Supplementary Materials for EditCLIP: Representation Learning for Image Editing

Anonymous ICCV submission

Paper ID 6088

A. Quantitative metrics

Here we explain the commonly-adopted metrics we used in the quantitative evaluation.

CLIP Score: is calculated as the cosine similarity between the embedding of the output image I_o and the text embedding of the description of the output image T_o; the embeddings are from the original CLIP image encoder $\mathcal{F}_\theta$ and CLIP text encoder $\mathcal{G}_\theta$. It can measure how much the output image is aligned with its description. The calculation is as follows:

$$\text{CLIP Score} = \cos\left(\mathcal{F}_\theta(I_o), \mathcal{G}_\theta(T_o)\right), \tag{1}$$

where $cos(A, B) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$,denoting the cosine similarity.

CLIP Directional Similarity: calculates the cosine similarity between the difference of the embeddings of the query image I_q and output image I_o, against the difference of the embeddings of query image description T_q and output image description T_o. The calculation is as follows:

$$\text{CLIP Direct. Similarity} = \cos\left(\mathcal{F}_\theta(I_q) - \mathcal{F}_\theta(I_o), \mathcal{G}_\theta(T_q) - \mathcal{G}_\theta(T_o)\right). \tag{2}$$

Alternatively, when the text instruction T is available, the calculation becomes:

$$\text{CLIP Direct. Similarity} = \cos\left(\mathcal{F}_\theta(I_q) - \mathcal{F}_\theta(I_o), \mathcal{G}_\theta(T)\right). \tag{3}$$

It can measure how much the change of the images matches the change of the text descriptions. Here, the change of the text descriptions (e.g., *A forest in the summer* $\rightarrow$ *A forest in the winter*) implicitly serve as an text instruction (*Change summer to winter*). Note that this is a similar counterpart to our proposed metric **`EC2T`**, while **`EC2T`** directly measure the change of the images against the change of the text instruction. Please refer to the main paper for the definition for **`EC2T`**.

s_{visual}**:** is a metric proposed in [4], which can be considered as a variant of the CLIP Directional similarity. It calculates the cosine similarity between the difference of the query image embedding and output image embedding, against the difference of the embeddings of the input image I_i and edit image I_e. The calculation is as follows:

$$S_{visual} = \cos\left(\mathcal{F}_\theta(I_q) - \mathcal{F}_\theta(I_o), \mathcal{F}_\theta(I_i) - \mathcal{F}_\theta(I_e)\right). \tag{4}$$

Note that this is a similar counterpart to our proposed metric **`EC2EC`**. Please refer to the main paper for the definition for **`EC2EC`**.

LPIPS: (Learned Perceptual Image Patch Similarity)[8] measures the perpetual similarity between the two images. Here we calculate it between the query image I_q and output image I_o. Different from the above mentioned metrics, LPIPS serves as a direct evaluation of how much the output image preserves the query image. A lower LPIPS score usually indicate better faithfulness to the query image. However, too low of LPIPS score may suggest insufficient edits.

B. More qualitative and quantitative results

B.1. Comparison between ours and baselines

We show more qualitative comparisons in Fig. 1.

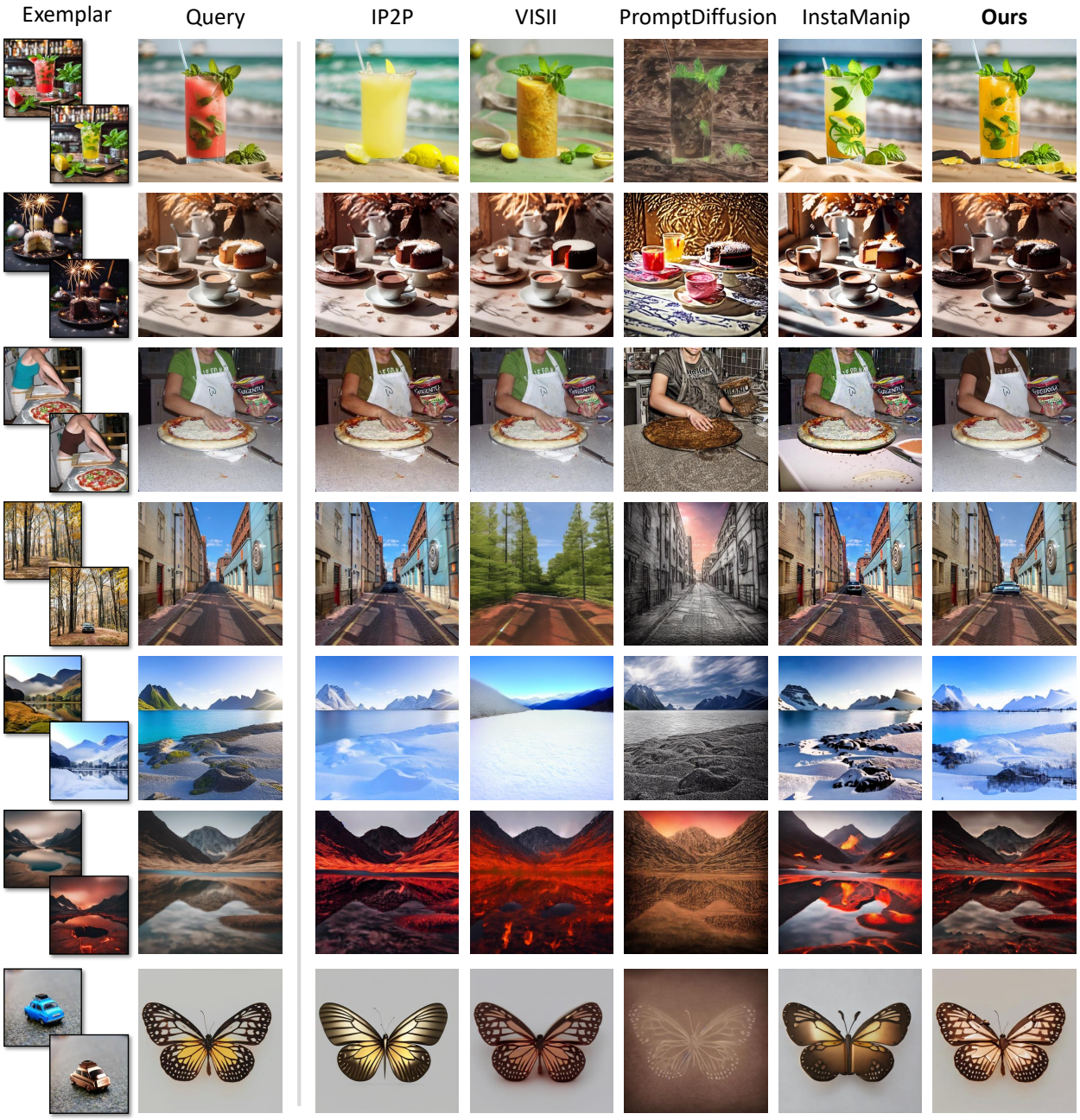

Figure 1. More qualitative comparisons between our method and the baselines.

B.2. Comparison with more baselines

Here we adopt more baselines apart from the ones we use in the main paper. We compare with InstructCLIP[2], IP-Adapter[7] and MDP[6] in Appendix B.2. InstructCLIP follows a similar idea which encodes the changes between input images and edited images through CLIP. IP-Adapter supports one reference image and one reference text, and in our case we use the input image as the reference image, and the description for the edited image as the reference text. MDP is a text-based editing method. We show consistent improvements over exemplar-based InstructCLIP and adapter-based IP-Adapter. Despite text-based MDP has better automatic metric scores, human votes show a 80% preference for ours over MDP.

		Text-based			*Exemplar-based*		*User-Study*		
	LPIPS ↓	CLIP ↑	**EC2T** ↑	CLIP-Dir. ↑	S_{visual}[4] ↓	**EC2EC** ↑	WR-Edit ↑	WR-Pres ↑	RT (s)
MDP[6]	0.424	0.249	0.184	0.231	0.712	0.372	79.68	80.65	60
InstructCLIP[2]	0.433	0.194	0.143	0.071	0.867	0.282	72.10	67.42	1.8
IP-Apdater[7]	0.690	0.198	0.162	0.068	0.906	0.292	82.74	83.39	1.8
EditCLIP (Ours)	**0.233**	**0.216**	**0.180**	**0.143**	**0.761**	**0.477**	-	-	**1.8**

Table 1. Quantitative results for exemplar-based image editing. WR-Edit and WR-Pres denote the winning rate of edit quality and input preservation of *our method against other methods* according to human evaluators. RT refers to runtime in seconds. We show the best one in **bold font**.

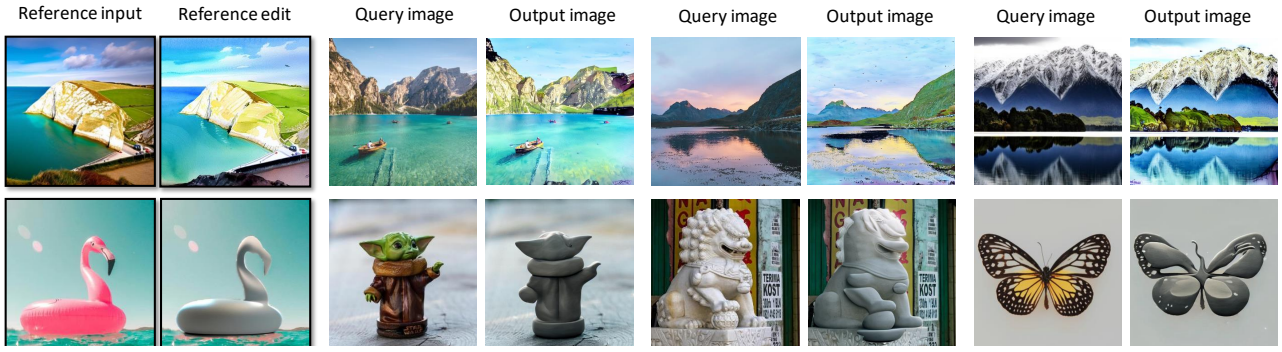

Figure 2. Transfer edits from a same exemplar to different test images.

B.3. Transferring edits to multiple test images

We show more visualization of transferring edits from an given exemplar to multiple different test images in Fig. 2. We show that the learned embedding of the edits are generalizable to different test images. Note that the test images do not have to be very similar to the exemplars in terms of the low-level structure or style, but rather share high-level similar semantics.

B.4. Comparison between VIT-B-32 and VIT-L-14

We compare the performance between VIT-B-32 and VIT-L-14 as backbone architecture for EditCLIP in Fig. 3. We observed that VIT-L-14 achieves a higher quality in most of the cases. While VIT-B-32 can encode the edit from the exemplar, the details of the output image may not be well-preserved (in the first row in Fig. 3), or the edit may not be of faithful (in the second row in Fig. 3). We conjecture that is because VIT-L-14 is a larger VIT model also with smaller patch sizes, which can capture more visual details compared to VIT-B-32. Therefore, we choose VIT-L-14 as the default backbone for EditCLIP. However, we do found that in some cases when VIT-L-14 struggles to maintain the details when doing global editing applications, VIT-B-32 can well-preserve the original layout details instead (in the third row in Fig. 3).

C. Visualization of the feature space

We employ t-SNE to visualize the EditCLIP embedding space, as shown in Fig. 4. First, we randomly sample 4 quartets of [reference input image, reference edited image, query image, ground truth image] from each of the 25 editing groups in TOP-Bench-X, resulting in 100 quartets. For each quartet, we compute two EditCLIP embeddings: (1) a reference pair embedding (from the reference input and edited images) and (2) an edited pair embedding (from the query and ground truth images). In the left panel of Fig. 4, we visualize these embeddings using t-SNE and connect each reference-edited pair from the same quartet with a purple line. The proximity of embeddings from the same quartet demonstrates that EditCLIP effectively captures and clusters semantically similar edits.

Next, we randomly select 100 input-edited pairs each from the IP2P and MagicBrush datasets and project their EditCLIP embeddings into the same space (right panel of Fig. 4). The visualization shows that EditCLIP not only aggregates reference-

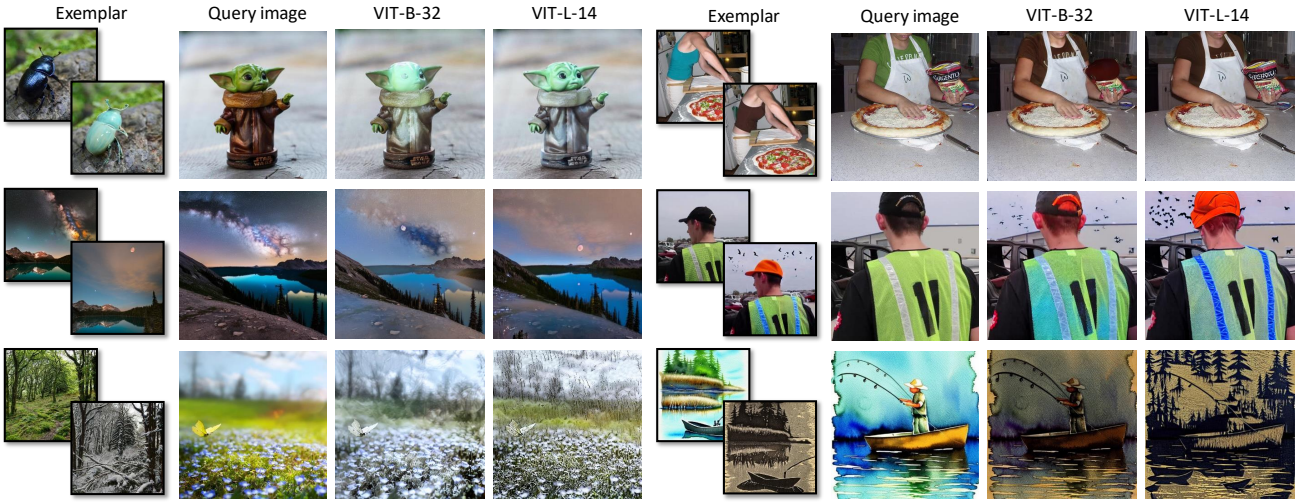

Figure 3. Compare the performance between VIT-B-32 and VIT-L-14 as backbone architecture for EditCLIP.

edited pairs but also distinguishes between different data sources.

For both experiments, we use t-SNE with consistent settings: a perplexity of 50 and PCA-based initialization.

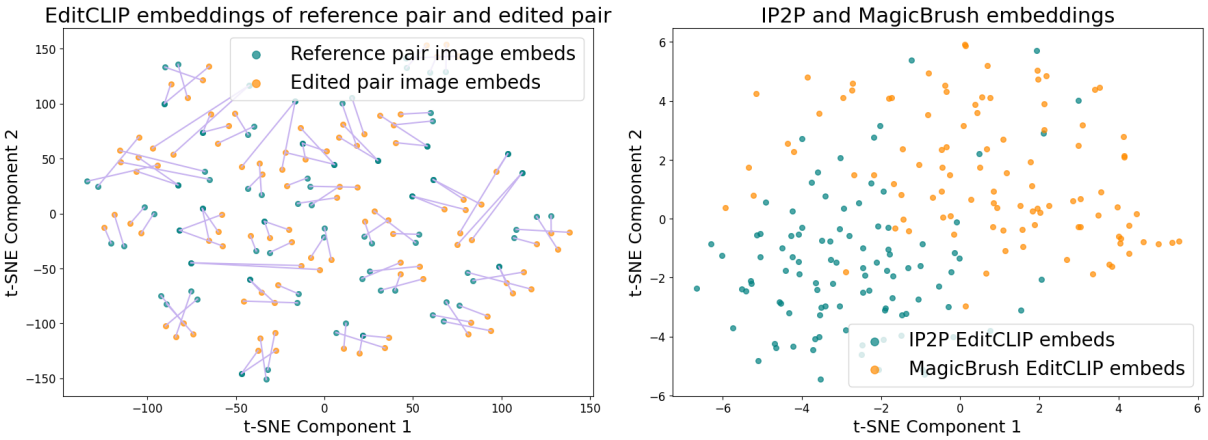

Figure 4. t-SNE visualization of the embedding space.

D. Zero-shot image retrieval

We conduct instruction-based input-edited image pair retrieval on full IP2P dataset. Given a pair of input and edited images from the IP2P dataset as the query, we retrieve the most relevant input-edited image pair from the rest of the IP2P dataset by computing the similarity of their edit embeddings produced by EditCLIP. We compare against the same retrieval task using CLIP directional similarity (CLIP Dir.) and CLIP embedding of edited image in Tab. 2. We show best results compared to the baselines.

E. Failure cases

For our base model that is only trained on IP2P dataset, we report four types of edits that this model fails to faithfully perform: deformation (in Fig. 5(a)), removal (in Fig. 5(b)), changing number of objects (in Fig. 5(c)), and changing positions of objects (in Fig. 5(d)). Training datasets which contain these types of edits and potential model architecture designs are needed in order to enable our model for a series of editing applications, such as pose transfer, virtual try-on and removing unwanted objects.

Table 2. Results of zero-shot image retrieval

	AUC ↑	MAP@10 ↑	Recall@10 ↑	Precision@10 ↑
CLIP	0.140	0.104	0.223	0.036
CLIP-Dir.	0.095	0.069	0.182	0.035
Ours	**0.201**	**0.160**	**0.349**	**0.081**

Figure 5. Failure cases of our method in exemplar-based image editing.

F. Additional training data for removal task

In order to see how our method can benefit from additional training data, we reverse the adding samples from IP2P dataset to simulate a **removal** dataset. We first train EditCLIP jointly on this removal dataset and the original IP2P data. Subsequently, we finetune only the second-stage diffusion model on the removal dataset for 2000 iterations, omitting the input preservation loss. As illustrated in Fig. 6, this finetuning on the second stage only significantly improves performance on removal tasks. These results suggest that training on a wider variety of edit types—such as deformation—could further enhance the model's generalization.

G. Additional ablation studies

G.1. Input loss preservation

We ablate on different values of λ_2 in Fig. 7, which control the strength of the input preservation loss against the diffusion denoising loss. When $\lambda_2 = 0$, it means no input preservation loss is applied. Intuitively, larger number of λ_2 will preserve more input layout, while a smaller one will allow more edits. We balance these two sides and choose 0.05 as the default value for λ_2.

G.2. Choice of EditCLIP Embedding Layer

Different from the common practice [5, 7] that uses the projected embedding from CLIP as the image condition, we found that using hidden states from the last transformer layer before going to the CLIP projection layer is more effective to transfer

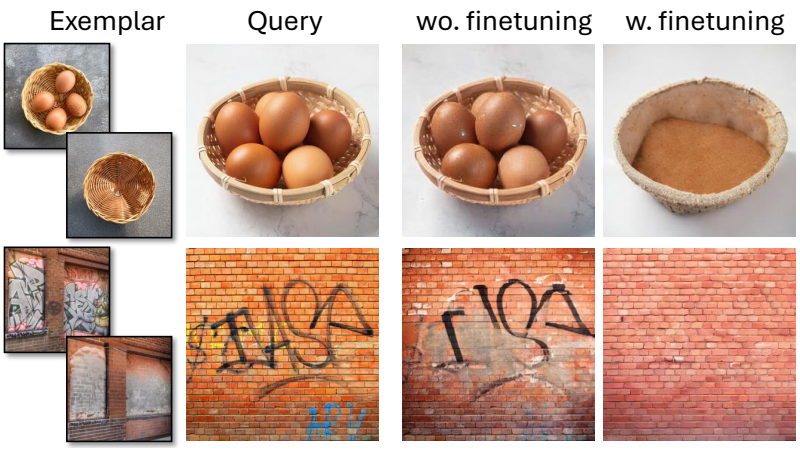

Figure 6. Results wo. and w. finetuning on removal sub-dataset.

Figure 7. The effects of different values of λ_2.

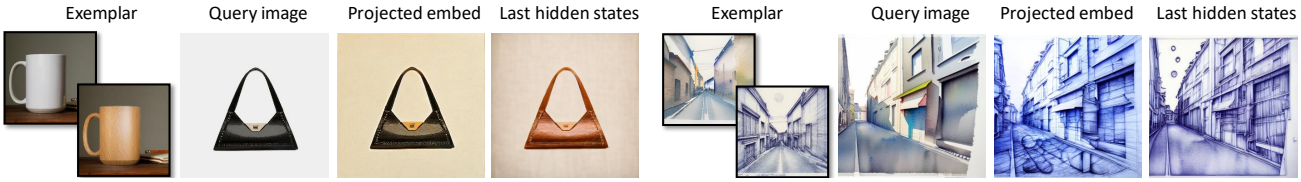

Figure 8. Ablation of using the projected embedding after projection layer or hidden states from the last transformer layer from EditCLIP for the embedding.

the edit while preserving the input layout. Figure 8 that in our task, we found We conjecture that it is because last hidden states contain more tokens, which encode more visual details and hence have higher capacity in general.

G.3. Guidance scale

As it is done in [1], our denoising UNet for exemplar-based editing is also conditioned on both the VAE input image $\mathcal{E}(I_i)$ and edit embedding E. Therefore, during inference, we could apply two separate guidance scales similar to [1], where edit guidance scale s_E controls how the output image follows the edits, and image guidance scale s_I controls how the output image resembles the input image.

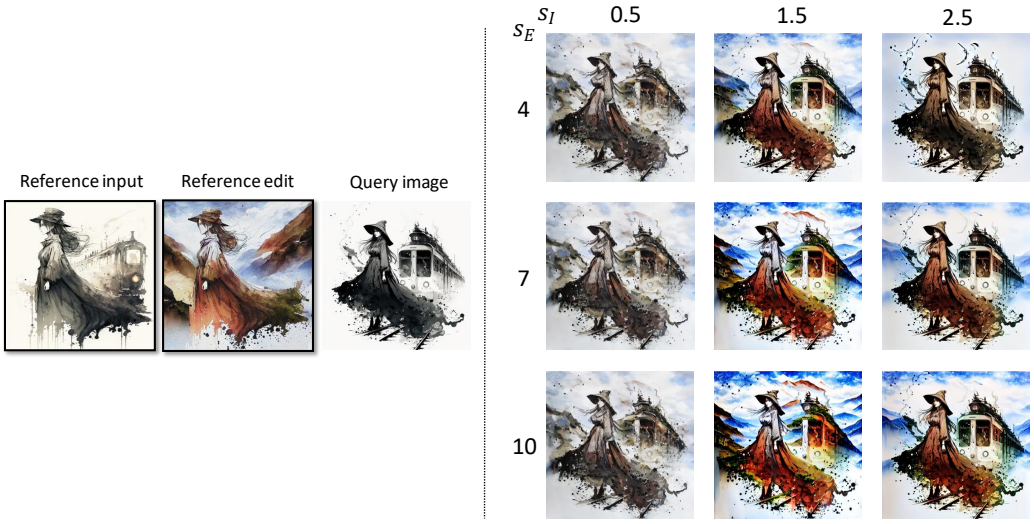

Figure 9. Ablation of using different values for guidance scales s_E and s_I.

The modified score estimate $\tilde{\epsilon}_\theta$ is as follows:

$$\begin{aligned}\tilde{\epsilon}_\theta(x_t, t, \mathcal{E}(I_i), E) =& \epsilon_\theta(x_t, t, \varnothing, \varnothing) \\ &+ s_I \left(\epsilon_\theta(x_t, t, \mathcal{E}(I_i), \varnothing) - \epsilon_\theta(x_t, t, \varnothing, \varnothing)\right) \\ &+ s_E \left(\epsilon_\theta(x_t, t, \mathcal{E}(I_i), E) - \epsilon_\theta(x_t, t, \mathcal{E}(I_i), \varnothing)\right) \end{aligned} \tag{5}$$

We show the ablation of the guidance scales in Fig. 9. In general, as s_E increases, the output images will have stronger editing effects; while when s_I increases, the output images will follow more the input image. By default, we set $s_E = 7$ and $s_I = 1.5$, which is the suggested practice in [1]. However, users can tune these hyperparameters to obtain desired results.

H. Benchmark statistics

We adapt the TOP-Bench dataset [9] for exemplar-based image editing and we denote it as *TOP-Bench-X*. TOP-Bench consists of different types of edits, where each type includes a set of training and test pairs. We use the training set to form exemplar pairs, denoted as $[I_i, I_e]$, while the test set provides the corresponding query image I_q. This results in a total of 1277 samples, comprising 257 unique exemplars and 124 unique queries. Edit types contain between 32 and 60 samples. Please refer to Tab. 3 for detailed numbers of pair samples under each editing groups. We visualize additional exemplar pairs with queries from the benchmark on Fig. 10, where we can see different types of edits present in the benchmark.

I. User study

The user study was conducted on Amazon MTurk with two alternative forced-choice (2AFC) layout as seen on Fig. 11. We use only participants with Master Qualification on the platform. There were a total of 53 unique participants, with the average time of each sample taking 40 seconds, and the average user did 89 samples with a total of 4712 comparisons. During evaluation, in addition to query-exemplar pairing, we perform multiple seeds per method, for the metric evaluation we include all the seeds. We randomly select 2 seeds (out of 5 seeds) for each inference.

Table 3. Statistics of pairs after creating exemplar pairs using Kawar et al. [3] benchmark.

Edit name	# of pairs	Edit name	# of pairs
boy2girl	50	watercolor	40
midnight	50	4dboy	50
seapainting	50	apple	50
sketchstyle	60	cake	50
summer	55	cloud_kitty	50
wallpaper	55	dog2cat	55
charcoal	60	juice	50
glasses	50	lava	50
painting	60	rain	50
paintingsnow	50	read_books	50
pencilsketch	50	smile	50
purple	32	traffic_lights	60
snow	50		
Total: 1277			

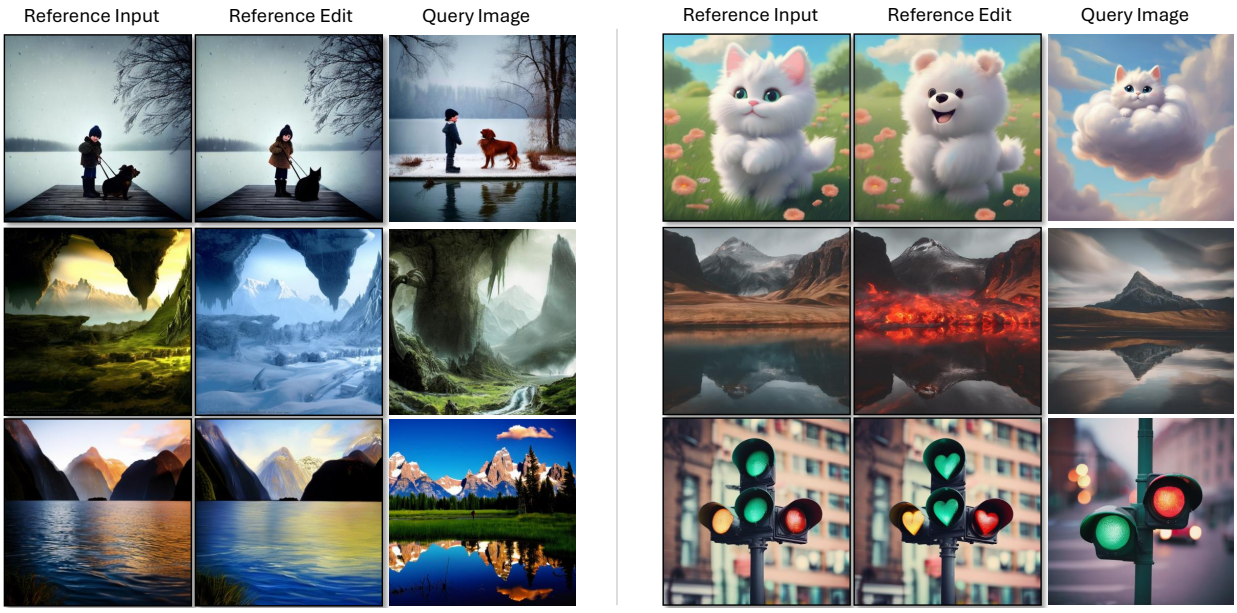

Figure 10. Additional visualization of exemplars present in the *TOP-Bench-X* variant.

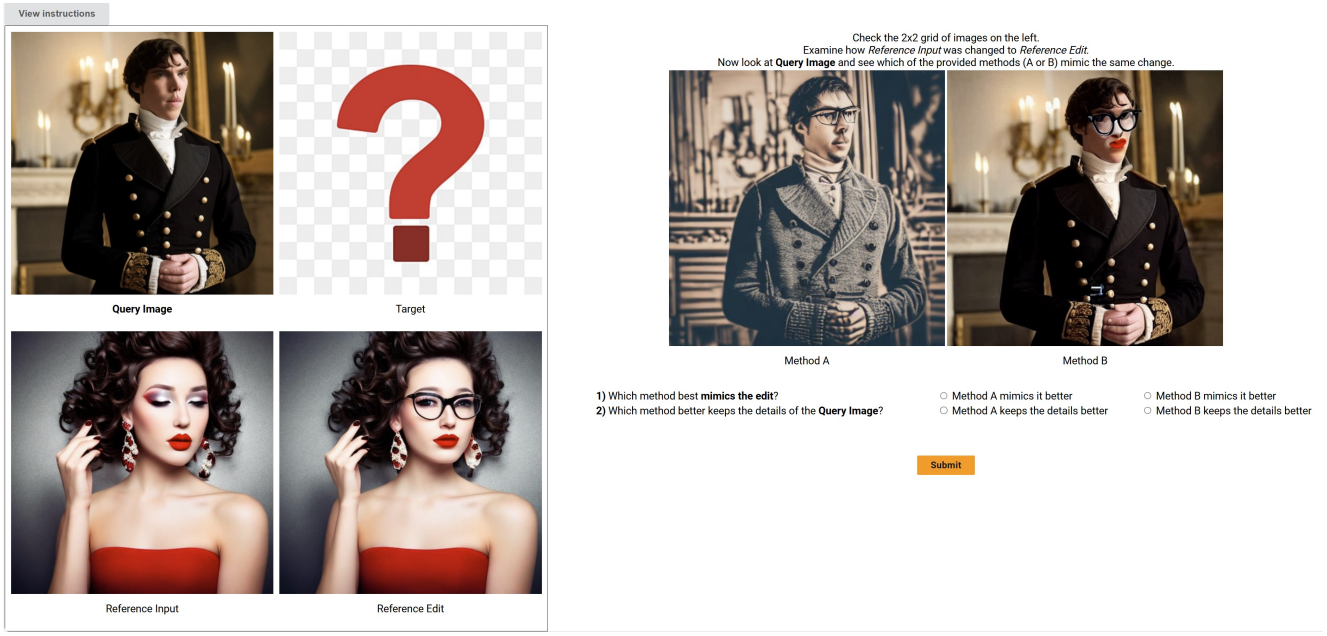

Figure 11. Single example of the 2AFC user study. Participants see the Query and Exemplar pairs on the left and two potential edits on the right. They are asked to select which method best mimics the edit and which better preserves the Query image details.

References

[1] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023. 6, 7

[2] Sherry X. Chen, Misha Sra, and Pradeep Sen. Instruct-clip: Improving instruction-guided image editing with automated data refinement using contrastive learning, 2025. 2, 3

[3] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6007–6017, 2023. 8

[4] Thao Nguyen, Yuheng Li, Utkarsh Ojha, and Yong Jae Lee. Visual instruction inversion: Image editing via image prompting. *Advances in Neural Information Processing Systems*, 36:9598–9613, 2023. 1, 3

[5] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *ArXiv*, abs/2204.06125, 2022. 5

[6] Qian Wang, Biao Zhang, Michael Birsak, and Peter Wonka. MDP: A generalized framework for text-guided image editing by manipulating the diffusion path. *Transactions on Machine Learning Research*, 2024. 2, 3

[7] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. 2023. 2, 3, 5

[8] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 1

[9] Ruoyu Zhao, Qingnan Fan, Fei Kou, Shuai Qin, Hong Gu, Wei Wu, Pengcheng Xu, Mingrui Zhu, Nannan Wang, and Xinbo Gao. Instructbrush: Learning attention-based instruction optimization for image editing. *arXiv preprint arXiv:2403.18660*, 2024. 7