Federal State Autonomous Educational Institution for Higher Education
National Research University Higher School of Economics

Faculty of Computer Science
Applied Mathematics and Information Science

BACHELOR'S THESIS

Research project

"Analysing Contemporary Diffusion Models for Generating Images"

Prepared by the student of group 193, 4th year of study,
Khalmatova Madina Ikramovna

Supervisor:
Lecturer, Doctoral Student, Alanov Aibek

Moscow 2023

Contents

Abstract

Diffusion models have recently demonstrated powerful synthesis capabilities for various types of data, including images. Producing highly realistic samples and introducing a guidance mechanism to control the generation process, these models have initiated new areas of research. The latter includes an ongoing field of study focused on utilizing pre-trained diffusion models to modify real images with consideration to a given textual prompt. Although existing methods have shown considerable results, they still require thorough investigation. In this paper, we analyse a recently introduced approach, Null-Text Inversion for Editing Real Images using Guided Diffusion Models, and propose significant improvements, targeting its challenging domains.

Аннотация

Диффузионные модели показали себя как передовой инструмент генерации различных данных, в том числе изображений. Создавая высокореалистичные примеры и позволяя контролировать процесс генерации с помощью механизма “guidance”, эти модели положили начало новым областям исследований. Последнее включает в себя использование предобученных диффузионных моделей для редактирования реальных изображений с учетом заданной текстовой подсказки. Несмотря на то, что существующие методы показывают многообещающие результаты, они все еще требуют тщательного исследования и доработки. В этой статье мы анализируем недавно представленный подход, Null-Text Inversion for Editing Real Images using Guided Diffusion Models, и предлагаем способы улучшить этот метод, решая его ключевые проблемы.

Keywords

Diffusion Models, Image Generation, Guidance Mechanism, Image Editing

1 Introduction

Diffusion models for image synthesis are capable of producing high quality samples by breaking down the generation process into sequential denoising steps (Ho, Jain, and Abbeel 2020). The formal definition of their generation process allows an unprecedented level of control by introducing guidance (Ho and Salimans 2021) and conditioning (Rombach et al. 2022) mechanisms.

As exceptional results in image generation have already been achieved (Rombach et al. 2022; Saharia et al. 2022; Ramesh et al. 2022), new areas of research have emerged, focusing on utilizing pre-trained diffusion models to modify or enhance real or generated images, instead of improving the quality of the generation.

A recently introduced method (Mokady et al. 2022) leverages Prompt-to-Prompt (Hertz et al. 2022) text-guided editing of real images. Given a real image and a corresponding textual prompt as input, the method obtains a diffusion trajectory by carefully inverting the image with DDIM inversion(Song, Meng, and Ermon 2021), which is then accurately reconstructed and edited using the Prompt-to-Prompt (Hertz et al. 2022) technique to match the target caption. This approach allows countless modifications to be made to a single image while maintaining its key features and realism.

The authors investigate a trade-off between preserving input image details and retaining high editability. Although their method outperforms other editing approaches, it still faces a number of obstacles, as it is strongly affected by the limitations of the underlying diffusion model and editing approach. As the authors claim, while StableDiffusion (Rombach et al. 2022) can produce artefacts when dealing with human faces and sometimes fails to associate words with proper regions of the image, Prompt-to-Prompt significantly narrows down the capabilities of the approach, as it does not allow complicated structural modifications.

In this work, we propose an editing approach, based on Null-text Inversion (Mokady et al. 2022). By performing editing by finetuning null-text embeddings with correspondence to the edited prompt, we eliminate Prompt-to-Prompt from

the editing pipeline and overcome the challenges caused by this technique.

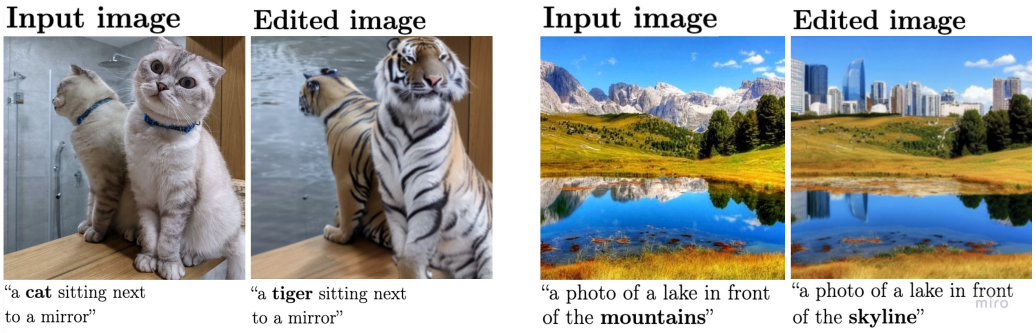


Figure 1.1: Examples of editing with a proposed method

2 Related work

2.1 Diffusion models

Generation is the process of obtaining a data sample from pure noise. Since this process requires knowledge of the data distribution, which is usually very complicated and impossible to estimate accurately, training a neural network to directly model a generation process is unstable and difficult to perform. However, the reverse process of transforming a data point into a noise sample is much clearer and often allows formal definition.

Diffusion models were introduced in (Ho, Jain, and Abbeel 2020). They break down the synthesis process into sequential steps, where each step denoises the result of the previous step by a small factor. The diffusion models learn to sample from a data distribution implicitly by learning a gradual noising process.

More formally, we fix a noise distribution as Standard Gaussian, $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Given an input image x_0, we turn it into a noise sample $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ by sequentially perturbing it T times, i.e. adding some Gaussian noise. The size of each step is controlled by the variance schedule $\{\beta_t\}_{t=1}^{T}$, where $\beta_t = \frac{(0.02-0.0001)\cdot t}{T}$:

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.1}$$

Let $\alpha_t = 1 - \beta_t$, and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. The Gaussian distribution is a stable

distribution, meaning that the sum of two independent Gaussian random variables is also Gaussian. Applying this property to (2.1), we obtain a closed form for noising an input image for t steps:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t \tag{2.2}$$

The diffusion model is trained to learn how much noise has been added at each step t. In a single training step, given an input image x_0, we sample a step number t, obtain a noisy image x_t, and minimise the Mean Square Error between the added noise and the prediction of the model:

$$\|\epsilon_t - \text{model}_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon_t, t)\|_2^2 \to \min_\theta \tag{2.3}$$

The chosen variance schedule ensures that we add a small amount of Gaussian noise at each step. This property guarantees that by learning a process of adding noise to the data, the model implicitly learns how to denoise a sample from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ into a realistic image, thus gaining image generation capabilities. Fig. 2.1 illustrates this idea.

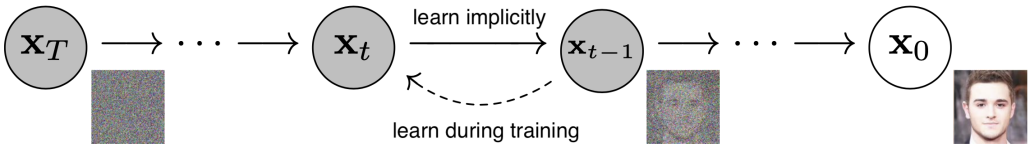


Figure 2.1: Illustration of diffusion model.

2.2 Latent Space Diffusion Models

While diffusion models set a new state-of-the-art result in generation quality, they take a long time to synthesise a single sample, which is a serious obstacle on the way to real-time applications. This problem was addressed in (Rombach et al. 2022; Vahdat, Kreis, and Kautz 2021), where the authors proposed to apply diffusion models in a much smaller latent space instead of a high-dimensional data space.

To obtain such a space, we use a powerful pre-trained Variational Autoencoder (Kingma and Welling 2014). The Variational Autoencoder is a neural network trained to map input data into a low-dimensional latent space containing information about the semantics of the data. The network consists of two parts, the encoder and the decoder. The encoder maps a given input to a corresponding distribution over the latent space, while the decoder aims to reconstruct the original input from the sample from that distribution. The model is trained to minimise the reconstruction error (the difference between the original and reconstructed inputs) with a regularization term, defined through Kulback-Leibler divergence. Fig. 2.2 illustrates this method.

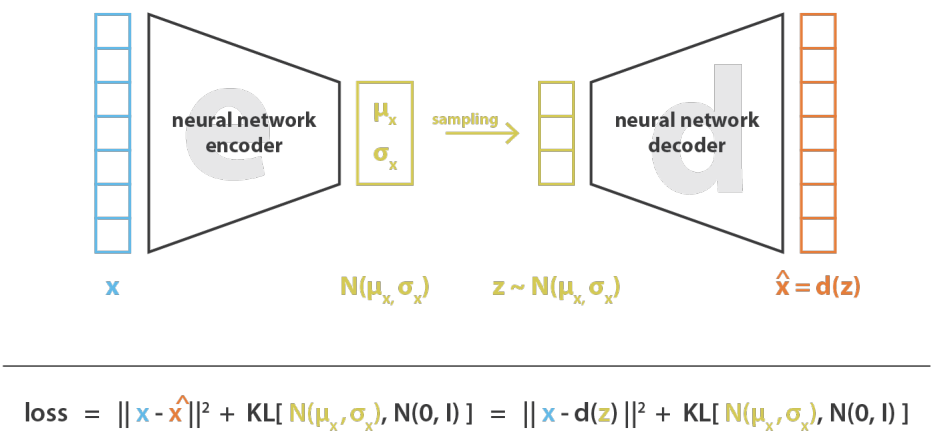


Figure 2.2: Illustration of variational autoencoder architecture and training.

2.3 Conditioning and Classifier-free Guidance

It is common for the input data to consist of images paired with some additional information, for example class labels or corresponding textual descriptions. We want to train a diffusion model to generate samples similar to those in the dataset. However, the original diffusion model only needs images to train, leaving the informative captions and labels out-of-use.

To utilize this information, we condition our diffusion model on that data. To condition the model on the text, given the output of the previous diffusion synthesis step and a corresponding caption, we run the caption through a textual

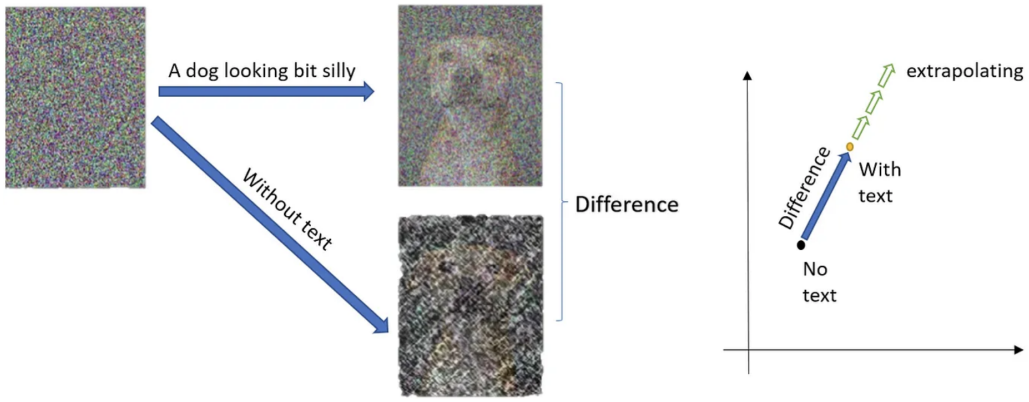


Figure 2.3: Illustration of classifier-free guidance.

encoder and obtain its vector representation, which is then concatenated with the noisy image. This new object is used as input to the diffusion model in the current step.

Classifier-free guidance (Ho and Salimans 2021) further enhances the influence of the additional data. With a trained diffusion model, we obtain both conditional and unconditional samples. The unconditional sample is then extrapolated in the direction of the conditional one, as described in Fig. 2.3.

2.4 Null-Text Inversion

A method of editing real images, proposed by (Mokady et al. 2022), relies on two key concepts: DDIM inversion and classifier-free guidance. Given an image z_0 and an initial prompt, describing this image, we obtain a diffusion trajectory $z_T \to \ldots \to z_0$ with DDIM inversion (Song, Meng, and Ermon 2021), where a diffusion model is conditioned on the text:

$$z_{t+1} = \sqrt{\frac{\alpha_{t+1}}{\alpha_t}} z_t + \left(\sqrt{\frac{1}{\alpha_{t+1}} - 1} - \sqrt{\frac{1}{\alpha_t} - 1} \right) \times \text{model}_\theta(z_t, t, \text{initial prompt}) \quad (2.4)$$

When editing, we give z_T, obtained in (2.4), as input to a diffusion model generation process and apply classifier-free guidance, guiding with a new prompt describing the changes we want to make to the original image.

However, DDIM inversion introduces a small error at each step This means

that an image generated from z_T with a diffusion model without any guidance will not be an accurate reconstruction of the original image. The editing also suffers, as classifier-free guidance only enhances the effect of the error.

To address this issue, the authors introduce trainable null-text embeddings $\{\varnothing_t\}_{t=1}^T$, vector representations of an empty prompt in each diffusion step, which are used in classifier-free guidance for unconditional generation. During training, we start with z_T and apply classifier-free guidance to generate z_{T-1}, using an initial prompt and a null text embedding $\varnothing_T$. This way we obtain a representation of the noisy original image z^*_{T-1}, which we move closer to z_{T-1} from (2.4) by minimising the Mean Square Error. The same procedure is applied sequentially to each step $t \in \{T, T-1, \ldots, 1\}$:

$$\|z_{t-1} - z^*_{t-1}\|_2^2 \to \min_{\varnothing_t} \tag{2.5}$$

Trained null-text embeddings are then used in classifier-free guidance when editing an image with Prompt-to-Prompt (Hertz et al. 2022). Fig. 2.4 illustrates this method.

Figure 2.4: Illustration of editing with Null-text Inversion.

2.5 Prompt-to-Prompt

Proposed in (Hertz et al. 2022), Prompt-to-Prompt is an editing approach, applicable to generated images only.

Cross-attention maps in text conditioned diffusion models contain information about semantic relations between image pixels and tokens. Based on this finding, the authors suggest injecting cross-attention maps of the source image into the generation process of the edited image.

Given initial prompt P and edited prompt P^*, we start the editing process by sampling noise $z_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and performing a denoising step with P starting from z_T, obtaining z_{T-1} and cross-attention map M_T. We then obtain the cross-attention map for the edited image generation, M_T^*, by performing a denoising step with P^* starting from $z_T^* = z_T$.

Generating images from the same noise sample with different text prompts will result in very different data samples. However, when editing, we want to preserve key features of the source image. To achieve this, the authors of Prompt-to-Prompt introduce cross-attention injection. More specifically, we want to inject some information from M_t into M_t^* and use a new attention map, $\hat{M}_t^*$, when generating an edited image. Several ideas have been suggested, such as replacing M_t^* completely with M_t in several denoising steps, or modifying only certain rows and columns of M_t^*.

When $\hat{M}_T^*$ is estimated, we do another denoising step starting at z_T^*, with prompt P^* and attention map $\hat{M}_T^*$, and arrive at z_{T-1}^*. By repeating the same process sequentially for $\{T-1, \ldots, 1\}$ we end up with a source image z_0 and its edited version z_0^*.

Although this method shows promising results, it still faces some challenges. A user has to come up with a set of two prompts that differ from each other in a specific way (e.g. by substituting a word or adding a few consecutive words). This significantly limits the editing capabilities of the method. In addition, Prompt-to-Prompt works with low-resolution cross-attention maps, which means that precise editing, such as adding or changing a small detail of the image, is difficult to perform accurately.

2.6 Imagic: Text-Based Real Image Editing with Diffusion Models

A different editing approach has been proposed in (Kawar et al. 2023). The authors divide the editing process into three steps: text embedding optimization,

model finetuning, and generation of the edited image with interpolation between text embeddings.

Given an initial real image and a prompt describing an edited version of it, we first obtain a textual embedding of the prompt, e_{tgt}. Using the initial image to obtain intermediate noisy samples $\{x_t\}_{t=1}^{T}$ and a pre-trained frozen diffusion model, we optimize the embedding $e_{opt} := e_{tgt}$ using a denoising objective, as in diffusion model training:

$$\|\varepsilon - \text{model}_\theta(x_t, t, e_{opt})\|_2^2 \to \min_{e_{opt}} \tag{2.6}$$

As a result, e_{opt} should correspond to an unedited version of the image. In practice, however, an image generated with e_{opt} is not an accurate reconstruction of the original one. To address this issue, the authors suggest finetuning the diffusion model on a single example, the initial image, using e_{opt} as a text embedding.

After finetuning the model, we can generate an edited image by interpolating between two embeddings, $e_{new} = \eta \cdot e_{tgt} + (1 - \eta)e_{opt}$, $\eta \in [0, 1]$. Fig. 2.5 shows an example of editing using the described approach.

Figure 2.5: Illustration of editing with Imagic.

3 Proposed approaches

We chose Null-text Inversion with Prompt-to-Prompt (Mokady et al. 2022) as the base approach for our work. By preserving the main features of the initial image very accurately, this method seems to be a promising step towards a general-purpose editing framework.

3.1 Null-text Inversion with Learned Embedding

One proposed approach is to combine null-text inversion with Imagic (Kawar et al. 2023). By exploiting the latter's idea of optimizing an embedding of the edited prompt, we aim to eliminate the need to construct a prompt describing an unedited image, thus achieving a more user-friendly pipeline. At the same time, we claim that null-text inversion can serve as a replacement for Imagic's model fine-tuning step, meaning that we can use interpolation between initial and optimized embeddings at the generation stage to make the editing process more flexible.

Given an initial image and a textual prompt describing an edited image, we first obtain a latent representation of the initial image, x_0, with a VAE encoder from a pre-trained StableDiffusion model. With a pre-trained textual encoder from the same model, we obtain e_{tgt}, an embedding of the edited prompt. As proposed by the authors of Imagic (Kawar et al. 2023), we optimize an embedding $e_{opt} := e_{tgt}$ by minimising a denoising objective. We apply regularization, which is a crucial step, as discussed in Experiment 4.2.2:

$$\|\varepsilon - \text{model}_\theta(x_t, t, e_{opt})\|_2^2 + \lambda \cdot \|e_{opt} - e_{tgt}\|_2^2 \to \min_{e_{opt}} \tag{3.1}$$

When e_{opt} is optimized, we perform DDIM inversion on the initial image and train null text embeddings, using e_{opt} as the textual embedding for conditional generation.

Finally, to perform editing, we apply classifier-free guidance using interpolation between two textual embeddings, $e_{new} = \eta \cdot e_{tgt} + (1-\eta) e_{opt}$, as an embedding for conditional generation, and learned null-text embeddings for unconditional generation, starting from the noise sample x_T obtained by DDIM inversion.

3.2 Null-text Inversion with Null-text Embeddings Fine-tuning

Another approach is to use the capabilities of learned null-text embeddings to perform editing. To edit an image, we replace prompt-to-prompt with finetuned null-text embeddings, using an initial image and an edited prompt. This way we eliminate the limitations imposed by the Prompt-to-Prompt method.

Given an initial image, an initial prompt P describing it, and an edited prompt P^*, we first perform Null-text Inversion using the initial image and prompt P.

Let x_0 be a latent representation of the input image. With DDIM inversion defined in (2.4), we obtain a diffusion trajectory of the input image, $\{x_T, \ldots, x_0\}$. For each timestep $t \in \{T, T-1, \ldots, 1\}$, we sequentially train a null-text embedding $\varnothing_t$ by minimizing the difference between x_{t-1} and x^*_{t-1} obtained with a single denoising step starting from x^*_t (where $x^*_T := x_T$). We apply classifier-free guidance, using an embedding of P for conditional generation and $\varnothing_t$ for unconditional generation.

To perform editing, we run Null-text Inversion again on the same image, using the prompt P^* and initializing the null-text embeddings with the values learned in the previous step. The more null-text embeddings we fine-tune, the closer the generated sample will be to the original image. In most cases, it is sufficient to fine-tune the first 20-30 null-text embeddings to obtain a sample that preserves the main features of the initial image and at the same time matches the edited prompt. It is worth mentioning that in order to achieve resemblance with the original image, each null-text embedding has to be finetuned for about 30 steps, which is three times more than when training these embeddings from scratch.

4 Experiments

4.1 Experimental setup

The aim of all the experiments is to investigate the strengths and weaknesses of the editing methods and to compare their capabilities. Each experiment uses a **Stable-Diffusion-v1-4** checkpoint of a pre-trained StableDiffusion model from the Diffusers library. For editing methods, we rely on the source code provided by the authors, if available. All experiments require about 20-30 GB of RAM.

4.2 Experimental evaluation

4.2.1 Null-text Inversion with Prompt-to-Prompt

In our experiments we aim at studying the following editing tasks:

- apply changes to the main object
- add something to the picture
- change the background
- change the style of the picture (for example, make it resemble a Picasso drawing)

Applying changes to the main object

For the first editing task we considered three images: an image of a cat sitting next to a mirror; an image of a man with a donut (these two images were considered by the authors of Null-Text Inversion, our aim was to reproduce their results); an image of a full face. The results of editing are shown in Fig. 4.1.

We can see that the method struggles a little with human faces. Although it makes the requested changes well enough to actually match the edited prompt, the features of the face are not exactly preserved, which is particularly evident in Fig. 4.1b. However, the resulting examples are quite realistic.

(a) A picture of a cat

(b) A picture of a man in the park

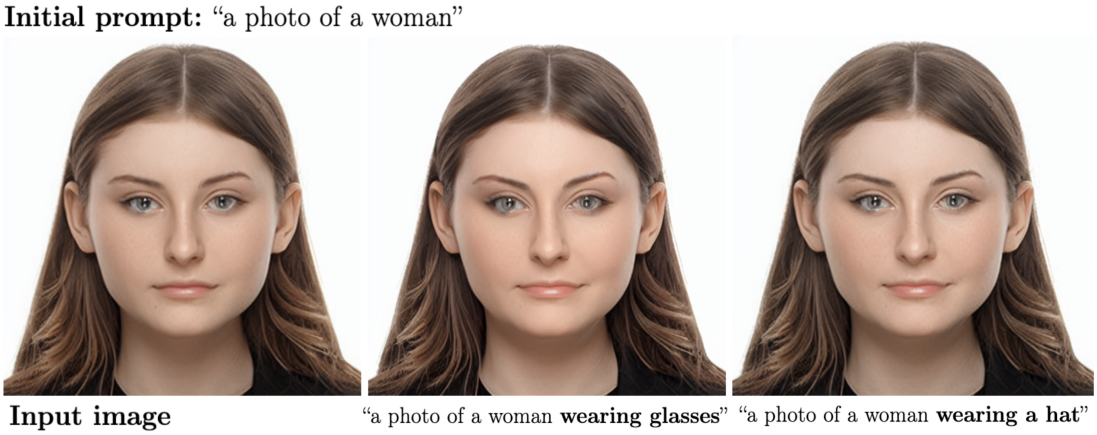
(c) A picture of a full face

Figure 4.1: Examples of applying changes to the main objects

Adding objects

In Fig. 4.2 we present examples of modifying an image by adding something new to it. While a picture of a landscape was successfully modified (Fig. 4.2a), the picture of the face was not modified (Fig. 4.2b). One of the possible reasons for the latter could be the difficulty the method has in dealing with human faces. Besides, this particular example may require more careful prompt engineering.

(a) A picture of a landscape



(b) A picture of a full face

Figure 4.2: Examples of adding objects to the picture

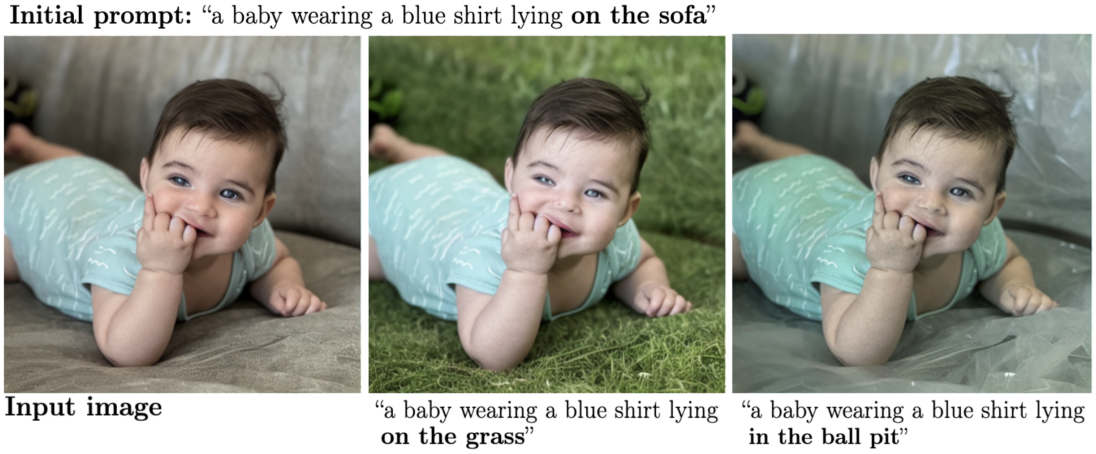
Changing the background

Modifying the background is a challenging task for this method as it tends to retain most of the objects from the original image. In Fig. 4.3a the trees from the park are present both in the desert and by the sea. An picture of the baby from Fig. 4.3b was used by the authors of Null-Text Inversion, but we were unable to reproduce their results, especially when turning the sofa into the ball pit.

Different styles

The various experiments with changing the style of the image were motivated by the success of the method in turning a photograph into a watercolour painting. However, Fig. 4.4 shows that the method struggles with a number of styles, such as anime or an artist's painting. We hypothesise that these problems are a direct

(a) A picture of a man in the park



(b) A picture of a baby

Figure 4.3: Examples of changing the background

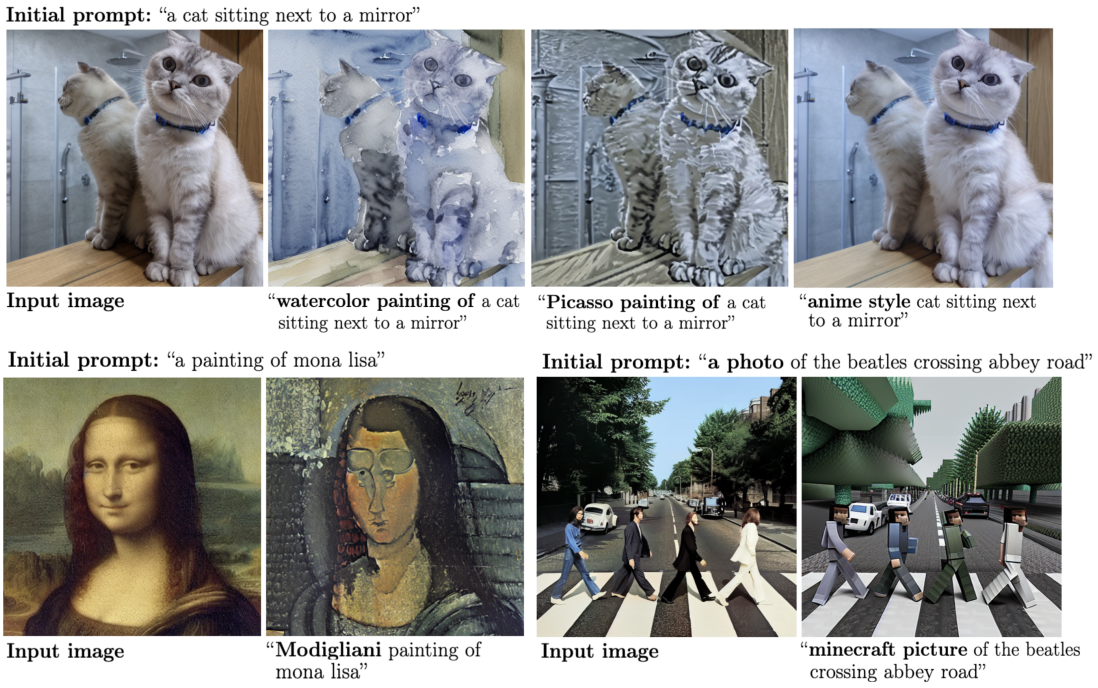


Figure 4.4: Examples of applying style changes

result of the limitations of StableDiffusion. However, this hypothesis remains to be tested.

4.2.2 Null-text Inversion with Learned Embedding

Preliminary Experiments

While conducting the experiments in the previous section, we noticed that Null-text Inversion successfully reconstructs the input image regardless of the exact structure of the initial prompt.

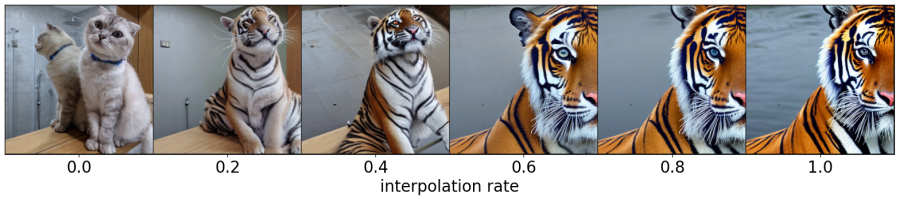
To exploit this finding, we decided to remove the initial prompt from the editing pipeline and replace it with an optimised embedding, as proposed in Imagic (Kawar et al. 2023). This way, we achieve a more user-friendly editing method where only one textual prompt is required (instead of two prompts as in the base approach).

Given two textual embeddings, we perform Null-text Inversion and obtain learned null-text embeddings and a diffusion trajectory.

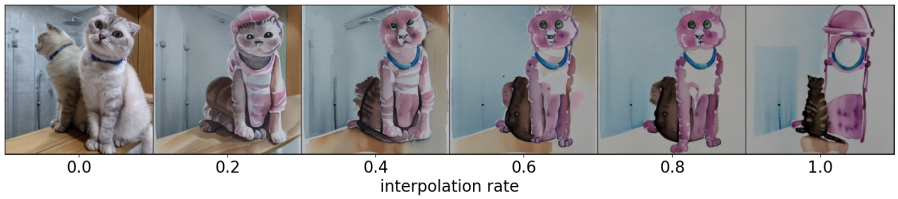
To generate an edited image, we use interpolation between initial and optimized text embeddings for conditional generation, and learned null-text embeddings for unconditional generation.

Fig. 4.5 presents examples of editing with a described approach for different interpolation coefficients. In order to obtain an image similar to the original one, the interpolation coefficient must be very small, approximately equal to 0.2. However, for some examples, even small interpolation coefficients lead to a noticeable difference between an edited and an original image. As fig. 4.6 shows, even small fluctuations in this coefficient tend to have a drastic effect on the final result, meaning that the editing process is unstable.

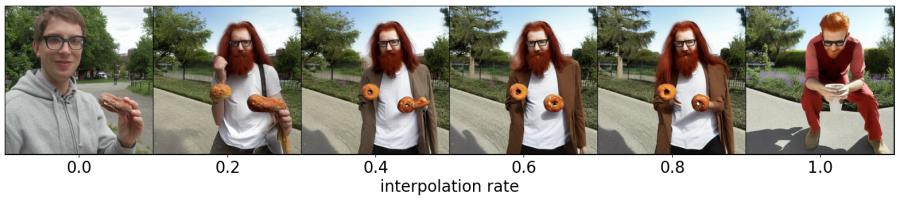
However, it is worth mentioning that this approach shows great results when applied to a full face photograph (Fig. 4.6c). Besides, it successfully adds an object to an image (Fig. 4.6c), which was a challenging task for the base approach.



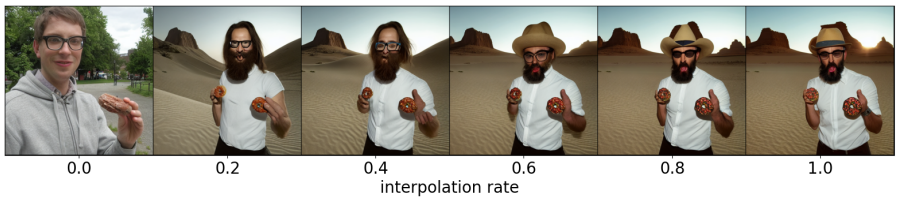
(a) cat → tiger



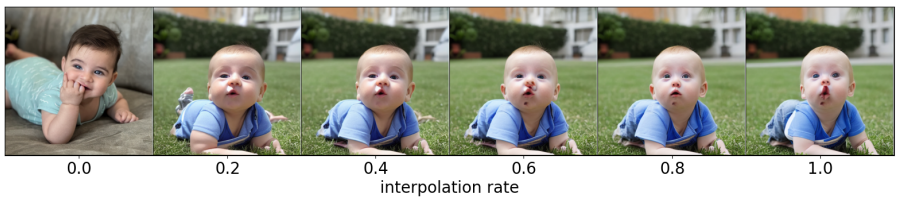
(b) turn into watercolor painting



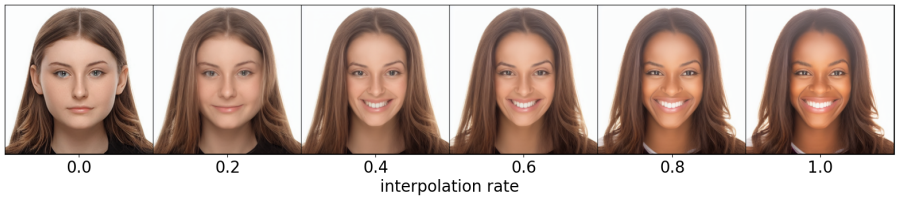
(c) make hair read



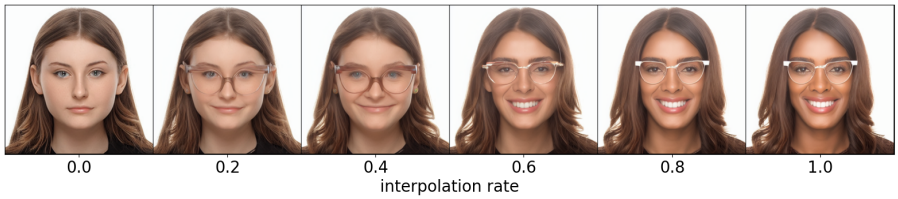
(d) change the background into desert



(e) change the background into grass

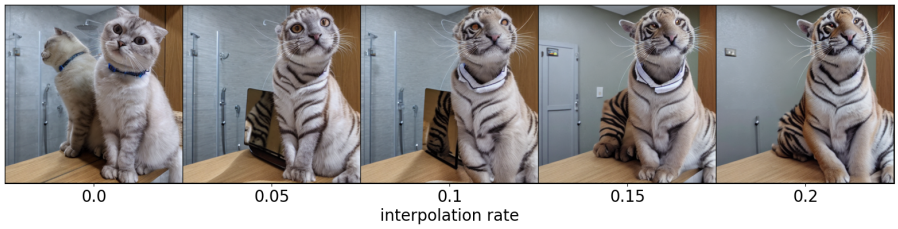


(f) turn into a smiling face

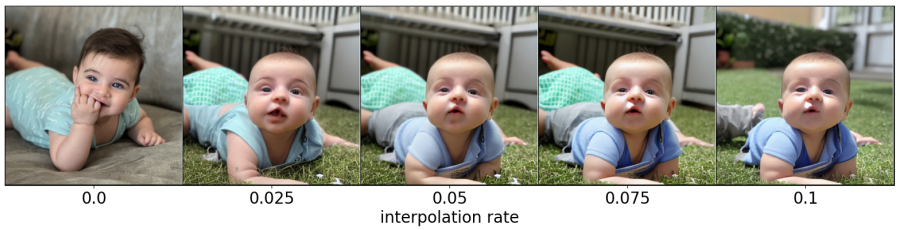


(g) add glasses

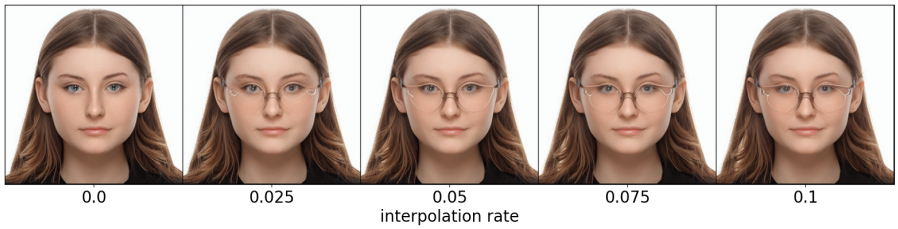
Figure 4.5: Examples of editing with Null-text Inversion with Learned Embedding



(a) cat → tiger



(b) change the background into grass



(c) add glasses

Figure 4.6: Editing with Null-text Inversion with Learned Embedding and small interpolation coefficients

Regularization in Embedding Optimization

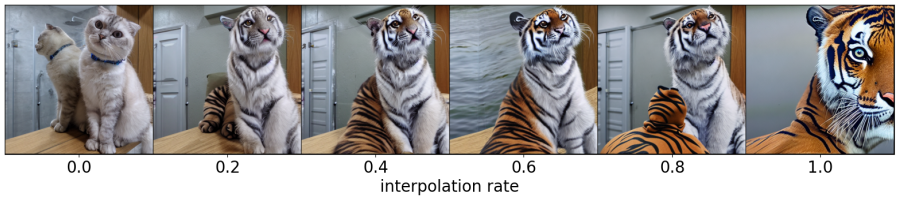
In order to achieve more stable editing results, we suggest applying regularization in embedding optimization.

In Fig. 4.5 we can see that images generated with unoptimized embedding instead of interpolation (or with an interpolation coefficient equal to 1) differ significantly from the initial images. We hypothesise that this could cause the observed instability.

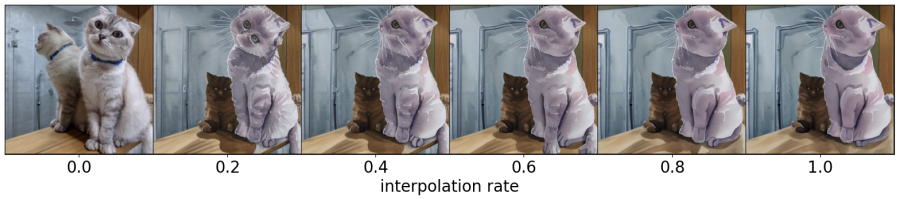
With regularization, an optimised embedding will stay closer to the initial one , meaning that an image generated with a large interpolation coefficient will be more similar to the original one than in a previous experiment. Examples generated with regularization are provided in Fig. 4.7.

Interpolation steps

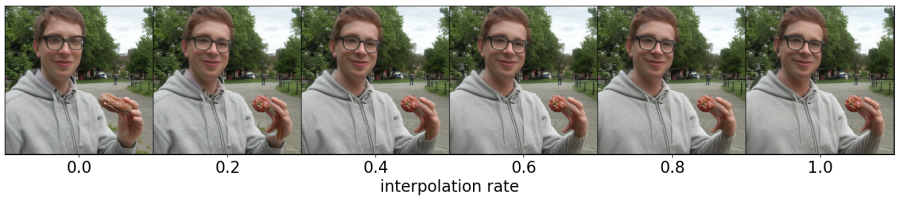
Although regularization improved the stability of the editing process, the results still lack correspondence with the initial images. To further improve the quality of editing, we suggest to reduce the number of synthesis steps that use



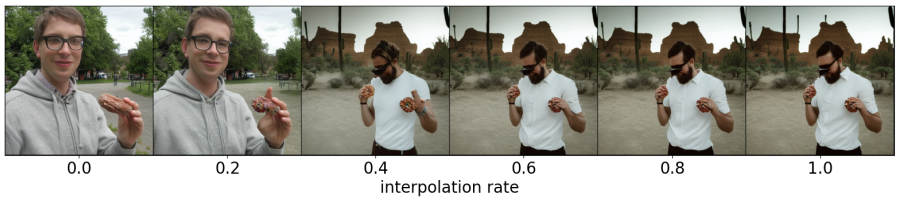
(a) cat → tiger



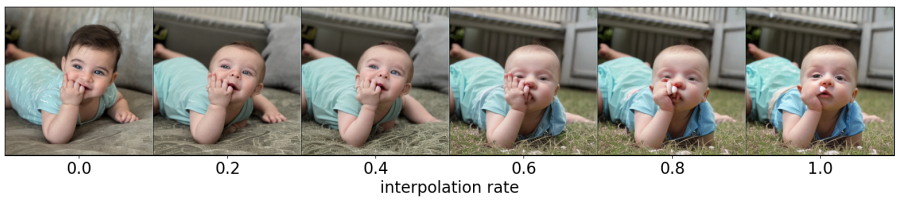
(b) turn into watercolor painting



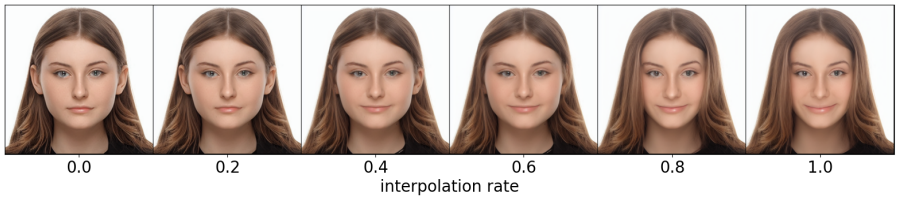
(c) make hair read



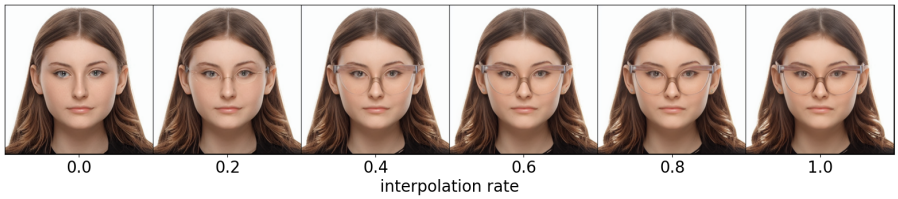
(d) change the background into desert



(e) change the background into grass



(f) turn into a smiling face



(g) add glasses

Figure 4.7: Examples of editing with Null-text Inversion with Learned Embedding and Regularization

interpolation of initial and optimized embeddings as textual embedding for conditional generation. The StableDiffusion synthesis process consists of 50 denoising steps. Let $N \in \{0, \ldots, 50\}$ be the number of *interpolation steps*. When generating an edited image, we use an optimized embedding for conditional generation in the first $50 - N$ steps, and use an interpolation result for the rest. Since learned null text embeddings combined with an optimized embedding lead to an accurate reconstruction of the initial image, we hypothesise that replacing an interpolation with an optimized embedding in several denoising steps might lead to more realistic edited images.

In this section we apply regularization in embedding optimization, as described previously.

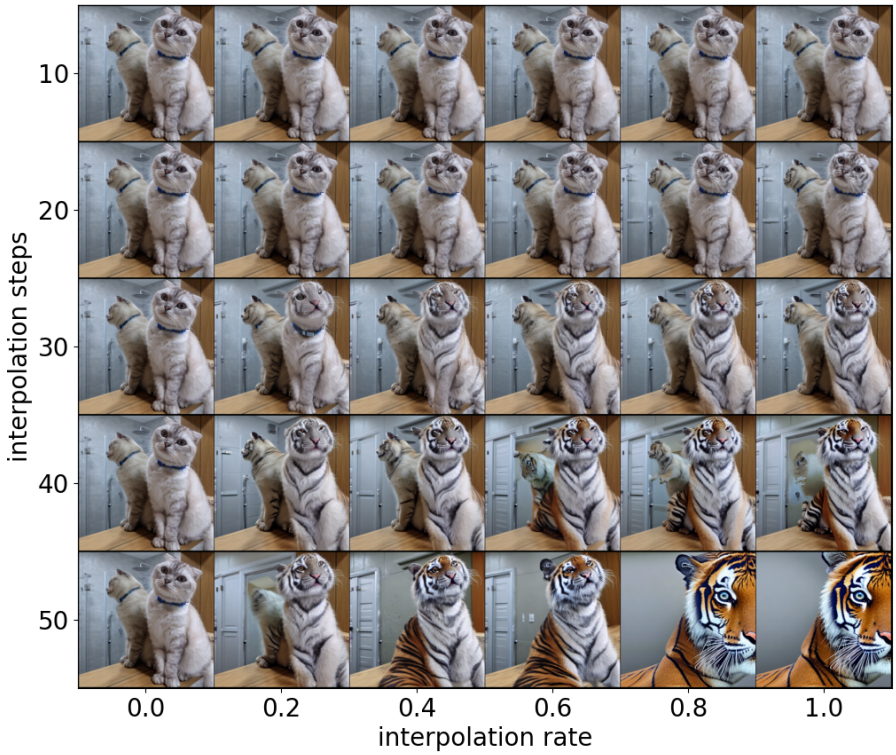


Figure 4.8: Example of turning a cat into a tiger, using *interpolation steps*

Fig. 4.8 shows a successful example of applying the proposed idea. Specifically, with 20 *interpolation steps* and a high interpolation coefficient, we achieve a realistic result when transforming a cat into a tiger.

However, there are still some limitations, for example when dealing with a lifestyle photo of a man and editing the background, as shown in Fig. 4.9.

For additional examples, refer to Appendix A.

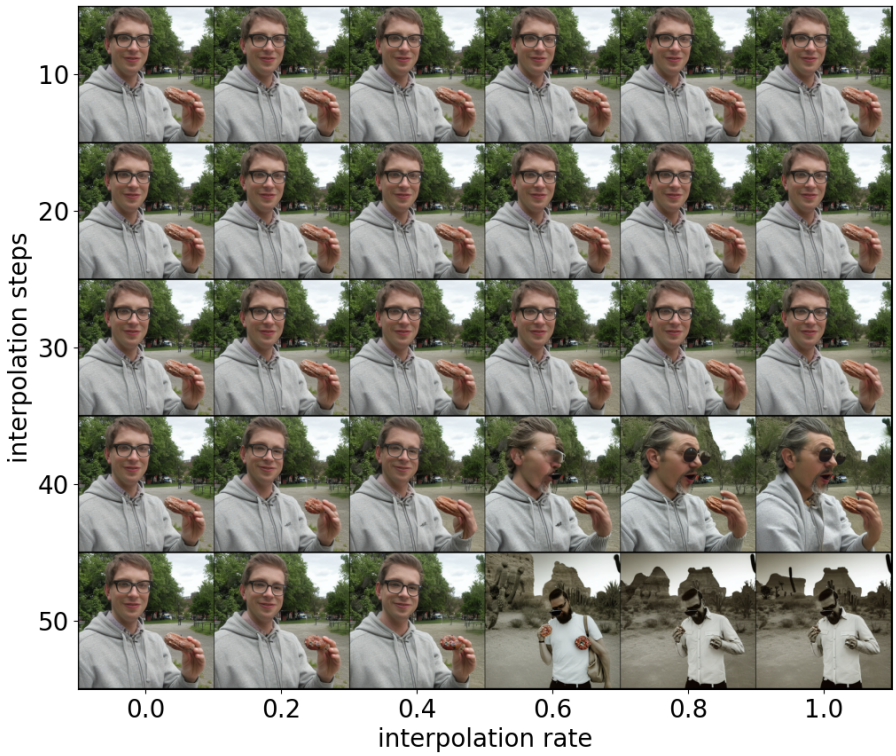


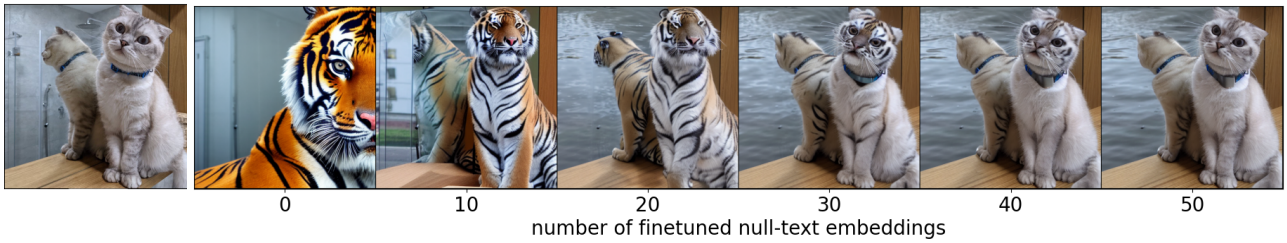
Figure 4.9: Example of changing the background into desert, using *interpolation steps*

4.2.3 Null-text Inversion with Null-text Embeddings Finetuning

In previous experiments, we observed that when using learned null-text embeddings and a textual embedding describing an edited image, a generated sample often differs significantly from the original image, which poorly affects the editing quality. This is an expected behaviour, since we train null-text embeddings with a different textual embedding.

To address this issue, we propose to fine-tune null-text embeddings using an edited prompt and an initial image. This way, we preserve the key features of the initial image and perform the editing. The more optimization steps we perform, the closer the result will be to the initial image and the less editing features it will have. We hypothesise that an edited image will appear in the middle of the fine-tuning of null-text embeddings.

Fig. 4.11 shows a detailed illustration of editing with finetuning null-text embeddings. We can see that after finetuning the first 20 null-text embeddings we achieve an edited image without any significant artefacts. For more examples see Fig. 4.11.



(a) Initial image

(b) Intermediate samples

Figure 4.10: Null-text embeddings finetuning process

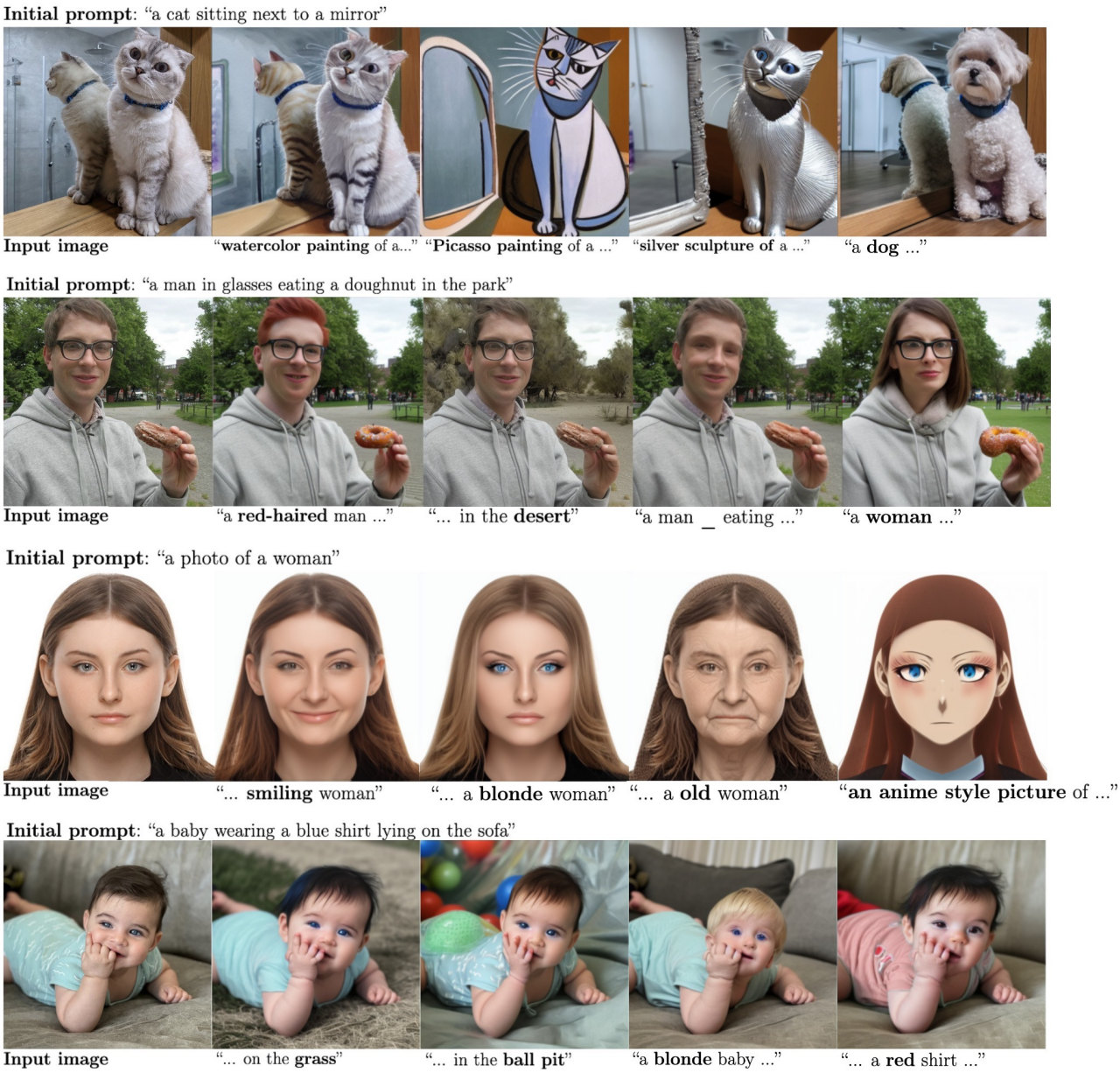


Figure 4.11: Null-text embeddings finetuning examples

4.2.4 Editing quality comparison

We compare our most successful approach, Null-text Inversion with Null-text Embeddings Finetuning, with the base approach of our work, Null-text Inversion

Figure 4.12: Methods quality comparison

with Prompt-to-Prompt (Mokady et al. 2022), as well as with Imagic (Kawar et al. 2023). Fig. 4.12 illustrates this comparison.

Our method significantly outperforms Imagic. While our approach is better at preserving the original image features, the editing it performs is more accurate

and prominent, which is especially evident from the last example in Fig. 4.12.

By removing Prompt-to-Prompt from the editing pipeline, we successfully solve editing tasks that are challenging for Null-text Inversion with Prompt-to-Prompt. For example, adding an object to the image (example 3 in Fig. 4.12).

4.2.5 Editing time comparison

Table 4.1: Methods time comparison (in seconds)

	Null-text Inversion with Prompt-to-Prompt (Mokady et al. 2022)	Imagic (Kawar et al. 2023)	**Ours**
Preliminary training	120	-	120
Editing	5	480	300
Generation	5	5	5
Total	130	485	425

Table 4.1 illustrates time comparison of the reviewed methods. Every editing pipeline is divided into three stages:

1 **Preliminary training**: this stage involves preliminary optimization, related to the initial image and the initial prompt (in case of Null-text Inversion with Prompt-to-Prompt (Mokady et al. 2022) and our method, this stage consists of DDIM inversion and training null-text embeddings)

2 **Editing**: this stage involves processing the edited prompt

3 **Generation**: this stage involves generating an edited sample

The division into stages is motivated by the fact that, when editing a single image with a number of different instructions, the "Preliminary training" stage only needs to be done once, followed by several "Editing" stages. However, this does not apply to Imagic (Kawar et al. 2023), where the whole pipeline has to be run from scratch.

For the "Editing" and "Generation" stages, we ran all methods with a single edited example as output.

Prompt-to-Prompt is the fastest editing method, but the disadvantage is that it always produces a single example and lacks the flexibility that our method introduces by controlling the number of finetuned null-text embeddings.

Our method is slightly faster than Imagic. Considering the fact that our method has a "Preliminary training" stage, we claim that it is more convenient.

5 Future plans

As the approach investigated in Experiment 4.2.3 showed the best results, it is suggested to base further work on it. The combination of finetuning null-text embeddings with textual embedding optimization is worth investigating as a promising idea that benefits from both approaches.

We are also considering replacing DDIM with another sampling approach (Karras et al. 2022). Since we haven't done a successful experiment yet, we still need to study the compatibility of StableDiffusion with other sampling approaches.

6 Conclusion

In this work, we analyze an approach to editing real images with a pre-trained diffusion model, Null-text Inversion with Propmt-to-Prompt (Mokady et al. 2022).

We perform several experiments with this method to find its limitations and determine its editing capabilities.

We propose an approach of editing, that combines Null-text Inversion with Imagic (Kawar et al. 2023). Using an optimized embedding from the latter as an embedding of the initial prompt in Null-text Inversion, we eliminate the necessity of constructing this prompt. At the same time, with substituting the model finetuning stage with Null-text Inversion, we obtain a more applicable and fast

method. However, this method has similar limitation to the base one.

Another suggested idea utilizes the capabilities of null-text embeddings for editing, eliminating the Prompt-to-Prompt technique. By finetuning null-text embeddings with an initial image and an edited prompt, we obtain realistic samples and manage to perform editing tasks, that are challenging for the base approach, such as removing or adding an object to the picture.

References

1. Hertz, A. et al. "Prompt-to-prompt image editing with cross attention control". In: (2022).
2. Ho, J., Jain, A., and Abbeel, P. "Denoising Diffusion Probabilistic Models". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.
3. Ho, J. and Salimans, T. "Classifier-Free Diffusion Guidance". In: *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. 2021.
4. Karras, T. et al. "Elucidating the Design Space of Diffusion-Based Generative Models". In: *Proc. NeurIPS*. 2022.
5. Kawar, B. et al. "Imagic: Text-Based Real Image Editing with Diffusion Models". In: *Conference on Computer Vision and Pattern Recognition 2023*. 2023.
6. Kingma, D. P. and Welling, M. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. 2014. arXiv: http://arxiv.org/abs/1312.6114v10 [stat.ML].
7. Mokady, R. et al. "Null-text Inversion for Editing Real Images using Guided Diffusion Models". In: *arXiv preprint arXiv:2211.09794* (2022).
8. Ramesh, A. et al. "Hierarchical Text-Conditional Image Generation with CLIP Latents". In: *ArXiv* abs/2204.06125 (2022).
9. Rombach, R. et al. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.
10. Saharia, C. et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding". In: *ArXiv* abs/2205.11487 (2022).

11. Song, J., Meng, C., and Ermon, S. “Denoising Diffusion Implicit Models”. In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=St1giarCHLP.

12. Vahdat, A., Kreis, K., and Kautz, J. “Score-based Generative Modeling in Latent Space”. In: *Neural Information Processing Systems (NeurIPS)*. 2021.

Appendix

A. Examples for Null-text Inversion with Learned Embedding, Regularization and Interpolation steps

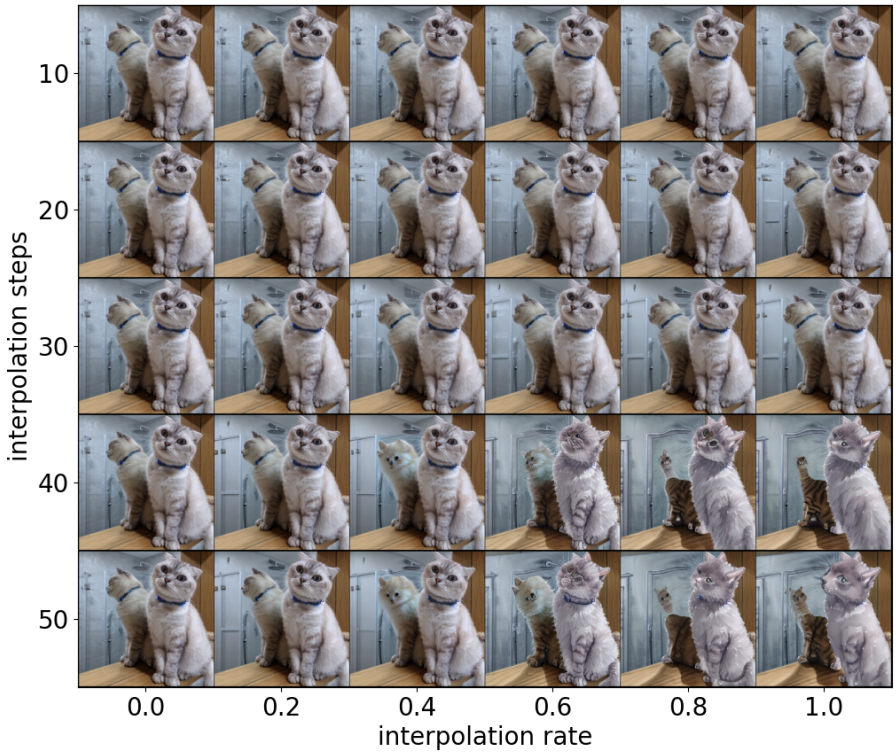


Figure 6.1: Example of turning a picture into a watercolor painting

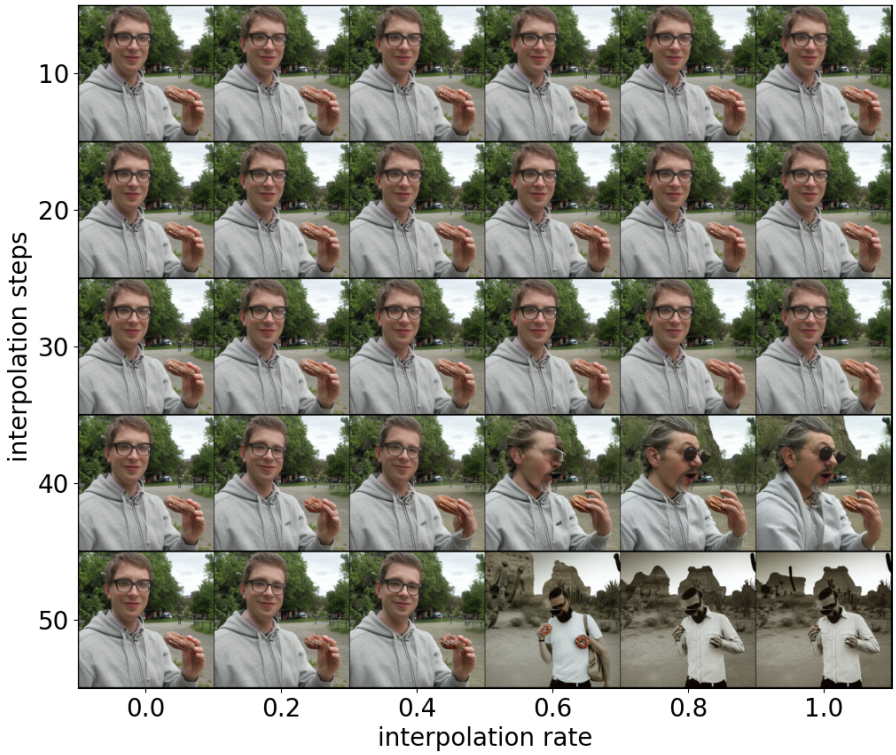


Figure 6.2: Example of making the man's hair red

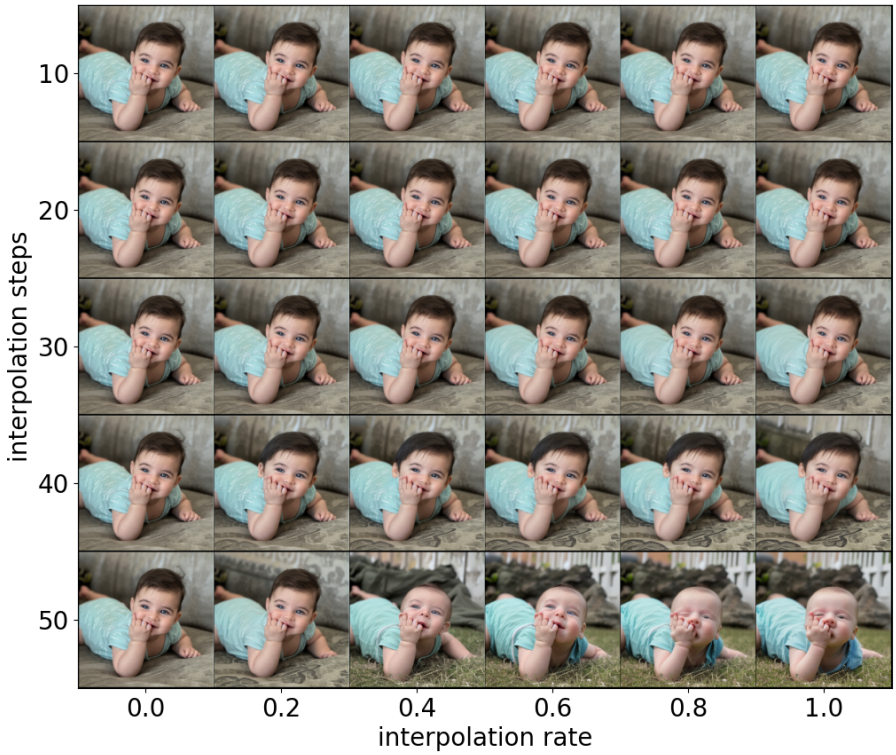


Figure 6.3: Example of changing the background into grass

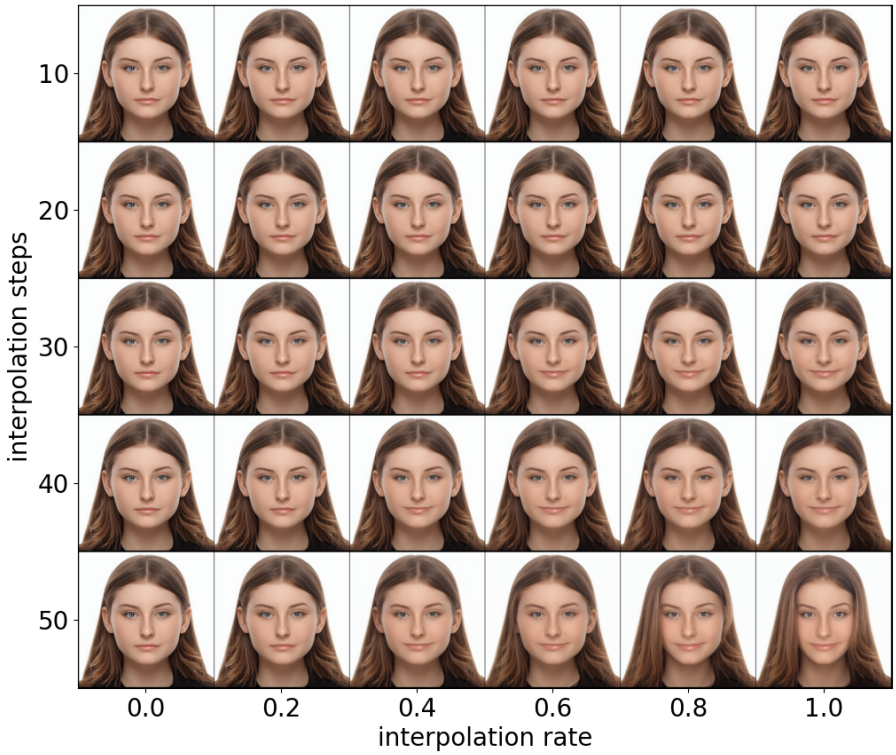


Figure 6.4: Example of turning a calm face into a smiling face

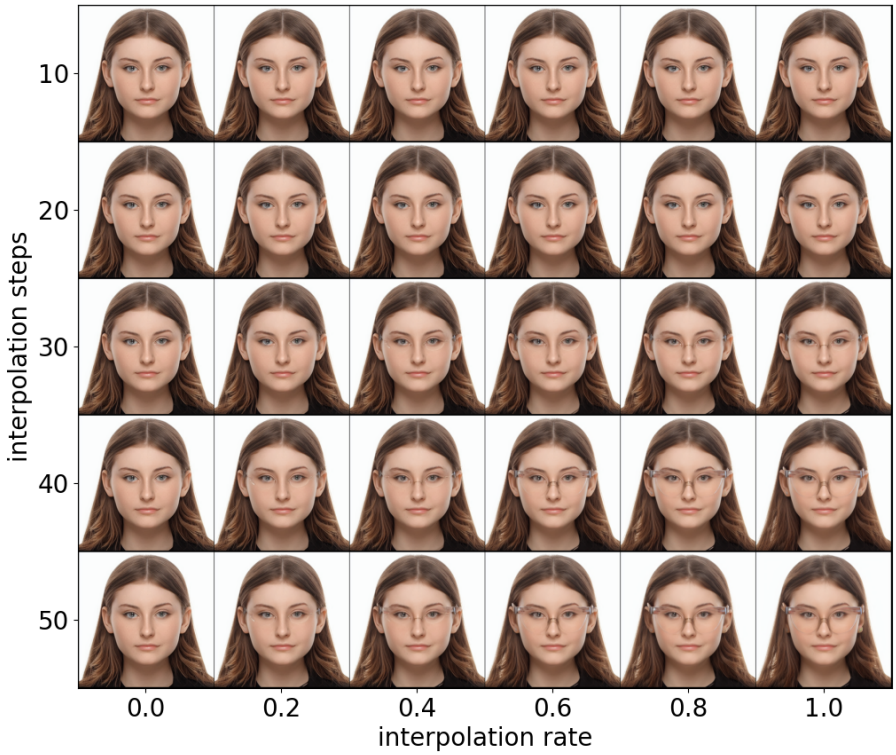


Figure 6.5: Example of adding glasses to the picture