Figure 7. LEDITS++ easily produces variations of an edit (different (styles of) sunglasses) by resampling the inversion process. (Best viewed in color)

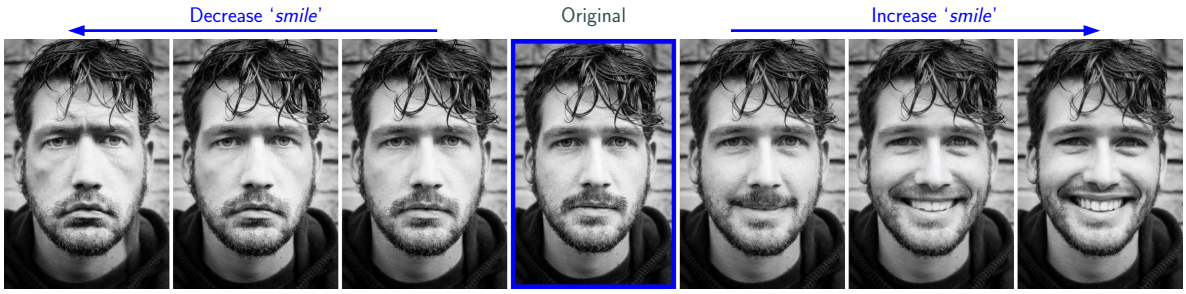


Figure 8. Monotonicity of editing scale with LEDITS++. The original image (middle) is edited with varying scales of the same edit ('smile'). The scale for 'smile' is semantically reflected in the images. (Best viewed in color)

Appendix

A. Broader (Societal) Impact

With LEDITS++, we aim to provide an easy-to-use image editing framework. It lowers the barrier of entry for experienced artists and novices alike, allowing them to unlock the full potential of generative AI in the pursuit of creative expression. Moreover, it puts the user in control for fruitful human-machine collaboration. Crucially, current text-to-image models [29, 33, 37] hold the potential to wield a profound influence on society. When applied in creative and design domains, their dual use offers both promise and peril, as highlighted by prior research [4, 11]. The models are trained on large amounts of data from the web [40], granting them the inherent capacity to generate content that may contravene societal norms, including the creation of inappropriate material like pornography [39], or content that violates law such as child abuse. More alarmingly, the inadvertent generation of inappropriate content is precipitated by spurious correlations within these models. Harmless prompts can lead to the creation of decidedly objectionable content [4, 11]. A prime example of this phenomenon lies in the correlation between specific phrases and the perpetuation of stereotypes, such as the connection between mentions of ethnicity and economic status. For example, an increase of the concept '*black person*' may inadvertently amplify the appearance of the concept '*poverty*.'

Conversely, methods like LEDITS++ also possess the potential to mitigate bias and inappropriateness, a prospect highlighted by prior research [10, 11], e.g. through dataset augmentation [36]. Furthermore, established strategies offer means to mitigate the generation of inappropriate content [13, 39] that could be deployed in tandem with LEDITS++. In summary, we advocate for a cautious approach to the utilization of these models, recognizing both the risks and promises they bring to the realm of AI-powered image editing.

B. Further Examples on LEDITS++ Properties

As discussed in Sec. 4, LEDITS++ versatility benefits from re-sampling to provide variations of edits. The example in Fig. 7 demonstrates the additional control non-deterministic variations provide to the user, which can select the preferred interpretation of the edit instruction.

The precision and versatility of LEDITS++ further benefit from the fact that the magnitude of an editing concept in the output scales monotonically with the edit scale s_e. In Fig. 8, we can observe the effect of increasing s_e. Both for positive and negative guidance, the change in scale correlates with the strength of the smile or frown. Consequently, any changes to the input image can be steered intuitively using the edit guidance scale.

C. Experimental Details

Subsequently, we provide further details on the experiments presented in the main body of the paper. We first provide

information on the reconstruction and runtime experiments (Sec. 4), followed by the masking evaluation (Sec. 5) and multi-conditioning experiments (Sec. 6). Details on the user study are independently described in App. E. All experiments were performed using the respective diffusers[4] implementation (version 0.20.2) with Stable Diffusion 1.5.

C.1. Properties

First, we go into detail on the reconstruction and runtime experiments presented in Tab. 1.

C.1.1 Reconstruction Error

Since the Stable Diffusion VAE already induces errors when reconstructing images, we considered the RMSE over the 64x64 latent image instead. We randomly sampled 100 images from the 2017 COCO validation dataset, which we attempted to reconstruct using the default configuration of each method as described in the respective paper or implementation. For methods that could potentially benefit from a descriptive target prompt of the input image, we considered an empty prompt, COCO caption as prompt, and unconditioned generation (no CFG) and reported the best score. Below, we outline the configuration for each method.

LEDITS++ (Ours) & **Edit-friendly DDPM [17]:** Perfect reconstruction for any hyperparameter combination. The error induced by machine precision is inconsequential.

Imagic [18]: 1000 embedding learning steps w/ learning rate $2e{-}3$ and 1500 model tuning steps w/ learning rate $5e-7$. Target prompt is the original image caption. We used 50 generation steps with α and guidance scale 0.0

Pix2Pix-Zero [30]: Inversion with 100 steps, no CFG, $\lambda = 20$ for auto correction and KL divergence, and 5 regularization and auto correction steps, respectively. 100 inference steps with cross-attention guidance 0.0 and no CFG. Source and target embeddings are null-vectors.

DDIM Inversion: 1000 inversion steps and 50 generation steps. Both without classifier-free guidance

SDEdit [26]: 40 diffusion steps (strength 0.8 at 50 default steps) with no CFG.

Notably, the small difference between DDIM and Pix2Pix-Zero only holds for the pure reconstruction of an image. Previous research has shown [28] that for DDIM inversion, the accumulated error increases drastically when using classifier-free guidance. Since CFG is necessary for editing, the "reconstruction" portion during editing becomes worse for DDIM and remains stable for Pix2Pix-Zero.

C.1.2 Runtime

For runtime measurements, we consider the wallclock runtime on a dedicated NVIDIA A100-SXM4-40GB GPU. As a proxy task, we considered applying an *´oilpainting'* style to a photograph. We only measured the inversion and generation loops, discarding any I/O or other processing. For each method, we considered 100 runs with hyperparameters based on the respective paper/official implementation, as outlined below.

LEDITS++ (Ours): 20 inversion and 20 generation steps with threshold $\lambda = 0.1$ and skip for $t = 0.75T$.

Edit-friendly DDPM [17]: 100 inversion steps and 64 generation steps (i.e. 36 skip steps)

Imagic [18]: 1000 text embedding optimization steps, 1500 model finetuning steps, 50 inference steps

Pix2Pix-Zero [30]: 100 steps at inversion and inference. 5 regularization steps and auto correction steps for each inversion step.

DDIM Inversion: 1000 inversion steps (w/o CFG) and 50 generation steps

SDEdit [26]: 40 diffusion steps (strength 0.8 at 50 default steps).

[4] https://huggingface.co/docs/diffusers

An interesting observation is the fact that LEDITS++, runs faster than SDEdit although both perform 40 diffusion steps overall. However, SDEdit requires 80 total U-Net evaluation (unconditioned and conditioned for each step) step, whereas LEDITS++ only requires 60 (unconditioned at each inversion step and unconditioned + conditioned at each inference step). Performing 2 evaluations of the U-Net is significantly slower than 1 evaluation even if performed as a batch.

C.2. Implicit Mask Quality

We have already provided detailed information on the experiment in Sec. 5. For further reference, we note that after removing duplicate/ambiguous objects the dataset contains 4983 images and 29307 segmentation objects.

C.3. Multi-conditioning

The multi-conditioning experiment presented in Sec. 6.1, used the following attributes:

- glasses
- smile
- hat
- wavy hair
- earrings

We used the first 100 images in CelebA that were labeled to not contain any of the five target attributes. Seeds were chosen at random but kept fixed across all experiments. For the LPIPS and CLIP scores, we relied on the default implementation from torchmetrics[5]. Consequently, we used the AlexNet variant with mean reduction and the original ViT-L/14 CLIP checkpoint from OpenAI[6]. For the CLIP scores, we calculated a dedicated score for each of the 3 applied edits and considered the mean for each image.

The hyperparameter variations of each method were run as a grid search over the hyperparameter ranges listed below. Other parameters were kept at their default values. For each method, we ran a grid search over a wider range of parameters to identify reasonable boundaries and subsequently discarded edge values leading to drops in performance.

LEDITS++ (Ours):	Skip between 0.2 and 0.3, Guidance scale between 10.0 and 15.0, Threshold between 0.7 and 0.9
Edit-friendly DDPM [17]:	Skip steps between 20 and 40, Guidance scale between 10.0 and 15.0
DDIM Inversion:	Guidance Scale between 1.0 and 15.0
Imagic [18]:	Guidance Scale between 2 and 6, and α between 0.1 and 1.3
Pix2Pix-Zero [30]:	Guidance Scale between 1.0 and 10.0 and cross guidance scale between 0 and 0.15
SDEdit [26]:	Guidance scale between 5.0 and 10.0 and strength between 0.2 and 0.8

D. TEdBench++

We propose TEdBench++[7], a revised version of TEdBench [18] which sets a new standard for benchmarking real text-based image editing. It is publicly available, including original images, edit instructions, and edited images with LEDITS++ for benchmarking new methods. Figs. 6 and 11 as well as Tab. 2 demonstrated our generated images with LEDITS++ to improve upon the previous SOTA method, Imagic, setting a new reference for benchmarking. Next to providing better-edited images, we also addressed several inconsistencies in the target texts and missing tasks.

We show several inconsistencies of TEdBench in Fig. 9. First, we corrected ambiguous text prompts such as a *standing* animal that is already standing Fig. 9 (top). This applied to multiple images (horse, cat, bear, etc.). Instead, we propose "an {animal} standing on hind legs" to specify the target text and thus ask for a clear but more challenging edit. Second, we correct for misspellings such as "enterance" which should be "entrance" instead. While this may appear negligible, DMs' tokenizers may provide completely different tokens in these cases: e.g., one token for the correct word but three tokens for the misspelled word. In Fig. 9, we show the impact of the corrections on the edit success on LEDITS++. Although we use the exact same parameters (seed, etc.), the edit of the original (left) to the middle fails, whereas it is successful for the corrected prompts (right). This way, we provide a higher-quality benchmark.

Further, we added novel tasks to the benchmark, making it more challenging and accounting for a broader range of tasks. In Fig. 6a, we show examples of the new tasks we added: i) multi-editing (adding multiple concepts at the

[5] https://lightning.ai/docs/torchmetrics/stable/
[6] https://huggingface.co/openai/clip-vit-large-patch14
[7] https://figshare.com/s/7adc2b0fe1e0388dd99f

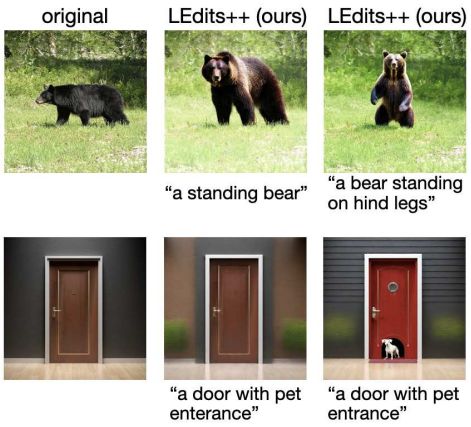


Figure 9. Exemplary inconsistencies in TEdBench and their corrections in TEdBench++. Left is the original image, and in the middle/right are images edited with LEDITS++ for the edit instructions above. All parameters (seed, etc.) are the same. As can be seen, the edit success heavily depends on the clear and correct writing of the words. In the middle column, it does not work (ambiguous (top) and misspelled (bottom)), whereas the edit is successful in the right image (clear and correctly spelled).

same time), ii) object removal (removing an object while staying consistent with the background and overall image composition), iii) style transfer (changing the whole image, i.e. all pixels without changing the overall image composition or object, only their style appearance), and iv) complex replacements (adding and removing multiple concepts at the same time).

With these extensions, we improve on the previous benchmark and propose a higher-quality version. This way, we hope to benefit the research community and set up a new standard for benchmarking text-based real-image editing techniques.

E. User Study

Next to evaluating with automated metrics such as CLIP and LPIPS scores, we also conducted a study with human evaluators. We focus the user study on TEdBench(++). First, we describe the experimental details for generating the images for the study. Then, we describe the setup of the user study.

Experimental details We followed the approach of Kawar *et al.* [18] and generated images for several seeds and hyperparameters and hand-selected the best fitting image (exemplary grid search shown in Fig. 13). Notably, we evaluated only three seeds, whereas Kawar *et al.* evaluated eight seeds. Furthermore, we limited the grid search to a decent but small range for each hyperparameter.

For LEDITS++ (with SD1.5 and SD-XL), we set the number of diffusion steps fixed to 50 steps and

Figure 10. Failure cases of LEDITS++ on TEdBench

grid-searched skip $[0.0, 0.1, 0.2, 0.4]$, masking threshold $[0.6, 0.75, 0.9]$, and guidance scale $[10, 15]$. As a result, we evaluated 72 images (= 3 seeds $\times$ 4 skips $\times$ 3 thresholds $\times$ 2 scales) per benchmark sample. All other hyperparameters correspond to the default values of the diffusers implementation. Consequently, the generated images with LEDITS++ could be even further improved when evaluating for more hyperparameters, e.g. more seeds (also see open question discussion on seed in App. F).

For Imagic with SD1.5, we relied on 3 seeds and 50 diffusion steps, too. We grid-searched the guidance scale $[5.0, 7.5, 10.0]$ and alpha value $[0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.2, 1.4, 1.6, 1.8, 2.0]$. As a result, we evaluated 108 images (= 3 seeds $\times$ 3 scales $\times$ 12 alphas) per benchmark sample. The remaining setup and parameters correspond to the default values of the original Imagic implementation with Imagen [18]. For Imagic with Imagen, we had to rely on their curated outputs of TEdBench since the model is not publicly available.

User study setup For the actual user studies, we chose the following setups (cf. Fig. 12) on the platform thehive.ai. Users had to pass a qualifying test, and during the actual labeling, 15% of the tasks a user saw were honeypot tasks (sanity check). Only if the qualifier test was passed error-free and the honeypot accuracy was permanently above 95%, we accepted the given answers to ensure high-quality evaluations. Users could zoom in/out and change several image parameters, such as brightness and contrast to further enable a high-quality assessment.

The first study (see Fig. 12a), which we also describe in the main text in Tab. 2, evaluates the success rate of an editing technique on TEdBench(++). To this end, we asked users to assess the overall success of edits, i.e., if an edit instruction was faithfully realized for a given input image. Fig. 12a shows the setup, in which a user had to choose between two options, whether the edit instruction has been

(a) Image editing or generating a new image?

(b) Coherence Trade-off: compositional robustness vs. object identity

Figure 11. Comparing failure cases of LEDITS++ and Imagic on TEdBench.

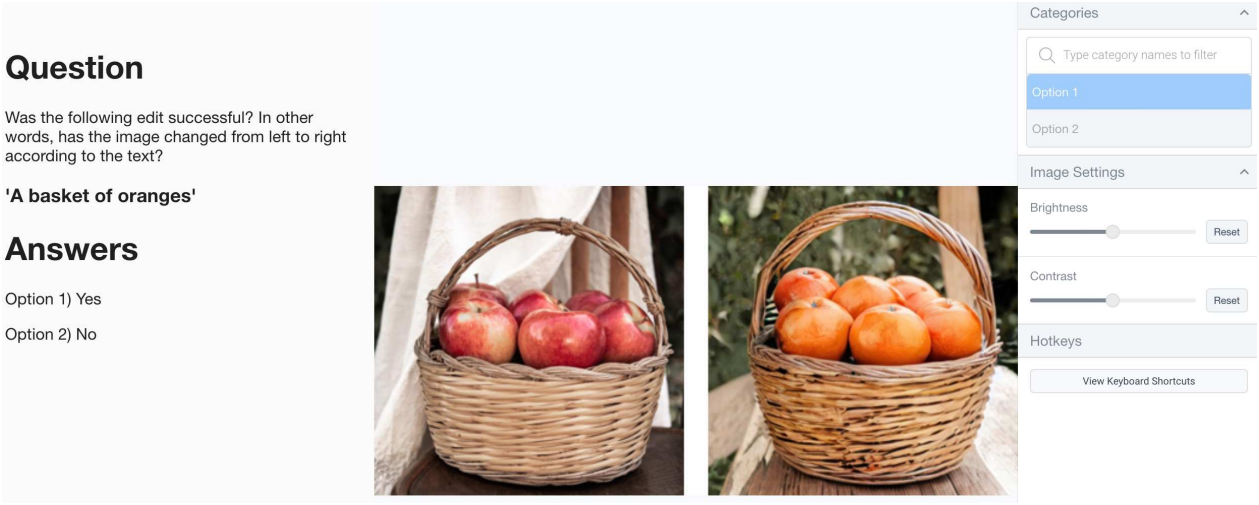
realized successfully or not. The setup consisted of a general question-and-answer setting for all examples alongside a specific edit text for each image pair. The original image (always left) and the edited image (always right) were shown in the center. Outputs from LEDITS++ and Imagic were interleaved at random. The result is given in Tab. 2 in which LEDITS++ clearly outperforms Imagic for both underlying DMs and both versions of the benchmark benchmarks.

We also conducted a second user study. In this study, we asked the user to assess the image similarity of two methods to a reference image, see Fig. 12b. With this human preference study, we investigate the image-to-image similarity after editing, i.e., if the edited images still look similar to the original one. Participants were shown the input image (middle) and were asked to choose the better editing result from one of the two methods (left and right), using the common practice of Two-Alternative Forced Choice (2AFC). The methods were randomly switched between left and right to avoid confounding factors. To this end, we compared LEDITS++ (with SD-XL) and Imagic (with Imagen) on TEdBench. For this comparison, we considered only images where both methods were labeled successful in the prior study. In that study, users preferred LEDITS++ over Imagic with 60% preference. As outlined previously, this again emphasizes the precision of our method, which preserves the overall image composition and results in high-quality edits. Yet, the preference seems smaller than the results with the LPIPS scores. We found this to be an artifact of imprecise user study design. A preference setup generally suffers from bias such as subjects replacing general questions with more specific ones [41] (e.g. “which is more similar?” might be replaced by “in which did the main object stay the same, regardless of the background?” or “in which are the background and overall image composition better preserved regardless of the main object?”). Hence, it is difficult to draw exact conclusions from this study, but a clear trend is still visible. Moreover, as shown in the main text in Tab. 2, we computed LPIPS scores, which further clarify the results of the user study. Additionally, we broadly discussed the similarity trade-off between object identity/coherence and overall image composition in the limitation sections of the main body (Sec. 7) and appendix (App. F).

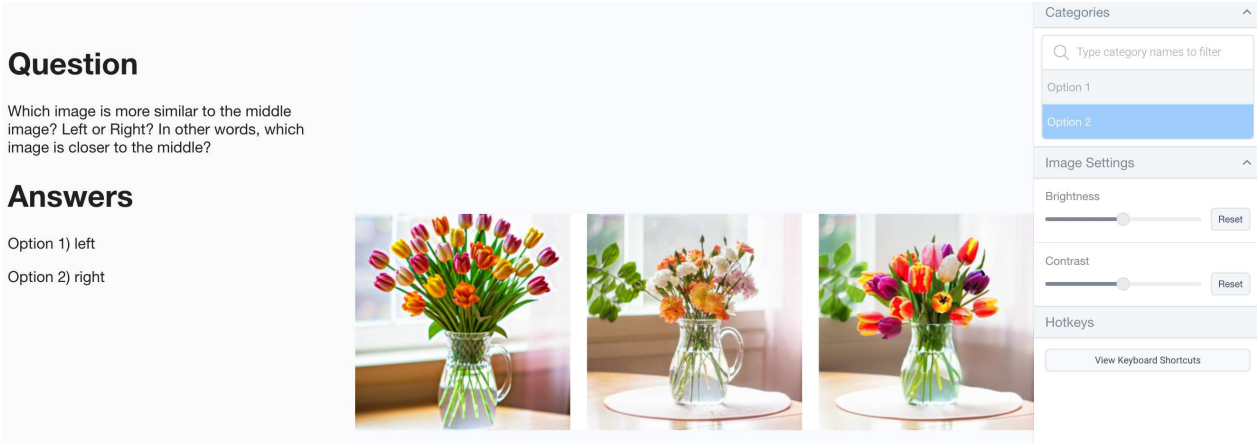
F. Limitations and Further Discussion

In the following, we extend the discussion of the main body with further examples and questions.

Model Dependency. In general, we observed the editing success to be dependent on the underlying DM. In Fig. 15, we show that the generation of a *sitting giraffe* depends on the underlying DM. For both editing techniques, the weaker SD1.5 variant fails but the more advanced variant succeeds. Upon further investigation, we realized that SD1.5 is incapable at all of generating images of *sitting* giraffes. In Fig. 16, we show exemplary images for the text prompt “a sitting giraffe” (we generated 100 and all showed the



(a) Setup for user study: "was the editing successful?"



(b) Setup for user preference study: "which edited image is closer to the original image?"

Figure 12. User study setups for both user studies conducted. The first user study evaluates the edit success of an image editing method. The second user study evaluates the user preference between two image editing methods regarding image-to-image similarity.

same result) and can see that none is actually sitting. In contrast, SD-XL is able to output images of *sitting* giraffes (cf. Fig. 17) and consequently enables LEDITS++ to perform the desired edit. This emphasizes the importance of choosing an apt underlying DM for real image editing and motivates future research to develop more powerful DMs from which editing techniques will benefit, too.

Failure Cases and Open Questions. In the following, we want to touch upon open questions and failure cases. In Fig. 10, we show several failure cases of LEDITS++. In the first case, the cat is indeed edited from a sitting to a standing cat. Yet, the identity of the cat has changed, i.e., the shape of the tail and the fur color have changed. We show further examples in Fig. 11. However, defining what makes an edit *acceptable* remains challenging and may differ between users, applications, and context. In general, however, there are two reasons for the discussed limitations. First, LEDITS++ limits its edits to the identified relevant regions. Consequently, the background and image composition will be preserved, but the edit within this region depends on various factors, including hyperparameter strength, underlying DM, and random seed. Second, a lack of descriptiveness of the editing prompt with respect to the specific identity of an object can lead to changes thereof. Especially if generic terms such as 'cat' are used to edit the image. To guarantee the preservation of the object's identity, methods like Textual Inversion [12], or Break-a-Scene [1] could be employed.

The other three examples in Fig. 10 show failure cases of LEDITS++beyond changes of object identity, i.e., cases in which the edit instruction is not or not sufficiently realized. There are several reasons for such failures, including a general lack of concept understanding in the under-

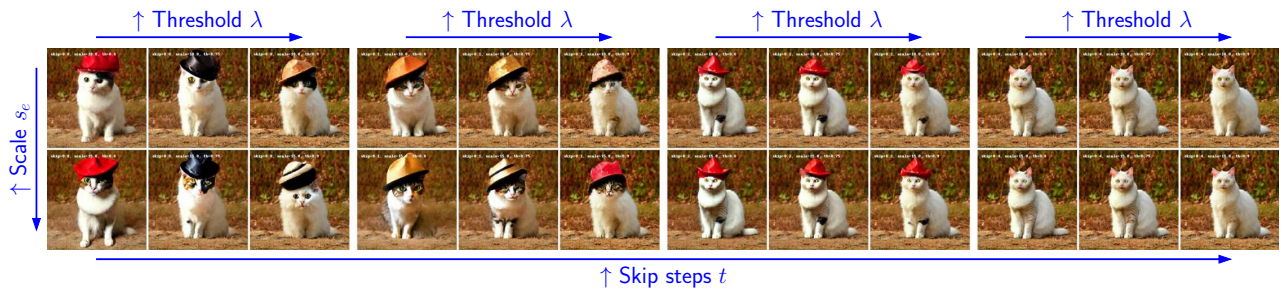


Figure 13. Grid search results for LEDITS++ for TEdBench(++). We show the grid search for the image "cat.jpeg" from the benchmark and the target text "a cat wearing a hat". From left to right the parameters are increased. We searched three parameters, the skip steps, the guidance scale, and the masking threshold. As can be seen, the stronger the parameters, the more changes are made to the image. On the other hand, too weak parameters do not change the image, i.e. do not realize the target text. This highlights the trade-off between edit success and preservation of the image composition and object identity. All images are generated for the same seed (for TEdBench(++) we evaluated 3 seeds per benchmark entry, whereas Imagic used 8 seeds).

lying DM (discussed above), incorrect masking of the relevant region, wrong hyperparameter choices, or challenging prompts. For example, what exactly is a "breakdancing dog" supposed to look like? We even considered removing this entry from the benchmark but found it very challenging at the same time and, therefore, kept it. Moreover, we found the edit success rate and quality to depend on the used seed. This is in line with current work on the impact of the used seed on the diffusion process [38]. Samuel *et al.* propose a new method to identify fitting seeds, which could be applied to LEDITS++ as well to find satisfying editing seeds.

Masking and User Interaction The automatically-inferred implicit masks allow for easy use of LEDITS++ without users tediously providing masks. Nonetheless, user intentions are diverse and cannot always be automatically inferred. Sometimes, individual user masks provide better control over the editing process. Such user masks can be easily integrated into LEDITS++. In Fig. 14, we show customized image editing with individual user masking. LEDITS++ can be easily extended with user masks to facilitate user preferences. Sometimes LEDITS++'s implicit masks do not meet user preferences or it is difficult to textually describe the relevant image region. Next to using dedicated models to obtain image masks, users can simply provide their own masks. In our setup, we did not evaluate this scenario as it drastically increases resources in terms of computation or human labor. Yet, it can be easily integrated into LEDITS++. Fig. 14 shows the original image can be edited well with LEDITS++. Moreover, dedicated user masks help focus on a specific image region. This is specifically helpful for logical and compositional instructions (current models struggle with "left"/"right" etc.), for one specific object if multiple are present (one specific "orange" from several ones), and for both combined. This way, LEDITS++ stays lightweight in its default with implicit masking, but can still and easily handle user masks and thereby implement individual user experience.

G. Further Results

Subsequently, we present further results, qualitative examples, and visual ablations.

G.1. Qualitative examples

We show further results in Fig. 21. We remove "cat" and add a diverse set of animals instead. Interestingly, this works for a variety of animals, that share no or only little similarity, such as "flamingo" or "parrot". Furthermore, the newly occurring background is inpainted semantically sensible, too. Additionally, we show more qualitative examples in Fig. 22.

G.2. Ablations

In Fig. 13, we show an ablation of LEDITS++ for TEdBench(++). This grid search illustrates the impact of different hyperparameters on the trade-off between edit success and the preservation of the overall image composition/ object identity.

G.3. Semantic Grounding Ablations

We performed extensive ablations on semantic grounding by re-running LEDITS++ on Sec. 5's benchmark without any grounding. The results in Fig. 18 show that LEDITS++ will still achieve strong instruction alignment (high CLIP score) without grounding, but semantic masking is key to keeping the generated image similar to the input (low LPIPS score). Moreover, grounding allows for a clearer trade-off between instruction alignment and image similarity in the first place. We believe these ablations foster a deeper understanding of the importance of semantic grounding in the LEDITS++ pipeline.

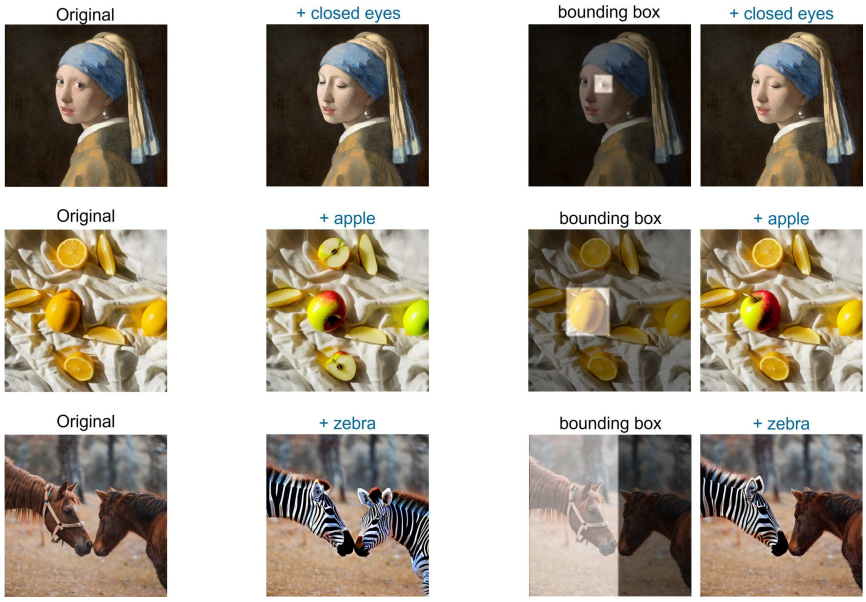


Figure 14. Customized and complex image editing with LEDITS++ for individual user masking. LEDITS++ can be easily extended with user masks to facilitate user preferences. The first column shows the original images and the second and fourth show the edited images with LEDITS++ and the target text above. The third column shows user-provided masks, marking the relevant region for the edit instruction. In the second column, LEDITS++ uses implicit masking as implemented in our default approach, and in the fourth column LEDITS++ uses the explicit user-provided masks from the third column.

Figure 15. Impact of underlying DM. The original image (leftmost) should be edited with the target text "a photo of a sitting giraffe". The edit success depends on the underlying DM: with SD1.5 it fails whereas it works fine with more advanced DMs (SD-XL and Imagen). This holds for both methods LEDITS++ and Imagic.

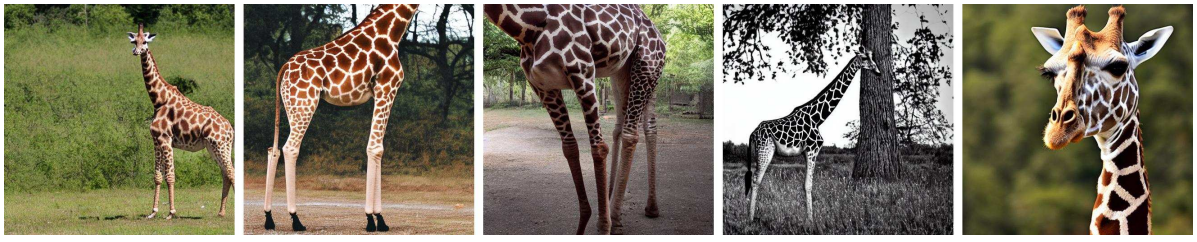


Figure 16. Generated Images with SD1.5 for "a photo of a sitting giraffe". The model consistently fails to generate a giraffe in that specific pose.

G.4. Explanatory Visualization of Monotonicity

The monotonicity of the LEDITS++ guidance scale is an important contribution of the method. Importantly, the inferred masks are mostly isolated from changes to the guidance scale. In the example shown in Fig. 19, we would expect the masks for '*smiling*' to always target the area around the mouth and eyes. Within these identified regions,

Figure 17. Generated Images with SDXL for “a photo of a sitting giraffe”.

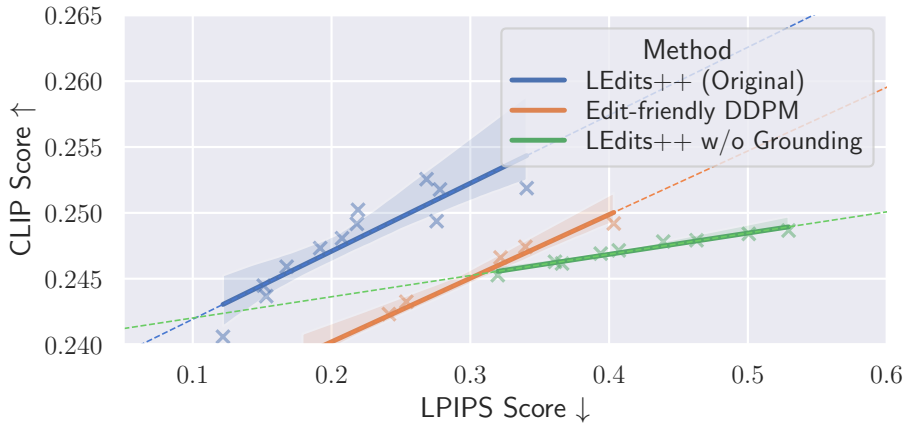


Figure 18. Semantic Grounding Ablations. Semantic grounding is an essential component of LEDITS++ that helps preserving overall image composition and realizing concise edit instructions. (Best viewed in color)

the magnitude of applied changes correlates directly with the changing scale, as evident from the provided heatmap. Here, we can also observe that different areas are prioritized depending on the magnitude of the change. The initial focus is clearly on the mouth, with strong changes in the eyes only appearing for larger scales.

G.5. Masking and Artifacts

LEDITS++ can faithfully edit reflections/shadows of objects, even with masking (cf. Fig. 1 & 20). This ability strongly depends on the underlying diffusion model. In the example in Fig. 6b, the underlying diffusion model simply failed to correctly correlate the couple and their shadow/reflection. However, in Fig. 20 where SDXL is applied, the reflection is edited as well.

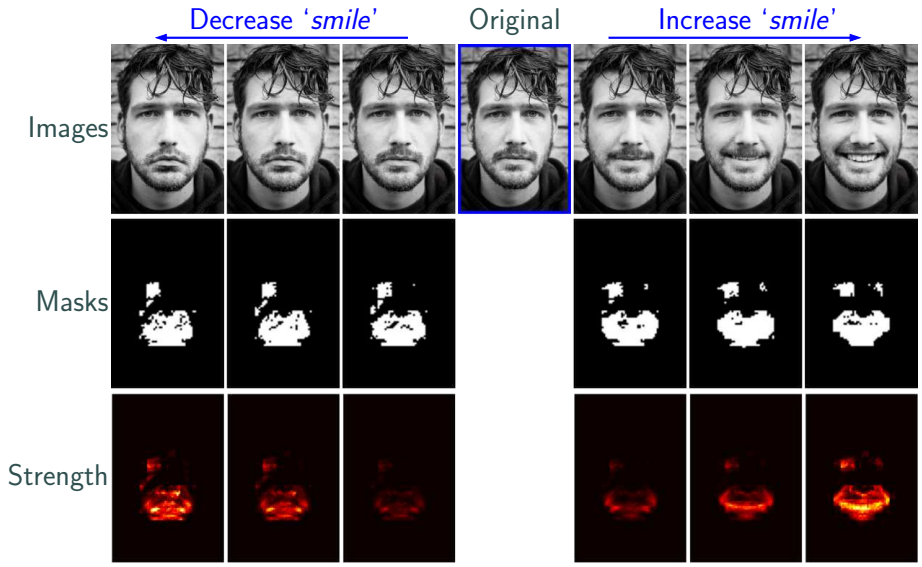


Figure 19. Masking and guidance scale. Within the identified edit regions, the magnitude of applied changes correlates directly with the changing scale. (Best viewed in color)

Figure 20. LEDITS++ can easily handle complex edits such as reflections. (Best viewed in color)

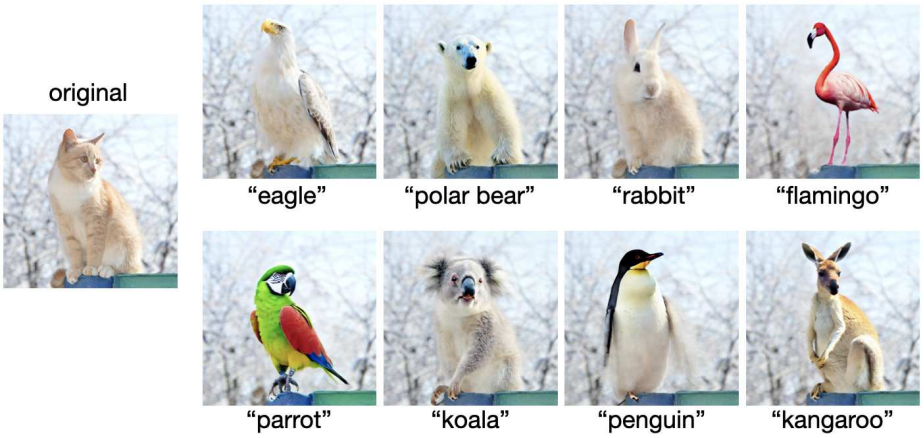


Figure 21. Object replacement with LEDITS++. The leftmost image is the original image and the other images are edited with LEDITS++ and the target text below. We apply diverse replacements of the main object with the overall image composition being preserved. Interestingly, the background is filled and interpolated very well, e.g. for "flamingo" or "parrot". (Best viewed in color)

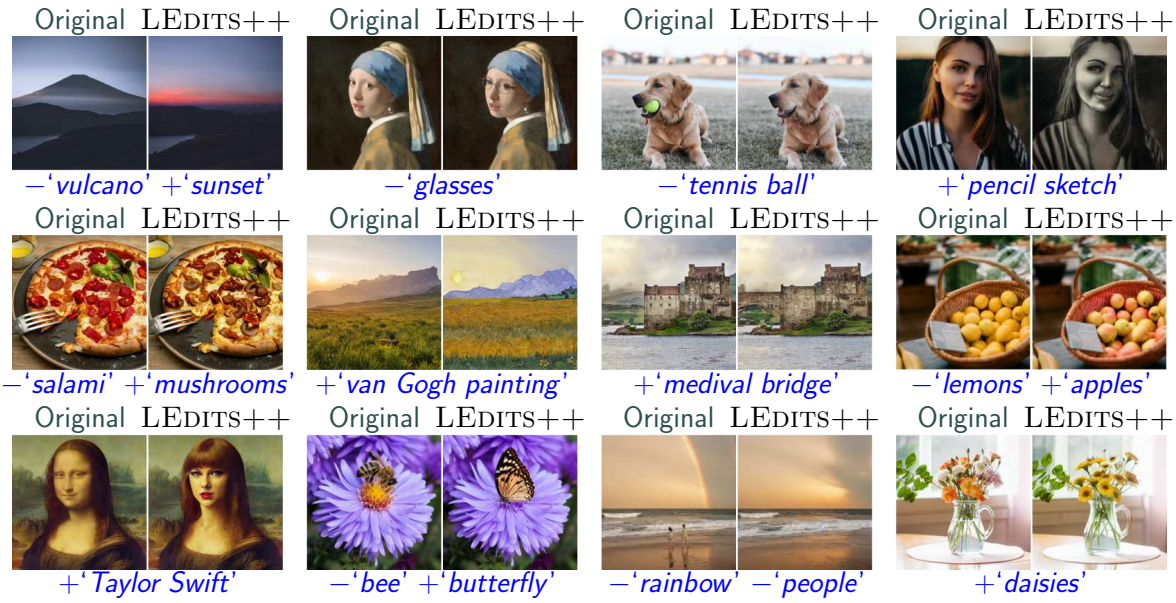


Figure 22. Further qualitative examples of image editing with LEDITS++, highlighting its versatility and precision (Best viewed in color)