Latent Space Editing in Transformer-Based Flow Matching

Vincent Tao Hu [1] **David W Zhang** [1] **Meng Tang** [2] **Pascal Mettes** [1] **Deli Zhao** [3] **Cees Snoek** [1]

Abstract

This paper strives for image editing via generative models. Flow Matching is an emerging generative modeling technique that offers the advantage of simple and efficient training. Simultaneously, a new transformer-based U-ViT has recently been proposed to replace the commonly used UNet for better scalability and performance in generative modeling. Hence, Flow Matching with a transformer backbone offers the potential for scalable and high-quality generative modeling, but their latent structure and editing ability are as of yet unknown. Hence, we adopt this setting and explore how to edit images through latent space manipulation. We introduce an editing space, we call u-space, that can be manipulated in a controllable, accumulative, and composable manner. Additionally, we propose a tailored sampling solution to enable sampling with the more efficient adaptive step-size ODE solvers. Lastly, we put forth a straightforward yet powerful method for achieving fine-grained and nuanced editing using text prompts. Our framework is simple and efficient, all while being highly effective at editing images while preserving the essence of the original content. We will provide our source code.

1. Introduction

The amazing realism demonstrated by large-scale text-to-image generative models (Rombach et al., 2022; Saharia et al., 2022; Ramesh et al., 2021; 2022) has garnered significant attention in the research community and beyond, leading to the development of various applications catering to non-expert users. For image editing, such models can be directly used in zero-shot fashion (Meng et al., 2021), enabling users to manipulate images without specific training on editing tasks (Bansal et al., 2023). The progress made in diffusion models (Ramesh et al., 2022; Saharia et al., 2022) has played a significant role in driving these advancements, prompting further investigations into the understanding of the learned latent space and its potential use for image editing tasks. While current works perform latent space editing on the original latent diffusion model setting and architecture (Kwon et al., 2023; Haas et al., 2023), not much is known about the structure of the latent space in the most recent advances in the field, specifically Flow Matching (Lipman et al., 2023; Liu et al., 2022; Neklyudov et al., 2022) and improved transformer backbones (Bao et al., 2023).

Flow Matching (Lipman et al., 2023) has positioned Continuous Normalizing Flow (CNF) as a strong contender to diffusion models for image synthesis. Flow Matching allows for simulation-free training of CNFs and offers improved efficiency compared to standard diffusion training and sampling techniques. Flow Matching has been quickly integrated in the field, with applications in image generation (Lipman et al., 2023), video prediction (Davtyan et al., 2022), point cloud generation (Wu et al., 2022), and manifold data (Chen & Lipman, 2023). Orthogonally, recent works have proposed enhancements to the traditional UNet architecture (Ronneberger et al., 2015) used in diffusion models, with the transformer-based U-ViT (Bao et al., 2023) demonstrating superior scaling performance. Since Flow Matching and U-ViT provide two new pieces of the puzzle towards better generative learning, a natural next step is to discover how images can be manipulated and edited via those techniques.

As a foundation for editing in transformer-based Flow Matching, we seek to uncover whether such an approach has semantic directions that can be manipulated. We take inspiration from investigations in GANs (Shen et al., 2020b; Song et al., 2023) and diffusion models (Kwon et al., 2023; Jeong et al., 2023; Haas et al., 2023), which have revealed there exists a latent space in such networks with semantic directions that can be adjusted and composed. We investigate whether transformer-based Flow Matching also induces a semantic space, that allows us to perform editing in a controllable, accumulative, and composable manner. Furthermore, to fix the misalignment between the forward and backward process in Flow Matching, we propose se-

[1]VISLab, University of Amsterdam [2]UC Merced [3]Alibaba, China. Correspondence to: Vincent Tao Hu <taohu620@gmail.com>.

Workshop on New Frontiers in Learning, Control, and Dynamical Systems at the International Conference on Machine Learning (ICML), Honolulu, Hawaii, USA, 2023. Copyright 2023 by the author(s).

mantic direction interpolating during the sampling process to reach a more exact and adaptive control over the semantic generation. To make latent space editing viable in practice, manipulation should be done at the input or prompt level. While early work generates new images from scratch based on updated prompts (Rombach et al., 2022; Ramesh et al., 2022), prompt-to-prompt (Hertz et al., 2022) allows for editing images locally while keeping the unedited part similar. Prompt-to-prompt is however specifically designed to work with the cross-attention of U-Net (Ronneberger et al., 2015). We show that local prompt editing becomes simple and intuitive in U-ViT (Bao et al., 2023). Latent space editing becomes as easy as replacing, removing, or appending prompts. Given an initial prompt and generated image, we can simply reweight tokens (e.g. *enlarge beard, remove the dog, shrink tree*), allowing us to locally manipulate images in an invasion-free and user-friendly manner while enabling us to make use of the more powerful transformer architecture.

2. Background: Flow Matching

In Flow Matching, we are given a set of samples from an unknown data distribution $q(\mathbf{x})$. The goal is to learn a *flow* that pushes the simple prior density $p_0(\mathbf{x})=\mathcal{N}(\mathbf{x}\,|\,0,1)$ towards a complicated distribution $p_1(\mathbf{x})\approx q(\mathbf{x})$ along the probability path $p_t(\mathbf{x})$. Formally, this is denoted using the push-forward operation as $p_t=[\phi_t]_* p_0$. The time-dependent flow can be constructed via a vector field $\mathbf{v}_t(\mathbf{x}) : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ that defines the flow via the neural ordinary differential equation (ODE):

$$\dot{\phi}_t(\mathbf{x}) = \mathbf{v}_t(\phi_t(\mathbf{x})), \qquad \phi_0(\mathbf{x}) = \mathbf{x}_0. \tag{1}$$

Given a predefined probability density path $p_t(\mathbf{x})$ and the corresponding vector field $\mathbf{w}_t(\mathbf{x})$, one can parameterize $\mathbf{v}_t(\mathbf{x})$ with a neural network with parameter θ and solve

$$\min_\theta \mathbb{E}_{t,p_t(\mathbf{x})} \|\mathbf{v}_t(\mathbf{x};\theta) - \mathbf{w}_t(\mathbf{x})\|^2. \tag{2}$$

However, directly optimizing this is infeasible, because we do not have access to $\mathbf{w}_t(\mathbf{x})$ in closed form. Instead, Lipman et al. (Lipman et al., 2023) propose to use the conditional vector field $\mathbf{w}_t(\mathbf{x}\,|\,\mathbf{x}_1)$ as the target, which corresponds to the conditional flow $p_t(\mathbf{x}\,|\,\mathbf{x}_1)$. Importantly, they show that this new *conditional Flow Matching* objective

$$\min_\theta \mathbb{E}_{t,p_t(\mathbf{x}\,|\,\mathbf{x}_1),q(\mathbf{x}_1)} \|\mathbf{v}_t(\mathbf{x};\theta) - \mathbf{w}_t(\mathbf{x}\,|\,\mathbf{x}_1)\|^2, \tag{3}$$

has the same gradients as Equation (2). By defining the conditional probability path as a linear interpolation between p_0 and p_1, all intermediate distributions are Gaussians of the form $p_t(\mathbf{x}\,|\,\mathbf{x}_1)=\mathcal{N}(\mathbf{x}\,|\,tx_1, 1-(1-\sigma_{\text{min}})t)$, where $\sigma_{\text{min}} > 0$ is a small amount of noise around the sample x_1. The corresponding target vector field is:

$$\mathbf{w}_t(\mathbf{x}\,|\,\mathbf{x}_1) = \frac{\mathbf{x}_1 - (1-\sigma_{\text{min}})\mathbf{x}}{1-(1-\sigma_{\text{min}})}, \tag{4}$$

Lipman et al. (Lipman et al., 2023) show that learning straight paths improves the training and sampling efficiency compared to diffusion paths. It allows to generate samples by first sampling $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{x}\,|\,0,1)$ and then solving Equation (1) using an off-the-shelf numerical ODE solver. Similarly, we can invert a data point to get its corresponding latent noise by solving the ODE in the other direction.

3. Latent Space Editing in Flow Matching

Transformer architecture. Our goal is to facilitate easy-to-use and powerful image editing within the framework of Flow Matching. Transformer architectures not only show promise in enhancing image generation capabilities (Bao et al., 2023; Peebles & Xie, 2022), but they also provide a straightforward editing approach by adjusting the tokens of user prompts. In light of this, we propose a general architecture for transformer-based Flow Matching, which is outlined in Figure 1. To efficiently generate high-resolution images we use a pretrained autoencoder to compress images $\tilde{\mathbf{x}}$ into representations $\mathbf{x}=E(\tilde{\mathbf{x}})$ of lower spatial resolution. While keeping the encoder and decoder parameters fixed, we train a U-ViT (Bao et al., 2023) using the Flow Matching objective in Equation (2). To generate new images, we first generate a sample $\mathbf{x}$ using the CNF and then map it back into the image space using the decoder $\tilde{\mathbf{x}}=D(\mathbf{x})$. In the context of a transformer-based Flow Matching framework, a crucial step to enable image editing through a semantic latent space is to identify a suitable latent space that exhibits semantic properties suitable for manipulation. While a straightforward approach might involve using the middle layer, similar to U-Nets, we have discovered that this approach does not yield a robust semantic latent space. Moreover, it can result in changes that are completely unrelated to the source image or produce images of poor visual quality. We attribute this discrepancy to the absence of a spatial compression in the vision transformer structure, in contrast to the conventional U-Net architecture (Ronneberger et al., 2015). We have found, by experiment, that the most effective space for semantic manipulation lies at the beginning of the U-ViT architecture. To distinguish it from the bottleneck layer, commonly referred to as the h-space in the U-Net (Kwon et al., 2023), we designate this space as u-space.

Semantic direction manipulation in u-space. In order to enable image editing through simple vector additions in the latent space we need to identify the directions in the vector space that correspond to semantically meaningful edits. We adopt a supervised approach to obtain interpretable semantic directions by contrasting two datasets: one dataset $\{\mathbf{x}_i^{k+}\}_i$

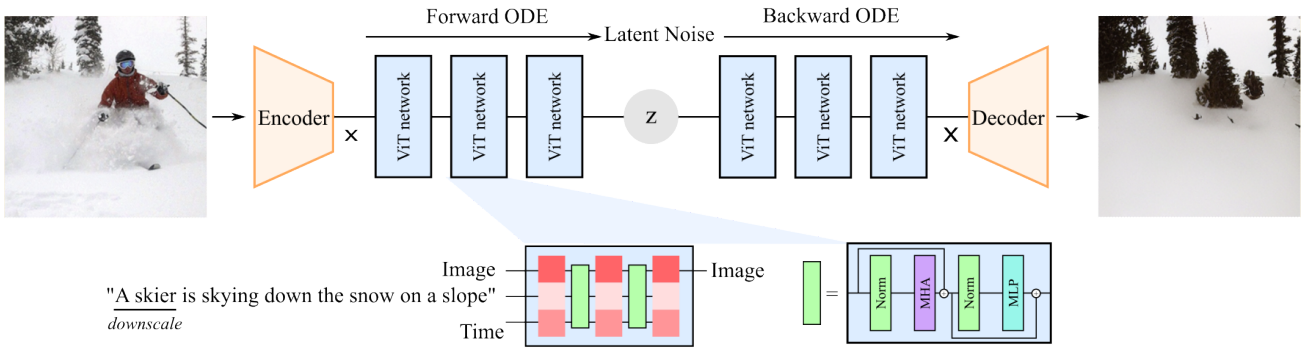


Figure 1. **Latent Flow Matching for image editing.** Starting from the original image, we extract the latent feature x from the frozen encoder. Then, Flow Matching is applied in latent space to transfer the trajectory between the latent feature and standard Gaussian noise by integration on the vector field. An editing operation can be triggered in u-space and `Prompt` by your own desire. The edited feature will be fed back to the decoder to generate the final edited image.

consisting of images that possess the desired attribute k, and the other dataset $\{\mathbf{x}_j^{k-}\}_j$ consisting of images without the desired attribute. Examples of attributes include age, gender, or smile. Importantly, we do not require any paired data, and the images in the two datasets can differ in other arbitrary ways. We compute a semantic direction $\mathbf{s}_t^k$ as follows:

$$\mathbf{s}_t^k = \frac{1}{n} \sum_{i=1,j=1}^{n} \left(\mathbf{u}(\mathbf{x}_{i,t}^{k+}) - \mathbf{u}(\mathbf{x}_{j,t}^{k-}) \right), \tag{5}$$

where $\mathbf{u}(\cdot)$ maps to the semantic latent space, t corresponds to the time variable in Equation (1), and i and j index the different images in the datasets.

We can collect semantic directions $\mathbf{s}^k$ from both real and generated images. In the former case, we gather the latent representations for different time steps through the forward process of the ODE. In the latter case, we utilize the backward process. In this work we focus on semantic directions acquired from real images. For generated images, we can identify the presence of specific attributes using an off-the-shelf attribute classifier (Shen et al., 2020b).

After collecting the sets of semantic directions, we can proceed to manipulate the sampling (backward) process by editing the u-space. This can be achieved through:

$$\tilde{\mathbf{u}}(\mathbf{x}_t, k, w) = \mathbf{u}(\mathbf{x}_t) + w \cdot \mathbf{s}_t^k, \tag{6}$$

where w is the semantic guidance strength. Image editing can be implemented in u-space by replacing $\mathbf{u}(\mathbf{x}_t)$ with $\tilde{\mathbf{u}}(\mathbf{x}_t, k, w)$ during the forward pass of the neural network. Additionally, we need to determine the appropriate time step t for performing the injection. Initially, one might consider injecting the guidance signal at every time step when the ODE solver calls the neural network. However, we have observed that this approach can result in a degradation of the visual quality of the sampled image. To address this issue, we limit the modification to time steps $0 < t < t_{edit}$. By focusing the injection on early integration steps, we can improve the visual quality while still achieving effective edits. The hyperparameter t_{edit} can be tuned to find the right balance between the consistency of the edits and the quality of the resulting image. For a fixed step ODE solver this can be written as:

$$\mathbf{x}_{t_{i-1}} = \begin{cases} \text{ODE}(\mathbf{x}_{t_i}, \tilde{\mathbf{u}}(\mathbf{x}_{t_i}, k, w)) & \text{if } 0 < t_i < t_{\text{edit}} \\ \text{ODE}(\mathbf{x}_{t_i}, \mathbf{u}_{t_i}) & \text{else} \end{cases} \tag{7}$$

where $t_i = \frac{i}{N}$ and $i \in 0, ...N-1$, N is the integration number for the ODE solver.

Semantic direction interpolation for adaptive step ODE. Existing methods for editing images using semantic latent directions are limited to fixed step-size ODE solvers. This is because an adaptive step-size ODE solver requires neural network evaluations at arbitrary time steps t, while the semantic directions $\mathbf{s}_t^k$ were only gathered at fixed intervals. This misalignment problem between the time steps keeps previous methods from making use of the more efficient adaptive step-size solvers. To address this problem, we propose an interpolation-based method to handle the misalignment between the two sequences of time steps. We first gather the semantic directions using a fixed step-size ODE solver, which only needs to be done once. During the editing process, when the neural network is called for a new time step t, we interpolate between the two closest semantic directions in time. Then we compute the semantic directions during the editing process as:

$$\mathbf{s}_t^k = \mathbf{s}_{\lfloor t \rfloor}^k + (\mathbf{s}_{\lceil t \rceil}^k - \mathbf{s}_{\lfloor t \rfloor}^k) \times (t - \lfloor t \rfloor), \tag{8}$$

where $\lfloor t \rfloor$, $\lceil t \rceil$ denotes the floor and ceiling values in the grid $[\frac{0}{N}, \frac{1}{N}, .., 1]$. As long as the total number of steps N is large enough, we can assume that the interpolation of

Algorithm 1 Semantic direction manipulation in u-space.

Input: parameter of pretrained network θ, fix step ODE solver ODE_f with step N, adaptive ODE solver ODE_a, target attribute k.
Output: An edited image $\mathbf{x}_1$
$\mathbf{x}_0 \sim \mathcal{N}(0, I)$ a unit Gaussian random variable
$m = 0$
for $i = 1, 2, \ldots, M$ **do**
  for $j = N, N-1, \ldots, 1$ **do**
    $\mathbf{u}^i_{t_{j-1}}, \mathbf{x}_{t_{j-1}} = \text{ODE}_f(\mathbf{x}_{t_j}; \theta)$; where $t_j = \frac{j}{N}$
  end for
end for
for $j = N, N-1, \ldots, 0$ **do**
  Calculate $\mathbf{s}^k_{t_j}$ by Equation (5)
end for
while $t_m \leq 1$ **do**
  if $0 < t < t_{\text{edit}}$ **then**
    Interpolate semantic direction $\mathbf{s}^k_t$ by Equation (8)
    $\mathbf{x}_{t_{m+1}}, t_{m+1} = \text{ODE}_a(\mathbf{x}_{t_m}, \mathbf{u}_{t_m} + w \cdot \mathbf{s}^k_{t_m}, t_m; \theta)$
  else
    $\mathbf{x}_{t_{m+1}}, t_{m+1} = \text{ODE}_a(\mathbf{x}_{t_m}, \mathbf{u}_{t_m}, t_m; \theta)$
  end if
end while
Return $\mathbf{x}_1$

the semantic direction can be as accurate as possible. In Algorithm 3, we provide the overall pipeline for semantic direction manipulation in u-space with adaptive step-size ODE solvers.

Steering by text-conditioned prompts. In the preceding segment of this section, we focused on image editing through the concept of feature steering, achieved by incorporating semantic offsets in a semantic latent space. Next, we will delve into the realm of text-conditioned generation in Flow Matching, where we aim to explore the potential for users to directly edit images by augmenting text prompts.

First, we revisit the text-conditioned U-ViT architecture (Bao et al., 2023). The architecture follows the standard Transformer encoder architecture with additional skip connections similar to a U-Net. The encoded image is patchified into tokens $\phi(x)$, which are then concatenated along the set dimension with the tokens from the text prompt $\psi(\mathcal{P})$ and the embedding of the time step $\mathcal{T}$.

In every attention layer the tokens $[\phi(x), \psi(\mathcal{P}), \mathcal{T}]$ are projected to a query matrix $Q = \ell_Q([\phi(x), \psi(\mathcal{P}), \mathcal{T}])$, a key matrix $K = \ell_K([\phi(x), \psi(\mathcal{P}), \mathcal{T}])$, and a value matrix $V = \ell_V([\phi(x), \psi(\mathcal{P}), \mathcal{T}])$, via learned linear projections ℓ_Q, ℓ_K, ℓ_V. The *attention map* is computed as

$$M = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right), \tag{9}$$

where d is the latent projection dimension of the keys and queries.

In the U-ViT architecture, each attention operation involves interactions between image tokens and the text prompt, which distinguishes it from the U-Net approach where such interactions are restricted to *cross*-attention layers only. Cross-attention forces every image token to attend to some token in the text prompt even when nothing in the text is relevant to that part of the image. Consequently, modifications to the text prompt can lead to significant image changes that go beyond the intended editing scope. Prompt-to-prompt (Hertz et al., 2022) addresses this issue by augmenting the attention map to retain a similar layout as in the unmodified case. However, this necessitates additional computation and storage of attention maps for the unmodified text prompt version. On the other hand, in self-attention, image tokens have the flexibility to choose whether or not to attend to any text token. This limits the impact of modifications made to the text tokens, especially when they are irrelevant to a specific image patch. Motivated by this observation, we explore a simpler form of local prompt editing.

In certain scenarios, there is a need for more precise and nuanced editing, such as modifying the magnitude of target concepts, adjusting colors, hairstyles, and so on, instead of making drastic changes like replacing, removing, or adding entire objects. One straightforward approach is to directly scale the representation of the corresponding token. However, this assumes that the tokens occupy a semantically interpretable space, which may not always hold true. To address this issue, we propose a simpler approach that makes weaker assumptions: scaling the attention value between the modified prompt tokens and image patches. For scaling a specific prompt, we first identify the target token IDs, denoted as $\mathbf{j}^*$. We then apply the following scaling operation:

$$(M^l_t)_{i,j} := \begin{cases} c \cdot (M^l_t)_{i,j} & \text{if } j \in \mathbf{j}^* \text{and i-th is token from image} \\ (M^l_t)_{i,j} & \text{otherwise.} \end{cases}$$

Here, the parameter c allows for fine-grained and intuitive editing by weakening or strengthening specific parts of the text prompt. The index l denotes the layer of the U-ViT model. In practice, applying this reweighting in every block of the U-ViT architecture yields the best results. We defer related analysis in Appendix. Similar to editing in the semantic latent space, we find that it is advantageous to limit the modifications to time steps $0 < t < t_{\text{edit}}$. In our experiments, we observe that this much simpler method can perform a multitude of different editing operations while preserving most of the source image.

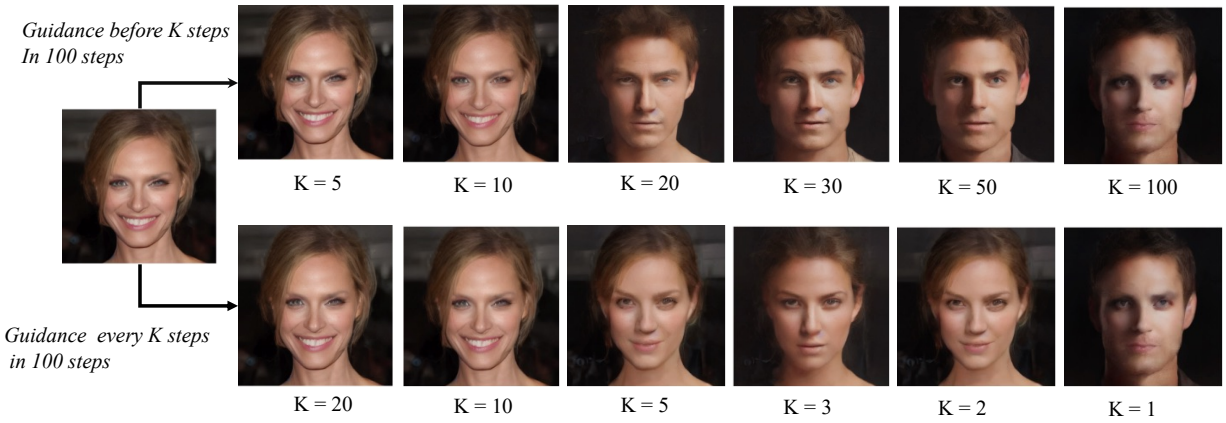


Figure 2. **The location ablation of the guidance injection** when editing *male*. Early time step injection is best, injection during full time step will lead to over-constraining. The first row is signal injection before the first 5, 10, 20, 30, 50, and 100 steps of the backward ODE process. The second row is guidance injection evenly throughout 100 steps.

4. Experiments

Experimental details. For the encoder and decoder architecture, we use convolutional VAEs (Kingma & Welling, 2014) with pretrained weights following (Rombach et al., 2022). For the experiments on semantic manipulation in the u-space, we mainly use the 256×256 CelebA-HQ (Xia et al., 2021) dataset. To reduce the training time, we finetune the U-ViT based on pretrained weights (Bao et al., 2023). We list all the hyperparameters in the supplementary. For t_{edit}, we found t_{edit}=0.5 works reasonably well. For the guidance strength w in Equation (7), we obvserve that $w \in (-2, 2)$ generally provides sufficient flexibility while still producing reasonable results. If not mentioned otherwise, we use the adaptive ODE solver `dopri5`.

For prompt-based editing, we conduct the experiments on the MultiModal-CelebA-HQ (Xia et al., 2021) and MS COCO (Lin et al., 2014) datasets, with image resolution 256×256. Both datasets are composed of text-image pairs for training. Typically, there are 5 to 10 captions per image in COCO and MultiModal-CelebA-HQ. Here too, we finetune starting from pretrained weights. For the reweighting parameter c in self-attention, we set $c \in (-10, 10)$. For editing real images, we choose the images from the validation set of MS COCO.

Semantic direction manipulation in u-space. In Figure 2, we investigate the optimal time interval to inject the guidance signal from the semantic direction. Using a fixed step-size ODE solver for a total of 100 steps, we explore signal injection during the first 5, 10, 20, 30, 50, and 100 sampling steps. We observe that injecting the signal for too few steps, for example for 5 or 10 steps, fails to perform the intended edits. Further extending the signal injection period to the first 30 or 50 steps, we find that the semantic direction can be manipulated more noticeably with the same guidance strength. However, if we extend the signal injection until the end it fails to retain anything from the original image.

To further validate whether steerability is related to the time step number, we conducted a controlled experiment in which we injected the guidance signal evenly throughout 100 steps, specifically at every (20, 10, 5, 3, 2, 1) time step. We find that splitting the injection step into those parts did not easily achieve semantic manipulation, even if we manipulated every 5th step into 100, which significantly indicates the importance of the early timestep involvement. We refer the reader to the progressive visualization in Appendix for more information.

Furthermore, in Figure 3, we demonstrate attribute editing on real and sampled images. Our method can achieve various manipulations, such as adding a *smile* or changing the gender to *male*. This not only indicates the effectiveness of the u-space but also demonstrates the flexibility of our method on both real and sampled images.

In the end, we validate the compositional ability of these semantic directions in Figure 4. We progressively apply the semantic injections one after another. We use the `euler` solver with 100 steps for the forward process, and `dopri5` solver for the backward process. By gradually adding the attributes of *smile*, *young*, and *male*, we demonstrate that the sampling process in u-space can be manipulated in an accumulative and composable manner.

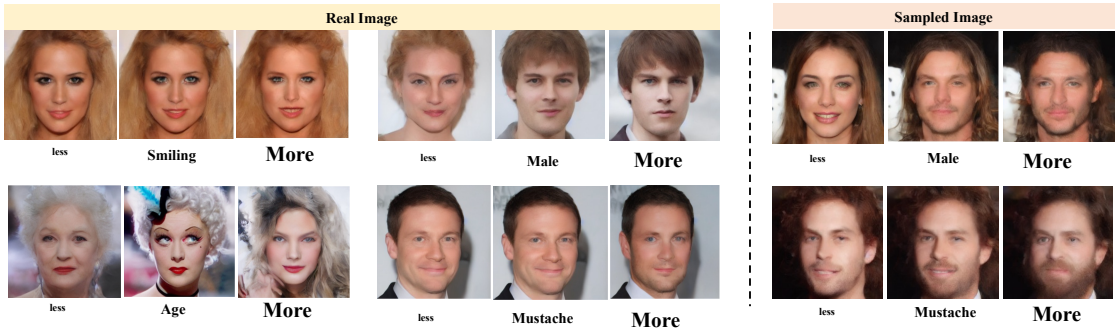


Figure 3. **Attribute editing on real images and sampled images.**

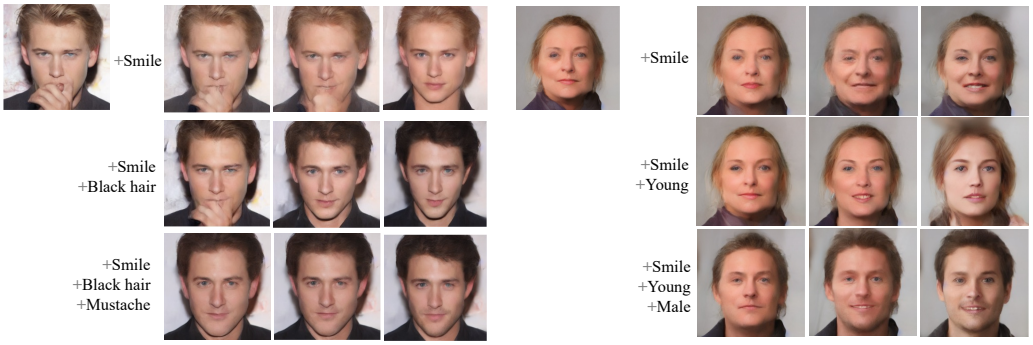


Figure 4. **Compose multiple attributes sequentially.** We gradually enforce three different semantic directions on a single reference image. The semantic direction of the three attributes is simply averaged by scale $w = 2$.

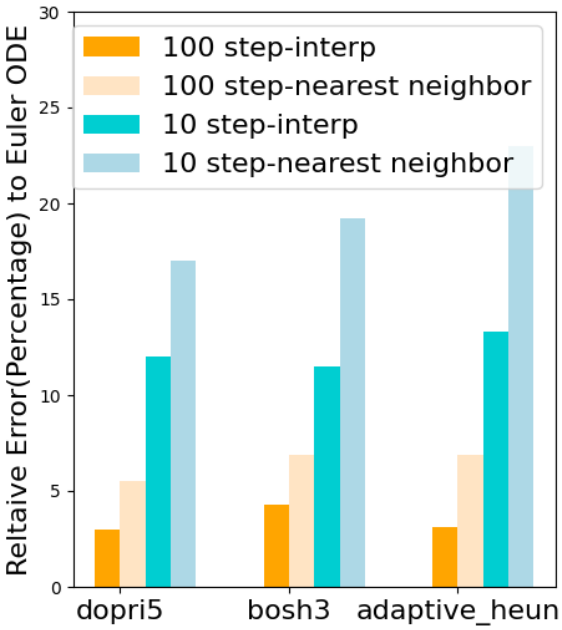


Figure 5. **Relative error comparison** between adaptive and `euler` ODE solvers with different time steps.

Semantic direction interpolation error analysis. In this study, we investigate the amount of error that may be introduced using semantic direction interpolation. We examine two factors: the number of time steps (N) and different ODE solvers, including `dopri5`, `bosh3`, and `adaptive_heun`. We compare the relative edit error to the editing performed by the `euler` solver with $N = 100$, using guidance strength $w = 1$ for the *male* attribute. Additionally, we compare our results to a baseline of nearest neighbor seeking in the grid of $[0, \frac{1}{N}, \frac{2}{N} ..., 1]$ for the semantic direction. In Figure 5, we observe that larger time steps (N) lead to smaller errors. The relative error is generally at the same level across all three solvers. However, too small of a time step, such as $N = 10$, can result in a drastically sizeable relative error. This indicates that using a too-small time step number(large step size) will lead to inaccurate integration for both methods. However, when the time step is large enough, the integration will be more stable and accurate.

Text-to-image editing. We first demonstrate the accumulation ability of our local-prompt method. In Figure 6, we demonstrate that by appending prompts sequentially based on the initial prompt, the sampled image retains its identity while gaining attributes that align well with the newly added prompts. To the best of our knowledge, a similar experiment

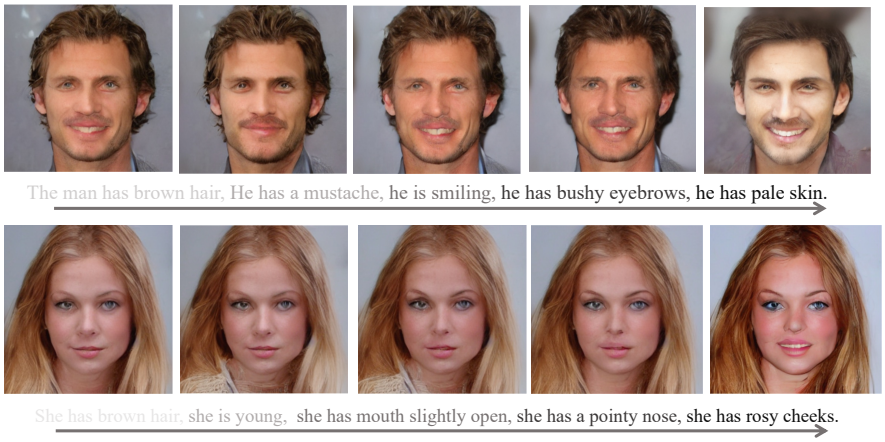


Figure 6. **Appending new prompts sequentially**. We can inject semantic meaning directly into the subject while keeping it almost unchanged. To do so, we start by sampling the first image based on the initial prompt. Then, we append the remaining prompts one by one.

Figure 7. **Prompt reweighting, removing, replacing in MM-CelebA-HQ dataset**. Our method can gradually scale the target prompt, remove the target prompt, and be robust to noise during prompt removal.

Figure 8. **Prompt reweighting** on the real images of MS COCO dataset by weighting down (left) and weighting up (right). Best viewed in color.

has not been conducted in prompt-to-prompt (Hertz et al., 2022). This suggests that our method can inject semantic meaning by altering raw prompts and that this injected semantic concept can be accumulated without compromising the identity.

Secondly, we investigate the effects of prompt replacement and removal, as depicted in Figure 7. Notably, we observe that concepts such as *wavy* and *beard* can undergo substantial alterations or even be entirely removed from the resulting image. Intriguingly, when less informative prompts like *he has* are removed, we find that the sampled image remains unchanged, thereby demonstrating the robustness of our method.

Lastly, we explore prompt reweighting. Specifically, we perform prompt reweighting on sampled images from MM-CelebA-HQ, as shown in Figure 7. By reweighting the attributes of *white* and *hair*, we can modify them effectively without altering the identity. In the top right row of Figure 7, we demonstrate a progressive and smooth evolution by changing the reweighting scale w from 2 to -10. The smooth change indicates a well-aligned embedding between text and image space. To explore the reweighting ability in a more complex dataset, we conduct experiments on MS COCO, shown in Figure 8. We used a pretrained latent flow-matching model on MS COCO and were able to easily invert the image from $\mathbf{x}_1$ to latent $\mathbf{x}_0$. We progressively conduct attention map editing as we revert the latent back from $\mathbf{x}_0$ to $\mathbf{x}_1$. As shown in the figure, we were able to add or remove concepts as we scaled up and down the related prompts.

5. Related Work

There has been a significant body of research dedicated to exploring the latent space of generative adversarial networks (GANs) (Bau et al., 2018; Shen et al., 2020b;a), which has provided valuable insights into enabling semantic editing of images through vector arithmetic in the latent space. Recent efforts have extended these explorations to diffusion models. Some recent works have extended similar explorations to diffusion models. Kwon et al. (Kwon et al., 2023) have made an interesting discovery, identifying the bottleneck of the UNet denoising model as a suitable space for semantic image manipulation in diffusion models. The discovery led to further exploration of the so-called h-space, resulting in supervised and unsupervised methods for discovering global semantic directions (Park et al., 2023; Haas et al., 2023), image-specific semantic directions (Haas et al., 2023), and style-aware semantic directions (Jeong et al., 2023), In a different approach, Huberman-Spiegelglas et al. (Huberman-Spiegelglas et al., 2023) propose an alternative latent noise space that deviates from the traditional Gaussian distribution. This non-Gaussian noise space exhibits improved amenability to semantic image editing. Our method differs in multiple aspects. In contrast to the aforementioned works, our method diverges in several key aspects. Firstly, we explore the semantic space of the U-ViT model (Bao et al., 2023) instead of the UNet. The U-ViT model has demonstrated favorable scaling behavior, but its dissimilarities from the UNet raise uncertainties and challenges in applying previous semantic editing methods that primarily rely on the spatially reduced latent representation at the bottleneck of the UNet. Secondly, our focus lies on the general continuous normalizing flow (CNF) setting, allowing the utilization of generic ODE solvers that include adaptive step-size techniques. In contrast, previous editing methods were limited to fixed step-size solvers, rendering them incompatible with the efficiency benefits of adaptive step-size solvers.

Specialized for text-to-image diffusion models, prompt-to-prompt (Hertz et al., 2022) leverages text-conditioned diffusion models for image editing by directly modifying the associated text prompt to achieve the desired semantic changes. The null-text inversion method (Mokady et al., 2022) extends this technique to real images by improving the inversion performance in the text-conditional setting. DiffusionClip (Kim et al., 2022) proposes to fine-tune a dedicated model for every type of editing, while DiffFit (Xie et al., 2023) enhances the efficiency of this by only fine-tuning a small subset of parameters. It is important to note that these methods are primarily designed for UNet architectures, where text conditioning is incorporated through cross-attention mechanisms. In contrast, our exploration, focuses on U-ViT (Bao et al., 2023) architectures, revealing that the conclusions drawn from UNets may not necessarily apply to other architectural choices, such as U-ViTs. Furthermore, their method requires recasting the diffusion model as an ODE during inference, while Flow Matching enables direct learning of the ODE that we use during editing.

6. Conclusion

In this paper, we explore the problem of image editing under the regime of Flow Matching with a transformer backbone. We discovered the u-space in this setting, which allows for editing in a controllable, accumulative, and composable manner. Furthermore, we make the editing method agnostic to the ODE solvers' choice of step size. To this end, we leverage the full-attention design in ViT and propose a simple and effective local-prompt method for augmenting the importance of specific parts of the prompt. As a result, this method can replace, remove, add, and scale the designated semantic concept. In future work, we plan to perform in-depth dissections to determine whether these conclusions hold true for other generative models.

References

Bansal, A., Chu, H.-M., Schwarzschild, A., Sengupta, S., Goldblum, M., Geiping, J., and Goldstein, T. Universal guidance for diffusion models. *arXiv*, 2023.

Bao, F., Li, C., Cao, Y., and Zhu, J. All are worth words: a vit backbone for score-based diffusion models. *CVPR*, 2023.

Bau, D., Zhu, J.-Y., Strobelt, H., Zhou, B., Tenenbaum, J. B., Freeman, W. T., and Torralba, A. Gan dissection: Visualizing and understanding generative adversarial networks. In *ARXIV*, 2018.

Chen, R. T. and Lipman, Y. Riemannian flow matching on general geometries. *arXiv*, 2023.

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. *NeurIPS*, 2018.

Davtyan, A., Sameni, S., and Favaro, P. Randomized conditional flow matching for video prediction. *arXiv*, 2022.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

Haas, R., Huberman-Spiegelglas, I., Mulayoff, R., and Michaeli, T. Discovering interpretable directions in the semantic latent space of diffusion models. *arXiv*, 2023.

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., and Cohen-Or, D. Prompt-to-prompt image editing with cross attention control. *arXiv*, 2022.

Hu, V. T., Zhang, D. W., Asano, Y. M., Burghouts, G. J., and Snoek, C. G. M. Self-guided diffusion models. In *CVPR*, 2023.

Huberman-Spiegelglas, I., Kulikov, V., and Michaeli, T. An edit friendly ddpm noise space: Inversion and manipulations. *arXiv*, 2023.

Jabri, A., Fleet, D., and Chen, T. Scalable adaptive computation for iterative generation. *arXiv*, 2022.

Jeong, J., Kwon, M., and Uh, Y. Training-free style transfer emerges from h-space in diffusion models. *arXiv*, 2023.

Kim, G., Kwon, T., and Ye, J. C. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *CVPR*, 2022.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *ICLR*, 2014.

Kwon, M., Jeong, J., and Uh, Y. Diffusion models already have a semantic latent space. *ICLR*, 2023.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *ICLR*, 2023.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv*, 2022.

Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2021.

Mokady, R., Hertz, A., Aberman, K., Pritch, Y., and Cohen-Or, D. Null-text inversion for editing real images using guided diffusion models. *arXiv*, 2022.

Neklyudov, K., Severo, D., and Makhzani, A. Action matching: A variational method for learning stochastic dynamics from samples. *arXiv*, 2022.

Park, Y.-H., Kwon, M., Jo, J., and Uh, Y. Unsupervised discovery of semantic latent directions in diffusion models. *arXiv*, 2023.

Peebles, W. and Xie, S. Scalable diffusion models with transformers. *arXiv*, 2022.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. In *ICML*, 2021.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. In *ARXIV*, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. In *ARXIV*, 2022.

Shen, Y., Gu, J., Tang, X., and Zhou, B. Interpreting the latent space of gans for semantic face editing. In *CVPR*, 2020a.

Shen, Y., Yang, C., Tang, X., and Zhou, B. Interfacegan: Interpreting the disentangled face representation learned by gans. *TPAMI*, 2020b.

Song, Y., Keller, A., Sebe, N., and Welling, M. Latent traversals in generative models as potential flows. *ICML*, 2023.

Wu, L., Wang, D., Gong, C., Liu, X., Xiong, Y., Ranjan, R., Krishnamoorthi, R., Chandra, V., and Liu, Q. Fast point cloud generation with straight flows. *arXiv*, 2022.

Xia, W., Yang, Y., Xue, J.-H., and Wu, B. Tedigan: Text-guided diverse face image generation and manipulation. In *CVPR*, 2021.

Xie, E., Yao, L., Shi, H., Liu, Z., Zhou, D., Liu, Z., Li, J., and Li, Z. Difffit: Unlocking transferability of large diffusion models via simple parameter-efficient fine-tuning. *arXiv*, 2023.

Yang, X., Shih, S.-M., Fu, Y., Zhao, X., and Ji, S. Your vit is secretly a hybrid discriminative-generative diffusion model. *arXiv*, 2022.

Zhang, E., Wang, K., Xu, X., Wang, Z., and Shi, H. Forget-me-not: Learning to forget in text-to-image diffusion models. *arXiv*, 2023.

Figure 9. **More sequential editing result by shuffling the order of the prompts.** The *same* set of semantic manipulations can be composed in different orders.

A. Supplementary Files

A.1. Extra main results

We further explore sequential editing in Figure 9. We shuffle the order of the prompts and find that the generated faces align with the corresponding prompt irrespective of the specific order. Interestingly, the final editing results (last column) are almost identical, indicating that the semantic edits can be composed through different prompt orders without altering the image layout.

A.2. Hyperparameters choices and details

We perform experiments on three datasets and document our hyperparameter choices in Appendix A.2. For sampling, we utilize the torchdiffeq (Chen et al., 2018) library with `atol=rtol=1e-5`.

We generate the initial image with a fixed seed and a given initial prompt. Then we concatenate additional captions to the prompt one by one and sequentially edit the image. Currently, we only support editing for images that have a caption. We leave the editing of images without captions to future work.

A.3. More related works

Cross-attention in generative models. Cross-attention is broadly applied in generative model to incorporate the guidance signal e.g., text (Rombach et al., 2022; Saharia et al., 2022), bounding box (Hu et al., 2023), segmentation mask (Rombach et al., 2022). Prompt-to-Prompt (Hertz et al., 2022) explores the cross-attention between the text and visual modalities to achieve controllable generation. Instead of fine-tuning based on CLIP difference in DiffusionClip (Kim et al., 2022), (Zhang

Dataset	CelebA 256×256	MM-CelebA-HQ 256×256	MS-COCO 256×256
Latent shape	32×32×4	32×32×4	32×32×4
U-ViT type	U-ViT-L/2	U-ViT-L/2	U-ViT-L/2
depth	20	20	20
embedding dim	1024	1024	1024
num of head	16	16	16
Batch size	256	256	512
Training iterations	100K	100K	200k
GPU	4 × A5000	4 × A5000	4 × A5000
Optimizer	Adam	Adam	Adam
Learning rate	1e-4	1e-4	1e-4
σ_{min}	1e-4	1e-4	1e-4

Table 1. **Hyperparameter Settings** of three datasets.

et al., 2023) utilize QKV attention Re-steering to achieve concept negation directly on pre-trained models. (Xie et al., 2023) only fine-tune attention's bias weight to conduct parameter-efficient fine-tuning to achieve transferability on other datasets. (Jabri et al., 2022) utilize cross-attention to exchange information between latent tokens and image tokens. On the side hand, ViT (Dosovitskiy et al., 2021) has been applied as a surrogate of UNet (Ronneberger et al., 2015) in diffusion models, e.g., U-ViT (Bao et al., 2023), DiT (Peebles & Xie, 2022), GenViT (Yang et al., 2022),RIN (Jabri et al., 2022). We focus on the exploration of the U-ViT structure in flow matching, and the U-ViT composed of full attention blocks has not been explored before.

A.4. Visual comparison between different ODE solvers for the semantic direction interpolation

In Figure 10, we invert the original image by solving the ODE in the forward direction using an `euler` solver for 100 steps. Then, we change the *gender* attribute by manipulating the latent noise with a weight of $w = 1$. Afterwards, we compare different ODE adaptive step-size solvers with the fixed step-size `euler` solver. We observe that the adaptive step size solvers (`dopri5`, `bosh3`, `adaptive heun`) achieve visually very similar results when compared to the more expensive `euler` method.

A.5. PCA in u-space

In detail, we perform PCA on 10,000 samples with shape $4 \times 32 \times 32$ in u-space. The two-dimensional features $\mathbf{u}(\mathbf{x}_{i,t})$ from the U-ViT are flattened into one dimensional vectors before applying PCA. We show some sample augmentations along the first four principal components in Figure 11. We observe that for different samples there is a difference in the type of semantic manipulation that occurs. We hypothesize that the lack of spatial compression in U-ViT leads to this phenomenon, where a linear augmentation along the principle components can augment multiple different semantic aspects of the image.

A.6. Semantic editing in other blocks of U-ViT

We ablate the effectiveness of the intermediate layer for semantic manipulation. The tensor shape is [T, C], where $T = 257$ and $C = 1024$. We find that the tensor is too large to calculate the semantic direction, which can lead to unstable results. In total, the overall dimensions are 64 times larger than u-space (e.g. $257 \times 1024 \approx 64 \times (4 \times 32 \times 32)$).

Based on the above analysis, we choose to perform semantic editing at the beginning of U-ViT, before the patchifying and attention blocks.

A.7. Attention map visualization in Local-Prompt

We visualize the attention map token by token and normalize the image value by image. We illustrate the attention map of different blocks in Figure 13 and the attention map of different time steps in Figure 12. We find that early timesteps demonstrate attention maps that clearly outline specific parts of the face in comparison to later time steps. This observation

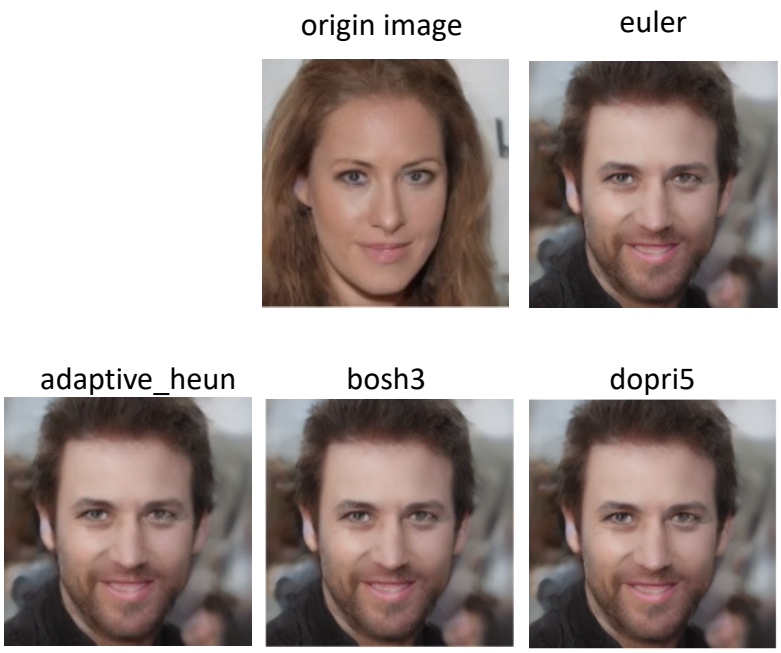


Figure 10. **The editing visualization by semantic direction interpolation.** The default ODE solver is `euler` with 100 time steps. We can find that interpolation indeed leads to visual similar manipulation.

aligns well with the conclusion presented in the main paper. In particular, at earlier stages during the sampling process, the transformer requires the text prompt to specify the to-be-generated content of the image. At later steps the self-attention can attend to other regions in the image instead of the text prompt. Secondly, we observe that the highest activation is achieved in *block 0* and that the most discriminative blocks are *block 4* and *block 8*.

A.8. Prompt rescaling in every block

As we saw in Figure 13, we observe that different blocks exhibit different magnitudes for the attention values. This highlights the significance of the magnitude of the attention and motivates us to perform attention-rescaling for the whole block.

A.9. Exploration in early time steps

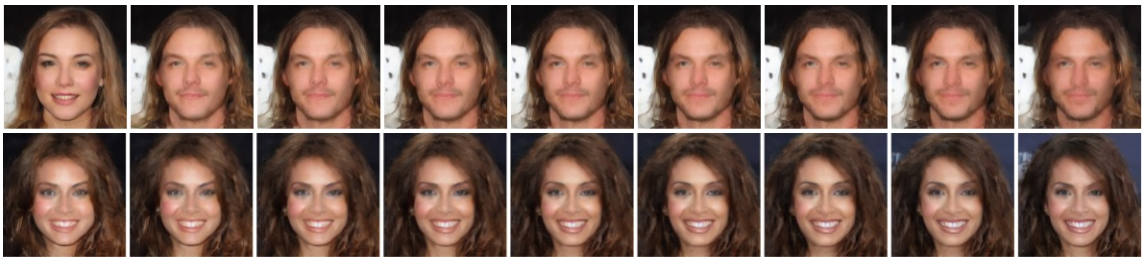
By visualizing the progressive sampling procedure, we explore how incorporating the augmentation at different timesteps during sampling affects the editing process. Our results show that incorporating the augmentations in the early timesteps works well, while evenly splitting the same number of manipulation operations throughout the entire period is not effective in manipulating the achieving the desired editing. Additionally, we investigate the effect of increasing the weight value w and find that this can disturb the final sample quality and discard the content of the source image. These findings are presented in Figure 15. Our study provides insights into effective techniques for semantic manipulation and highlights the importance of early timestep guidance in this process.

A.10. More visualization

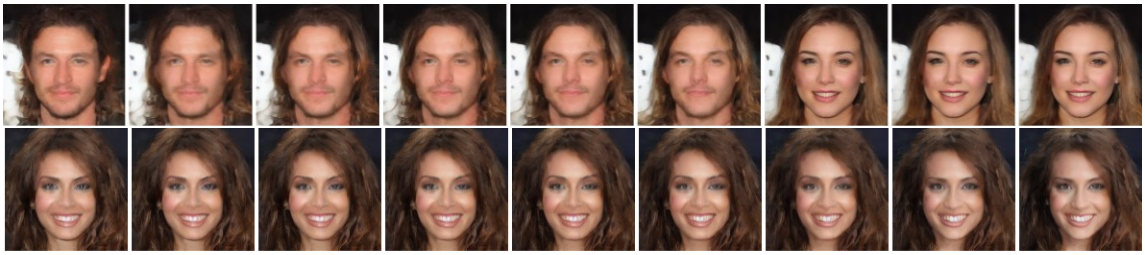
We add a noise prompt to the MS COCO dataset and find that it did not significantly alter the layout, as shown in Figure 16.

We visualize the samples with the same latent noise and prompt during the training process in Figure 14.

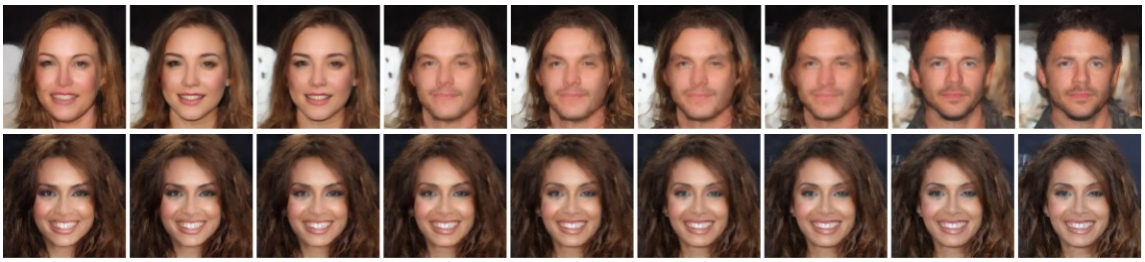
1th Component



2th Component



3th Component



4th Component

Figure 11. **Semantic direction from the top 4 components of PCA.** We generate manipulated samples of two identical images by adjusting the guidance strength range to [-2, -1.5, -1, -0.5, 0, +0.5, +1, +1.5, +2]. These manipulated samples demonstrate that the directions found by PCA do not yield consistent semantic manipulations.

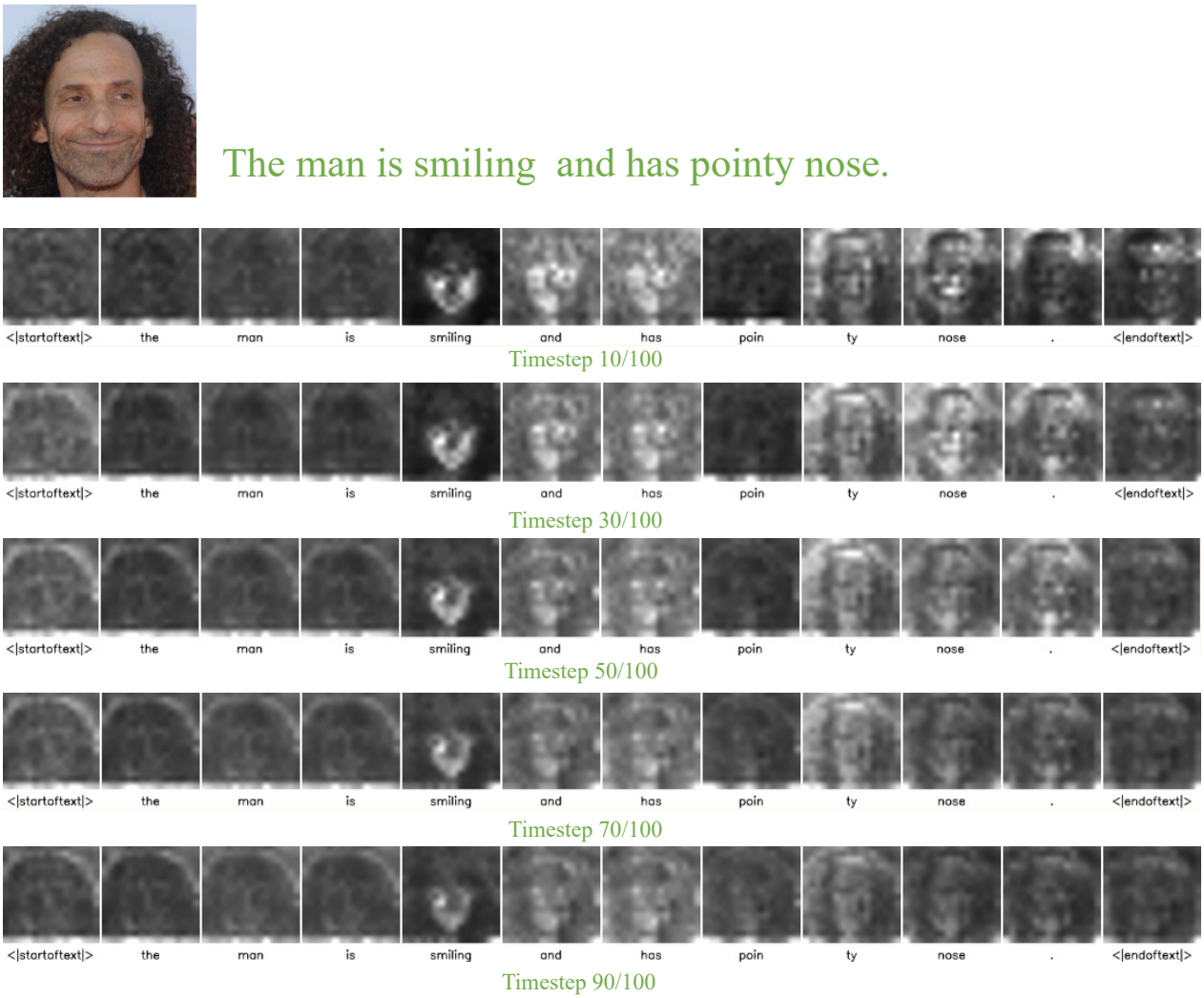


Figure 12. **Attention map of different timestep of *3th*-Block in U-VIT.**

Figure 13. **Attention map of different blocks of *10th* step in 100 timesteps in U-ViT.**

Figure 14. **Training samples with the same latent noise and same prompt.** The first row contains examples of image generation, while the second and third rows contain examples of text-conditioned image generation. Best viewed in color.

A.11. Failure case

If the scaling factor s during self-attention is too large, it can lead to over-constraining. This is demonstrated in Figure 17.

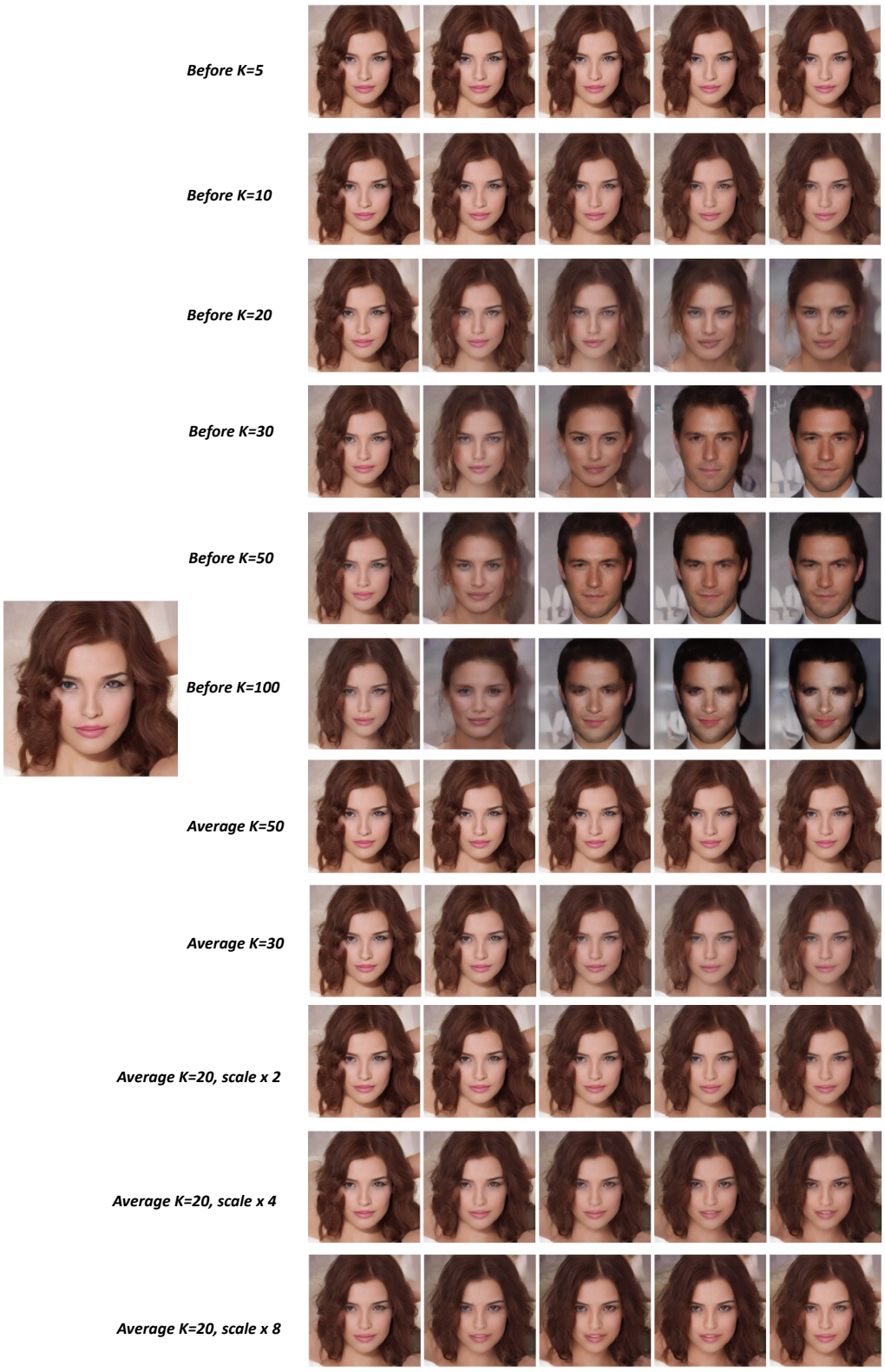


Figure 15. **Early timestep has semantic editing ability.** The default scale w is [0.5, 1, 1.5, 2, 2.5]. The edited attribute is *male*.

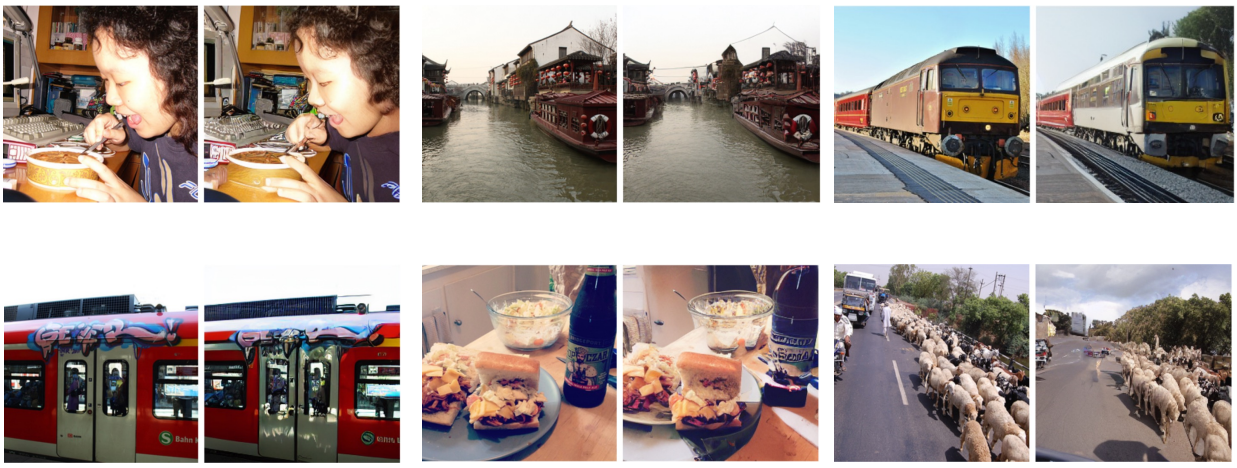


Figure 16. **Adding noise prompts does not significantly change the layout.** However, please note that the content may be slightly altered due to the marginal semantic relationship between noise prompts and visual tokens.

The man has **wavy(↺)** hair.

Figure 17. **Too large values for the self-attention rescaling will lead to over-constraining.** The *wavy* attribute can cause the generated image to be *female* if the scaling factor is too large, e.g., $s = 25$.