-diffusion-inpainting

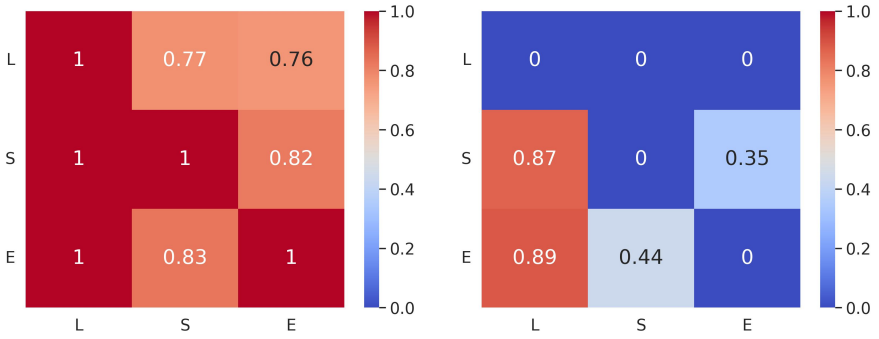

Figure 10: "L", "S" and "E" stand for language processor, segmenter and image editor, respectively. Left matrix shows when a component performs correctly, the percentage of other components also performs correctly. Right matrix show when a component performs incorrectly, the percentage of other components performs correctly.

4.7 Video editing

We show that our framework can also be used for video editing in Fig. 9. We adopt CoDef Ouyang et al. (2023), which decomposes a video into a canonical image and a temporal deformation field recording the transformations. After reconstructing a video, we apply InstructEdit to edit the canonical image and generate the edited video by reconstrucing a video with the edited image and the deformation field.

5 Ablation study

We analyze the accuracy rates of the three components (language processor, segmenter and image editor). We collect 85 real images, and for each image, we perform one editing case ranging from changing object(s), adding object(s), removing object(s) or mixing one object with another. For each user instruction, we randomly select three variations to test if the language processor is robust to the different paraphrases. For each editing case, we test encoding ratios 0.5, 0.7 and 0.9, both with BLIP2 assistance and without.

The overall accuracy rates of the language processor, segmenter and image editor are 97.3%, 71.0% and 69.4%, respectively. We observe that across different variations of the same user instruction, the accuracy rate of each component is almost the same. We also show in Fig. 10 how each component influences one another. Generally, there are accumulated errors for the next component when the last component fails. If the language processor fails to parse the user instruction well, the segmenter and the image editor fail in all the cases. However, interestingly, when the segmenter fails, in 35% of the cases the image editor can still correctly edit the image. In those cases, when the segmenter overshoots the correctly edited region, the image editor can still only edit the region of interest given a correctly inverted prompt and edited prompt. We therefore show that our language processor is robust to instruction variations, and the image editor is also robust to the generated masks.

We also show that when objects are added, BLIP2 helps in 62.5% of the cases as usually very little information is given in the user instructions for adding. 74.6% of the correctly edited cases are sensitive to the choice of encoding ratio, i.e., under a different encoding ratio the edit fails.

6 Limitations, discussions and Conclusions

We show the failure cases in Fig. 11. We find that there are some cases where individual components may fail to perform the instruction, therefore leading the overall failure of the edited image.

There are several limitations in our work. Firstly, the editing capabilities of our system are limited. Further, our framework generally cannot perform well in global editing applications such as “Change the summer to winter” or “Make the photo look like a painting”. Our framework cannot perform deformation. In the future,

(a) "Change the green car to white" (b) "Change the right dog to a cat" (c) "The orange cat should be dancing"

Figure 11: From (a) to (c) we show the failure cases of the Language processor, Segmenter and Image editor, respectively. In (a), the description provided by BLIP2 is "Photo of the lamborghini huracan - a new supercar", which is irrelevant to the context and therefore confuses the parsing of the user instruction. In (b), Segmenter fails to correctly locate the dog. In (c), the Image editor has limited ability to deform the cat.

we want to extend our framework to more editing applications. Secondly, we show the editing performance is sensitive to the choice of encoding ratio, which is a hyper-parameter that the user needs to tune. One future direction can be automatically picking an optimal encoding ratio. More failure cases of each component can be found in the Supplementary Materials.

In this work, we proposed a framework termed InstructEdit that can do fine-grained editing and directly accepts user instructions as input. We use a language processor to process user instructions, a segmenter to generate high-quality masks, and an image editor to do the mask-guided editing. Experiments show that our method outperforms previous editing methods when doing fine-grained edits. We improve the mask quality over DiffEdit and thus improve the quality of edited images. Our framework can also be combined with NeRF or video editing pipelines.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Omri Avrahami, Ohad Fried, and Dani Lischinski. Blended latent diffusion. *arXiv preprint arXiv:2206.02779*, 2022a.

Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18208–18218, 2022b.

Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, Tero Karras, and Ming-Yu Liu. ediff-i: Text-to-image diffusion models with ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800*, 2022.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL `https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf`.

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, 2023.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=r1xMH1BtvB`.

Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL `https://doi.org/10.18653/v1/n19-1423`.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. A survey on in-context learning, 2023.

Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models, 2023.

Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators, 2021.

Anchit Gupta, Wenhan Xiong, Yixin Nie, Ian Jones, and Barlas Oğuz. 3dgen: Triplane latent diffusion for textured mesh generation. *arXiv preprint arXiv:2303.05371*, 2023.

Ayaan Haque, Matthew Tancik, Alexei Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

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.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. Clipscore: A reference-free evaluation metric for image captioning, 2022.

Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models, 2022a.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv:2204.03458*, 2022b.

Lianghua Huang, Di Chen, Yu Liu, Yujun Shen, Deli Zhao, and Jingren Zhou. Composer: Creative and controllable image synthesis with composable conditions, 2023a.

Xinyu Huang, Youcai Zhang, Jinyu Ma, Weiwei Tian, Rui Feng, Yuejie Zhang, Yaqian Li, Yandong Guo, and Lei Zhang. Tag2text: Guiding vision-language model via image tagging. *arXiv preprint arXiv:2303.05657*, 2023b.

Ka-Hei Hui, Ruihui Li, Jingyu Hu, and Chi-Wing Fu. Neural wavelet-domain diffusion for 3d shape generation. In *SIGGRAPH Asia 2022 Conference Papers*, pp. 1–9, 2022.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. *arXiv preprint arXiv:2210.09276*, 2022.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.

Gihyun Kwon and Jong Chul Ye. Diffusion-based image translation using disentangled style and content representation. *arXiv preprint arXiv:2209.15264*, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023a.

Yuheng Li, Haotian Liu, Qingyang Wu, Fangzhou Mu, Jianwei Yang, Jianfeng Gao, Chunyuan Li, and Yong Jae Lee. Gligen: Open-set grounded text-to-image generation. 2023b.

Jun Hao Liew, Hanshu Yan, Daquan Zhou, and Jiashi Feng. Magicmix: Semantic mixing with diffusion models. *arXiv preprint arXiv:2210.16056*, 2022.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

Oscar Mañas, Pau Rodriguez, Saba Ahmadi, Aida Nematzadeh, Yash Goyal, and Aishwarya Agrawal. Mapl: Parameter-efficient adaptation of unimodal pre-trained models for vision-language few-shot prompting. *arXiv preprint arXiv:2210.07179*, 2022.

Chenlin Meng, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on Image Processing*, 2012.

OpenAI. Introducing chatgpt, 2023. URL https://openai.com/blog/chatgpt.

Hao Ouyang, Qiuyu Wang, Yuxi Xiao, Qingyan Bai, Juntao Zhang, Kecheng Zheng, Xiaowei Zhou, Qifeng Chen, and Yujun Shen. Codef: Content deformation fields for temporally consistent video processing, 2023.

Dong Huk Park, Grace Luo, Clayton Toste, Samaneh Azadi, Xihui Liu, Maka Karalashvili, Anna Rohrbach, and Trevor Darrell. Shape-guided diffusion with inside-outside attention. *arXiv*, 2022.

Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation, 2023.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

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 Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, 2023.

Dani Valevski, Matan Kalman, Yossi Matias, and Yaniv Leviathan. Unitune: Text-driven image editing by fine tuning an image generation model on a single image, 2022.

Can Wang, Ruixiang Jiang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Nerf-art: Text-driven neural radiance fields stylization. *IEEE Transactions on Visualization and Computer Graphics*, 2023a.

Qian Wang, Biao Zhang, Michael Birsak, and Peter Wonka. Mdp: A generalized framework for text-guided image editing by manipulating the diffusion path, 2023b.

Ruihan Yang, Prakhar Srivastava, and Stephan Mandt. Diffusion probabilistic modeling for video generation, 2022.

Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *arXiv preprint arXiv:2301.11445*, 2023.

Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection, 2022.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric, 2018.

Xin-Yang Zheng, Hao Pan, Peng-Shuai Wang, Xin Tong, Yang Liu, and Heung-Yeung Shum. Locally attentional sdf diffusion for controllable 3d shape generation. *arXiv preprint arXiv:2305.04461*, 2023.

A Reproducibility

Our implementation can be found in https://anonymous.4open.science/r/InstructEdit_Anonymous-D2CC.

B DiffEdit

DiffEdit is a mask-based text-guided image editing model that can automatically generate masks. During the denoising process, different text prompts will guide the diffusion model to yield different predictions. By contrasting two predictions, the difference can give information for which image regions the input caption and edited caption have different estimates. We refer to the text prompt that is used to generate or inverse the input image as input caption $\mathbf{c}^{(i)}$, and to the text prompt that is used to describe the edited image as edited caption $\mathbf{c}^{(e)}$. The predicted noise at each timestep t guided by $\mathbf{c}^{(i)}$ and $\mathbf{c}^{(e)}$ are $\boldsymbol{\epsilon}_t^{(i)} = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}^{(i)}, t)$ and $\boldsymbol{\epsilon}_t^{(e)} = \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}^{(e)}, t)$, respectively. The difference of the prediction $\boldsymbol{\epsilon}_t^{(d)}$ at timestep t is then calculated as:

$$\boldsymbol{\epsilon}_t^{(d)} = \left| \boldsymbol{\epsilon}_t^{(i)} - \boldsymbol{\epsilon}_t^{(e)} \right|, \tag{5}$$

$\widetilde{\boldsymbol{\epsilon}}_t^{(d)}$ is finally decoded from the latent to a binary mask image $\mathbf{M}$ with a threshold θ.

DiffEdit uses DDIM inversion to invert the input image $\mathbf{x}_0$ into the initial noise $\mathbf{x}_T$ which can generate $\mathbf{x}_0$. Each inversion step is calculated as:

$$\mathbf{x}_{t+1} = \sqrt{\alpha_{t+1}} \cdot f_\theta(\mathbf{x}_t, \mathbf{c}, t) + \sqrt{1 - \alpha_{t+1}} \cdot \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, \mathbf{c}, t). \tag{6}$$

We iteratively apply this formula until we obtain $\mathbf{x}_T$. However, if we stop the inversion step at timestep $r \leq T$, we encode $\mathbf{x}_0$ into a less noised version $\mathbf{x}_r$. r is called the encoding ratio as in DiffEdit. A larger value of r indicates a stronger edit effect, making the edited image guided more by the edited caption but less similar to the input image.

C Editing types

In this work, we focus on image edits restricted to certain object categories or the number of occurrences in the input image. We consider replacing objects with other objects and changing the attributes of objects. The edited image should faithfully follow the user instructions while also preserving the layout and the appearance of the other parts in the original image.

- Single-object: there is a foreground object consisting of multiple parts and we want to edit one part of that object.
- Multi-object of the same type: there are several objects of the same type and we want to edit one or more of them.
- Multi-object of different types: there are several objects of different types and we want to edit one of them.

D Implementation details

D.1 Example of task template

We show one task template we provide to ChatGPT:

For example, if the user says "Change the dog to a cat", you need to give the segmentation model only the keyword "Dog". You also need to give the image editing model two text prompts: "Photo of a dog", and "Photo of a cat". Your answer should be in the form of: Segmentation prompt: Dog. Editing prompt 1: "Photo of a dog". Editing prompt2: "Photo of a cat".

Here, *Editing prompt 1* is the input caption $\mathbf{c}^{(i)}$ and *Editing prompt 2* is the edited caption $\mathbf{c}^{(e)}$.

D.2 Baselines

For MDP-$\boldsymbol{\epsilon}_t$, we fix the interpolation factor as 1 as default and vary the editing starting timestep and ending timestep; For InstructPix2Pix, we only tune the classifier-free guidance factor for text condition; For DiffEdit and InstructEdit, as they both share the same generation process after obtaining a mask, we tune the encoding ratio r. For DiffEdit we also tune the additional parameter θ which controls the threshold when computing a binary mask. For InstructPix2Pix and InstructEdit, we use the same user instruction as input, while for MDP-$\boldsymbol{\epsilon}_t$ and DiffEdit we manually design the input caption and edited caption based on the user instruction.

For MDP-$\boldsymbol{\epsilon}_t$, we use the official implementation from https://github.com/QianWangX/MDP-Diffusion. For InstructPix2Pix, we use the official implementation from https://github.com/timothybrooks/instruct-pix2pix. For DiffEdit, as there is no official implementation available, we refer to https://github.com/johnrobinsn/diffusion_experiments/blob/main/DiffEdit.ipynb and modify the parts that are not consistent with the paper Couairon et al. (2022).

D.3 NeRF editing

We use Instruct-NeRF2NeRF as our editing pipeline. The original Instruct-NeRF2NeRF uses InstructPix2Pix as the image editing backbone. Here we replace it with InstructEdit. We refer to the implementation from https://github.com/ayaanzhaque/instruct-nerf2nerf which is developed based on the Nerfstudio Tancik et al. (2023) framework. For the NeRF reconstruction stage, we use the default method for static scenes in Nerfstudio called *nerfacto* for 30000 iterations. For the NeRF editing stage, we use the *in2n* config for the InstructPix2Pix backbone but the *in2n_tiny* config for the InstructEdit backbone due to memory constraints caused by pre-trained models. Please refer to the documentation of Nerfstudio for the settings of these two configurations. For both methods, we perform dataset updating for around 3000 iterations.

D.4 Video editing

We use CoDef Ouyang et al. (2023) as video representation framework. CoDef learns a canonical image and a deformation field for an input video. By applying InstructEdit as an image editing method to the canonical image, we can obtain a consistent edited version of the original video. We use the official implementation from https://github.com/qiuyu96/CoDeF. We use the provided datasets *beauty_0*, *beauty_1* and *scene_0*, official checkpoints and their corresponding configurations.

E More results

E.1 More testing examples from InstructPix2Pix

We show more testing examples from InstructPix2Pix in Fig. 13. Interestingly, in the third test example "*What would it look like if it were snowing?*", which belongs to global editing, our generated mask covers the whole region of the input image, which is therefore able to create an effect similar to global editing.

E.2 More editing applications

Despite focusing on fine-grained scale localized editing, in Fig. 14 we additionally show the performance of our method on three more kinds of editing applications: global editing, deformation, and object swapping, while the latter two are generally considered more difficult. We show that for global editing, our method is able to generate a mask covering large areas in order to edit the global style. However, for deformation and object swapping, our method can only generate sub-optimal edits or completely fail to understand the editing instructions. There are two limitations of our framework behind the failure cases: first, the edit is performed within the area of the generated mask, which may not be optimal to modify the shape of the object; second, the mask-guided editing method we adopted is purely text-based, which is not able to transfer image-based appearance from one object to another on an image. While the first limitation can be alleviated by inflating or compressing the automatic mask by some designed factor, the second limitation needs more

careful design to under image-based prompting. Future research can be conducted on how to enable more interesting editing applications under our end-to-end image editing framework.

E.3 Additional NeRF editing results using NeRF-Art

We show the rendered images from the edited NeRF using the same example *"Color the lips purple"* in Fig. 8 by Wang et al. (2023a) in Fig. 12. Compare edited NeRF by our method and NeRF-Art, the results obtained by ours have accurate edited region (lips), while the results of NeRF-Art have artifacts and overshot edited region.

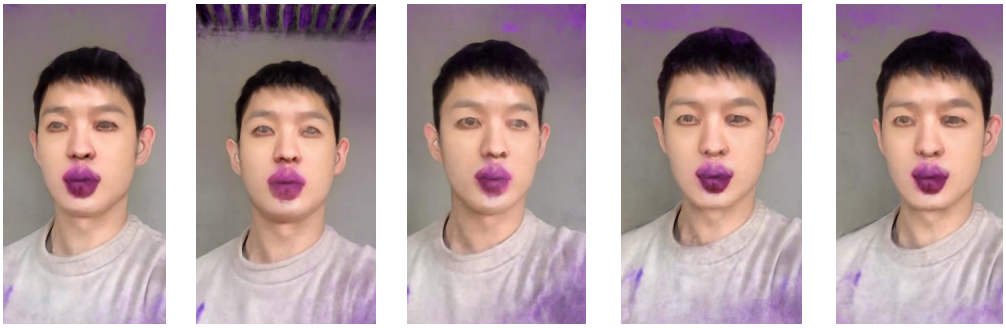

Figure 12: More NeRF editing results using NeRF-Art Wang et al. (2023a). The example is the same as shown in Fig. 8.

E.4 Qualitative and quantitative comparison

We show more qualitative results in Figs. 16 and 17. We show that InstructEdit performs better than the baseline methods in the majority of cases.

For the user study, we could acquire 26 workers on MTurk who were presented with triplets of images, each triplet consisting of one input image and two edited images. One of the edited images always was the output from our method, the second one was a randomly picked edited image from one of the baseline methods. We presented triplets of images as done in the main paper, each triplet consisting of one input image and two edited images. One of the edited images was the output from our method, while the second one was a randomly picked edited image from one of the baseline methods. For every example, we ask the participant a question: *"Which image applied the instruction more appropriately?"*. We show a screenshot of the user study interface in Fig. 15.

E.5 Mask improvement

We show more results for mask improvement of InstructEdit over DiffEdit in Fig. 18. The quality of the automatic masks generated by DiffEdit is highly affected by the mask threshold θ. Nevertheless, the mask area generated by DiffEdit can be completely off the region that should be edited under different θ. With the help of the grounding model and segmentation model, however, the generated mask can accurately outline the region to be edited without tuning θ.

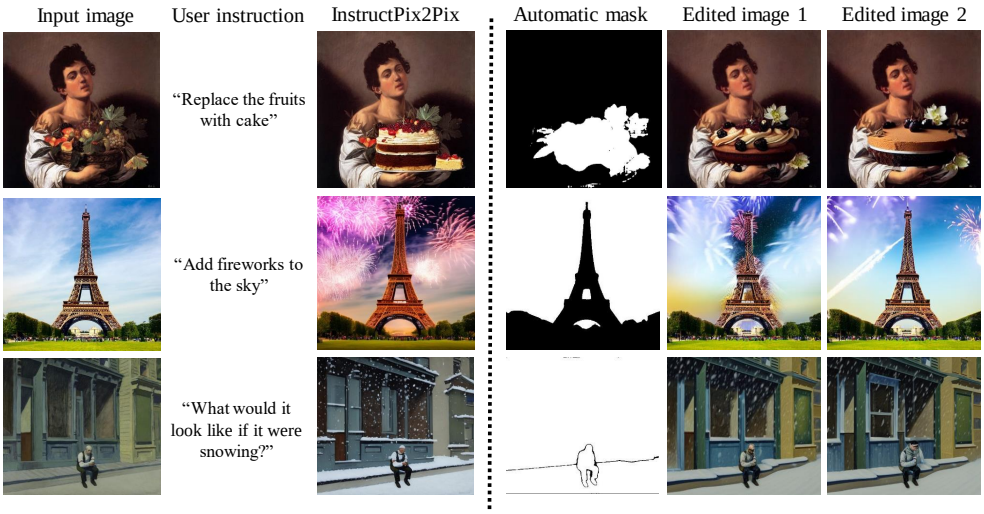

Figure 13: More testing examples from InstructPix2Pix. We show the automatic mask and two edited images variations generated by our method in column 4 to 6, respectively.

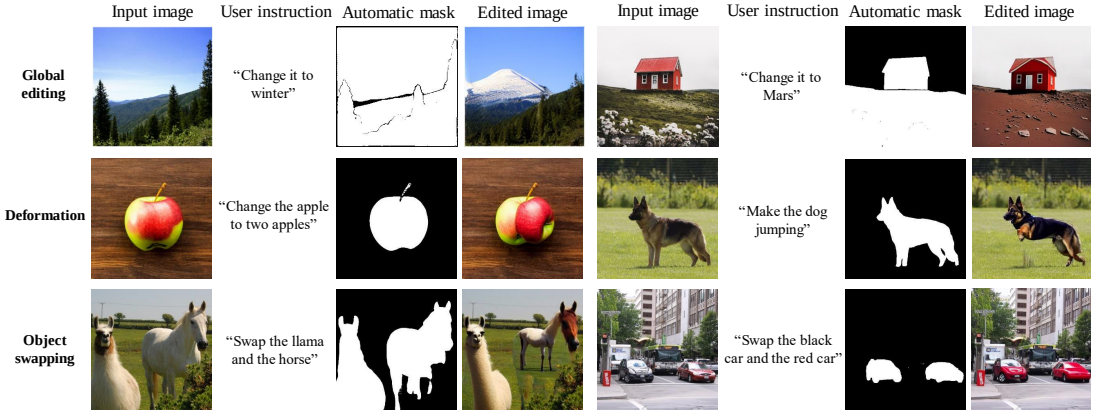

Figure 14: The results of our method on more editing applications.

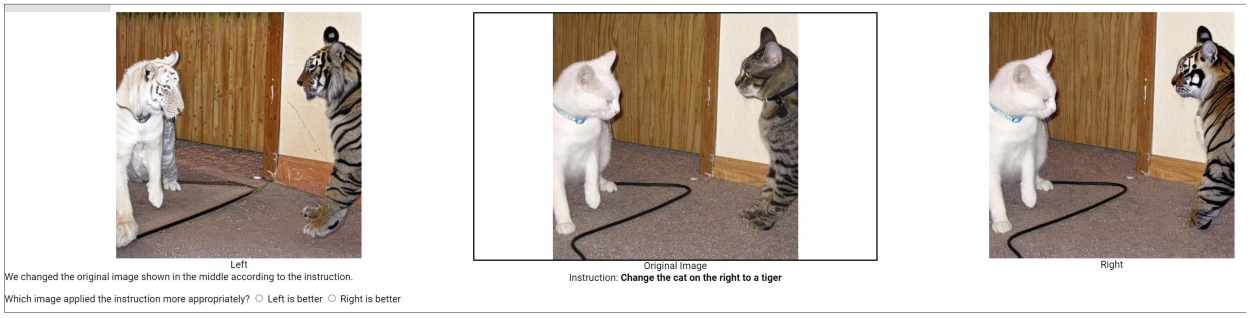

Figure 15: Screenshot of the user study interface.

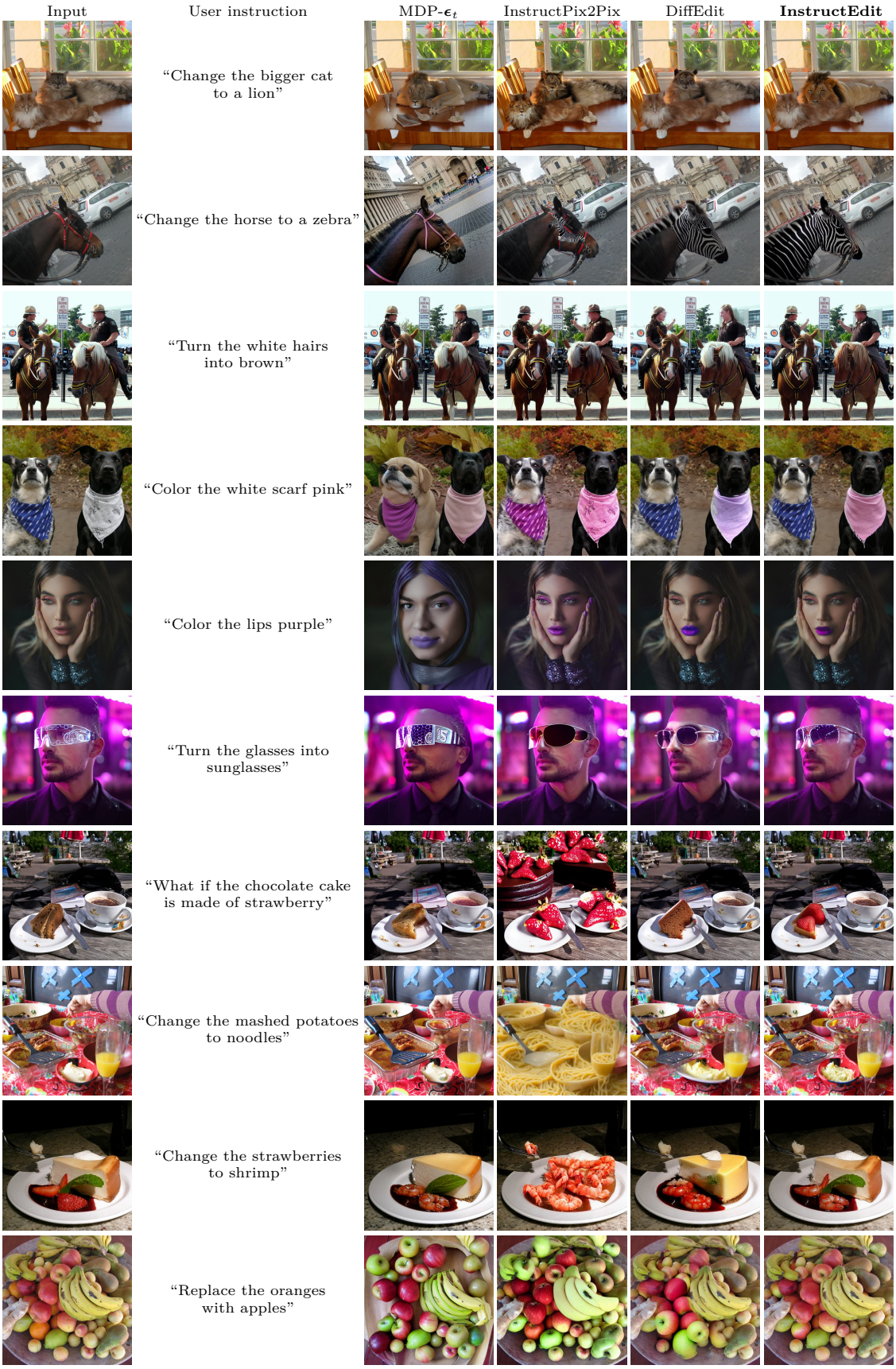

Figure 16: Qualitative results of baselines and our method.

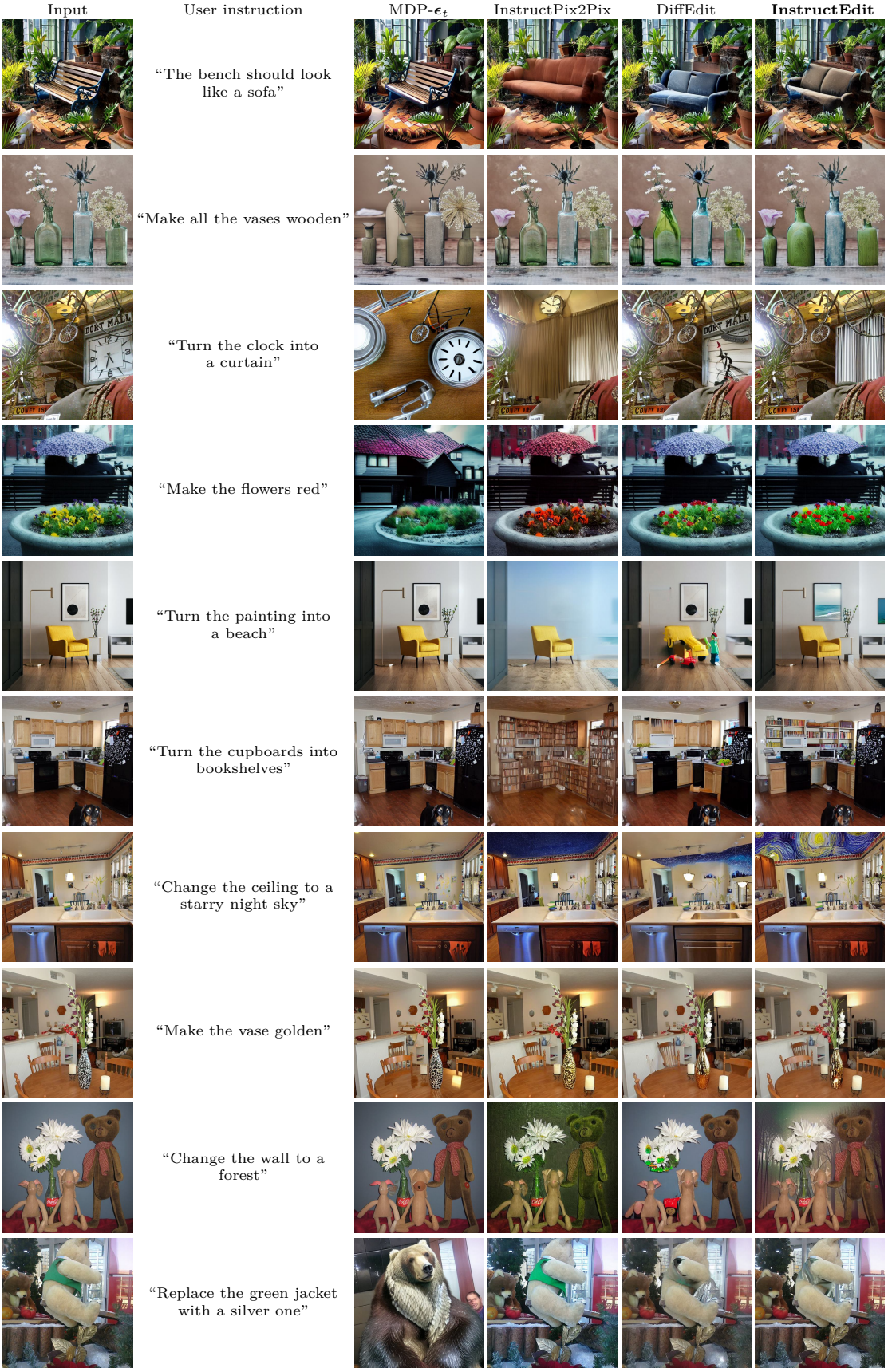

Figure 17: Qualitative results of baselines and our method.

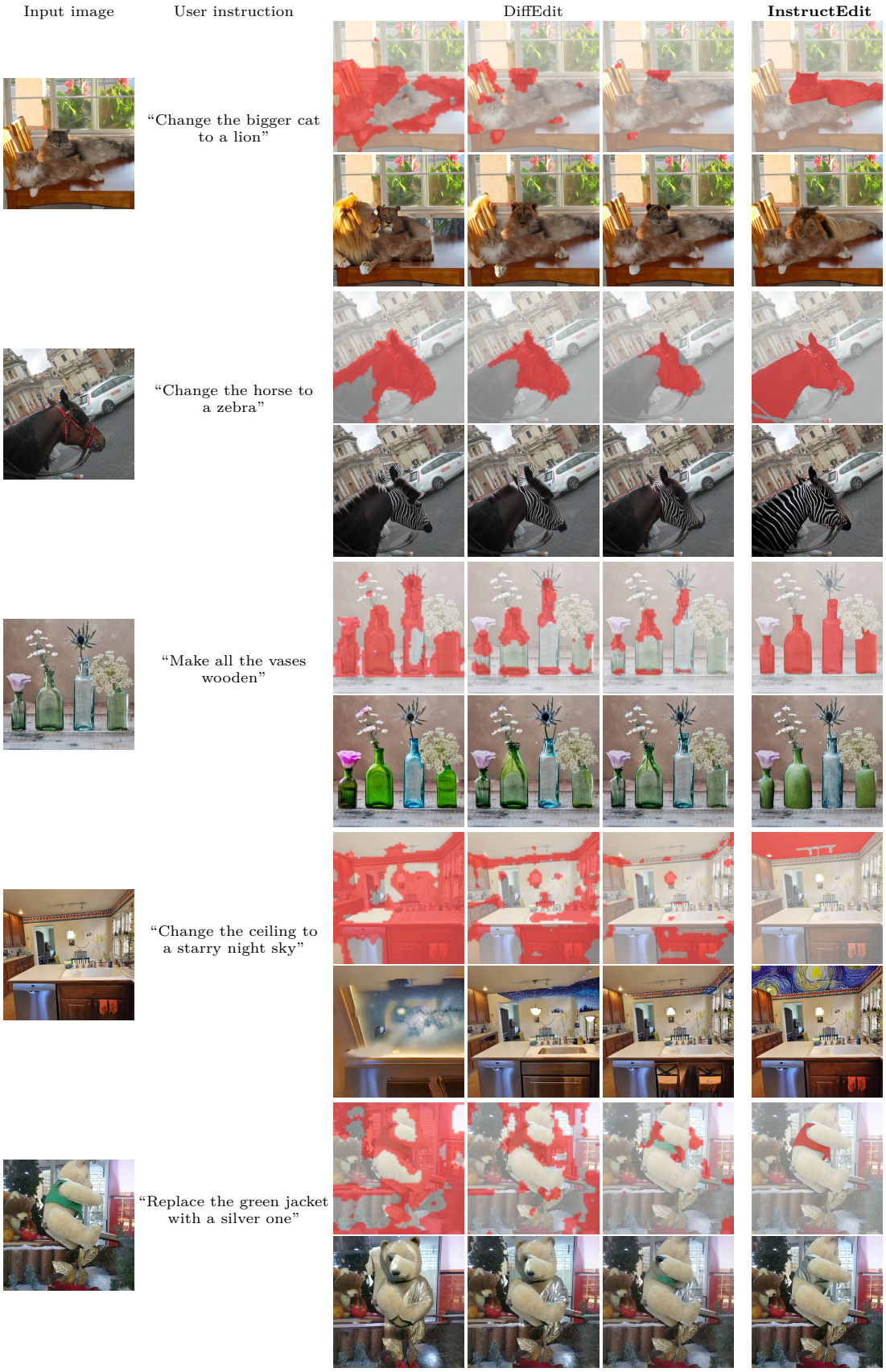

Figure 18: Comparison of the masks (colored in red and blended with the input image) and the corresponding edited image (below each mask) generated by DiffEdit and InstructEdit.