IMAGE-TO-IMAGE TRANSLATION FOR FACE ATTRIBUTE EDITING WITH DISENTANGLED LATENT DIRECTIONS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

By

Yusuf Dalva

June 2023

Image-to-Image Translation for Face Attribute Editing With Disentangled Latent Directions
By Yusuf Dalva
June 2023

We certify that we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Ayşegül Dündar Boral(Advisor)

Selim Aksoy

Ramazan Gökberk Cinbiş

Approved for the Graduate School of Engineering and Science:

Orhan Arıkan
Director of the Graduate School

ABSTRACT

IMAGE-TO-IMAGE TRANSLATION FOR FACE ATTRIBUTE EDITING WITH DISENTANGLED LATENT DIRECTIONS

Yusuf Dalva
M.S. in Computer Engineering
Advisor: Ayşegül Dündar Boral
June 2023

We propose an image-to-image translation framework for facial attribute editing with disentangled interpretable latent directions. Facial attribute editing task faces the challenges of targeted attribute editing with controllable strength and disentanglement in the representations of attributes to preserve the other attributes during edits. For this goal, inspired by the latent space factorization works of fixed pretrained GANs, we design the attribute editing by latent space factorization, and for each attribute, we learn a linear direction that is orthogonal to the others. We train these directions with orthogonality constraints and disentanglement losses. To project images to semantically organized latent spaces, we set an encoder-decoder architecture with attention-based skip connections. We extensively compare with previous image translation algorithms and editing with pretrained GAN works. Our extensive experiments show that our method significantly improves over the state-of-the-arts.

Keywords: Image-to-image translation, Generative Adversarial Networks, Latent Space Manipulation, Face Attribute Editing.

ÖZET

AYRIŞTIRILMIŞ ÖRTÜLÜ VEKTÖRLERLE YÜZ ÖZELLİKLERİ DÜZENLEME İÇIN RESİMDEN RESİME ÇEVİRİ

Yusuf Dalva
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Ayşegül Dündar Boral
Haziran 2023

Bu çalışmada yüz özelliklerini düzenlemek için, ayrıştırılmış ve yorumlanabilir örtülü vektörler ile bir resimden resime çeviri sistemi önerilmektedir. Çalışmamızın odak noktası olan yüz özelliklerini düzenleme görevi, belirli bir özelliği kontrol edilebilir bir miktarda düzenleme ve bu süreçte diğer özellikleri koruyan ayrışık gösterimler öğrenme gibi zorluklara sahiptir. Bu zorlukları aşabilmek amacıyla önerilen sistemimizde, önceden eğitilmiş çekişmeli üretken ağlar (GAN) üzerinde uygulanan örtülü uzay ayrıştırması çalışmalarından ilham alarak, sistemimizde bulunan ve her biri farklı bir özelliği modelleyen diğer vektörlere dik bir doğrusal vektör elde ediyoruz. Sistemimiz, bu doğrusal vektörleri öğrenmek için diklik ve ayrıştırma üzerine optimizasyon fonksiyonları kullanmaktadır. Önerilen sistem, düzenlenmek istenen yüz resimlerini anlamsal olarak düzenlenmiş örtülü uzaya yansıtmak amacıyla kodlayıcı ve kod çözücüden oluşan ve dikkat mekanizması ile atlamalı bağlantılar içeren bir ağ mimarisi kullanmaktadır. Önerilen sistemin etkinliğini göstermek için, yüz özelliklerini düzenleme görevinde en iyi performansı gösteren modellerle detaylı karşılaştırmalar sunulmaktadır. Karşılaştırmalarımızda da görüldüğü gibi, çözümümüz bu görev için kullanılan güncel modellerden daha iyi bir performans göstermektedir.

Anahtar sözcükler: Resimden resime çeviri, Çekişmeli Üretken Ağlar, Örtülü Uzay Manipulasyonu, Yüz Özellikleri Düzenleme.

Acknowledgement

I would like to express my deepest gratitude to my advisor Ayşegül Dündar Boral for her invaluable guidance and continuous support throughout my studies. I will forever be indebted to her for giving me an opportunity to prove myself and encouraging me to push my limits every day.

I sincerely appreciate Prof. Selim Aksoy and Asst. Prof. Gökberk Cinbiş for accepting to be my thesis jury.

I would like to thank my parents, İzak Dalva and Klara Dalva, for their guidance and unconditional support, even in extraordinary circumstances. I couldn't have wished for better mentors. I also thank Sevim Dalva Aydemir and Kadir Aydemir for their encouragement towards my studies.

I am forever grateful to my teammates from Bilkent Image Generation Lab, for all of our discussions, feedback, and the collaborations we did. I also appreciate my friends Batıkan Karlar, Gökhan Akkaya, and Zeynep Ebru Arısoy for their support.

Contents

1 Introduction **1**

2 Related Work **6**

2.1 Image-to-Image Translation Networks 6

2.2 Editing with pretrained GANs 8

3 Method **10**

3.1 Generator Architecture 10

3.2 Translation Module 13

3.3 Attention-based Skip Connections 16

3.4 Training pathways 17

3.5 Learning objectives 19

3.5.1 Adversarial Objective 19

3.5.2 Shift Reconstruction Objective 20

3.5.3 Image Reconstruction Objective 20

3.5.4 Orthogonality Objective 21

3.5.5 Disentanglement Objective 21

3.5.6 Sparsity Objective 22

3.5.7 Full Objective 22

4 Experiments 23

4.1 Dataset and Settings 23

4.1.1 Comparing with End-to-End Approaches 23

4.1.2 Comparing with StyleGAN2-based Methods 25

4.2 Metrics 26

4.3 Ablation Study 30

4.3.1 Experiments on the Base Version 30

4.3.2 Experiments on the Extended Version 33

4.4 Comparisons with Competing Methods 35

5 Analysis and Discussions 42

5.1 Comparing End-to-end Image Translation Networks versus StyleGAN Inversion-based Methods 42

5.2 Analysis of Projected Styles 43

5.3 Limitations 45

6 Conclusion **49**

A Additional Results **56**

B Classifier Details **61**

C Additional Architecture Details **62**

C.1 Resampling Block Architecture 62

C.2 Discriminator Architecture 63

List of Figures

1.1 VecGAN image translation results. 2

2.1 Existing approaches for facial attribute editing task 7

3.1 Organization of Labels 11

3.2 Overview of VecGAN Pipeline 12

3.3 Generator architecture of VecGAN 14

3.4 Architecture for Skip Network S 17

3.5 Overview of cycle-translation path 18

4.1 Sample generation procedure for evaluation 25

4.2 FID calculation for smile addition edit 28

4.3 Identity Preservation (ID) metric calculation for smile addition task 29

4.4 Background Preservation (BG) metric calculation for smile addition task 29

4.5 Qualitative results of ablation study for orthogonality loss 32

4.6 Ablation study between VecGAN$_{\text{base}}$ and VecGAN$_{\text{final}}$ 33

4.7 FID curves on smile addition task for models trained with and without disentanglement loss 34

4.8 Qualitative results of bangs addition edit of VecGAN$_{\text{final}}$ 37

4.9 Feature strength interpolation results 39

4.10 Plots of FID, ID, and BG metrics with changing editing intensity 40

4.11 Qualitative results of ours and competing methods 41

5.1 Comparison of α_s distributions learned by VecGAN$_{\text{base}}$ and VecGAN$_{\text{final}}$ 44

5.2 Attribute scale plots 46

5.3 Visualization of attribute scale histograms 47

5.4 Failure cases of VecGAN 48

A.1 Supplementary Results for bangs edits. 56

A.2 Supplementary Results for hair color edits. 57

A.3 Supplementary Results for age edits. 58

A.4 Supplementary Results for gender edits. 59

A.5 Supplementary Results for smile edits. 60

B.1 Confusion matrices for the classifiers used in evaluation Setting II. 61

C.1 Architecture of up/downsampling blocks in VecGAN 63

C.2 Discriminator architecture of VecGAN 64

List of Tables

4.1 Ablation study on setting I 30

4.2 Quantitative results for setting I 36

4.3 Quantitative results for setting II for local edits 38

4.4 Quantitative results for setting II for global edits 38

Chapter 1

Introduction

The task of image-to-image translation has experienced a significant amount of progress which aims to learn a mapping between two different imaging domains [1, 2, 3, 4, 5, 6, 7, 8]. As a sub-field of the task, facial attribute editing aims to translate a face image concerning a given semantic while preserving the other properties of the image. With the use cases of the task in applications focused on entertainment, there have been significant improvements in the methods focusing on Generative Adversarial Networks (GANs) based solutions [9, 10, 11, 12, 13, 14]. As the task aims to edit the given face image in a way that the target semantic is edited, but the remaining would be preserved, facial attribute editing is considered one of the most challenging translation tasks where humans can easily assess the quality of the edit on whether the identity of the input changes or not. A visual overview of the task is shown in Fig. 1.1.

The current efforts on the problem approach the task from two different viewpoints. One aims to learn an end-to-end network for the target translation, and the other focuses on manipulating the latent codes of pretrained GANs. In the first end, different architectures for the image translation task are offered, which typically involve two networks, where one is for encoding the style of the image and the other is for performing the translation by injecting the extracted styles

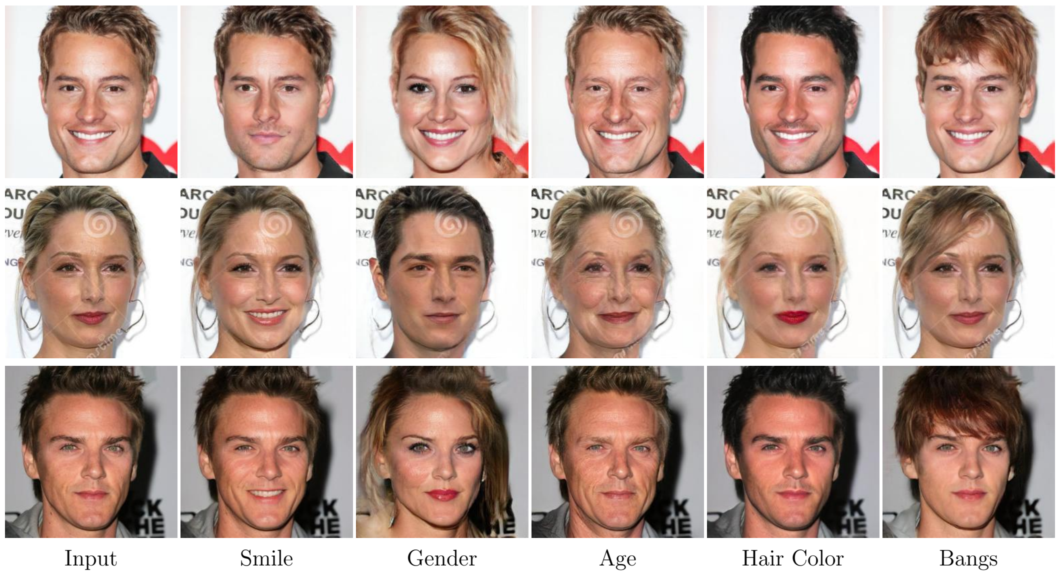


Figure 1.1: VecGAN image translation results. The first column shows the source images, and the other columns show the result of editing a specific attribute. Each edited image has a semantic value opposite to the input image with respect to the target attribute. Only for the hair color semantic, we translate the image to one of three semantics, black, blonde, and brown hair, where we illustrate black hair and blonde hair edits above.

[9, 15, 13]. In this approach, a style or an attribute is usually encoded from another image or sampled from a distribution. As facial attribute editing aims to achieve disentangled translations, the attributes in both the editing and encoding phases must be disentangled.

With this objective, works focus on style encoding and iterate using a shared style code, SDIT [16], to eclectic style codes, StarGANv2 [9], to hierarchical disentangled styles, HiSD [15]. Among these works, HiSD [15] learns styles of each attribute independently, namely for the bangs, eyeglasses, and hair color semantics, and introduces a translator network to apply local edits with attention masks, which enables avoiding global edits. Even though such an approach proves its success on three local editing tasks, it is not tested for tasks targeting global edits, e.g., age, smile, and gender. In addition, such methods cannot modify the style codes to control the intensity of the edited attribute (e.g., blondness in hair color editing) straightforwardly.

The second class of methods for facial attribute editing task builds on well-trained generative models with the motivation of benefiting from the well-organized feature space of such networks. In recent approaches, several studies select StyleGAN2 [17] as the pretrained generator, relying on its ability to organize the latent space as disentangled representations with semantically meaningful directions. Such approaches have two steps; in the first step, an input image is embedded in the generative model's latent space by training an additional encoder for the task [18, 19, 20, 21] or via latent optimization [22, 23]. Following this step, the embedded latent code is modified based on the discovered latent directions such that a transformation with such directions results in an edited image after decoding. The task of embedding images to generative models' latent space experienced significant improvements on the fixed pretrained GANs [24, 25, 26, 27, 28]. These models are referred to as *StyleGAN inversion-based methods* throughout this thesis.

However, such models are not trained end-to-end. Therefore, the encoding process cannot guarantee a latent representation that is accurate enough to represent the input image in the latent space of the selected pretrained generator, which limits editability. As a result of such limitation, the encoding process may lack the ability to faithfully reconstruct the input image, which fails to match the objective of preserving identity during editing. Additionally, since the generator is not explicitly trained for such a translation task, there is no guarantee that the directions learned for different semantics would be disentangled (e.g., eyeglasses versus age).

To overcome the challenges mentioned above in the facial attribute editing task, we propose a novel image-to-image translation framework, VecGAN, an end-to-end approach empowered by interpretable latent directions learned in training time. Unlike the other efforts in end-to-end image translation approaches, our framework does not require a separate style encoder. Furthermore, it performs the translation in the latent space directly by the latent directions learned. The directions for attribute editing are learned in the latent space with a disentanglement-focused objective, which encourages learning linearly independent directions to perform identity-preserving translations. The other component of our framework is the controllable strength of semantic change, which models the editing intensity with a scalar value. This scalar can be sampled from a distribution or extracted from a reference image by projecting it into the latent space. With such a design, VecGAN achieves significant improvements over state-of-the-art methods for global and local edits and provides a control mechanism for editing strength.

VecGAN is motivated by combining the two approaches in image editing, where an end-to-end trainable network is offered that perform edits in the latent space. This design is encouraged by the findings that well-trained generative models organize their latent space as disentangled representations with semantically meaningful directions [26, 24, 29]. These works show that images can be mapped to the GANs' latent space, and edits can be achieved by manipulating latent space. However, as these models are not trained end-to-end, and the input image may not match the imaging distribution learned by the pretrained generator, the results are sub-optimal, where we address the problem by adopting end-to-end training.

To enable VecGAN, we adopt a deeper encoder-decoder architecture compared to the preceding image-to-image translation methods. Previous methods, such as HiSD [15], use a network consisting of a small encoder-decoder structure that downsamples the input image only by four times. As we want to achieve an organization in the latent space to enable us to learn attribute-specific latent directions, images should be encoded into a spatially smaller feature space. We enable this behavior with a deep encoder-decoder architecture, allowing our network to understand the image completely. Even though such an architecture enables understanding an input image at the semantic level with the improved receptive field, the network then faces the problem of reconstructing all details in the input image. To solve this problem, we use an attention-based skip connection between the encoder and decoder, which enables information flow with feature selection capabilities. In summary, our main contributions are:

- We propose VecGAN, an image-to-image translation framework trained end-to-end with interpretable latent directions. Unlike previous works, our approach achieves local and global edits with a single deep encoder-decoder structure instead of a separate style network.

- VecGAN enables both attribute strength manipulation and attribute strength copy with the control knob proposed on the translation strength. As we rely on a latent code learned by an encoder for editing, we can perform reference-guided manipulations on a shared encoder. To do so, we obtain the latent vectors corresponding to the input and reference images and then manipulate the desired attribute by projection using the learned directions.

- To train VecGAN, we propose a novel objective to enable both learning linearly independent latent directions and stable training for our framework.

- We introduce an attention-based skip connection mechanism to enable information selection in the residual connection.

- We compare our method with several state-of-the-art image translation methods, especially with popular pretrained GAN-based models. We provide results with an extensive number of metrics for quality, attribute edit accuracy, identity, and background preservation. Our results show the effectiveness of our framework with significant improvements over the state-of-the-arts.

- We report metrics as the strength of editing increases for our and competing methods. We also analyze the projected style codes and show that they can classify the targeted attributes of images, e.g., hair color and smile.

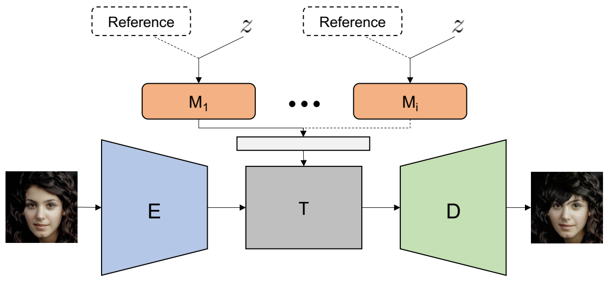
Chapter 2

Related Work

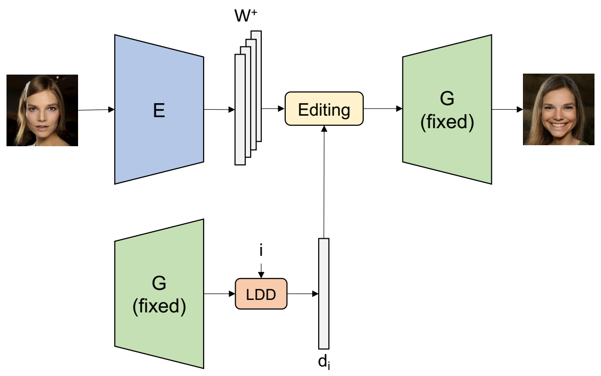
2.1 Image-to-Image Translation Networks

Image-to-image translation algorithms aim at preserving a given content from the input image while changing targeted attributes. They find a wide range of applications from translating semantic maps into RGB images [2, 4, 30], RGB images to portrait drawings [31] and very popularly to editing faces [9, 10, 11, 12, 15, 32, 33, 34]. These algorithms set an encoder-decoder architecture and train the models with reconstruction and adversarial losses [35]. For facial attribute editing, the problem is formulated as the target domain contains the target attribute and the input domain does not, where the translation occurs from the input domain to the target domain [15].

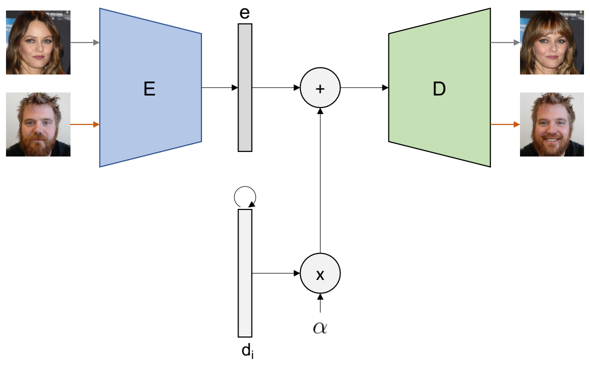
When the translation is designed in a unimodal setting, images are processed with an encoder and decoder to output translated images from one domain to the other [2] where the mapping between domains is one-to-one. The shortcoming of such a setup is that a single input image may correspond to multiple possible outputs, which makes the translation problem ambiguous. Because of that, multi-modal image translation models are proposed in which style is encoded separately from another image or sampled from a distribution [36, 9]. In such approaches, the



(a) End-to-end trainable translation networks, where semantic-specific mapper networks learn style codes.



(b) StyleGAN inversion-based models, where editing directions are learned over a fixed generator with a latent direction discovery (LDD) method.



(c) Our approach, where we learn an end-to-end trainable pipeline with latent directions that are learned in training time.

Figure 2.1: Overview of different Image-to-Image translation methods for facial attribute editing compared to our proposed approach. Combining end-to-end approaches with inversion-based methods, we learn an end-to-end trainable translation network with learnable latent directions.

generator, decoder, receives style and content information. This information can be combined either by a channel-wise concatenation [37], or by combining with a mask [15]. Alternatively, style and content information may be fed separately to the decoder network, where features representing the content pass through convolutional layers and style features goes through instance normalization blocks [36, 38]. Such works use two encoders, one for encoding style features and the other for encoding content features [15, 39]. However, such a combination of features makes the definition of style ambiguous. Usually, style is referred to the domain attributes one wants to change, and the content is the rest of the attributes, which is a vague problem definition. In this work, we design the attributes as learnable linear directions in the latent space, and we do not employ separate style and content encoders. Instead, we use a single encoder, resulting in a more intuitive framework.

2.2 Editing with pretrained GANs

Facial attribute editing is also shown to be possible with pretrained GANs. State-of-the-art GAN models organize their latent space with interpretable directions [40, 17, 41] such that moving along the direction only changes one image attribute. These directions are explored in supervised [24] and unsupervised ways [25, 26, 27], and many directions are found for face editing, e.g., directions that change the smile, pose, age attributes are found, to name a few. To edit a facial attribute of an input image, one needs to project the image to a latent code in GANs' latent space such that the generator reconstructs the input image from this latent code. Various architectures [19, 20] and objectives proposed to project an image to GANs' embedding. However, they suffer from reconstruction-editability trade-off [18]. That is, in one end where faithful reconstruction is promised, it may not lie in the actual distribution of GANs' latent space. Therefore, the directions do not work as expected, which prevents editing the image. Conversely, if the projection is close to the actual distribution, then the reconstruction is poor. We also show this behavior in Sec. 4.4 when comparing our method with state-of-the-art editing with pretrained GANs methods.

Even though these methods are not as successful as end-to-end trainable image-to-image translation networks, it is still quite remarkable when the generative network is only taught to synthesize realistic images; it organizes the use of latent space such that linear shifts on them change a specific attribute. Inspired by these findings, our image-to-image translation framework is designed similarly so that a linear shift in the encoded features is expected to change a single attribute of the input image. Unlike previous works, our framework is trained end-to-end for the translation task, allowing reference-guided attribute manipulation via projection, and does not suffer from the reconstruction-editability trade-off. We compare our approach with the existing methods in Fig. 2.1.

Chapter 3

Method

In this section, we provide an overview of the generator architecture and the training set-up. We follow the hierarchical labels defined by [15]. For a single image, its attribute for tag $i \in \{1, 2, ..., N\}$ can be defined as $j \in \{1, 2, ..., M_i\}$, where N is the number of tags (semantic categories) and M_i is the number of attributes for tag i. For example, i can be the hair color tag, and attribute j can take the value of black, brown, or blonde. A visual overview of the tags and attributes included in our framework is provided in Fig. 3.1.

Our framework has two main objectives. As the main task, we aim to perform the image-to-image translation task in a feature (tag) specific manner. While performing this translation, as our secondary objective, we also want to obtain an interpretable feature space that allows us to perform tag-specific feature interpolation.

3.1 Generator Architecture

For the image-to-image translation task, we set an encoder-decoder based architecture with a latent space translation module in the middle, as given in Fig. 3.2. We perform the translation in the encoded latent space, e, which is obtained

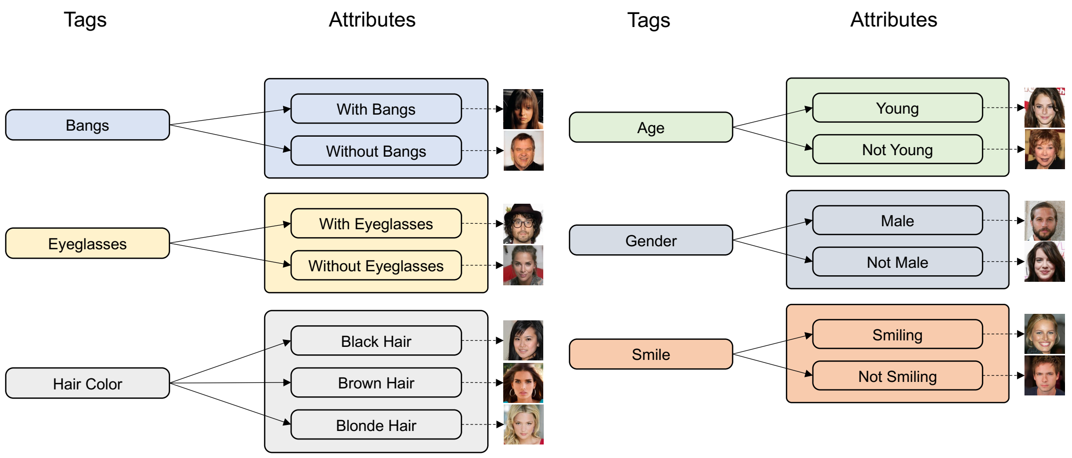


Figure 3.1: Label organization for the proposed framework. Inspired by the hierarchical labeling introduced in [15], we define our labels for bangs, eyeglasses, hair color, gender, age, and smile tags. Next to the labels, we provide a sample image for each tag-attribute pair.

by $e = E(x)$ where E refers to the encoder (Throughout this thesis, we use e for encoded features and d for decoded features, for the subjected feature resolution). The encoded features go through a transformation T, which is discussed in the next section. The transformed features are then decoded by the decoder G to reconstruct the translated images. The image generation pipeline following feature encoding is described in Eq. 3.1.

$$\begin{aligned} e' &= T(e, \alpha, i) \\ x' &= G(e') \end{aligned} \tag{3.1}$$

Previous image-to-image translation networks [15, 39, 9] set a shallow encoder-decoder architecture to translate an image while preserving the content and a separate deep network for style encoding. In most cases, the style encoder includes separate branches for each tag.

The shallow architecture used to translate images prevents the model from making drastic changes in the images, which helps to preserve the person's identity. Our framework is different as we do not employ a separate style encoder and instead have a deep encoder-decoder architecture for translation. As we would like

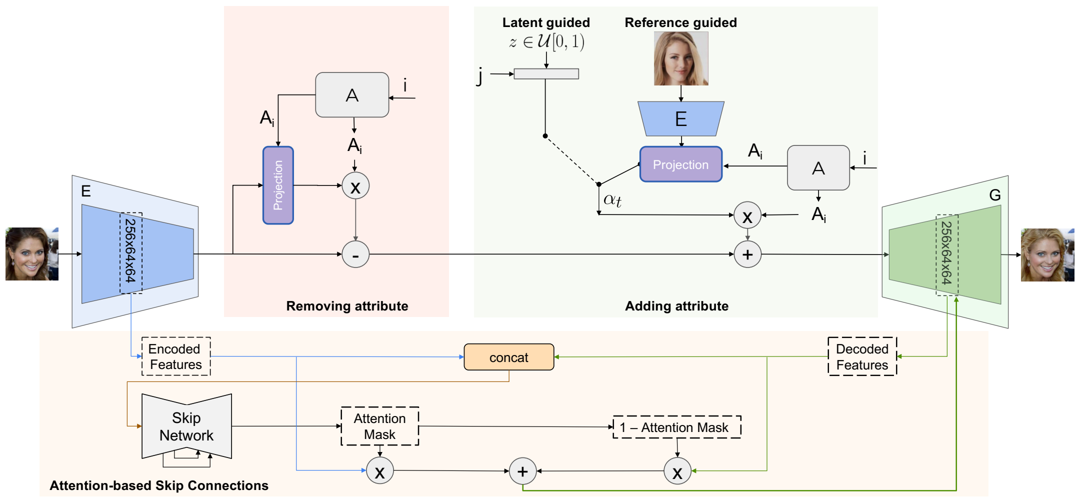


Figure 3.2: Our translator is built on the idea of interpretable latent directions. We encode images with an Encoder (E) to a latent representation from which we change a selected tag (i), e.g. hair color with a learnable direction A_i and a scale α. To calculate the scale, we subtract the target style scale (α_t) from the source style scale (α_s). This operation corresponds to removing an attribute in the amount present in the source and adding an attribute in the amount of the target. To remove the image's attribute, the source style is encoded and projected from the source image. To add the target attribute, the target style scale is sampled from a distribution that is mapped for the given attribute (j), e.g. black, blonde, or encoded and projected from a reference image. We also propose an attention-based skip connection module to transfer selected features without an information bottleneck to the decoder.

to organize the latent space in an interpretable way during training, our framework requires a full understanding of the image and, therefore, a larger receptive field which results in a deeper network architecture. A deep architecture with decreasing feature size, on the other hand, faces the challenges of reconstructing all the fine details from the input image.

With the motivation of helping the network in preserving tag-independent features such as the fine details from the background, we use attention-based skip connections between our encoder and decoder as described in Sec. 3.3.

The architectural details of the encoder and decoder are as follows: For the encoder, following a 1×1 convolution, we use 8 successive blocks that perform downsampling, which reduces feature map dimensions to 1×1. In our decoder, we have an architecture symmetric to the encoder, which is composed of 8 successive upsampling blocks. Except for the last downsampling block and the first upsampling block, we use instance normalization denoted as (+IN). The channels increase as {32, 64, 128, 256, 512, 512, 512, 1024, 2048} (for output resolution 256×256) in the encoder and decrease in a symmetric way in the decoder. The complete architecture for the generator is provided in Fig. 3.2 illustrating the translation module and in Fig. 3.3 illustrating the building blocks. Each DownBlock and UpBlock has a residual block with 3×3 convolutional filters followed by a downsampling or an upsampling layer, respectively. For downsampling, we use average pooling; and for upsampling, we use nearest-neighbor. We use the LeakyReLU activation layer (with a negative slope of 0.2) and instance normalization layer in each convolutional module with the exception of the last downsampling and first upsampling block. The details of the building blocks of our framework are provided in Fig. C.1.

3.2 Translation Module

To achieve a style transformation, we perform tag-based feature manipulation in a linear fashion on the latent space. First, we set a feature direction matrix A, which contains learnable feature directions for each tag as its rows. In our formulation, we show the direction learned for tag i as A_i. The direction matrix A is randomly initialized and learned during training which enables us to obtain directions that are compatible with the latent space learned.

Our translation module is formulated in Eq. 3.2, which adds the desired shift on top of the encoded features e similar to [25].

$$T(e, \alpha, i) = e + \alpha \times A_i \tag{3.2}$$

We compute the shift by subtracting the target style from the source style as given in Eq. 3.3 where they are represented as α_t and α_s respectively.

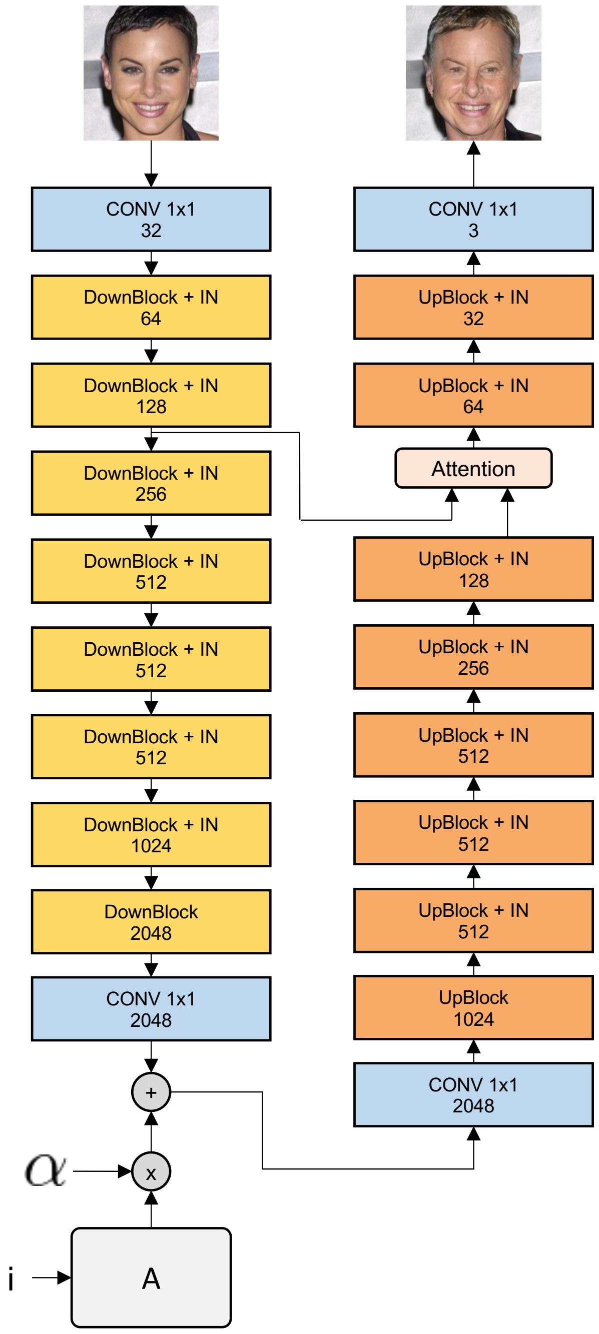


Figure 3.3: Generator architecture of VecGAN, illustrating the building blocks for the encoder and decoder networks shown over an example translation for age editing. We provide the interior details of UpBlock and DownBlock structures in Fig. C.1. In addition to structure names, we also show the presence of instance normalization (+IN) and the number of output channels for a given block.

$$\alpha = \alpha_t - \alpha_s \tag{3.3}$$

Since the translations are designed as linear steps in learnable directions, we find the style shift by subtracting the target attribute scale from the source attribute scale. This way, the same target attribute α_t can have the same impact on the translated images independent from the attributes of the original image. For example, if our target scale corresponds to brown hair, the source scale can be coming from an image with blonde or black hair, but since we take a step in the difference of the scales, they can both be translated to an image with the same shade of brown hair.

There are two alternative pathways to extract the target scale α_t for a given feature (tag) i. The first pathway, named the latent-guided path, samples a $z \in \mathcal{U}[0,1)$ and applies a linear transformation $\alpha_t = w_{i,j} \cdot z + b_{i,j}$, where α_t denotes sampled shifting scale for tag i and attribute j. We learn linear transformation parameters $w_{i,j}$ and $b_{i,j}$ in training time. As we aim to obtain a continuous translation over attributes (a scale modeling a continuous change in color for the case of hair color tag), we reformulate this sampling equation as $\alpha_t = (a_{i,j+1} - a_{i,j}) \cdot z + a_{i,j}$ which corresponds to sampling a point from a line segment. Since we design the consecutive intervals with shared endpoints, we succeed in learning a continuous scale distribution for tag i.

Here tag i can be hair color, and attribute j can be blonde, brown, or black hair. We learn a different transformation module for each attribute, denoted as $M_{i,j}(z)$, which contains the line endpoints as its parameters. Since we learn a single direction for every tag, e.g. hair color, this transformation module can put the initially sampled z's into the correct scale in the linear line based on the target hair color attribute. As the other alternative pathway, we encode the scalar value α_t in a reference-guided manner. We extract α_t for tag i from a provided reference image by first encoding it into the latent space, e_r, and projecting e_r by A_i as given in Eq. 3.4.

$$\alpha_t = P(e_r, A_i) = \frac{e_r \cdot A_i}{||A_i||} \tag{3.4}$$

In the reference guidance set-up, we do not use the information of attribute j, since it is encoded by the tag-specific features of the image that are found via projection.

The source scale, α_s, is obtained in the same way we obtain α_t from the reference image. We perform the projection for the tag we want to manipulate, i, by $P(e, A_i)$, where the latent encoding for the input image is projected on the target latent direction for the given tag. We formulate our framework with the intuition that the scale controls the amount of features to be added. Therefore, especially when the attribute is copied over from a reference image, the amount of features that will be added will be different based on the source image. For this reason, we find the amount of shift by subtraction as given in Eq. 3.3. Our framework is intuitive and relies on a single encoder-decoder architecture.

3.3 Attention-based Skip Connections

We benefit from attention-based skip connections that merge encoded and decoded features with an attention mask to enable feature-aware residual connections. Our architecture includes a skip network S, which calculates an attention map from the concatenation of encoded and decoded features (feature resolution is 64x64 as illustrated in Fig. 3.2). The equation summarizing our approach is provided as Eq. 3.5 for encoded features e and decoded features d with a feature resolution of 64x64.

$$d' = e \cdot \sigma(S(e||d)) + d \cdot (1 - \sigma(S(e||d))) \tag{3.5}$$

Here, $(e||d)$ refers to concatenation, and σ is the sigmoid function. In the skip network architecture, we aim to compute an attention mask that reflects

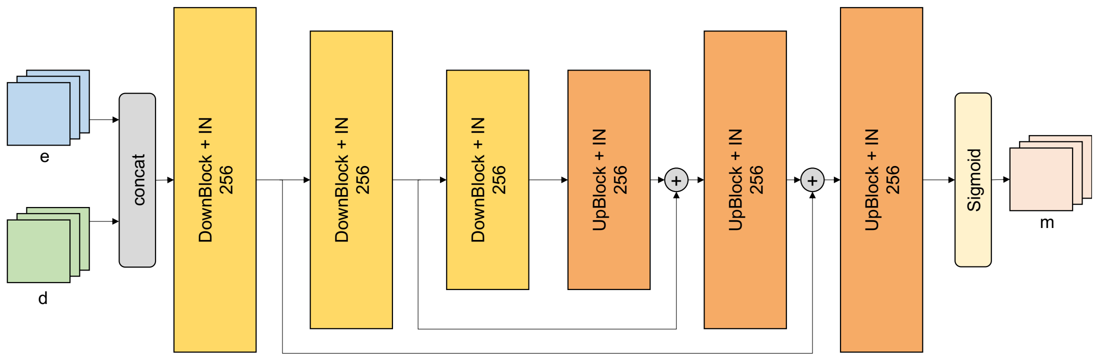


Figure 3.4: Architecture of Skip Network S, which is used for obtaining the attention mask in our attention-based skip connections. The encoded features and decoded features are represented as e and d, where both have feature dimensions $256 \times 64 \times 64$. Taking the concatenation of these tensors as an input, skip network S outputs a $256 \times 64 \times 64$ attention mask m, which filters out the information to be passed through the residual connection.

a maximal understanding of the image. In order to achieve this, we use an architecture inspired by U-Net[42], which downscales the concatenated features to a resolution of 256x8x8. By doing this, we achieve a mask computed with a considerable amount of receptive field while preserving the input features with the residual connections. We use the residual blocks from the original generator to upsample and downsample the features (where instance normalization is enabled) in the skip network [14] with channel size 256. A visual overview of the skip network S is provided in Fig. 3.4.

The skip network takes both encoded and decoded features to identify which parts should be taken from the encoded features from the original image, and which parts should be taken from the edited decoded features.

3.4 Training pathways

We train our network using two different paths by modifying the translation paths defined in [15]. For each iteration during training, we sample a tag i for the shift direction, a source attribute j as the current attribute, and a target attribute

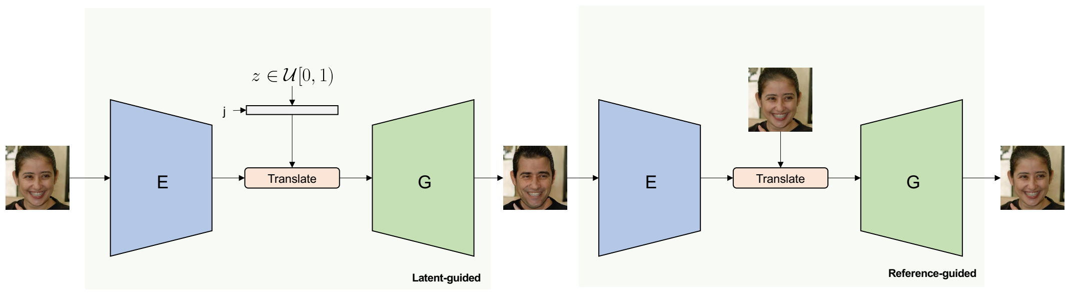


Figure 3.5: Overview of cycle-translation path. To enable a self-supervised cycle, we first perform a latent-guided edit using a $z \in \mathcal{U}[0,1)$, which results in the generation of x_t. Following this initial translation, we attempt to translate back the edited image using the input image as a reference which gives us the final output of this path, x_c.

$\hat{j}$, which is kept constant over a training batch. The two paths that the input images undergo during training are defined below.

Non-translation path. To ensure that the encoder-decoder structure preserves the images' details, we reconstruct the input image without applying any style shifts. The resulting image is denoted as x_n as given in Eq. 3.6.

$$x_n = G(E(x)) \tag{3.6}$$

Cycle-translation path. We apply a cyclic translation to ensure we get a reversible translation from a latent guided scale by using only one image as an input. In this path, we first apply a style shift by sampling a $z \in \mathcal{U}[0,1)$ and obtaining target scale α_t with $M_{i,\hat{j}}(z)$ for target attribute $\hat{j}$. The translation step uses α as the shift amount, which is obtained by subtracting the target style α_t from the source style α_s. We obtain the source style scale α_s by projecting the image encoding to the target direction ($\alpha_s = P(e,i)$). The decoder then generates an image, x_t, as given in Eq. 3.7 where e stands for the encoded features from input image x, $e = E(x)$.

$$x_t = G(T(e, M_{i,\hat{j}}(z) - P(e,i), i)) \tag{3.7}$$

Then by using the original image x as a reference image, we aim to reconstruct the original image by translating x_t. Overall, this path attempts to reverse a latent-guided style shift with a reference-guided shift. The second translation is given in Eq. 3.8 where $e_t = E(x_t)$. A visual overview of this cyclic translation is provided in Fig. 3.5.

$$x_c = G(T(e_t, P(e, i) - P(e_t, i), i)) \tag{3.8}$$

In our learning objectives, we use x_n and x_c for reconstruction and x_t and x_c for adversarial losses, and $M_{i,j}(z)$ for the shift reconstruction loss and for the adversarial loss. Details about the learning objectives are given in the next section.

3.5 Learning objectives

Given an input image $x_{i,j} \in \mathcal{X}_{i,j}$, where i is the tag to manipulate and j is the tag-specific attribute for that image, we optimize our model with the following objectives. In our equations, $x_{i,j}$ is shown as x.

3.5.1 Adversarial Objective

We learn a discriminator using an architecture with decreasing feature resolution and increasing channel size. Like the generator, we build our discriminator with channel sizes of {32, 64, 128, 256, 512, 512, 512, 1024, 2048}, reducing the feature resolution to 1x1. We concatenate the extracted style α_s from the input image to this latent code and apply a 1x1 convolution. This final convolution is specific to each tag-attribute pair, so the model can use this information.

During training, our generator performs a style shift either in a latent-guided or a reference-guided way, resulting in a translated image. In our adversarial loss, we receive feedback from the two steps of the cycle-translation path. As the first component of the adversarial loss, we feed a real image x with tag i and attribute j to the discriminator as the real example. To give adversarial feedback to the latent-guided path, we use the intermediate image generated in the cycle-translation path, x_t. Finally, to provide adversarial feedback to the reference-guided path, we use the final output of the cycle-translation path x_c. Only x acts as a real image; both x_t and x_c are translated images and are treated as fake images with different attributes. The discriminator aims to classify whether an image is real or fake, given its tag and attribute. The overall adversarial objective is given in Eq. 3.9.

$$\begin{aligned}\mathcal{L}_{adv} = 2log(D_{i,j}(x)) + log(1 - D_{i,\hat{j}}(x_t)) \\ +log(1 - D_{i,j}(x_c))\end{aligned} \tag{3.9}$$

3.5.2 Shift Reconstruction Objective

As the cycle-consistency loss performs reference-guided generation followed by latent-guided generation, we utilize a loss function to make these two methods consistent with each other [43, 36, 44, 15]. Specifically, we would like to obtain the same target scale, α_t, both from the mapping and the encoded reference image generated by the mapped α_t. The loss function penalizing the distance of the mentioned style scales is given in Eq. 3.10.

$$\mathcal{L}_{shift} = ||M_{i,j}(z) - P(e_t, i)||_1 \tag{3.10}$$

The parameters formulating our distance function, $M_{i,j}(z)$ and $P(e_t, i)$, are calculated using the cycle-translation path that is described in Eq. 3.7 and 3.8.

3.5.3 Image Reconstruction Objective

In all of our training paths, the purpose is to be able to regenerate the original image again. To supervise this behavior, we use $\mathcal{L}_1$ loss as our reconstruction objective. In our formulation, x_n and x_c are outputs of the non-translation and cycle-translation paths, respectively. Formulation of the reconstruction objective is provided in Eq. 3.11.

$$\mathcal{L}_{rec} = ||x_n - x||_1 + ||x_c - x||_1 \tag{3.11}$$

3.5.4 Orthogonality Objective

To encourage the orthogonality between directions, we use a soft orthogonality regularization term based on the Frobenius norm, which is given in Eq. 3.12. This orthogonality further encourages disentanglement in the learned style directions.

$$\mathcal{L}_{ortho} = \|A^T A - I\|_F \tag{3.12}$$

3.5.5 Disentanglement Objective

We intend to change the scale for the desired semantic in each translation. As a reflection of this criteria, we penalize the changes in the attributes that are not subjected to any translation. For translated tag i, using input scales α, and edited scales α', the disentanglement loss is formulated in Eq. 3.13 where we penalize the scale changes that are intended to be preserved. In the formulation, α_k represents the semantic scale for tag k with a minor abuse in notation. Scales are calculated based on the projection operation described in Eq. 3.4.

$$\mathcal{L}_{dis} = \sum_{k \neq i} ||\alpha_k - \alpha'_k|| \tag{3.13}$$

This disentanglement objective complements our orthogonality objective by encouraging linear independence between the learned directions. When the model is trained with an additional disentanglement loss, we observe that orthogonality loss drops to a lower value. In addition to this effect, disentanglement loss also encourages stability during training. As incompatible weight updates on the latent directions and the encoder module would disrupt the latent space significantly after the subjected translation, disentanglement loss achieves stability by penalizing such changes. The stabilizing effect of the disentanglement loss is discussed in Sec. 4.3.2.1 with experimental results.

3.5.6 Sparsity Objective

During training, we also include an additional loss term to encourage the sparsity of the learned latent directions to maximize the information represented by them. To do so, we add an $\mathcal{L}_1$ loss term on matrix A, which contains the latent directions learned during training as its rows. The sparsity loss is formulated in Eq. 3.14 where N represents the total number of directions and L represents the dimensionality of the latent directions learned.

$$\mathcal{L}_{sparse} = \sum_{i}^{N} \sum_{l}^{L} ||A_{i,l}|| \tag{3.14}$$

3.5.7 Full Objective

Combining all of the loss components described, we reach the overall objective for optimization as given in Eq. 3.15. We construct our overall objective with the following hyperparameters; $\lambda_a = 1$, $\lambda_{rec} = 1.5$, $\lambda_s = 1$, $\lambda_o = 1$ and $\lambda_{sp} = 0.1$. We use a learning rate of 10^{-4} and train our model for 600K iterations with a batch size of 4 on a single GPU.

$$\begin{aligned}\min_{E,G,M,A} \max_{D} \lambda_a \mathcal{L}_{adv} + \lambda_s \mathcal{L}_{shift} + \lambda_r \mathcal{L}_{rec} \\ + \lambda_o(\mathcal{L}_{ortho} + \mathcal{L}_{dis}) + \lambda_{sp}\mathcal{L}_{sparse}\end{aligned} \tag{3.15}$$

Chapter 4

Experiments

4.1 Dataset and Settings

We train our model on the CelebA-HQ dataset [45], which contains 30,000 face images. To extensively compare with state-of-the-arts, we follow two training & evaluation protocols as follows:

4.1.1 Comparing with End-to-End Approaches

In our first setting, we follow the setup from HiSD [15] to compare our method with end-to-end trainable image translation algorithms. Following HiSD, we use the first 3000 images of the CelebA-HQ dataset as the test set and 27000 as the training set. These images include annotations for different attributes from which we use hair color, the presence of glass, and bangs attributes for translation tasks in this setting. The images are resized to 128×128. Throughout this thesis, this setup is referred to as setting I.

Following the evaluation protocol proposed by HiSD [15], we compute FID scores on the bangs addition task. For each test image without bangs, we translate them to images with bangs using latent-based and reference-based guidance. In our evaluation setup, we use the images that do not contain the target attribute as the source images to perform the translation. Using these generation results, we report the FID metric by comparing them with images from the test set containing the target attribute.

In all of our experiments, we calculate FID on five different generation sets to make our measurements robust to the effect of randomness we introduce in the sampling process. These sets include the translations of the same test images, with randomly sampled target style scales, α_t. The evaluation procedure for latent-guided and reference-guided synthesis are as follows:

- **Latent-guided evaluation:** In this setup, we generate 5 images for a given input to be edited. For each sample we generate, we sample a random $z \in \mathcal{U}[0, 1)$.
- **Reference-guided evaluation:** Similar to the latent-guided setup, we generate 5 images per input to construct the five evaluation sets we average on. For each of these samples, we sample a random reference image with the target tag-attribute pair from our dataset, to extract the target style scale α_t.

After constructing our evaluation sets by generating the required samples, we calculate the FID score for each of them. Then, to report the final FID score, we average these scores. The procedure for the construction of the mentioned image sets is shown in Fig. 4.1.

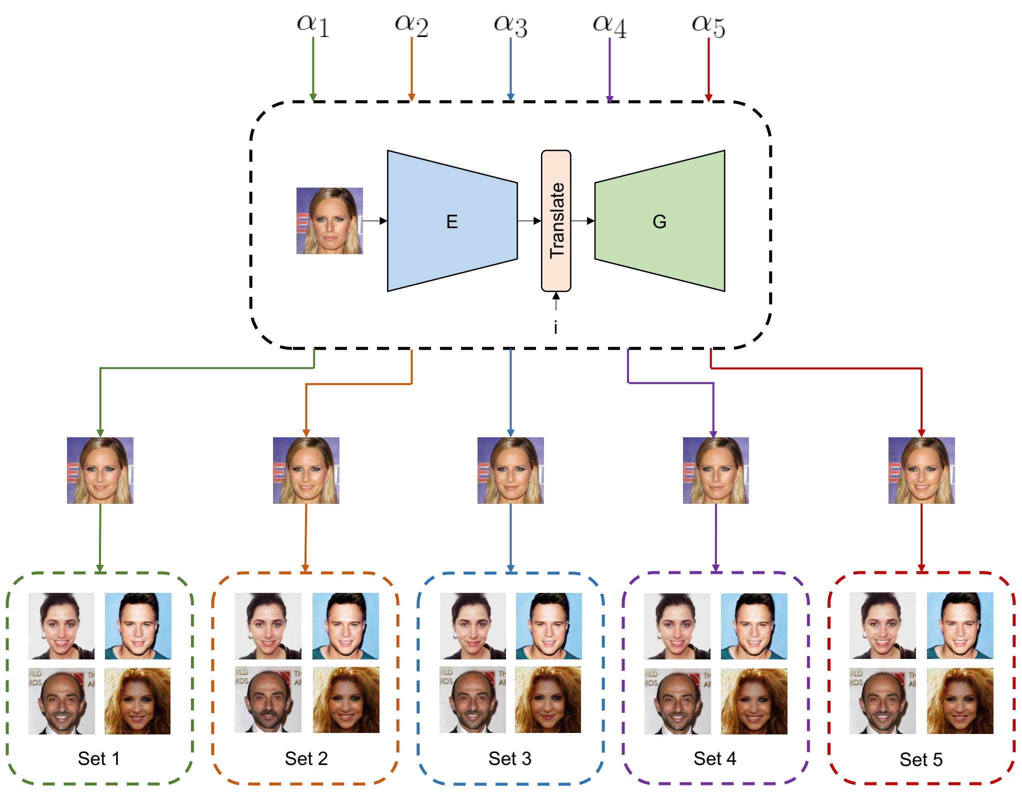


Figure 4.1: Constructing evaluation sets for smile addition task. On both Setting I and II, we calculate our metrics by averaging over 5 sets of generated images. To do so, we use 5 randomly sampled α_t values as the target style scale for translation. These scale values are obtained either from a $z \in \mathcal{U}[0, 1)$ for latent-guided experiments or an image with the target tag-attribute pair for reference-guided experiments.

4.1.2 Comparing with StyleGAN2-based Methods

We use our second setting to comprehensively compare our method with StyleGAN2-based inversion and editing methods. For this setting, the training/test split is obtained by re-indexing each image in CelebA-HQ back to the original CelebA dataset and following the standard split of CelebA. This results in 27,176 training and 2,824 test images. Our models are trained for hair color, the presence of glasses, bangs, age, smiling, and gender attributes, where all of these translations are learned simultaneously. Images are resized to 256×256 resolution, which is the dimension StyleGAN2-based inversion methods use. This setup is mentioned as setting II in this thesis.

We evaluate our model on bangs, hair color, gender, age, and smile editing tasks using the FID metric. Using this setup, we aim to evaluate our translations with a mixed amount of feature strengths. As our translations include both global and local edits, we demonstrate the effectiveness of our framework in terms of image understanding. In addition to evaluating these edits by using randomly sampled scales, as used in setting I, we also evaluate our model with translations with increasing strength for smile removal and bangs addition tasks. To achieve this, we set the feature strength as a constant for each iteration and increment the target scale α_t in every step.

This setup also uses additional evaluation metrics as mentioned in section 4.2 to evaluate the quality of the translations and identity preservation. In this regard, we select the smile editing task as it requires a general understanding of the input image and the bangs editing as it requires preserving the identity significantly by performing local edits. By benchmarking our model in such a way, we evaluate our framework with varying editing strengths in both local and global editing tasks.

4.2 Metrics

We mostly build our evaluation on the FID metric as in previous works. Additionally, for smile removal and bangs addition tasks, we evaluate our results on other metrics, such as classification accuracy, identity preservation, and background preservation for setting II. For each of the metrics, we report the average of the scores for the five image sets containing our generation results, except for our analysis on translation strength with fixed style scales. We summarize the image generation procedure in Fig. 4.1. The metrics we use are described as follows:

Frechet Inception Distance (FID): For the FID metric [46], we calculated the fréchet distance between the feature tensors of original and generated images using their mean μ and covariance matrix Σ, which are obtained by the Inception-V3 model. The FID calculation is formulated in Eqn. 4.1. In this formulation, μ_r, Σ_r are feature statistics for original images, and μ_g, Σ_g are statistics for the generated images. The operation $Tr(\cdot)$ stands for the trace of the given matrix.

$$FID(\mu_r, \Sigma_r, \mu_g, \Sigma_g) = ||\mu_r - \mu_g||^2 + Tr(\Sigma_r + \Sigma_g + 2(\Sigma_r \Sigma_g)^{1/2}) \tag{4.1}$$

We set up the FID evaluation based on the attributes that are edited. For example, for smile addition attribute edit, we edit the images that have a negative smile attribute from the validation set. Those become our generated images. For the ground-truth distribution, we filter the images in our validation set for the ones that have a positive smile attribute. Therefore, for the FID to be lower, the edit needs to be applied since our source images come from non-smiling images and target distribution images are smiling ones. The calculation procedure is visualized in Fig. 4.2.

Accuracy (Acc): We train two classifiers for smile and bangs tags on the training split of the CelebA-HQ dataset. We use an ImageNet pretrained ResNet-50 model and fine-tune it for the corresponding classification tasks. The resulting smile and bangs classification models achieve 95% and 96% accuracy, respectively. We use these classifiers to evaluate the accuracy of the generated images to test if the attribute is correctly manipulated. By doing so, we also evaluate the success of the attribute strength manipulation capabilities of our framework by evaluating the generated samples on changing target scales, α_t. Further details about the classification models trained are provided in Appendix B.

Identity Preservation (ID): To calculate the ID metric, we use the CurricularFace model [47] to calculate the similarity between the original and generated

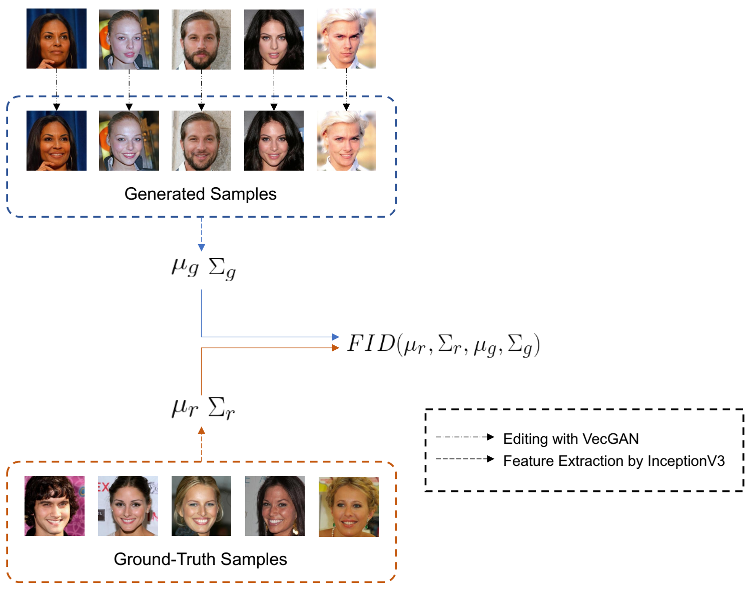


Figure 4.2: FID calculation for smile addition edit. Using the images with a negative smile attribute from our validation set as our source distribution, we initially translate them using our framework. Following this step, the generated images are compared with the ones having a positive smile attribute over fréchet inception distance (FID).

images. The CurricularFace model uses ResNet-101 as a backbone for the feature extraction and outputs an embedding representing the identity of the input image. We calculate the cosine similarity score between the output embeddings of each original and edited image pair. The scoring process is illustrated visually in Fig. 4.3.

Background Preservation (BG): For the BG metric, we first use the segmentation masks to separate the parts of faces from the CelebAMask-HQ dataset [48] to form background masks for images in our validation set. Using these masks, we calculate the mean structural similarity index (MSSIM) [49] between the backgrounds of the original and edited images, which are obtained after masking the images with the obtained background segmentation masks. The process of calculating the BG metric is summarized in Fig. 4.4.

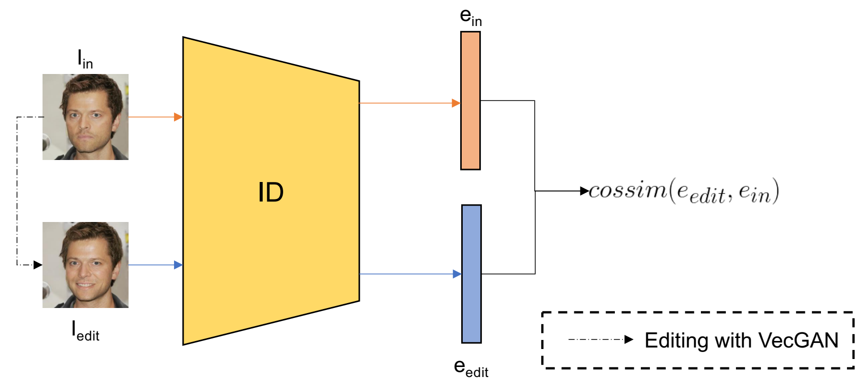


Figure 4.3: Identity Preservation (ID) metric calculation for smile addition task. After editing the input with a negative smile attribute with smile addition edit, we feed both the generated image I_{edit} and input image I_{in} to the CurricularFace model. Then by using the embeddings outputted from the input image e_{in} and the generated image e_{edit}, we report the cosine similarity between them as the ID score.

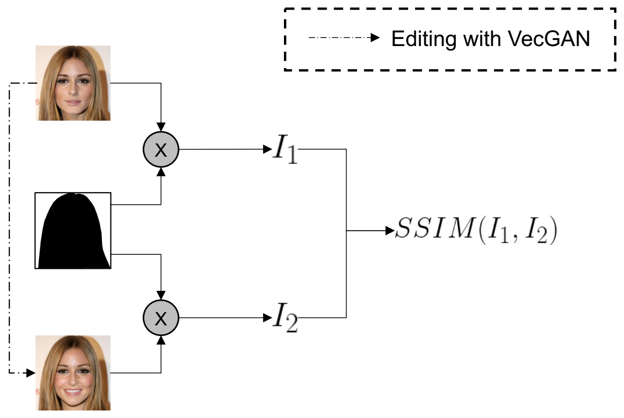


Figure 4.4: Background Preservation (BG) metric calculation for smile addition task. After editing the input image with a negative smile attribute, we multiply both the generated image and input image with the background mask obtained from the CelebAMask-HQ dataset [48]. Then by using the outputs of these two multiplications, we calculate the structural similarity index [49] in between to get the BG score for a given input image.

Abl. Index	Method	Lat.	Ref.
I-A	Shallow	21.30	20.94
	Deep w/o skip	88.62	127.65
	Deep w/ all skip	273.80	273.97
	Deep w/ single skip (VecGAN$_{\text{base}}$)	20.17	20.72
I-B	w/o Orthogonality	21.98	22.50
	w/o Sparsity	24.07	22.43
II-A	w/ Disent.	20.23	20.57
II-B	w/ Disent. + Attn. Skip	20.15	20.08
	w/ Disent. + Attn. UNet Skip	19.98	19.87
	w/ Disent. + Attn. UNet Skip w/ dec.	**19.65**	**19.62**

Table 4.1: Ablation study on setting I. Each ablation is labeled with an index representing the category of the experiment. We label latent and reference-guided results as Lat. and Ref. respectively.

4.3 Ablation Study

We conduct ablation studies on network architecture and loss objectives as given in Table 4.1 using evaluation setting I, which is explained in Sec. 4.1.1. We present our ablations in two parts, where we experiment on the base version first in Sec. 4.3.1, which excludes attention-based skip connections and disentanglement loss as presented in [14]. Following these ablations, we experiment on the proposed disentanglement loss and attention-based skip connections in Sec. 4.3.2.

4.3.1 Experiments on the Base Version

The vanilla version of the proposed framework is labeled as the base version throughout this section. Different than the final version of VecGAN, the disentanglement loss and attention-based skip connections are excluded in this version. We present ablations from two different perspectives in this section. First, we investigate the network architecture by experimenting with the network depth and skip connection frequency (I-A). Following these ablations, we experiment on the orthogonality loss and sparsity losses (I-B), which are two optimization objectives that are fundamental to the success of our framework. The ablations of the base version are indexed as I.

4.3.1.1 Ablations on Network Architecture (I-A)

We first experiment with a shallower architecture without a skip connection in between. This version is labeled as shallow since the encoder decreases the input dimension from 128×128 to 8×8. Even if this version gives reasonable scores for bangs addition edit, we are interested in a better latent space organization to perform global edits. To do so, we use a deeper encoder-decoder architecture where encoded latent space resolution goes as low as 1x1, and decoded to 128x128. We refer to this architecture as deep.

Taking the deep version without any skip connections as a starting point, we experiment with the skip connection frequency. Without any skip connections, our framework is unable to minimize the reconstruction loss, which results in a high FID value. On the other extreme, we conduct experiments by including skip connections at each resolution from the encoder to the decoder. With such an architecture, our framework is able to reconstruct the input well. However, the latent space is not well organized since the model tends to pass all the information through the skip connections, which instabilizes the training that results in a high FID value. Our architecture with a single skip connection after downsampling the input two times (using 32×32 features with input resolution 128×128) provides a good balance between the information flow from the encoder to the decoder and the latent space bottleneck.

In our experiments focusing on skip connection frequency, we formulate these connections with a summation of the encoded (e) and decoded features (d), which is formulated in Eq. 4.2.

$$d' = e + d \tag{4.2}$$

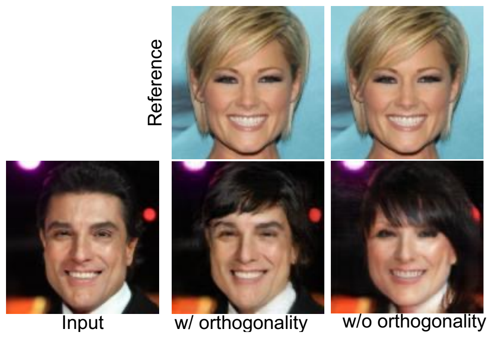


Figure 4.5: Qualitative results of ablation study for orthogonality loss. The bangs tag transferred from the reference image for the provided samples. In the presence of orthogonality loss, we observe that the bangs attribute can get disentangled from the gender tag, which is not the case when the orthogonality loss is not included.

4.3.1.2 Ablations on Orthogonality and Sparsity Objectives (I-B)

Following the experiments on the network architecture of the base version of our framework, we perform ablations on the effect of loss functions. First, in order to test our promise of learning linearly independent directions with an orthogonality objective, we remove the orthogonality loss that is effective on matrix A, which contains directions that are learned in training time. This change in our objective results in worse FID scores, but more importantly, we observe that the styles are not disentangled, e.g. changing bangs attribute changes also adds feminine features to the input image as it can be seen in Fig. 4.5. Even without this loss function, we observe that the orthogonality loss of matrix A decreases but to a higher value compared to when this loss is added to the final objective. This is because the framework and other loss objectives also encourage the disentanglement of attribute manipulations, and it shows the significance of the orthogonality of direction vectors. This also shows the importance of orthogonality in style disentanglement, and this targeted loss helps improve that significantly. We also observe that sparsity loss applied on the directional vectors stabilizes the training and lowers the FID values.

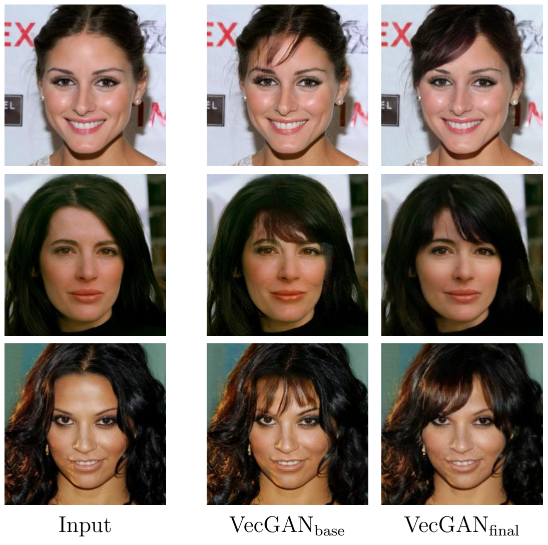


Figure 4.6: Ablation study between VecGAN$_{\text{base}}$ and VecGAN$_{\text{final}}$ on bangs editing task. In addition to quantitative improvements, we also achieve more realistic edits over our ablations.

4.3.2 Experiments on the Extended Version

Taking the base version of the framework that is presented in [14] as a starting point, we perform two groups of experiments where one is focused on the disentanglement loss and the other focuses on attention-based skip connections. With these two incremental changes, we form the final version of our framework. A visual comparison between the final model and the base model is provided in Fig. 4.6. The ablations building on top of the base version are indexed as II.

4.3.2.1 Ablations on Disentanglement Loss (II-A)

As our initial increment, we add the disentanglement loss to our learning objective to further encourage learning linearly independent latent directions, in addition to orthogonality loss. In our experiments, we observe that this loss is complementary to orthogonality loss and lowers the orthogonality loss value even more compared to the base model.

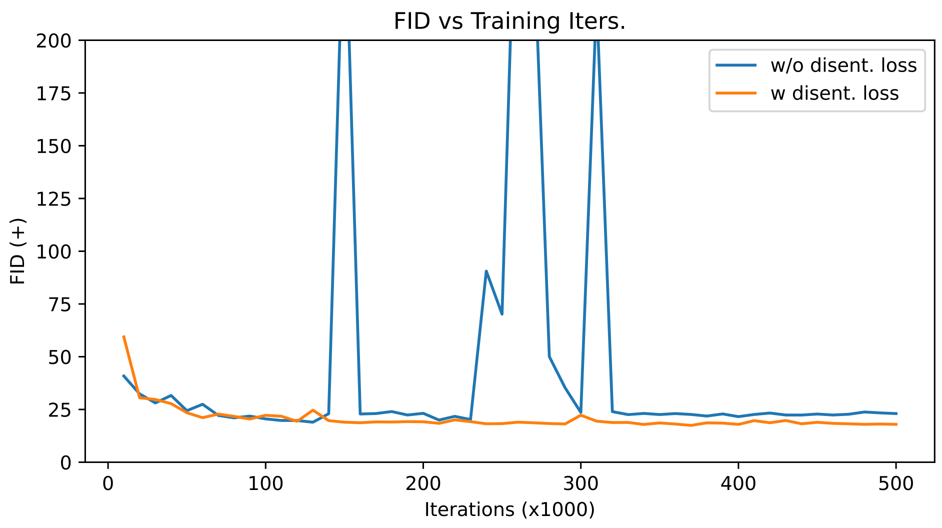


Figure 4.7: FID curves on smile addition task for models trained with and without disentanglement loss. In the absence of disentanglement loss, our framework experiences sudden peaks in FID over training iterations. By adding this additional loss term, we eliminate such instabilities in training.

Different than the orthogonality loss, this objective is sensitive to the extracted scales rather than the angles between the directions on its own, which enables us to stabilize the training further. We explain this behavior with the fact that the proposed loss function penalizes weight updates that harm the latent space organization for a given direction, which results in changing scales for the other directions. In case of updates on direction weights that disrupt the latent space, we experience sudden peaks in FID values, which means instability in training. We also experience the same effect in our experiments with Setting II, where the image resolution is set to 256×256. The change in the FID metric versus training iterations is provided for the smile addition task in Fig. 4.7 as proof of training stability.

4.3.2.2 Ablations on Attention-based Skip Connections (II-B)

In the base version of our framework, we include skip connections from the encoder to the decoder, which is formulated as a summation of encoded features with the decoded features. To enhance this mechanism, we introduce an approach enabling feature selection in the skip connections to ease the learning process for our framework. We experiment with different variants of the attention-based skip connection architectures in Table 4.1. We first experiment with attention-based feature summation as given in Eq. 3.5 but without the U-Net architecture, instead with a single layer module as well as without feeding the decoded features to the skip network S.

As it is also shown in Table 4.1, enabling this attention mechanism impacted our results in a positive way which lowers the FID values for bangs addition edits. As our next step, we experiment with the effect of the receptive field on the attention-based skip connections. To do so, we replace the attention module with a skip network inspired by the U-Net architecture [42] as explained in Section 3.3. Finally, we also feed the decoded features to this network with a concatenation operation, so that the model can explore the impact of edits on feature tensors to identify which features should be passed through the skip connection for high-quality edits. Each of these trials resulted in incremental improvements. Relying on our ablation study, we finalize our framework with attention-based skip connections that use both encoded and decoded features.

The results of our ablation study are summarized in Table 4.1, where the metrics are calculated by averaging the FID values over 5 runs, as explained in Sec. 4.1.1.

4.4 Comparisons with Competing Methods

We extensively compare our results with competing image translation methods. We present our comparisons with end-to-end approaches using our first evaluation setting, which is explained in Sec. 4.1.1, in Table 4.2. In this setup, we compare our method with SDIT [16], StarGANv2 [9], Elegant [11], and HiSD [15] models, which are all end-to-end trainable image-to-image translation models. Among these methods, HiSD learns a hierarchical style disentanglement, whereas StarGANv2 learns a mixed style code. Therefore StarGANv2, when translating images, also edits other attributes and does not strictly preserve the identity. HiSD achieves disentangled style edits with its tag-specific translation networks. However, HiSD learns feature-based local translators, an approach known to be successful on local edits, e.g. bangs, and their model is trained for bangs, eyeglasses, and hair color attributes. VecGAN achieves significantly better quantitative results than HiSD both in latent-guided and reference-guided evaluations, even though they are compared on a local edit task.

Fig. 4.8 shows reference-guided results of our final model and HiSD. As shown in the figure, both methods achieve attribute disentanglement, they do not change any other attribute of the image than the bangs tag. It is important to note that, HiSD learns feature-based local translators, which is a successful approach for local edits, e.g. bangs, eyeglasses, and hair color, but not smile, age, or gender. Our method achieves comparable visual and better quantitative results than HiSD on this local task and can also achieve global edits.

In our second second evaluation setup, setting II, we compare our method with state-of-the-art StyleGAN2 inversion-based methods, e4e [18], HyperStyle [19], HFGI [20], and StyleTransformer [21] in Table 4.3 and 4.4. We compare the methods on local (e.g. bangs, hair color) and global (e.g. age, smile, gender) attribute manipulations using the FID metric. For the smile and age edits, we use the directions explored by the InterfaceGAN method [24]. For the others (e.g. hair color, gender, bangs), we use the directions discovered by the StyleCLIP method [29]. The strength attribute is set to the one that achieves the best FID scores for a fair comparison.

Method	**Latent**	**Reference**	**Avg.**
SDIT [16]	33.73	33.12	33.42
StarGANv2 [9]	26.04	25.49	25.77
Elegant [11]	-	22.96	-
HiSD [15]	21.37	21.49	21.43
VecGAN$_{\text{base}}$ [14]	20.17	20.72	20.45
VecGAN$_{\text{final}}$	**19.65**	**19.62**	**19.64**

Table 4.2: Quantitative results for setting I on the bangs addition task. The provided scores are obtained using the FID metric.

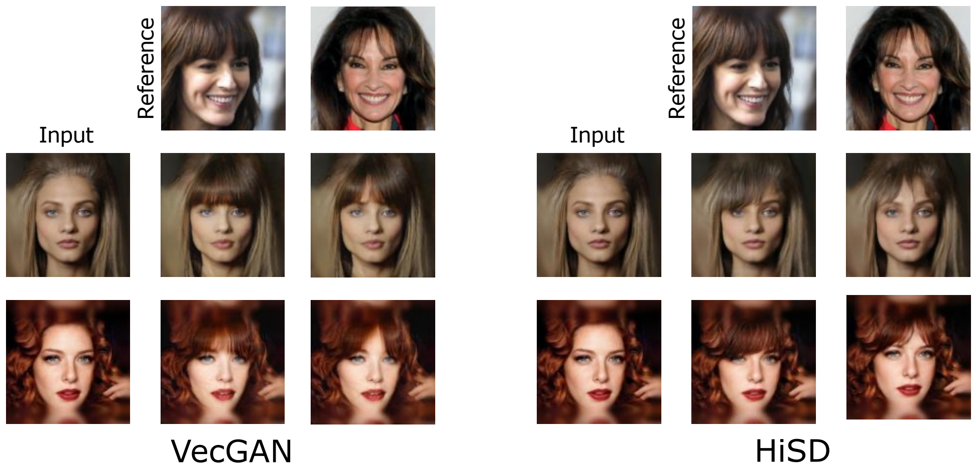


Figure 4.8: Qualitative results of bangs addition edit of our final model VecGAN$_{\text{final}}$ and HiSD. Given the reference images, methods extract reference attributes and edit input images accordingly. All methods achieve high-quality results. It is important to note that, HiSD learns feature-based local translators, which is a successful approach for local edits, e.g. bangs, eyeglasses, and hair color, but not smile, age, or gender. Our method achieves comparable visual and better quantitative results than HiSD on this local task and can also achieve global edits.

We achieve significant improvements on the presented attributes with state-of-the-art StyleGAN2 inversion-based methods. The quantitative results comparing our framework with the StyleGAN2 inversion-based methods are presented in Table 4.3 for local edits and Table 4.4 for global edits. Similar to our previous experiments, we average over five experiments while reporting the metrics. As the editing strength has been kept constant for StyleGAN2 inversion-based methods, we do not apply any repetitions for reporting FID metric for them.

Our method and StyleGAN inversion-based methods both provide a knob to control the editing attribute intensity. We obtain plots provided in Fig. 4.10 by changing the editing attribute intensity on smile and bangs editing tasks, which corresponds to setting the target style scale α_t to a constant in our framework. As we increase the intensity, edits become more detectable. We measure that with a tag-specific classifier; the details of these classifiers are explained in Sec. 4.2. Therefore, we plot FID, ID (Identity), and BG (Background Preservation) scores with respect to the attribute intensity measured by the accuracy of the classifier.

	Bangs		Hair Color		
Method	(+)	(-)	Blonde	Black	Brown
e4e [18]	53.29	53.62	69.26	64.92	40.93
HyperStyle [19]	41.37	47.32	60.50	59.63	35.31
HFGI [20]	40.54	45.06	67.92	60.09	35.15
StyleTran. [21]	44.66	51.91	55.96	66.64	35.15
VecGAN	**27.09**	**31.63**	**37.18**	**49.48**	**34.98**

Table 4.3: Quantitative results for setting II for local edits. Among the edits performed by our framework, we label bangs and hair color edits as local editing tasks. The results presented are in terms of the FID metric.

	Smile		Age		Gender	
Method	(+)	(-)	(+)	(-)	Female	Male
e4e [18]	35.01	37.91	52.28	60.38	36.55	62.27
HyperStyle [19]	25.25	24.64	52.23	44.60	42.77	62.11
HFGI [20]	23.49	26.58	45.30	46.20	43.98	64.84
StyleTran. [21]	27.64	32.71	55.24	55.67	46.34	57.81
VecGAN	**17.43**	**18.24**	**35.14**	**25.22**	**30.24**	**52.56**

Table 4.4: Quantitative results for setting II for global edits. The edits that cause global changes in the face, such as age, gender, and smile, are presented here. Similar to local edits, we present the results in terms of the FID metric.

Specifically, for VecGAN, we set α_t given in Eq. 3.3 to $\{0.0, 0.33, 0.5, 0.66, 1.0\}$, as five different intensity values for our edits. We provide qualitative results using this sampling strategy in Fig. 4.9 for the smile and bangs addition tasks. As we do not enforce any sign restriction on the direction of tag-specific vectors, we label the interpolated images with strength values as {1,2,3,4,5}.

For StyleGAN inversion-based methods, we set the strength parameter to $\{1, 2, 3\}$ for both bangs and smile edits. These models usually set the strength to 3 for successful smile edits. As shown in Fig. 4.10, VecGAN achieves better FID scores compared to others consistently. With the highest strength, where the accuracy of the classifier goes to 100% for all models, we observe that FID scores for StyleGAN inversion-based model scores drastically get worse, whereas our results are robust. We find that the ID score worsens as the edit strength increases for all models. That results from changes in the person and the limitations of the CurricularFace model. VecGAN achieves significantly better scores

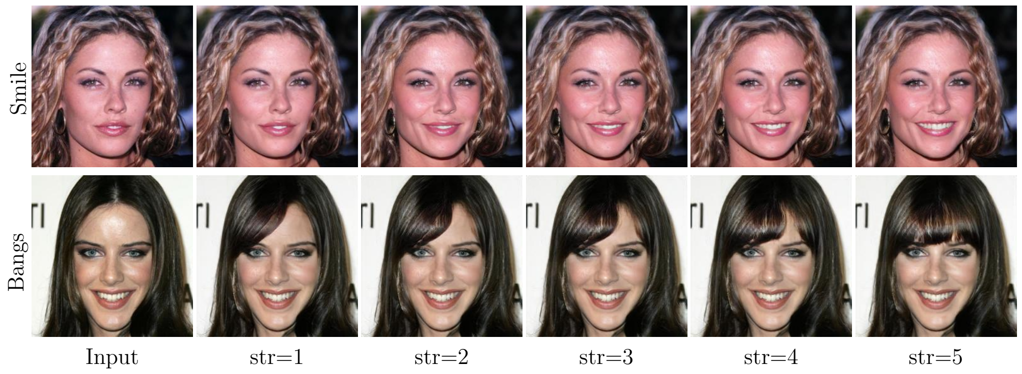


Figure 4.9: Feature strength interpolation results for bangs and smile addition edits. We use the z values of {0.0, 0.33, 0.5, 0.66, 1.0} to interpolate the editing intensity to obtain the target scales.

for the BG metric than StyleGAN inversion-based models.

We provide qualitative comparisons in Fig. 4.11. StyleGAN inversion-based methods do not faithfully reconstruct input images. They miss many details from the background and foreground. In contrast, VecGAN achieves high fidelity compared with the original images, with only targeted attributes manipulated naturally and realistically. This can be observed clearly in age edits, where VecGAN achieves successful edits by not making any additional changes to the input. In contrast, StyleGAN inversion-based methods add eyeglasses very frequently. This shows that VecGAN provides better disentanglement between correlated attributes, e.g., age and eyeglasses. This is because our models are trained end-to-end with labeled datasets for this task. We provide a more detailed analysis of the representations learned in the next section.

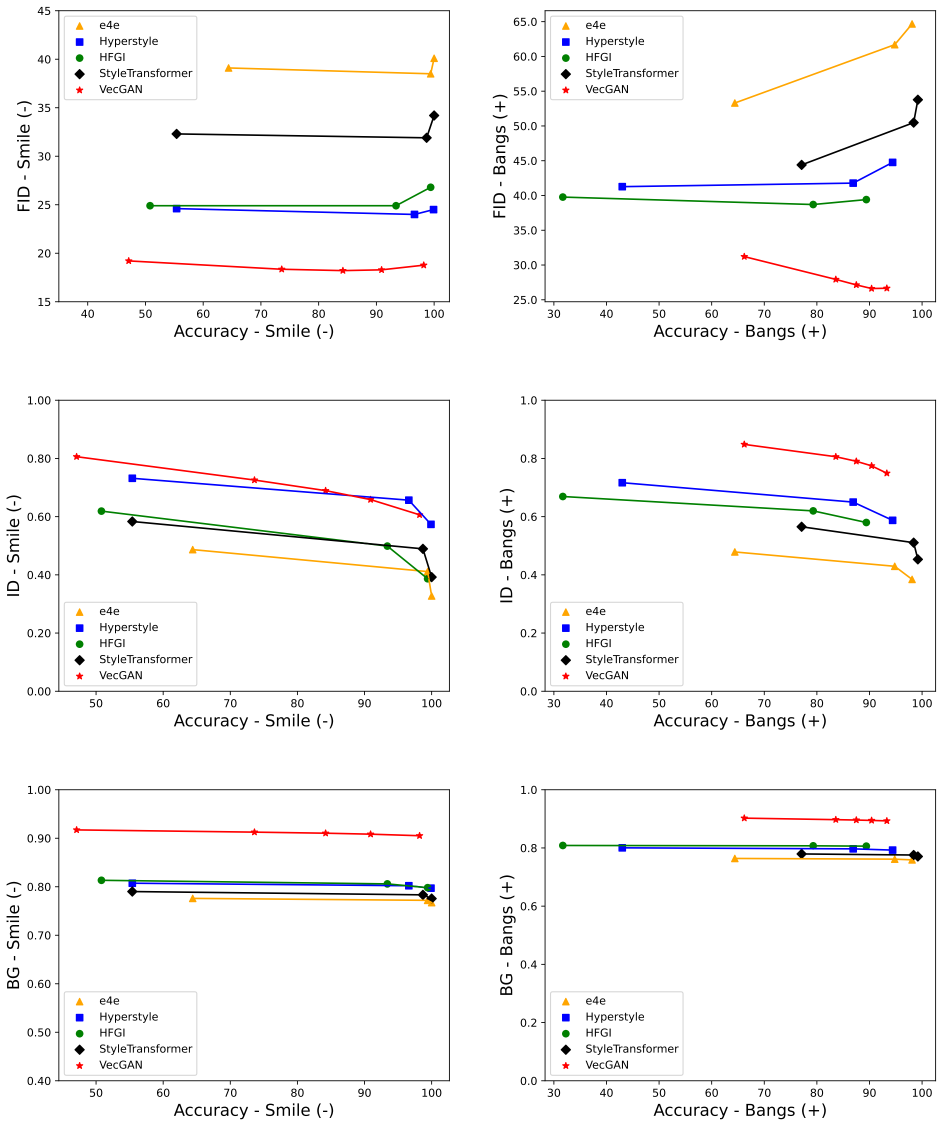


Figure 4.10: Plots of FID, ID, and BG metrics as we change the intensity of the attributes. For each intensity, we measure the attribute accuracy in the x-axis. The first column plots present results for smile removal (global attribute), and the second column presents them for bangs addition (local attribute).

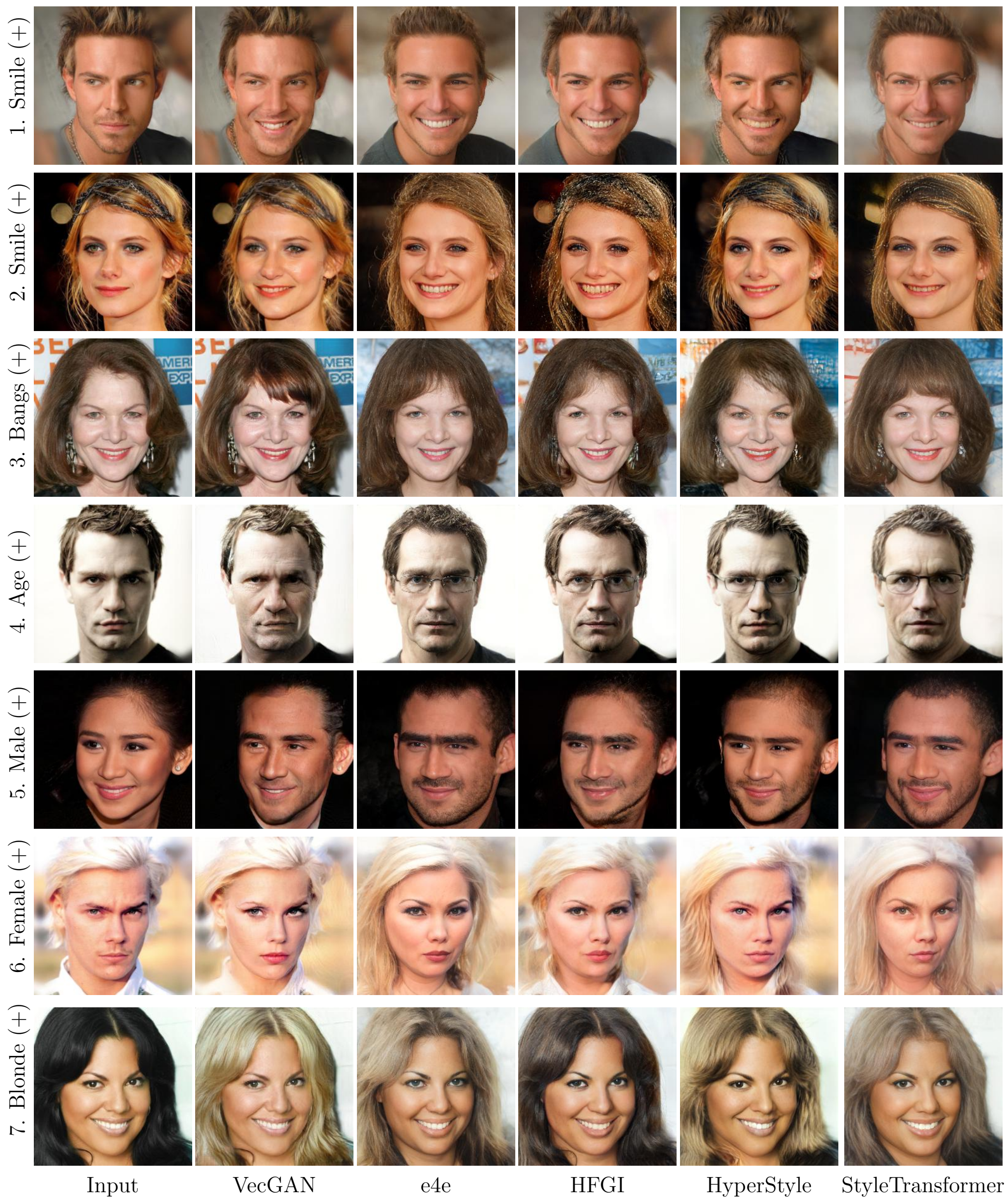


Figure 4.11: Qualitative results of ours and competing methods. StyleGAN inversion-based methods do not faithfully reconstruct input images. VecGAN achieves high fidelity to the originals with only targeted attributes manipulated naturally and realistically.

Chapter 5

Analysis and Discussions

5.1 Comparing End-to-end Image Translation Networks versus StyleGAN Inversion-based Methods

We propose an end-to-end trained image translation network in this thesis and extensively compare our method with StyleGAN inversion-based methods. We note the different advantages and disadvantages of both approaches.

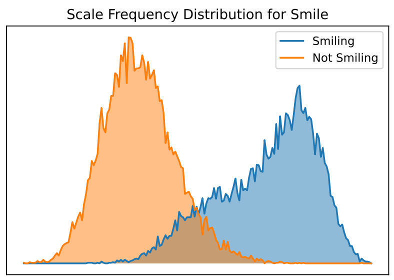
We observe that end-to-end trained image translation networks, especially our proposed framework, do not suffer from the reconstruction and editability trade-off. This trade-off is pointed out for StyleGAN inversion-based methods [18]. That is, when the inversion parameters are optimized to reconstruct the input images faithfully, they do not lie in the natural StyleGAN distribution space, and therefore the edit quality gets poor for those high-fidelity inversions. The advantage of our method is that it is trained end-to-end, and we learn both reconstruction and editing together. This way, our framework learns to reconstruct and edit the images in the distribution space of our generator, and the gap observed in StyleGAN inversion-based methods is not present.

StyleGAN inversion-based methods enjoy many editing capabilities, whereas our framework only achieves pre-defined edits for which it is trained for. Those methods that use pre-trained StyleGANs rely on StyleGAN's semantically rich feature organizations. The editing directions are discovered after StyleGAN is trained. Some methods discover directions in supervised and unsupervised ways. Supervised methods, e.g. InterfaceGAN, require labeled datasets the same as ours. On the other hand, with unsupervised methods and text-based editing methods, directions are explored for those that do not have labeled datasets. For example, with the GANSpace method [26], editing directions are found for different expressions, and with the StyleCLIP method [29], editing directions are found for different hairstyles (Mohawk hairstyle, Bob-cut hairstyle, Afro hairstyle, e.g.). That is an advantage of StyleGAN inversion-based methods.

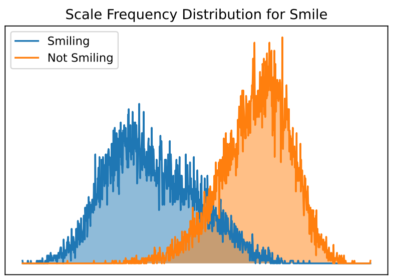
5.2 Analysis of Projected Styles

We explore the behavior of encoded scales from reference images, α_s. These scales are supposed to provide information about the attribute of the image (whether a person smiles or not) and its intensity (how big the smile is). We plot the histograms of α_s values from the dataset images for bangs, eyeglasses, hair color, age, gender, and smile tags and use orange and blue colors (the green color is also used for hair color) depending on their ground truth attributes from the dataset as shown in Fig. 5.2 for VecGAN. Even if we sample our target scales with a uniform distribution $z \in \mathcal{U}[0, 1)$, we observe that the extracted scales are closer to a Gaussian distribution. We explain this behavior with the fact that a majority of events in nature can be approximated with such a distribution.

For the smiling tag, α_s values are mostly disentangled with a small intersection. When we plot these plots for the base version of our framework, VecGAN$_{\text{base}}$, we remove the outliers for visualization purposes. There are some encoded scales far away from the clusters. On the other hand, in the final version of our framework, we do not have such a problem and do not remove any data points. As we enable stability in training for the final version of our framework, it is assumed that the



(a) Histogram α_s values for smile tag of VecGAN$_{\text{base}}$



(b) Histogram α_s values for smile tag of VecGAN$_{\text{final}}$

Figure 5.1: Comparison of α_s distributions learned by VecGAN$_{\text{base}}$ and VecGAN$_{\text{final}}$, for the smiling tag. As both of these plots illustrate, both versions of our framework learn an interpretable distribution for the smiling scale. In addition, VecGAN$_{\text{final}}$ succeeds in learning the smiling scale on the desired scale range, where VecGAN$_{\text{base}}$ contains many examples that are out of the defined range, which changes the number of bins lying in a fixed scale.

framework can learn the desired scale distribution more effectively. We compare the α_s histograms for VecGAN$_{\text{base}}$ and VecGAN$_{\text{final}}$ in Fig. 5.1, for the smiling tag.

Fig. 5.3a shows a visualization of the dataset images plotted based on their α_s values extracted for the smiling tag. We visualize a few samples from each bin from the histogram provided in Fig. 5.1b using the frequency values in the scale histogram. The visualization shows that α_s values encode the intensity of the smile. The rightmost samples have large smiles, and the leftmost samples look almost angry. On the other hand, the images in the middle space are confusing. We also observe many wrong labeling in the CelebA-HQ dataset by going through the middle space.

We repeat the same analysis for the hair color tag as provided in Fig. 5.3b since the hair color tag is a challenging one as it is expected to have a continuous scale with no clear separation between classes. Observing the given distributions for the hair color, we conclude the base version of our framework struggles to cluster the three classes for the hair color tag. The final version improves on it by presenting separate clusters for three classes for the hair color tag (considering the means of the Gaussian-approximated distributions), however, the learned distribution is still imperfect to separate the images. We observe many examples in the overlapping region, especially for the samples having the class label of "black hair". We visualize the dataset images based on their α_s values extracted for the hair color tag in Fig. 5.3b.

The images go from black hair to brown hair to blonde hair. We observe that the shade of hair goes lighter, but we also note that the extracted scales are not perfect, and there is room for improvement.

5.3 Limitations

In this section, we present the failure cases of our algorithm. We find our model to struggle with edits when the face is present with poses that are not very common in the dataset. For example, as shown in Fig. 5.4, in the examples in the first row and the first example from the second row, faces are rotated and tilted, and the model does the edits poorly in these images. We observe in some edits, artifacts may exist, such as in the bangs removal. Sometimes they are not completely erased. Based on our studies and our inspection of other state-of-the-art face editing methods we present in this thesis, we conclude that even though great improvements are accomplished, face editing remains an open problem.

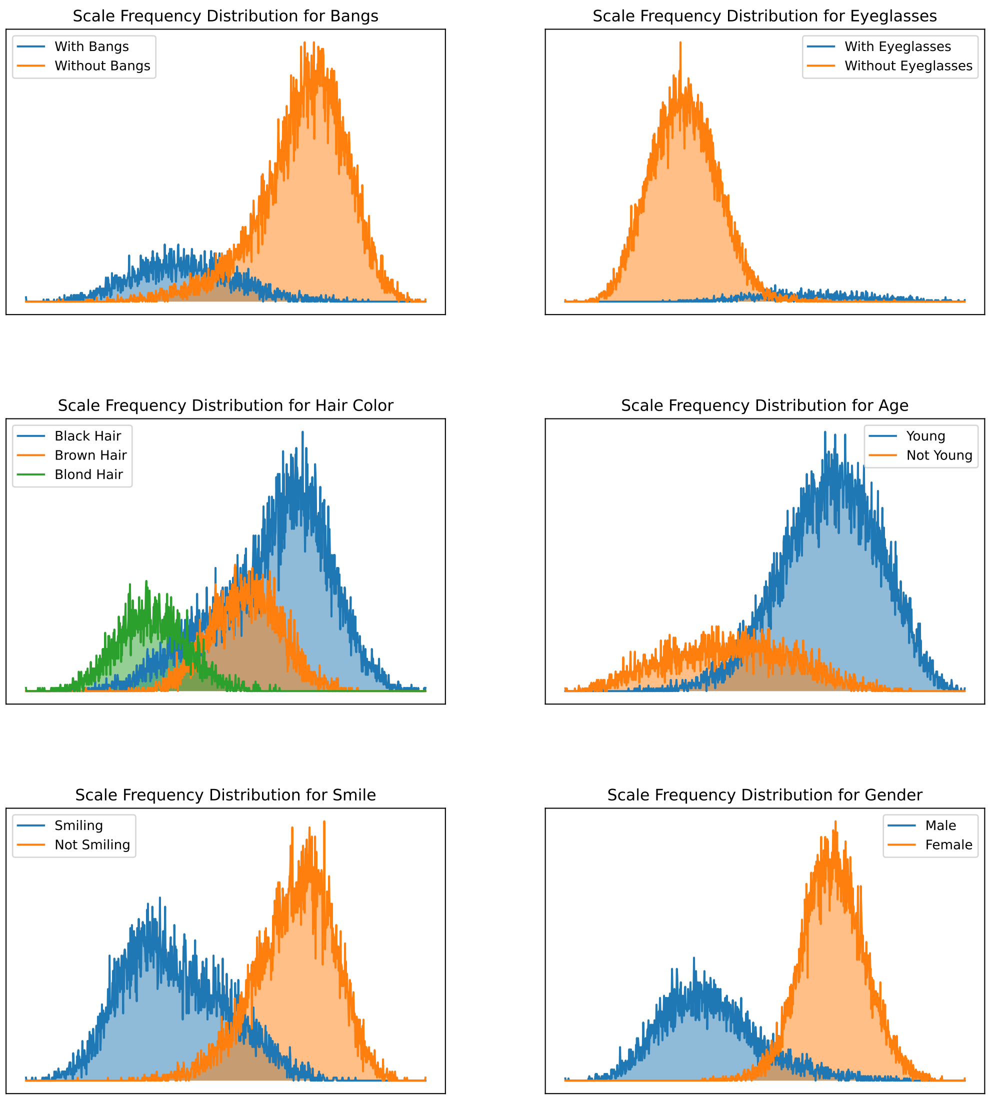
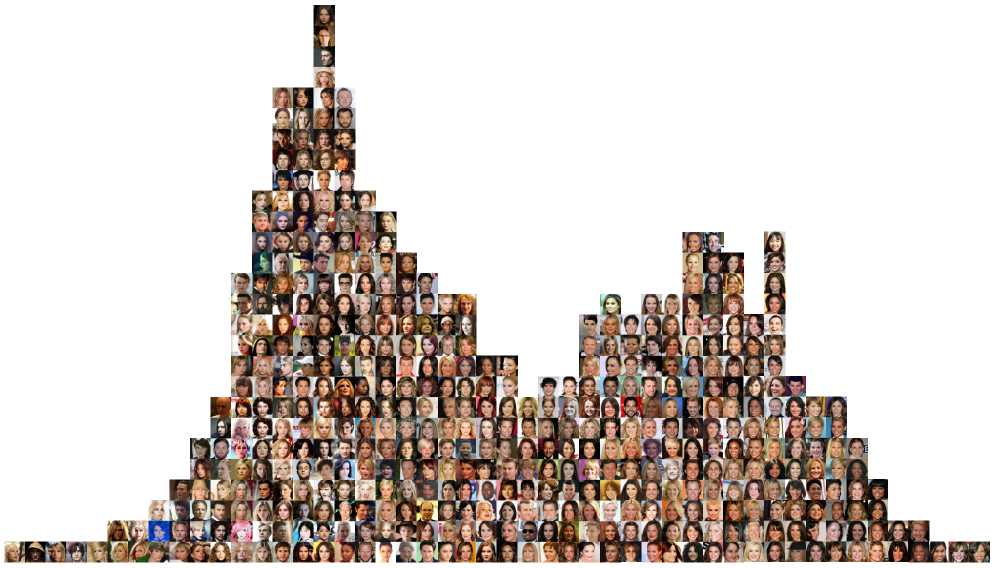
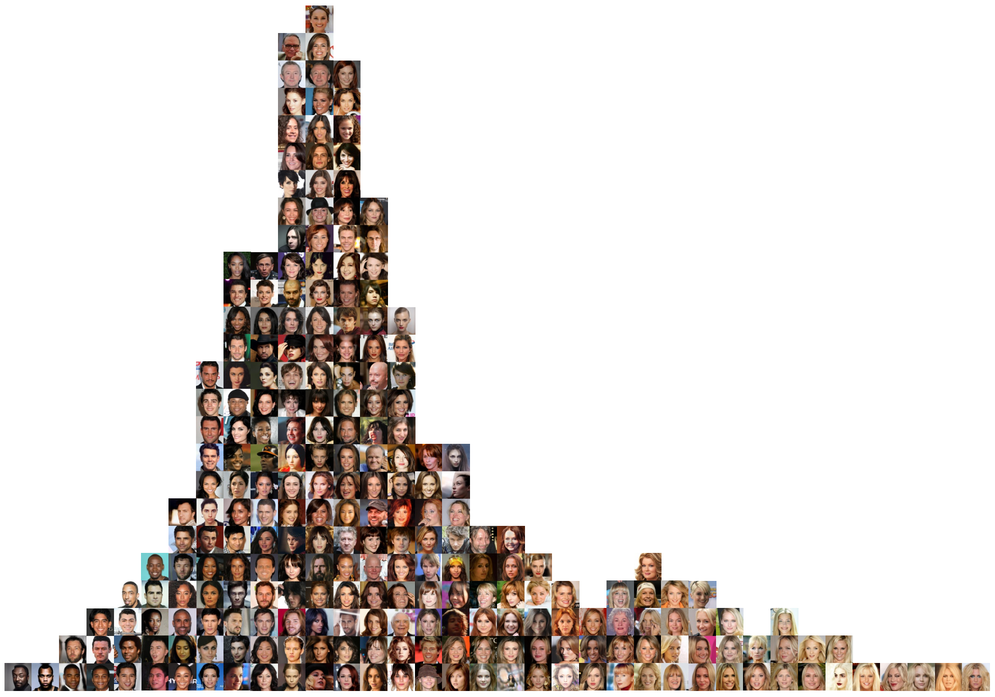


Figure 5.2: Attribute scale plots for the tags translated by our framework. As the scale plots show, our framework successfully clusters images w.r.t. their age and smile. Additionally, we can also cluster the remaining classes even if the learned distribution can be improved further.



(a) Histogram α_s values for smile tag of VecGAN$_{\text{final}}$.



(b) Histogram α_s values for hair color tag of VecGAN$_{\text{final}}$

Figure 5.3: Attribute scale histograms for smile and hair color tags. As the ordering of the images show, the extracted source style scales represent a semantic meaning for both tags.

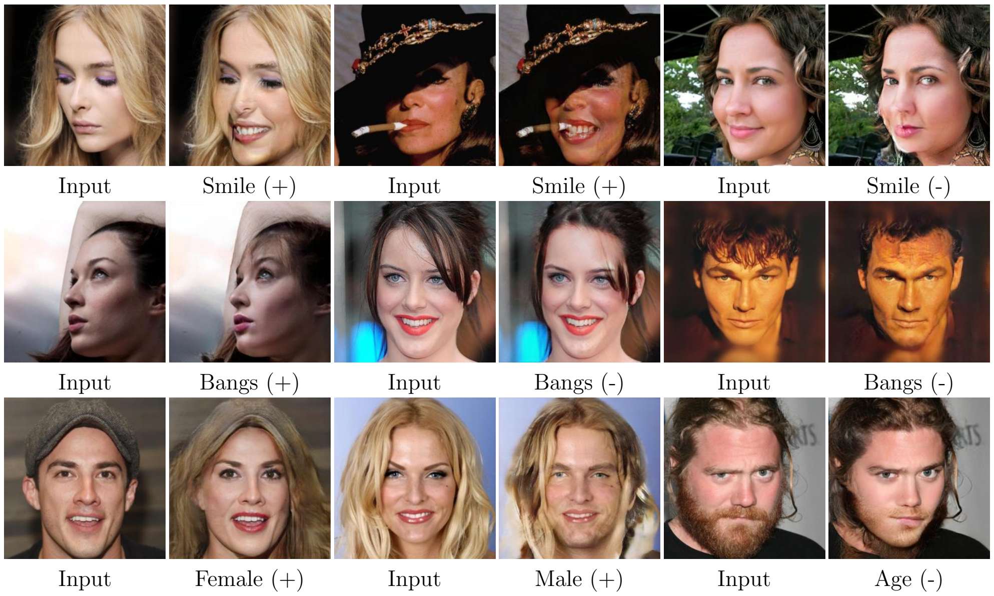


Figure 5.4: Failure cases of VecGAN. In our experiments, we observe that our model struggles in cases where the face image has a pose orientation that is uncommon in the dataset. Additionally, we also observe failure cases on ones where the removal of an attribute changes the face significantly. In these cases, our model tends to generate artifacts on the generated outputs.

Chapter 6

Conclusion

This thesis introduces VecGAN, an image-to-image translation framework with interpretable latent directions. This framework includes a deep encoder and decoder architecture with latent space manipulation in between. Latent space manipulation is designed as vector arithmetic, where for each attribute, a linear direction is learned. This design is encouraged by the finding that well-trained generative models organize their latent space as disentangled representations with meaningful directions in a completely unsupervised way. Therefore, we also extensively compare our method with StyleGAN inversion-based methods and point out their advantages and disadvantages compared to our method. Each change in the architecture and loss function is extensively studied and compared with state-of-the-arts. Experiments show the effectiveness of our framework.

Bibliography

[1] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017.

[2] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2337–2346, 2019.

[3] R. Yi, Y.-J. Liu, Y.-K. Lai, and P. L. Rosin, “Apdrawinggan: Generating artistic portrait drawings from face photos with hierarchical gans,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10743–10752, 2019.

[4] A. Dundar, K. Sapra, G. Liu, A. Tao, and B. Catanzaro, “Panoptic-based image synthesis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8070–8079, 2020.

[5] M.-Y. Liu and O. Tuzel, “Coupled generative adversarial networks,” *Advances in neural information processing systems*, vol. 29, pp. 469–477, 2016.

[6] Z. Yi, H. Zhang, P. Tan, and M. Gong, “Dualgan: Unsupervised dual learning for image-to-image translation,” in *Int. Conf. Comput. Vis.*, 2017.

[7] M.-Y. Liu, T. Breuel, and J. Kautz, “Unsupervised image-to-image translation networks,” in *Adv. Neural Inform. Process. Syst.*, 2017.

[8] M. Mardani, G. Liu, A. Dundar, S. Liu, A. Tao, and B. Catanzaro, "Neural ffts for universal texture image synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14081–14092, 2020.

[9] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "Stargan v2: Diverse image synthesis for multiple domains," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[10] W. Shen and R. Liu, "Learning residual images for face attribute manipulation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4030–4038, 2017.

[11] T. Xiao, J. Hong, and J. Ma, "Elegant: Exchanging latent encodings with gan for transferring multiple face attributes," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 168–184, 2018.

[12] G. Zhang, M. Kan, S. Shan, and X. Chen, "Generative adversarial network with spatial attention for face attribute editing," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 417–432, 2018.

[13] W. Chu, Y. Tai, C. Wang, J. Li, F. Huang, and R. Ji, "Sscgan: Facial attribute editing via style skip connections," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pp. 414–429, Springer, 2020.

[14] Y. Dalva, S. F. Altindis, and A. Dundar, "Vecgan: Image-to-image translation with interpretable latent directions," *Proceedings of the European conference on computer vision (ECCV)*, 2022.

[15] X. Li, S. Zhang, J. Hu, L. Cao, X. Hong, X. Mao, F. Huang, Y. Wu, and R. Ji, "Image-to-image translation via hierarchical style disentanglement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8639–8648, 2021.

[16] Y. Wang, A. Gonzalez-Garcia, J. van de Weijer, and L. Herranz, "Sdit: Scalable and diverse cross-domain image translation," in *Proceedings of the 27th ACM International Conference on Multimedia*, pp. 1267–1276, 2019.

[17] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8110–8119, 2020.

[18] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for stylegan image manipulation," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–14, 2021.

[19] Y. Alaluf, O. Tov, R. Mokady, R. Gal, and A. Bermano, "Hyperstyle: Stylegan inversion with hypernetworks for real image editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18511–18521, 2022.

[20] T. Wang, Y. Zhang, Y. Fan, J. Wang, and Q. Chen, "High-fidelity gan inversion for image attribute editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11379–11388, 2022.

[21] X. Hu, Q. Huang, Z. Shi, S. Li, C. Gao, L. Sun, and Q. Li, "Style transformer for image inversion and editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11337–11346, June 2022.

[22] J. Xie, H. Ouyang, J. Piao, C. Lei, and Q. Chen, "High-fidelity 3d gan inversion by pseudo-multi-view optimization," *arXiv preprint arXiv:2211.15662*, 2022.

[23] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan: How to embed images into the stylegan latent space?," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4432–4441, 2019.

[24] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9243–9252, 2020.

[25] A. Voynov and A. Babenko, "Unsupervised discovery of interpretable directions in the gan latent space," in *International Conference on Machine Learning*, pp. 9786–9796, PMLR, 2020.

[26] E. Härkönen, A. Hertzmann, J. Lehtinen, and S. Paris, "Ganspace: Discovering interpretable gan controls," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[27] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1532–1540, 2021.

[28] Z. Wu, D. Lischinski, and E. Shechtman, "Stylespace analysis: Disentangled controls for stylegan image generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12863–12872, 2021.

[29] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, "Styleclip: Text-driven manipulation of stylegan imagery," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2085–2094, 2021.

[30] G. Liu, A. Dundar, K. J. Shih, T.-C. Wang, F. A. Reda, K. Sapra, Z. Yu, X. Yang, A. Tao, and B. Catanzaro, "Partial convolution for padding, inpainting, and image synthesis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[31] R. Yi, Y.-J. Liu, Y.-K. Lai, and P. L. Rosin, "Unpaired portrait drawing generation via asymmetric cycle mapping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8217–8225, 2020.

[32] P.-W. Wu, Y.-J. Lin, C.-H. Chang, E. Y. Chang, and S.-W. Liao, "Relgan: Multi-domain image-to-image translation via relative attributes," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5914–5922, 2019.

[33] Y. Gao, F. Wei, J. Bao, S. Gu, D. Chen, F. Wen, and Z. Lian, "High-fidelity and arbitrary face editing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16115–16124, 2021.

[34] X. Hou, X. Zhang, H. Liang, L. Shen, Z. Lai, and J. Wan, "Guidedstyle: Attribute knowledge guided style manipulation for semantic face editing," *Neural Networks*, vol. 145, pp. 209–220, 2022.

[35] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[36] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," *Eur. Conf. Comput. Vis.*, 2018.

[37] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Multimodal image-to-image translation by enforcing bi-cycle consistency," in *Advances in neural information processing systems*, pp. 465–476, 2017.

[38] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, "Sean: Image synthesis with semantic region-adaptive normalization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5104–5113, 2020.

[39] G. Yang, N. Fei, M. Ding, G. Liu, Z. Lu, and T. Xiang, "L2m-gan: Learning to manipulate latent space semantics for facial attribute editing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2951–2960, 2021.

[40] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4401–4410, 2019.

[41] N. Yu, G. Liu, A. Dundar, A. Tao, B. Catanzaro, L. S. Davis, and M. Fritz, "Dual contrastive loss and attention for gans," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6731–6742, 2021.

[42] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and*

Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18, pp. 234–241, Springer, 2015.

[43] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 35–51, 2018.

[44] X. Li, J. Hu, S. Zhang, X. Hong, Q. Ye, C. Wu, and R. Ji, "Attribute guided unpaired image-to-image translation with semi-supervised learning," *arXiv preprint arXiv:1904.12428*, 2019.

[45] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[46] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[47] Y. Huang, Y. Wang, Y. Tai, X. Liu, P. Shen, S. Li, and F. H. Jilin Li, "Curricularface: Adaptive curriculum learning loss for deep face recognition," in *CVPR*, pp. 1–8, 2020.

[48] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[49] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

Appendix A

Additional Results

In this section we present additional results on the edits learned by our framework. Namely, we present results from bangs, hair color, age, gender and smile edits.

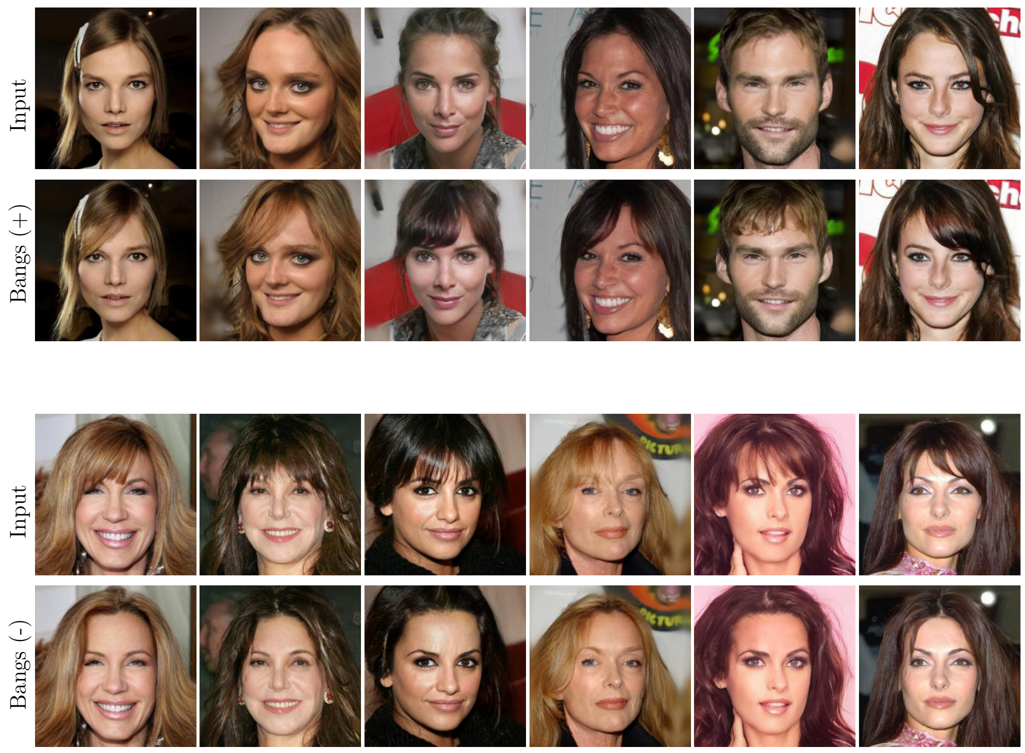


Figure A.1: Supplementary Results for bangs edits.

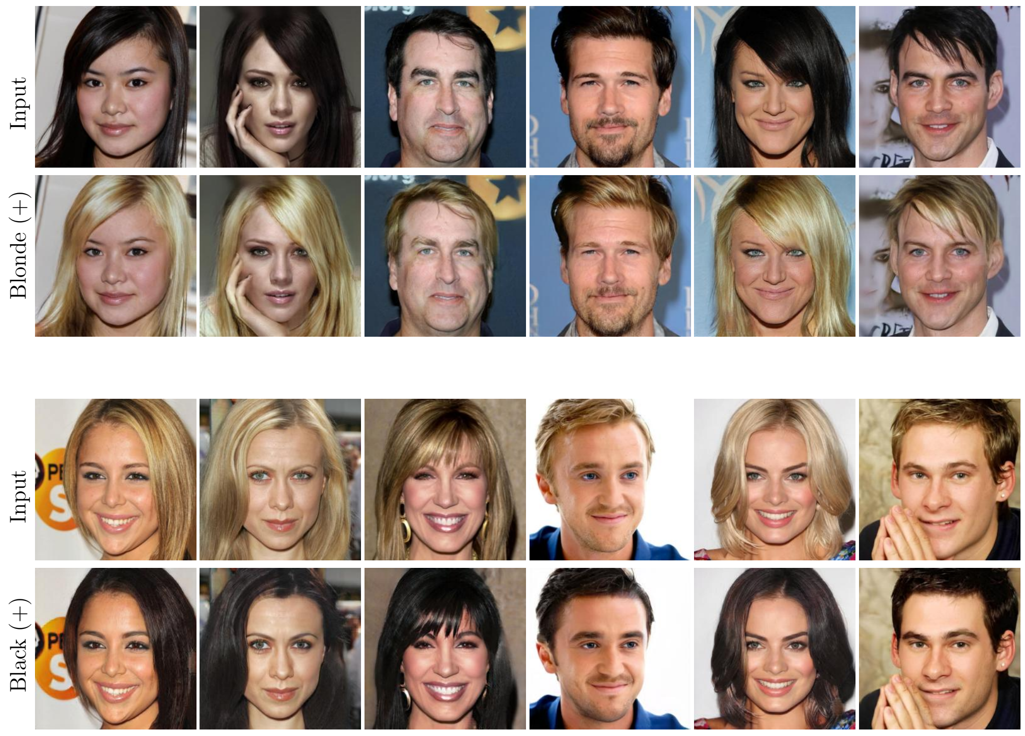


Figure A.2: Supplementary Results for hair color edits.

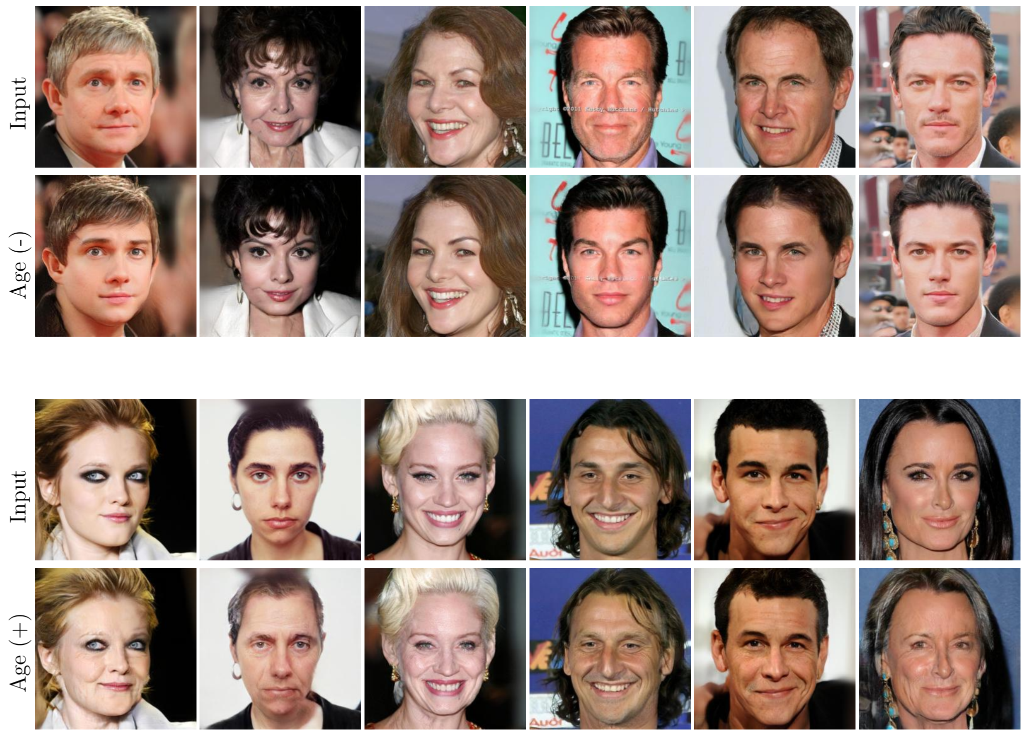


Figure A.3: Supplementary Results for age edits.

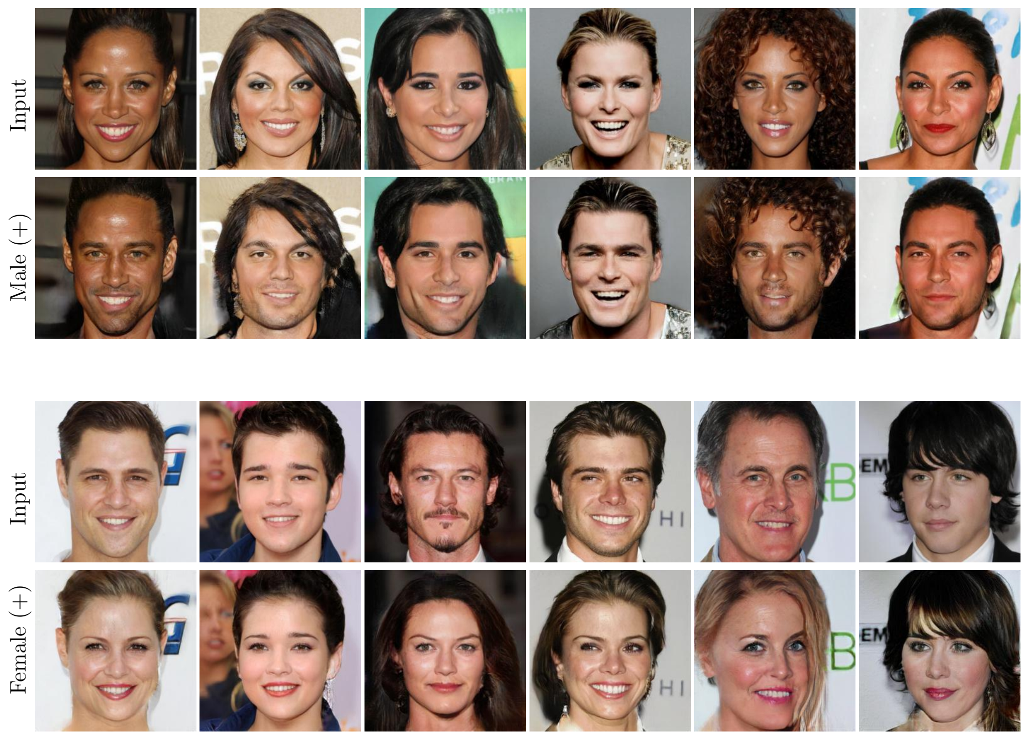


Figure A.4: Supplementary Results for gender edits.

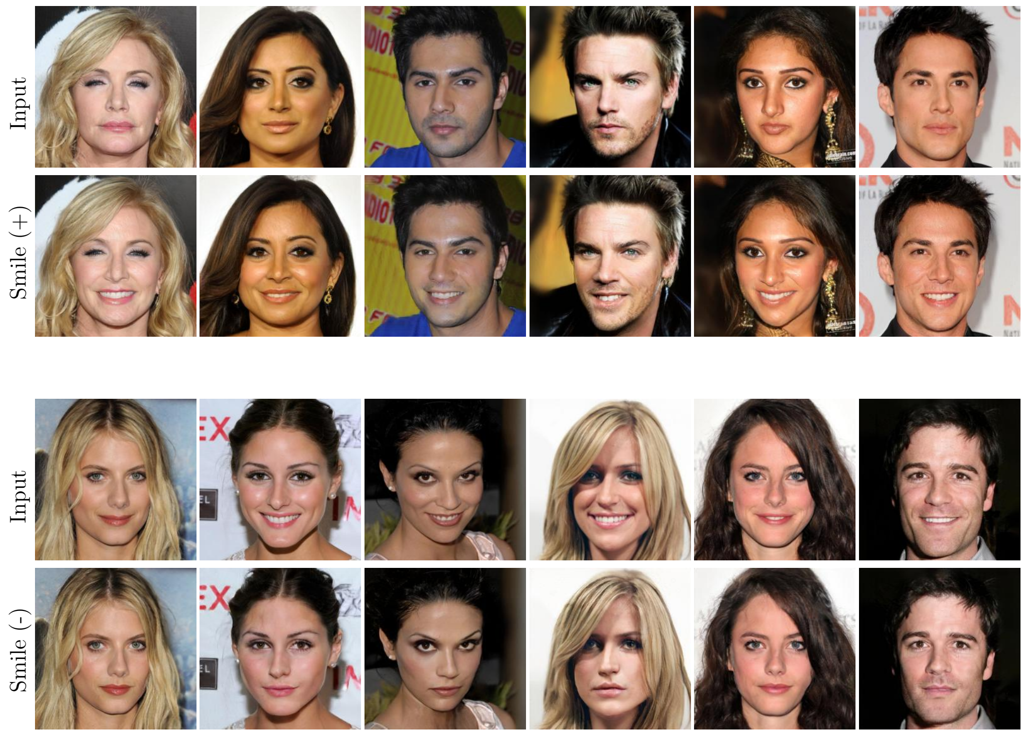
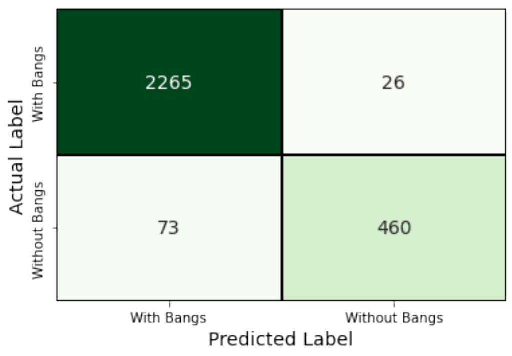


Figure A.5: Supplementary Results for smile edits.

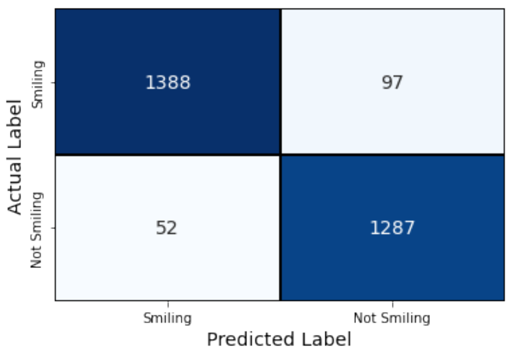
Appendix B

Classifier Details

To provide more details about the classifiers we use for evaluating our framework, we provide the details of the classifiers in this section. We use two separate classifiers for separating the outputs of bangs and smile edits, where the results are presented in Fig. 4.10. For both classifiers, we use a ResNet-50 backbone with a classification head using cross-entropy loss. We provide the confusion matrices of the bangs and smile classifiers in Fig. B.1. These classifiers achieve 96% and 95% accuracy, respectively.



(a) Confusion matrix for bangs tag



(b) Confusion matrix for smile tag

Figure B.1: Confusion matrices for the classifiers used in evaluation Setting II.

Appendix C

Additional Architecture Details

In addition to the generator architecture presented in Sec. 3.1, we provide additional details about our framework in this section. We define our resampling (upsampling/downsampling) blocks in C.1 and the discriminator used in our framework in C.2.

C.1 Resampling Block Architecture

We build our upsampling and downsampling blocks over a shared structure shown in Fig. C.1. This structure uses different resampling layers to downsample or upsample the input features. For the case of upsampling, we use bilinear interpolation and max pooling (with a stride of 2) for downsampling. The blocks are named as UpBlock and DownBlock, respectively. We build our blocks with residuals using 3×3 convolutional filters followed by resampling layers. These blocks include LeakyReLU as activation functions with a negative slope value of 0.2. These blocks also have an instance normalization option, which is enabled for the generator and disabled for the discriminator.

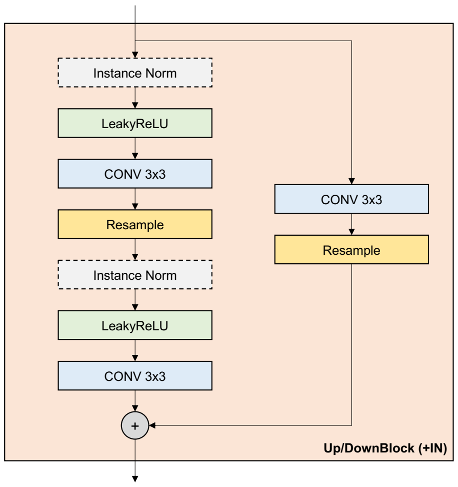


Figure C.1: Architecture of up/downsampling blocks in VecGAN. We use bilinear interpolation for upsampling and max pooling for downsampling blocks. Optionally, the residual blocks include instance normalization layers. In case they are enabled, we apply them before the LeakyReLU activations.

C.2 Discriminator Architecture

Like our generator architecture, the discriminator also uses an architecture with increasing channel size and decreasing resolution. We build our discriminator with the channel sizes used in the generator, which downsamples our 256×256 input to 1×1 features. Unlike the generator, our discriminator does not use any instance normalization layers in the DownBlock structures it contains.

Following the convolutional counterpart, we implement an attribute-specific fully connected layer to retrieve the final output from the discriminator network. We implement this with 1×1 convolutional layers, with separate layers for each tag-attribute pair. Our fully connected layers take the concatenation of the downsampled features and the target style scale as input, achieving adversarial learning sensitive to the target style scale.

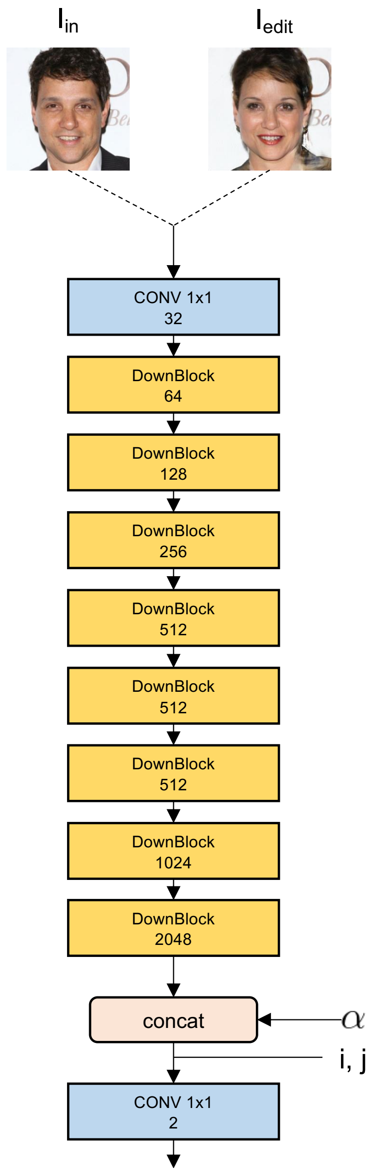


Figure C.2: Discriminator architecture of VecGAN. Like our generator, we build our discriminator with residual downsampling blocks shown in Fig. C.1 where instance normalization is disabled. We output two different values from our discriminator, which are used to learn latent and reference-guided synthesis. The outputs are separated as we wanted to learn these two translations independently. In our experiments, we experienced that this strategy results in better convergence.