REAL IMAGE EDITING WITH STYLEGAN

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
Hamza Pehlivan
September 2023

Real Image Editing with StyleGAN
By Hamza Pehlivan
September 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

REAL IMAGE EDITING WITH STYLEGAN

Hamza Pehlivan
M.S. in Computer Engineering
Advisor: Ayşegül Dündar Boral
September 2023

We present a novel image inversion framework and a training pipeline to achieve high-fidelity image inversion with high-quality attribute editing. Inverting real images into StyleGAN's latent space is an extensively studied problem, yet the trade-off between image reconstruction fidelity and image editing quality remains an open challenge. The low-rate latent spaces are limited in their expressiveness power for high-fidelity reconstruction. On the other hand, high-rate latent spaces result in degradation in editing quality. In this work, to achieve high-fidelity inversion, we learn residual features in higher latent codes that lower latent codes were not able to encode. This enables preserving image details in reconstruction. To achieve high-quality editing, we learn how to transform the residual features for adapting to manipulations in latent codes. We train the framework to extract residual features and transform them via a novel architecture pipeline and cycle consistency losses. We run extensive experiments and compare our method with state-of-the-art inversion methods. Qualitative metrics and visual comparisons show significant improvements.

Keywords: Generative Adversarial Networks (GANs), GAN Inversion, Image Editing with GANs.

ÖZET

STYLEGAN ILE GERÇEK RESIM DÜZENLEME

Hamza Pehlivan
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Danışmanı: Ayşegül Dündar Boral
Eylül 2023

Bu tezde yüksek kaliteli resim düzenleme elde etmek için yeni bir görüntü yeniden yapılandırma metodu ve yeni bir yapay sinir eğitim hattı sunuyoruz. Gerçek görüntüleri StyleGAN'ın gizli alanına cevirmek kapsamlı bir şekilde incelenen bir sorundur. Ancak görüntü yeniden yapılandırma ve resim düzenlenebilirliği arasında bir takas vardır. Düşük oranlı gizli alanların, yüksek doğrulukta yeniden yapılanma için ifade gücü sınırlıdır. Öte yandan, yüksek oranlı gizli alanlar düzenleme kalitesinin bozulmasına neden olur. Bu çalışmada, yüksek doğruluklu ters çevirme elde etmek için, daha düşük gizli kodların kodlayamadığı, yüksek gizli kodlardaki artık özellikleri öğreniyoruz. Bu, yeniden yapılandırma sırasında görüntü ayrıntılarının korunmasını sağlar. Yüksek kaliteli düzenleme elde etmek için, gizli kodlardaki manipülasyonlara uyum sağlamak üzere artık özelliklerin nasıl dönüştürüleceğini öğreniyoruz. Artık özellikleri çıkarmak için ağımızı eğitiyor ve bunları yeni bir eğitim hattı ve döngü tutarlılığı kullarak karşımıza çıkan düzenlemelere göre değiştiriyoruz. Yöntemimizin kapasitesini göstermek adına kapsamlı deneyler yapıp onu en son teknolojiye sahip ters çevirme yöntemleriyle karşılaştırdık. Kalitatif ölçümler ve görsel karşılaştırmalar önemli gelişmeler göstermektedir.

Anahtar sözcükler: Çekişmeli Üretken Ağlar (GAN'lar), GAN ile Görüntü Yeniden Yapılandırma, GAN ile Resim Düzenleme.

Acknowledgement

I appreciate the financial support provided by The Scientific and Technological Research Council of Turkey (TUBITAK). This project is funded by TUBITAK, 3501 Research Project under Grant No 121E097.

I am deeply grateful to my advisor Asst. Prof. Ayşegül Dündar Boral, whose guidance and support made this project possible. Her continuous encouragement and dedication to pushing boundaries have empowered me to excel beyond my own expectations.

I extend my appreciation to Prof. Dr. Selim Aksoy and Assoc. Prof. Ramazan Gökberk Cinbiş for their willingness to join as members of my thesis jury.

I am profoundly thankful to my parents, İbrahim and Meryem Pehlivan, whose support has been the key of my success since childhood. Their enduring love and guidance have shaped not only this endeavor but also the very essence of who I am today.

My heartfelt thanks extend to my colleagues at the Bilkent Image Generation Lab, especially Ahmet Burak Yıldırım and Yusuf Dalva. Their collaboration have enriched this journey.

I am also grateful to my dearest friend, Meryem Efe, whose belief in me has been a constant source of motivation, even during the toughest times.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Generative Adversarial Networks (GANs) achieve high quality synthesis of various objects that are hard to distinguish from real images [1, 2, 3, 4, 5]. These networks also have an important property that they organize their latent space in a semantically meaningful way; as such, via latent editing, one can manipulate an attribute of a generated image. This property makes GANs a promising technology for image attribute editing and not only for generated images but also for real images. However, for real images, one also needs to find the corresponding latent code that will generate the particular real image. For this purpose, different GAN inversion methods are proposed, aiming to project real images to pretrained GAN latent space [6, 7, 8, 9, 10]. Even though this is an extensively studied problem with significant progress, the trade-off between image reconstruction fidelity and image editing quality remains an open challenge.

The trade-off between image reconstruction fidelity and image editing quality is referred to as the *distortion-editability* trade-off [11]. Both are essential for real image editing. However, it is shown that the low-rate latent spaces are limited in their expressiveness power, and not every image can be inverted with high fidelity reconstruction [14, 15, 11, 12]. For that reason, higher bit encodings and more expressive style spaces are explored for image inversion [14, 16]. Although with these techniques, images can be reconstructed with better fidelity, the editing

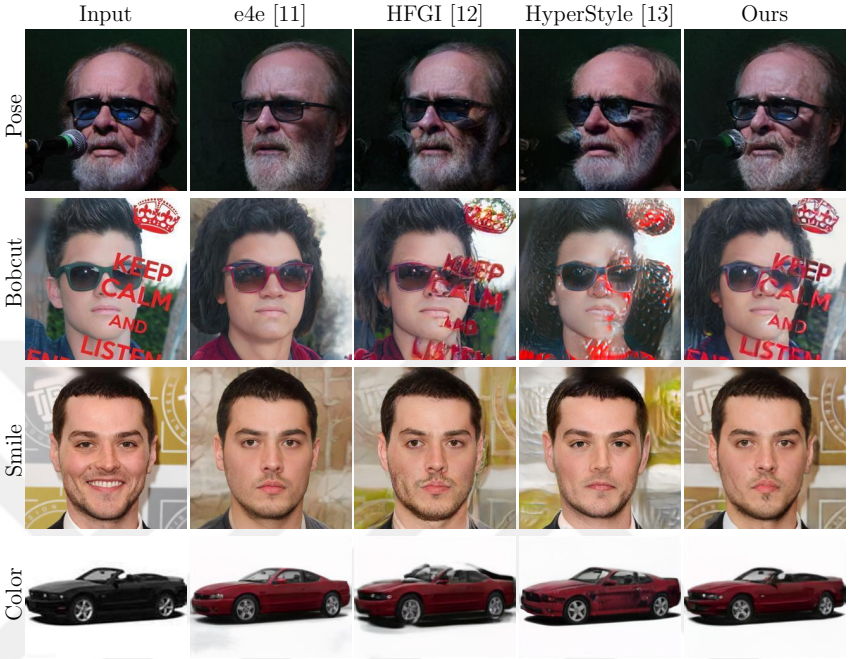


Figure 1.1: Comparison of our method with e4e, HFGI, and HyperStyle for the pose, bobcut hairstyle, smile removal, and color change edits. Our method achieves high fidelity to the input and high quality edits.

quality decreases since there is no guarantee that projected codes will naturally lie in the generator's latent manifold.

In this work, we propose a framework that achieves high fidelity input reconstruction and significantly improved editability compared to the state-of-the-art. This work is presented in IEEE/CVF Conference on Computer Vision and Pattern Recognition with the name "StyleRes: Transforming the Residuals for Real Image Editing with StyleGAN" [17]. We learn residual features in higher-rate latent codes that are missing in the reconstruction of encoded features. This enables us to reconstruct image details and background information which are difficult to reconstruct via low rate latent encodings. Our architecture is single stage and learns the residuals based on the encoded features from the encoder and generated features from the pretrained GANs. We also learn a module to transform the higher-latent codes if needed based on the generated features (e.g. when the low-rate latent codes are manipulated). This way, when low-rate latent codes are edited for attribute manipulation, the decoded features can adapt to

the edits to reconstruct details. While the attributes are not edited, the encoder can be trained with image reconstruction and adversarial losses. On the other hand, when the image is edited, we cannot use image reconstruction loss to regularize the network to preserve the details. To guide the network to learn correct transformations based on the generated features, we train the model with adversarial loss and cycle consistency constraint; that is, after we edit the latent code and generate an image, we reverse the edit and aim at reconstructing the original image. Since we do not want our method to be limited to predefined edits, during training, we simulate edits by randomly interpolating them with sampled latent codes.

The closest to our approach is HFGI [12], which also learns higher-rate encodings. Our framework is different as we learn a single-stage architecture designed to learn features that are missing from low-rate encodings and we learn how to transform them based on edits. As shown in Fig. 1.1, our framework achieves significantly better results than HFGI and other methods in editing quality. In summary, our main contributions are:

- We propose a single-stage framework that achieves high-fidelity input embedding and editing. Our framework achieves that with a novel encoder architecture.
- We propose to guide the image projection with cycle consistency and adversarial losses. We edit encoded images by taking a step toward a randomly sampled latent code. We expect to reconstruct the original image when the edit is reversed. This way, edited images preserve the details of the input image, and edits become high quality.
- We conduct extensive experiments to show the effectiveness of our framework and achieve significant improvements over state-of-the-art for both reconstruction and real image attribute manipulations.

Chapter 2

Related Work

GAN Inversion. GAN inversion methods aim at projecting a real image into GANs embedding so that from the embedding, GAN can generate the given image. Currently, the biggest motivation of the inversion is the ability to edit the image via semantically rich disentangled features of GANs; therefore models aim for high reconstruction and editing quality. Inversion methods can be categorized into two; 1) methods that directly optimize the latent vector to minimize the reconstruction error between the output and target image [18, 14, 16, 4, 19], 2) methods that learn encoders to reconstruct images over training images [20, 11, 12, 21]. Optimization based methods require per-image optimization and iterative passes on GANs, which take several minutes per image. Additionally, overfitting on a single image results in latent codes that do not lie in GAN's natural latent distribution, leading to poor editing quality. Among these methods, PTI [19] takes a different approach, and instead of searching for a latent code that will reconstruct the image most accurately, PTI fine-tunes the generator in order to insert encoded latent code into well-behaved regions of the latent space. It shows better editability; however, the method still suffers from long run-time optimizations and tunings. Encoder based methods leverage the knowledge of training sets while projecting images. They output results with less fidelity to the input, but edits on the projected latents are better quality. They also operate mostly in real-time. They can project the latent code with a single

feed-forward pass [20, 15, 11]. Among them, e4e [11] uses a latent code discriminator to bring the output distribution of the generator's mapping network and encoder's output closer. These methods usually suffer from low-fidelity results since they only predict low-dimensional latent code. The high-rate GAN inversion models can utilize the low-rate models in their pipeline [12, 13], and the low-rate models are usually called base encoders. In this work, we also utilize e4e as our base encoder. High-rate GAN inversion models recover the deficiencies of base encoders with various approaches. HFGI [12] uses two stage feed-forward passes where the first encoder is e4e, and the second encoder learns to reconstruct the missing details of e4e. They first find an error map by subtracting the input image from the inverted image and align this error map according to the incoming edit. They train their aligners using ground truth error maps and their perspective transformation augmented versions. The aligner learns how to undo the perspective transformation with respect to the inverted image. However, this causes a problem when applied to edited images since the edits are more complicated than linear transformations. For instance, if the edit is smile removal, their network can still try to place the teeth in the output since linear transformations do not handle disappearing features. On the other hand, our model will use randomly sampled directions to simulate incoming edits, overcoming the shortcomings of HFGI. Our pipeline is also faster than HFGI despite having more parameters since their method requires three passes on the generator. Another high-rate GAN inversion work, HyperStyle [13], uses a hypernetwork to change the weights of the generator, similar to PTI. The problem with changing the generator is the known edits may not be applicable anymore. One can always extract the editing directions for the new generators; however, this would be very time consuming since we end up with a new generator for each input image. As shown in this paper, methods that change the generator can suffer from unrealistic textures. In this work, we propose a novel encoder architecture that achieves significantly better results than state-of-the-art via a single feed-forward pass on the input image.

Image Translation. There is quite an interest in image translation algorithms that can change an attribute of an image while preserving a given content

[22, 23, 24], especially for face editing [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35]. These algorithms set an encoder-decoder architecture and train the models with reconstruction and GAN losses [36, 25, 37, 29, 38]. Even though they are successful at image manipulation, they are trained for a given task and rely on predefined translation datasets. On the other hand, it is shown that GANs that are trained to synthesize objects in an unconditional way can be used for attribute manipulation [2, 39, 4] via their semantically rich disentangled features. This makes pretrained GANs promising technology for image editing, given that via their ability of image generation, they can achieve many downstream tasks simultaneously. Many methods have been proposed for finding latent directions to edit images [6, 7, 8, 9, 10]. These directions are explored both in supervised [6, 33] and unsupervised ways [7, 8, 9, 10] resulting in exploration of attribute manipulations beyond the predefined attributes of labeled datasets. However, finding these directions is only one part of the problem; to achieve real image editing, there is a need for a successful image inversion method which is the topic of this paper.

Latent Spaces of StyleGAN. GAN inversion models pick one or more latent spaces as their target and invert the images on those spaces. The most popular spaces are W, W+, S+, and F.

StyleGAN's mapping network uses randomly sampled z vectors to create the W space. The initial inversion models use this space to invert images [40]. For 1024x1024 resolution StyleGAN, W space vectors are repeated 18 times and fed to different points in the generator. The repeated vectors form the W+ space. In general, GAN inversion models use this space [11, 15] since it provides a good balance between editing and reconstruction. Once a GAN inversion model inverts images to this latent space, the resulting vectors are different than each other despite the original design. Vectors in S+ space are retrieved when the W+ vectors pass through fully connected layers. Lastly, F space corresponda to high resolution StyleGAN features, and recent GAN inversion models use this space to keep the high-rate information [12], including our model.

The dimensionality of the vectors in these spaces increases as we move from

W to F space. As a result, the reconstruction-editability trade-off occurs, where the W has the most editable but low fidelity vectors. On the other hand, F space provides high-quality reconstruction but low editability. This paper aims to exploit the high-quality reconstructions of F space while adopting it to incoming edits.

Chapter 3

Method

In Section 3.1, we describe the motivation for our approach. The model architecture and the training procedure are presented in Sections 3.2 and 3.3, respectively.

3.1 Motivation

Current inversion methods aim at learning an encoder to project an input image into StyleGAN's natural latent space so that the inverted image is editable. However, it is observed by previous works that when images are encoded into GAN's natural space (low-rate W or W+ space), their reconstructions suffer from low fidelity to input images, as shown in Fig. 3.1(b). On the other hand, if the image is encoded to higher feature maps of pretrained GANs, they will not be editable since they are not inverted to the semantic space of GANs. On the other hand, low-rate inversion is editable, as shown in Fig. 3.1(c).

StyleGAN, during training and inference, relies on direct noise inputs to generate stochastic details. These additional noise inputs provide stochastic image details such as for face images, the exact placement of hairs, stubble, freckles, or skin pores which are not modeled by W+ space. During inversion to W+ space alone, that mechanism is discarded. One can tune the noise maps as well via

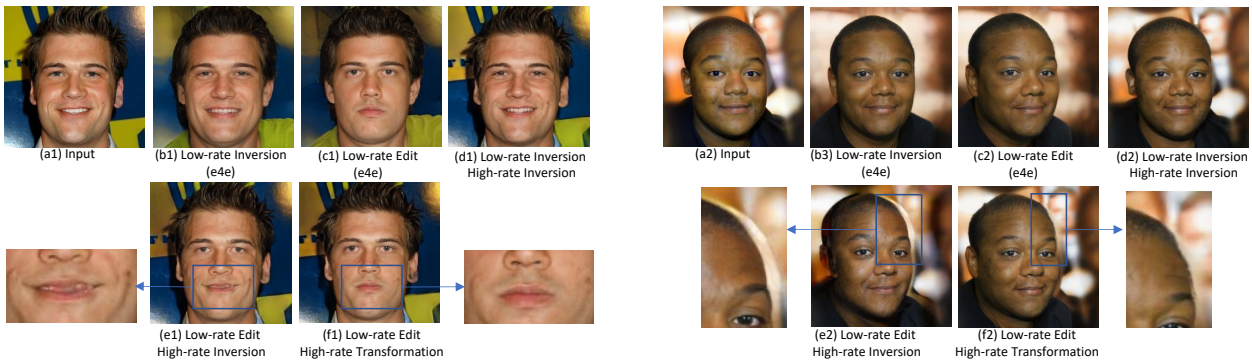


Figure 3.1: When images are encoded to W+ space, as shown in (b), the reconstructions miss many image details. However, the edits are in good quality (c). Additional features can be learned in higher-rate latent codes. For example, skip connections from the encoder to the generator in higher resolution features can enable high-fidelity reconstruction to input images as shown in (d). However, if the high-rate features do not transform with the edits, they result in ghosting effects as shown in (e). In this work, we propose high-rate encoding and transformation for successful inversions and edits (f).

the reconstruction loss; however, then there will be no mechanism to adapt those stochastic details to attribute manipulation. For example, the freckles should move as the pose is manipulated, but with noise optimization alone, it will not be possible.

We propose to embed images to both low-rate and high-rate embeddings. Consistent with the design of StyleGAN, we aim at encoding the overall composition and high-level aspects of the image to W+ space (low-rate encoding). Our goal is to embed the stochastic image details and the diverse background details, which are difficult to reconstruct from W+ space to higher latent codes. However, this setting also requires a mechanism for encoded image details to adopt manipulations in the image. Otherwise, it will cause a ghosting effect, as shown in Fig 3.1(e). For example, in Fig 3.1(e1), the smile is removed by W+ code edit; however, the higher-rate encodings stay the same and cause artifacts around the mouth area. In Fig. 3.1(e2), the pose is edited but higher detail encodings did not move and caused blur.

In this work, we design an encoder architecture and training pipeline to achieve both learning residual features (the ones W+ space could not reconstruct) and

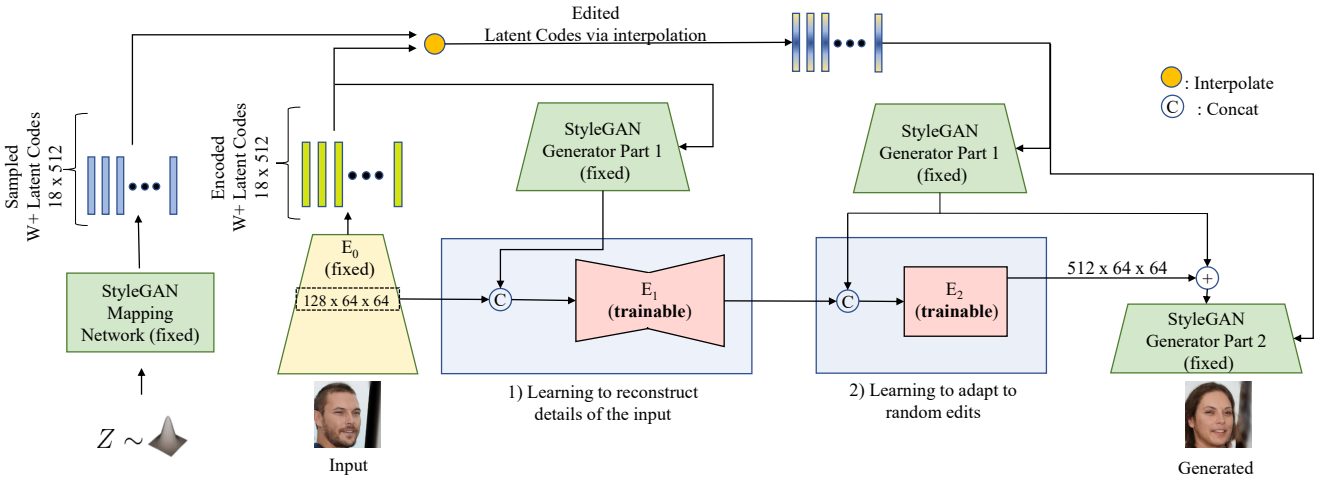


Figure 3.2: StyleRes encodes missing features for high-fidelity reconstruction of given input via the first encoder, E_1. Those encoded features are the ones which could not be encoded to low-rate W+ space via E_0 due to the information bottleneck. Through the second encoder, E_2, StyleRes learns to transform features based on the manipulated features. During training, latent codes are edited by interpolating encoded W+'s with randomly generated ones by StyleGAN's mapping network. During inference, they are edited with semantically meaningful directions discovered by methods such as InterfaceGAN and GANSpace. Note that StyleGAN generator is shown as two parts just for the ease of visualizing the diagram. First part includes the layers that generate features to 64×64 and the second part generates the higher resolution features and final image.

how to transform them to be consistent with the manipulations.

3.2 StyleRes Architecture

Our method utilizes an encoder E_0 that can embed images into $W+$ latent space and a StyleGAN generator G. In our setup, we utilize a pretrained encoder for E_0 [11] and StyleGAN2 generator [4] and fix them in our training. Because it is difficult to preserve image details only from low-rate latent codes, we also extract high level features from the encoder. First, the high and low rate features are extracted as $F_0, W^+ = E_0(x)$ using the encoder E_0 from the input image x as shown in Fig. 3.2. F_0 has the spatial dimension of 64×64 and provides us with more image details. Next, from F_0, our aim is to encode residuals that are missing

from the image reconstruction. For this goal, we set an encoder, E_1, which takes F_0 as input. Since the goal of E_1 is to encode residual features, we also feed the generated features with W^+ from the StyleGAN generator, $G_W = G_{0\rightarrow n}(W)$, where the arrow operator indicates the indices of convolutional layers used from G.

$$F_a = E_1(F_0, G_{W+}) \tag{3.1}$$

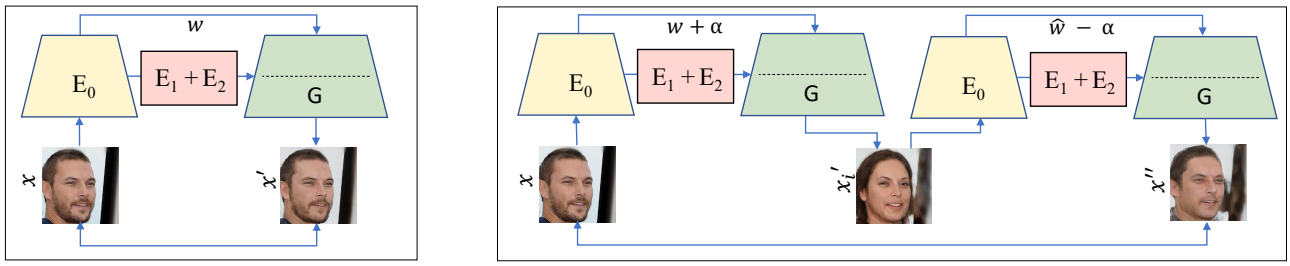
With the inputs of F_0 and G_{W+}, E_1 can learn what the missing features are by comparing the encoded features F_0 and generated ones, G_{W+}, so far. While E_1 can learn the residual features, it is not guided on how to transform them if images are edited. For that purpose, we train E_2, which takes F_a and edited features from the generator. Since we do not target predefined edits (e.g., smile, pose, age), we simulate the edits by taking random directions. Specifically, we sample a z vector from the normal distribution and obtain W_r^+ by StyleGAN's mapping network, M; $W_r^+ = M(z)$. Next, we take a step towards W_r^+ to obtain a mixed style code W_α^+.

$$W_\alpha^+ = W^+ + \alpha \frac{W_r^+ - W^+}{10} \tag{3.2}$$

where α controls the degree of the edit. During the training, we assign α to 0 (no edit) or a value in the range of $(4, 5)$ with 50% chance. We adjust F_a to the altered generator features $G_\alpha = G_{0\rightarrow n}(W_\alpha)$ using a second encoder E_2 and obtain the final feature map F:

$$F = E_2(F_a, G_\alpha) \tag{3.3}$$

Finally, F and G_α are summed, and given as an input to next convolutional layers in the generator. Architectural details of E_1 and E_2 are given in Supplementary.



(a) No Editing: Reconstruction Loss

(b) Cycle-consistency: Reconstruction Loss

Figure 3.3: We train StyleRes with (a) no editing based reconstruction and (b) cycle consistency based reconstruction losses. Many details from the inference pipeline are omitted for brevity. With reconstruction losses, the model learns to preserve the image details. We additionally apply adversarial losses on x'_i; this way, when the image is edited, the transformation network learn to output realistic images. With cycle consistency based reconstruction, the network is regularized to keep the input details also during edits. We use additional losses as explained in Section 3.4.

3.3 Training Phases

To train our model with the capabilities of high-fidelity inversion and high-quality editing, we use two training phases, as shown in Fig. 3.3.

No Editing Path. In this path, we reconstruct images with no editing on encoded W^+. This refers to the case where $\alpha = 0$ in Eq. 3.2. This is also the training path other inversion methods solely rely on. While this path can teach the network to reconstruct images faithfully, it does not include any edits and cannot guide the network to high-quality editability.

Cycle Translation Path. In this path, we edit images by setting α a value in the range of $(4, 5)$, which we found to work best in our ablation studies. Via the edit, the generator outputs image x'_i. Next, we feed this intermediate output image to the encoder and reverse the edit by inverting the addition presented in Eq. 3.2. The generator reconstructs x'', which is supposed to match the input image x. The cycle translation path is important because there is no ground-truth output for the edited image, x'_i. Adversarial loss can guide x'_i to look realistic but will not guide the network to keep the input image details if edited.

3.4 Training Objectives

Reconstruction Losses. For both no editing and cycle consistency path outputs, our goal is to reconstruct the input image. To supervise this behavior, we use L_2 loss, perceptual loss, and identity loss between the input and output images. The first reconstruction loss is as follows, where x is the input image, x' and x'' are output images, as given in Fig. 3.3:

$$\mathcal{L}_{rec-l2} = ||x' - x||_2 + ||x'' - x||_2 \tag{3.4}$$

We use perceptual losses from VGG (Φ) at different feature layers (j) between these images from the loss objective as given in Eq. 3.5.

$$\mathcal{L}_{rec-p} = ||\Phi_j(x') - \Phi_j(x)||_2 + ||\Phi_j(x'') - \Phi_j(x)||_2 \tag{3.5}$$

Identity loss is calculated with a pre-trained network A. A is an ArcFace model [41] when training on the face domain and a domain specific ResNet-50 model [11] for our training on car class.

$$L_{rec-id} = (1 - \langle A(x), A(x')\rangle) + (1 - \langle A(x), A(x'')\rangle) \tag{3.6}$$

Adversarial Losses. We also use adversarial losses to guide the network to output images that match the real image distribution. This objective improves realistic image inversions and edits. We load the pretrained discriminator from StyleGAN training, D, and train the discriminator together with the encoders.

$$\begin{aligned} L_{adv} = 2\log D(x) + \log\left(1 - D(x')\right) \\ + \log\left(1 - D(x_i')\right) \end{aligned} \tag{3.7}$$

Feature Regularizer. To prevent our encoder from deviating much from the original StyleGAN space, we regularize the residual features to be small:

$$L_F = \sum_{F \in \phi} \|F\|_2 \tag{3.8}$$

Full Objective. We use the overall objectives given below. The hyper parameters are provided in Supplementary.

$$\begin{aligned} \min_{E_1,E_2} \max_{D} \lambda_a \mathcal{L}_{adv} + \lambda_{r1} \mathcal{L}_{rec-l2} + \lambda_{r2} \mathcal{L}_{rec-p} \\ + \lambda_{r3} \mathcal{L}_{rec-id} + \lambda_f \mathcal{L}_F \end{aligned} \tag{3.9}$$

3.5 Training Details

We use e4e [11] as the basic encoder and invert StyleGAN2 [4] generator. The high level features F_0 are the input of the first map2style layer used in e4e. F_0 has the spatial dimension of $128 \times 64 \times 64$. The intermediate StyleGAN features are extracted as $G_W = G_{0 \to 8}(W)$, where the arrow operator indicates the indices of convolution layers used. G_W has the spatial dimension of $512 \times 64 \times 64$.

When we choose the *no editing path*, we set $\lambda_{r1} = 1.0$, $\lambda_{r2} = 0.001$, $\lambda_{r3} = 0.1$ for the face and $\lambda_{r3} = 0.5$ for the car dataset. When we choose the *cycle translation path*, we set $\lambda_{r1} = 0.0$, meaning we do not use cycle consistency at the pixel level, $\lambda_{r2} = 0.0001$, $\lambda_{r3} = 0.01$ for the face and $\lambda_{r3} = 0.05$ for the car dataset. At both paths, we set $\lambda_a = 0.1$. The regularizer coefficient is set to $\lambda_f = 5.0$ for the face dataset and $\lambda_f = 3.0$ for the car dataset. The network is trained with Adam optimizer, with a learning rate equal to 0.0001. We halved the learning rate at iterations 5000, 10000 and 15000.

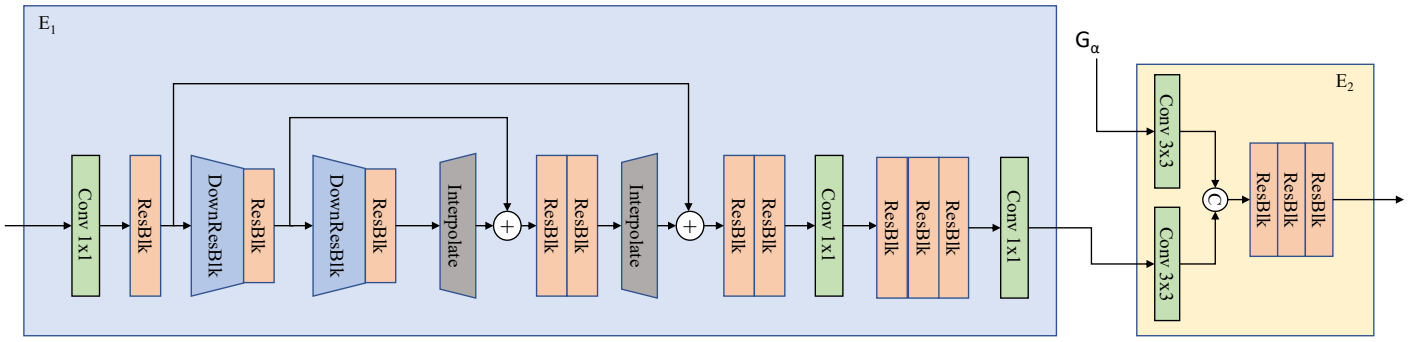


Figure 3.4: Detailed architecture of our encoders. More information regarding dimensions are given in the corresponding text.

3.6 Architecture Details

Our model consists of residual layers, which are visualized in Fig. 3.5. Encoder E_1 takes the concatenated features F_0 and G_w, which has a spatial dimension of 640x64x64. It then forwards the concatenated features to E_2 using an encoder-decoder architecture. The first convolution layer sets the channel size as 512, and it is not changed in the rest of the layers. *DownResBlk* reduces the resolution to half, and *Interpolate* layer doubles the resolution, by using linear interpolation.

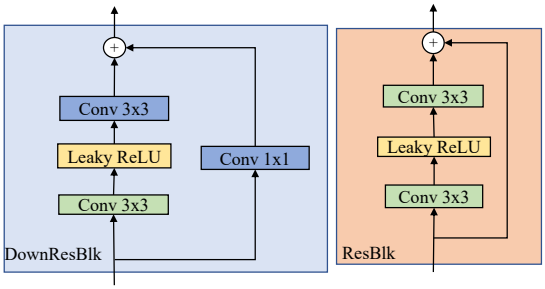


Figure 3.5: Building blocks of our encoders. On the left, we demonstrate the downsampling layers used in E_1. On the right, we show standard residual layers used in both E_1 and E_2.

E_2 takes both F_a and G_α as inputs, and learns how to adapt encoded features to the generator features. Before concatenating the input features, their channel sizes are reduced to 256 using 3x3 convolution layers. After the concatenation, the channel size becomes 512, and it does not change throughout the E_2. The detailed architecture of our encoders is given in Fig. 3.4.

Chapter 4

Experiments

4.1 Set-up

For datasets and attribute editing, we follow the previous work [12]. For the human face domain, we train the model on the FFHQ [2] dataset and evaluate it on the CelebA-HQ [42] dataset. For the car domain, we use Stanford Cars [43] for training and evaluation. We run extensive experiments with directions explored with InterfaceGAN [6], GANSpace [8], StyleClip [44], and GradCtrl [45] methods.

4.2 Evaluation Procedure

We report metrics for reconstruction and editing qualities. For reconstruction, we report Frechet Inception Distance (FID) metric [46], which looks at the realism by comparing the target image distribution and reconstructed images, Learned Perceptual Image Patch Similarity (LPIPS) [47] and Structural Similarity Index Measure (SSIM), which compares the target and output pairs at the feature level of a pretrained deep network and pixel level, respectively. Additionally, FIDs

are calculated for adding and removing the smile attributes on the CelebA-HQ dataset. By using the ground-truth attributes, we add smile to images that do not have smile attribute and calculate FIDs between smile addition edited images and smiling ground-truth images. The same set-up is used for smile removal. On the Stanford cars dataset, we change the grass and color attributes of images, and since there is no ground-truth attribute, we calculate the FIDs between the edited and original images.

For the face images, the reconstruction evaluations are performed on the last 2000 images of the CelebA-HQ dataset, same as the most of the inversion methods. For smile addition and removal, we use corresponding CelebA-HQ images to the CelebA test set since the ground truth labels are given in this dataset. To add or remove a smile, we use the boundary obtained by InterfaceGAN [6], with a factor of 3 and -3 respectively. For the car images, the reconstruction and editing evaluations are performed on the first 2000 images of the Stanford Cars test set. Because we do not have ground-truth labels, we directly calculate FID between the original and edited images. The edited images are obtained using GanSpace [8] directions.

For HyperStyle and Restyle, we use 5 iterative iterations in the reconstruction and editing, which is consistent with their training scheme. For PTI, we deploy locality regularization, as introduced in their work, for both reconstruction and editing.

4.3 Baselines

We compare our method with state-of-the-art image inversion methods pSp [15], e4e [11], ReStyle [48], HyperStyle [13], HFGI [12], StyleTransformer [49], and FeatureStyle [50]. We use the author's released models. Hence, for the Stanford car dataset, some methods are omitted from comparisons if the models are not released. Among those, we only train HFGI for car model with author's released code since we base our main comparisons with it. Additionally, we compare our

method with PTI [19], which is an optimization-based inversion approach.

4.4 Quantitative Results

	Reconstruction			Editing - FIDs	
Method	FID	SSIM	LPIPS	Smile(+)	Smile(-)
PTI [19]	10.64	**0.92**	**0.07**	30.29	30.11
pSp [15]	23.86	0.75	0.17	32.47	34.0
e4e [11]	30.22	0.71	0.21	38.58	39.68
ReStyle [48]	24.82	0.73	0.20	30.35	33.69
HyperStyle [13]	16.08	0.83	0.11	26.43	25.26
HFGI [12]	12.17	0.85	0.13	25.22	27.10
StyleTransformer [49]	21.82	0.75	0.17	34.32	34.61
FeatureStyle [50]	11.33	0.90	0.10	27.20	26.15
StyleRes (Ours)	**7.04**	0.90	0.09	**23.52**	**21.80**

Table 4.1: Quantitative results of reconstruction and editing on CelebA-HQ dataset. For reconstruction, we report FID, SSIM, and LPIPS scores. For editing, we report FID metrics for smile addition (+) and removal (-). PTI is an optimization based method, whereas the others are encoder based.

In Table 4.1, we provide reconstruction and editing scores on the CelebA-HQ dataset. Our method achieves better results than all competing methods on all metrics. Most significantly, we achieve better FID scores on both reconstruction and editing qualities. While FeatureStyle achieves comparable SSIM and LPIPS scores on reconstruction, the editing FIDs are worse than our model. As shown in Table 4.2, we achieve significantly better results than previous methods on the Stanford Car dataset as well.

We also compare the runtime of our method and previous methods. Our method is significantly faster than HyperStyle (0.125 sec vs. 0.439 sec). That is because HyperStyle refines its predicted weight offsets gradually via multiple iterations. Our method is also faster than HFGI (0.125 sec vs. 0.130 sec), in addition to achieving better scores. We run a single stage network inference, whereas HFGI first generates an image via e4e and StyleGAN and provides the

	Reconstruction			Editing - FIDs	
Method	FID	SSIM	LPIPS	Grass	Color
e4e [11]	14.04	0.50	0.32	18.02	29.79
ReStyle [48]	13.38	0.57	0.30	16.01	21.34
HyperStyle [13]	11.64	0.63	0.28	17.13	26.30
HFGI [12]	9.41	0.83	0.16	14.84	26.65
StyleTransformer [49]	14.01	0.57	0.28	19.47	19.94
StyleRes (Ours)	**7.60**	**0.83**	**0.14**	**10.64**	**18.86**

Table 4.2: Quantitative results of reconstruction and editing on the Stanford Cars Dataset. For reconstruction, we report FID, SSIM, and LPIPS scores. For editing, we report FID metrics for grass addition and color change.

error map to a second architecture. We also provide comparisons with PTI. PTI runs optimization for each image and achieves better reconstruction scores in terms of SSIM and LPIPS. However, our method achieves significantly better edit FIDs and reconstruction FIDs. Furthermore, running evaluation with PTI method takes more than 2 days on a single GPU, whereas our method finishes within 5 minutes. The running times are obtained by averaging the reconstruction times of 2000 samples with batch size equal to 1 on a single NVIDIA GeForce GTX 1080 Ti GPU. The table for runtimes is provided in Table 4.3.

Method	Runtime (sec)
PTI [19]	97.94
pSp [15]	0.088
e4e [11]	0.089
ReStyle [48]	0.365
HyperStyle [13]	0.437
HFGI [12]	0.130
StyleTransformer [49]	0.063
FeatureStyle [50]	0.526
StyleRes (Ours)	0.125

Table 4.3: Inference time of the competing methods are given.

We also provide FID scores on age and pose edits among the encoder based GAN inversion methods. For these edits, we obtain FID score between the original and edited images, which is calculated using the entire evaluation dataset. Pose

Method	Pose(+)	Pose(-)	Age(+)	Age(-)
pSp	22.95	22.13	35.89	28.52
e4e	26.92	26.83	42.21	40.36
ReStyle	21.65	22.43	25.82	26.62
HyperStyle	15.59	15.83	16.19	22.81
HFGI	15.37	15.97	19.28	18.19
StyleTransformer	20.26	21.06	42.22	35.21
FeatureStyle	30.00	24.24	14.80	23.59
StyleRes (Ours)	**11.31**	**10.73**	**12.94**	**14.91**

Table 4.4: Age and pose editing FIDs of competing methods are given.

and age edits are obtained by InterfaceGAN, with a factor of 2. The comparisons are given in Table 4.4.

4.5 Qualitative Results

We show visuals of inversion and editing results of our method, e4e, HFGI, and HyperStyle in Fig. 4.1. We provide further comparisons in Appendix with other attribute editings and with other methods. Compared to previous methods, our method achieves significantly better fidelity to the input images and preserves the identity and details when edited. It is the only method in these comparisons that preserves the background, earings (second row), hands (second-fourth rows), and hats (fourth row). Our method also achieves facial detail reconstruction; for example, in the fifth row, the person has a mole at the corner of her mouth. Among the inversions, our method is the only one preserving that. Furthermore, during the pose edit, it is transformed correctly. On the sixth row, our method is the only one that achieves the correct inversion and edit. In car examples, we again achieve high fidelity to the input images both in inversion and editing. e4e and HyperStyle do not reconstruct the image faithfully. On the other hand, HFGI outputs artefacts during edits. We additionally show results with edits explored by StyleClip [44] and GradCtrl [45] methods in Figs. 4.2 and 4.3, respectively. Note that since our network is trained with editing in mind, we can directly apply

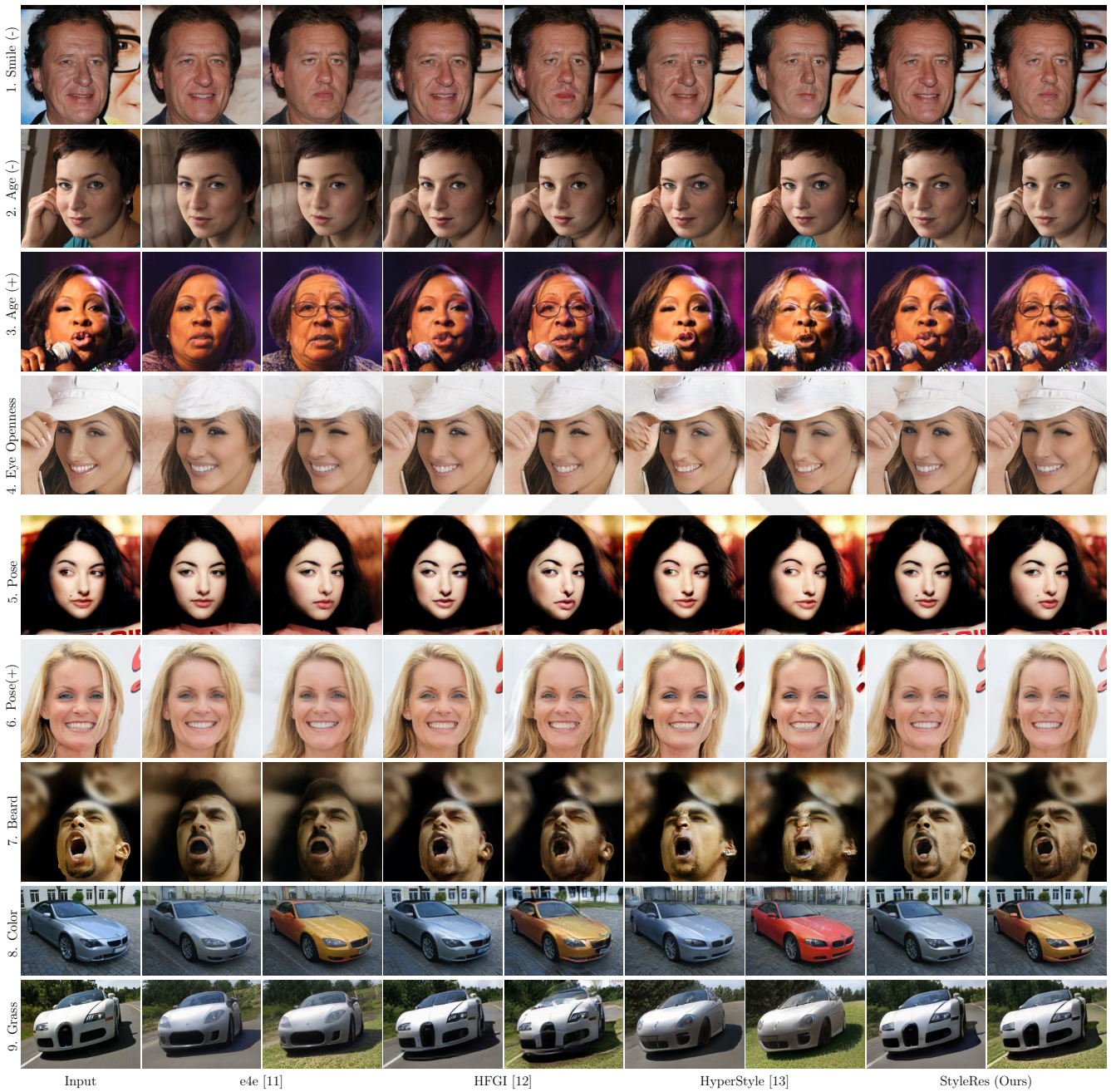


Figure 4.1: Qualitative results of inversion and editing. For each method, the first column shows inversion, and the second shows InterfaceGAN [6] and GANSpace [8] edits.

boundaries found by different editing methods.

Figure 4.2: Comparison of our method with e4e, HFGI, and HyperStyle with StyleClip edits [44]. The first row shows eyeglasses addition, the second and third rows show bangs addition, and the last row shows bobcut hairstyle results.

4.6 More Qualitative Results

We provide visual comparisons on:

1. smile editing with StyleTransformer, PTI, and FeatureStyle in Figs. 4.4 and 4.5, by using InterfaceGAN.

2. smile editing with e4e, HFGI and HyperStyle in Figs. 4.6 and 4.7, by using InterfaceGAN.

3. age editing with e4e, HFGI and Hyperstyle in Figs. 4.8 and 4.9, by using InterfaceGAN.

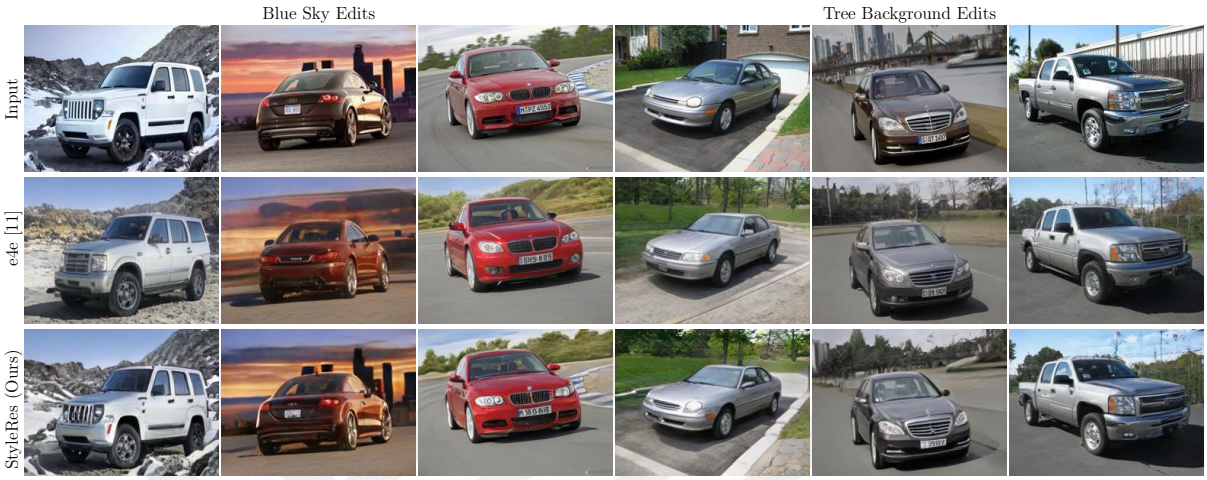


Figure 4.3: Additional results of our method and e4e with GradCtrl edits [45]. The first three columns show blue sky edit, and the others show tree background edits.

4. pose change with e4e, HFGI and HyperStyle in Figs. 4.10 and 4.11, by using InterfaceGAN.

5. beard addition with e4e, HFGI and HyperStyle in Fig. 4.12, by using GanSpace.

6. eye openness with e4e, HFGI and HyperStyle in Fig. 4.13, by using GanSpace.

7. lipstick addition with e4e, HFGI and HyperStyle in Fig. 4.14, by using GanSpace.

8. eyeglasses addition with e4e, HFGI and HyperStyle in Fig. 4.15, by using StyleClip.

9. bangs addition with e4e, HFGI and HyperStyle in Fig. 4.16, by using StyleClip.

10. bobcut hairstyle with e4e, HFGI and HyperStyle in Fig. 4.17, by using StyleClip.

11. car color change with e4e, HFGI and HyperStyle in Fig. 4.18, by using GanSpace.

12. grass addition with e4e, HFGI and HyperStyle in Fig. 4.19, by using GanSpace.

Figure 4.4: Comparisons with StyleTransformer, PTI and FeatureStyle on smile addition. For each method, first column shows inversion, and second shows editing.

Figure 4.5: Comparisons with StyleTransformer, PTI and FeatureStyle on smile removal. For each method, first column shows inversion, and second shows editing.

Figure 4.6: Comparisons with e4e, HFGI and HyperStyle on smile addition. For each method, first column shows inversion, and second shows editing.

Figure 4.7: Comparisons with e4e, HFGI and HyperStyle on smile removal. For each method, first column shows inversion, and second shows editing.

Figure 4.8: Comparisons with e4e, HFGI and HyperStyle on age reduction. For each method, first column shows inversion, and second shows editing.

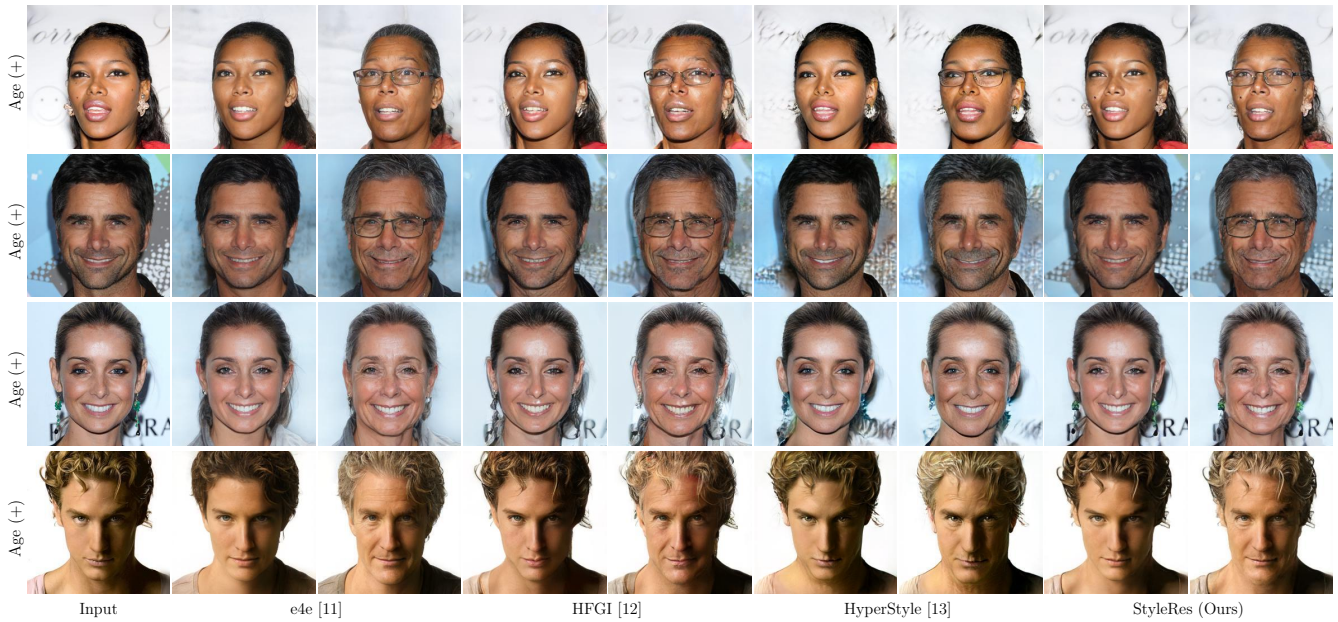


Figure 4.9: Comparisons with e4e, HFGI and HyperStyle on age increase. For each method, first column shows inversion, and second shows editing.

Figure 4.10: Comparisons with e4e, HFGI and HyperStyle on pose editing. For each method, first column shows inversion, and second shows editing.

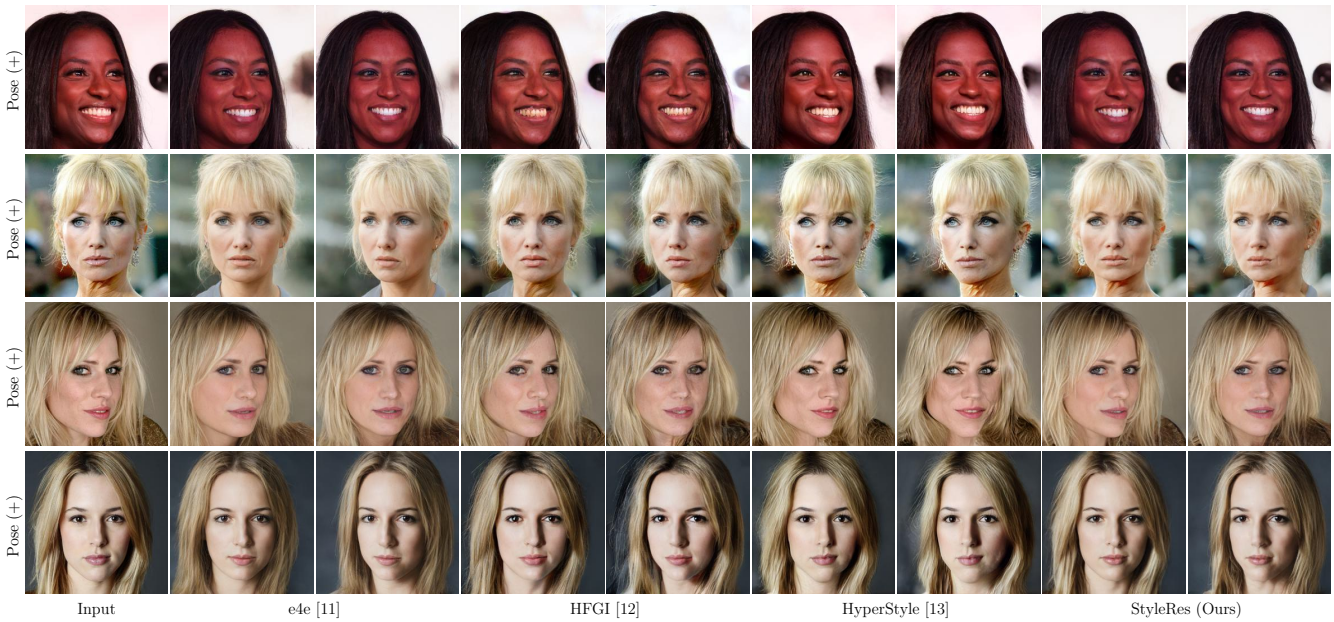


Figure 4.11: Comparisons with e4e, HFGI and HyperStyle on pose editing. For each method, first column shows inversion, and second shows editing.

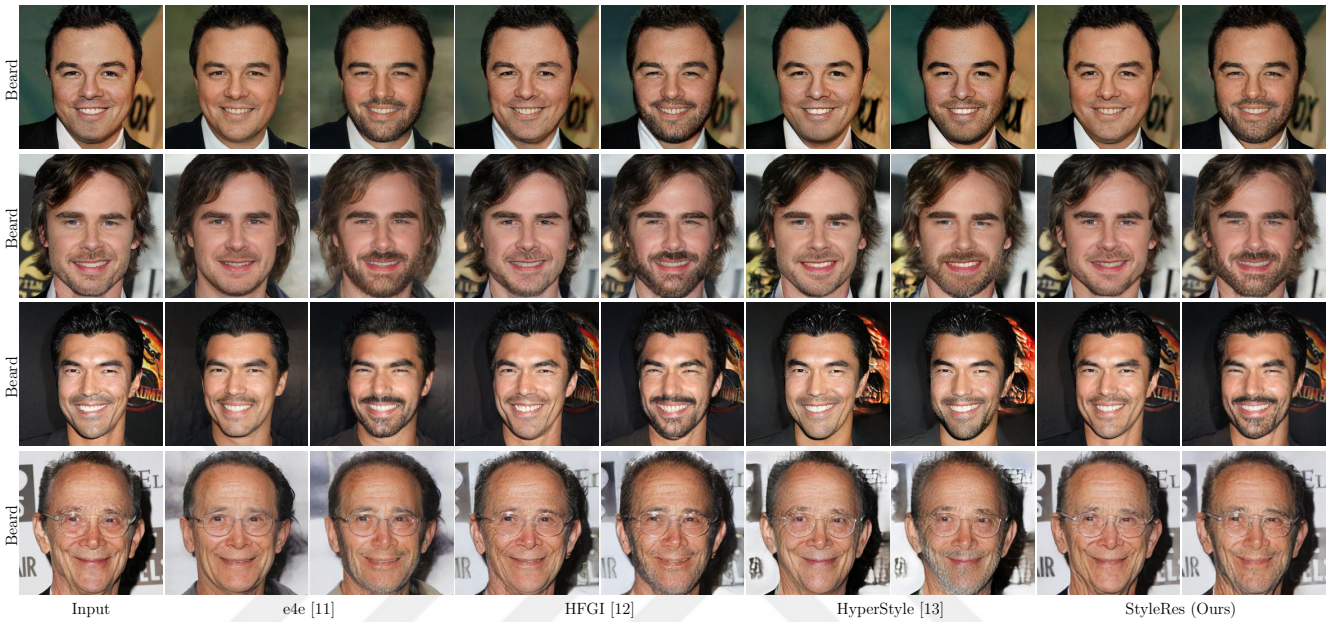


Figure 4.12: Comparisons with e4e, HFGI and HyperStyle on beard addition. For each method, first column shows inversion, and second shows editing.

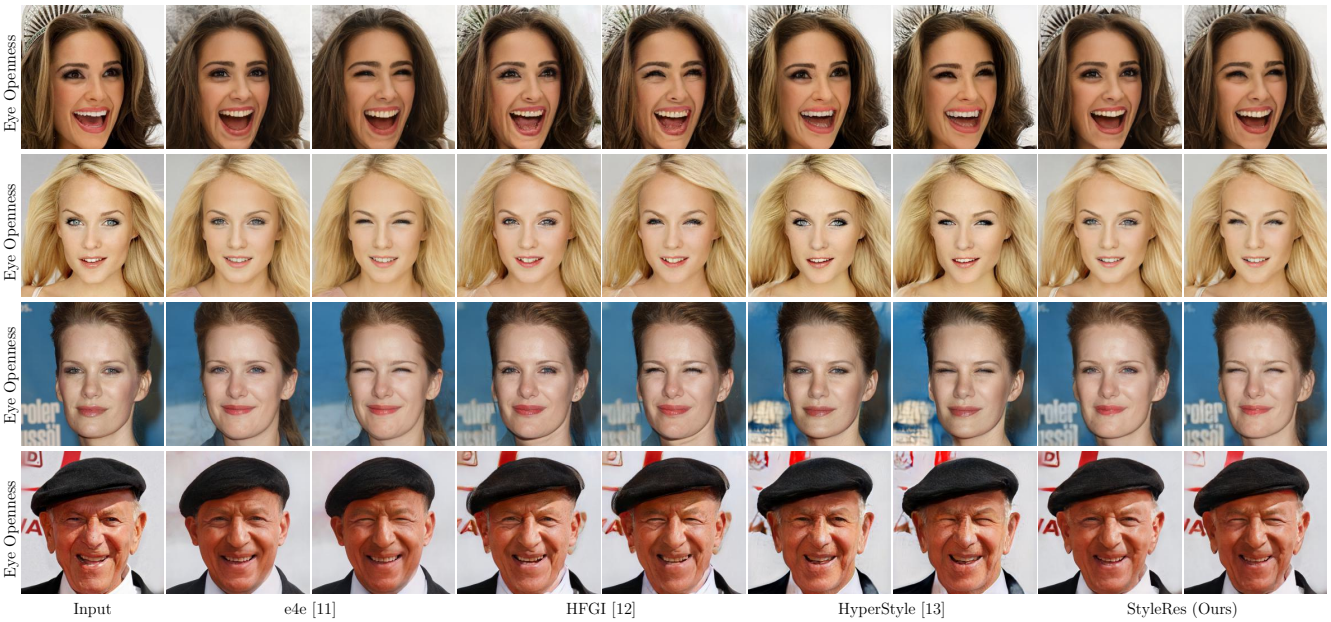


Figure 4.13: Comparisons with e4e, HFGI and HyperStyle on eye-openness. For each method, first column shows inversion, and second shows editing.

Figure 4.14: Comparisons with e4e, HFGI and HyperStyle on lipstick addition. For each method, first column shows inversion, and second shows editing.

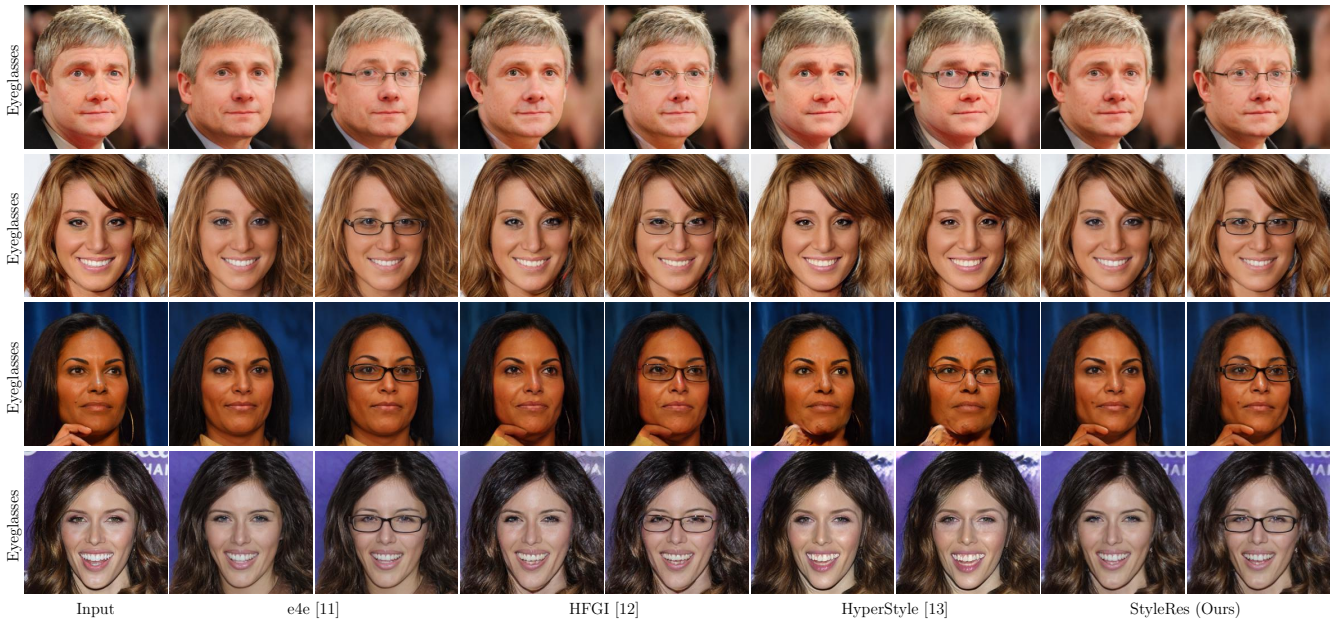


Figure 4.15: Comparisons with e4e, HFGI and HyperStyle on eyeglasses addition. For each method, first column shows inversion, and second shows editing.

Figure 4.16: Comparisons with e4e, HFGI and HyperStyle on bangs addition. For each method, first column shows inversion, and second shows editing.

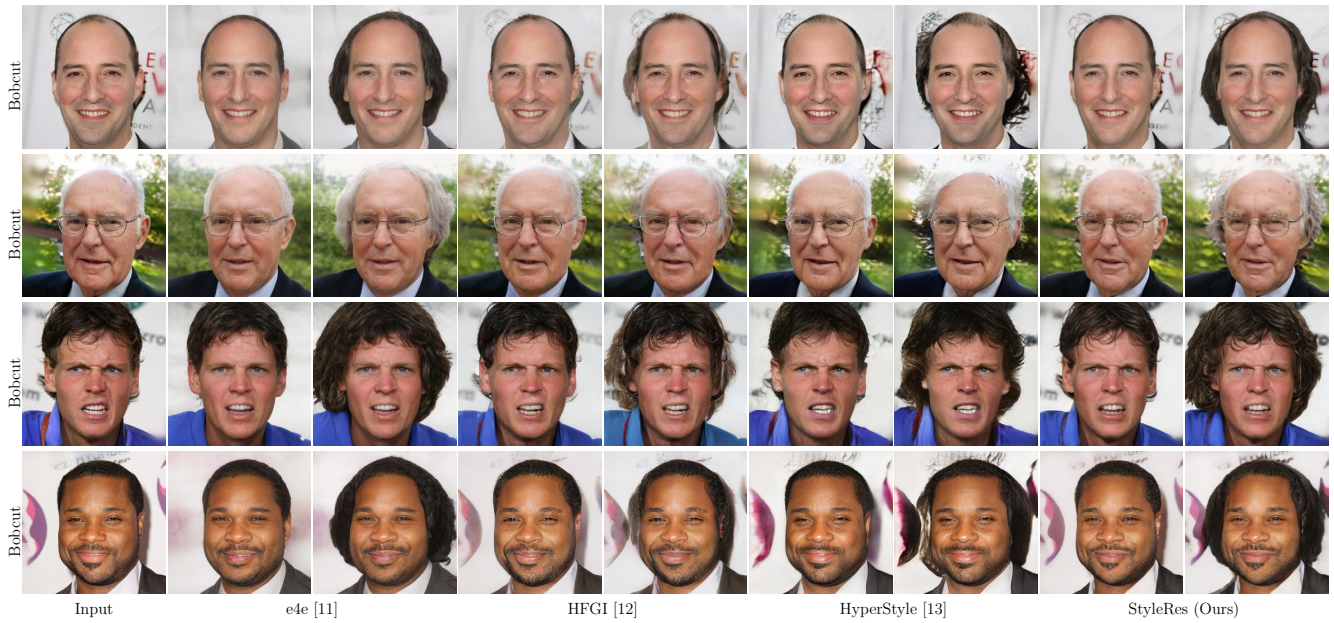


Figure 4.17: Comparisons with e4e, HFGI and HyperStyle on bobcut hairstyle change. For each method, first column shows inversion, and second shows editing.

Figure 4.18: Comparisons with e4e, HFGI and HyperStyle on car color change. For each method, first column shows inversion, and second shows editing.

Figure 4.19: Comparisons with e4e, HFGI and HyperStyle on grass addition. For each method, first column shows inversion, and second shows editing.

4.7 Discussions and Ablation Study

We run extensive experiments to validate the role of each proposed design choice. We first experiment with an architecture that does not have E_1 and E_2 modules. This experiment refers to the case where we learn a network that does not take input from StyleGAN Part 1 features, neither the original (G_{W+}) nor the edited features (G_α). The network directly takes higher layer features (F_0) from the encoder and outputs them to the generator. As shown in Fig. 4.20, the network still tries to learn residual features to achieve reconstructions; however, it achieves poor edits. Without E_2 refers to the experiment with E_1 directly outputting features to Generator Part 2 (as shown in Fig. 3.2). Without E_1 refers to the experiment where E_2 directly takes input from the encoder (F_0) and (G_α) as defined in Sec. 3.2. None of the networks are able to learn the residual features and how to transform them as well as our final architecture since they are not designed to take the original and edited features separately. We also experiment without cycle consistency losses. Fig. 4.21 shows visual results of methods trained with and without cycle consistency constrain. We observe that with cycle consistency constrain, our network achieves preserving fine image details even when images are edited.

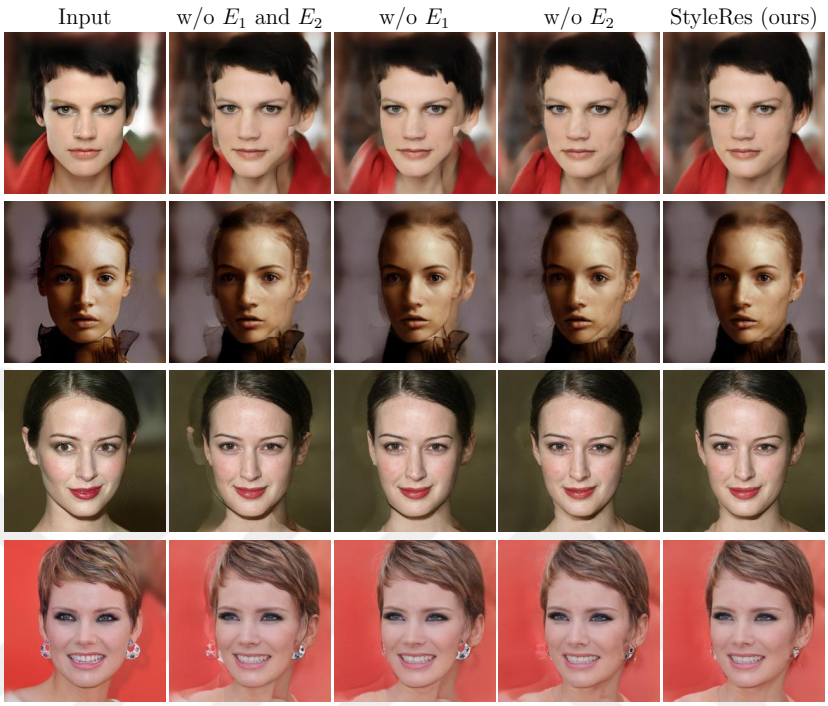
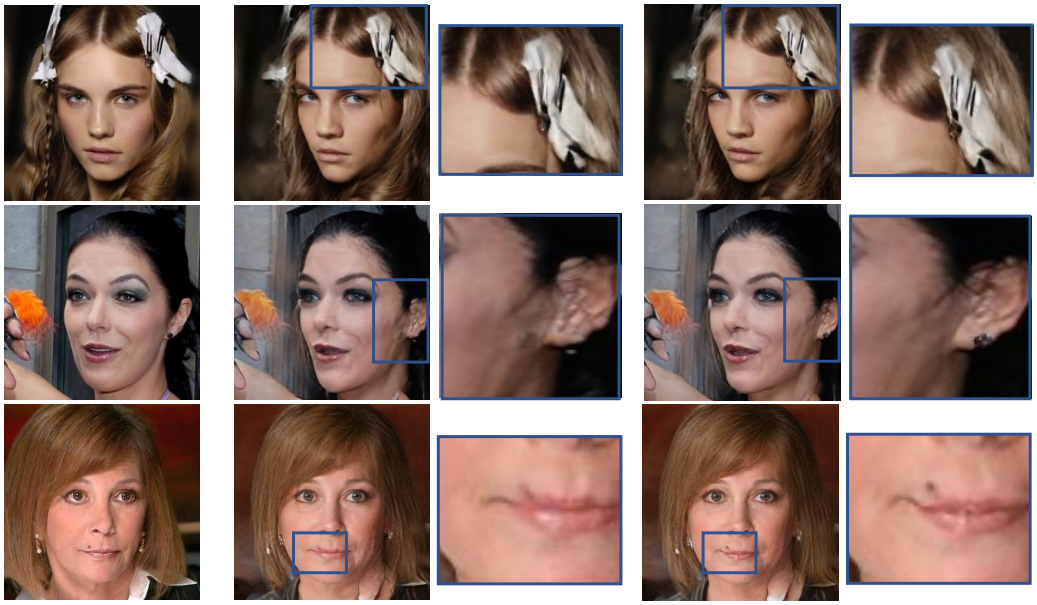


Figure 4.20: Ablation Study. Pose edit outputs of our final architecture and architecture with missing modules. w/o E_1 or E_2, networks struggle to transform features correctly.



(a) Inputs (b) w/o Cycle Consistency (c) w/ Cycle Consistency

Figure 4.21: Ablation Study. Pose edit outputs of models trained with and without cycle consistency loss. The model trained with cycle consistency is better at preserving the details as shown in enlarged boxes.

Chapter 5

Limitations

A common problem with high-rate inversion models is they may not be able to adapt to viewpoint or structure changes. Although our model is trained to adapt better to edits, it can still produce unpleasant results for edits with large misalignments like large pose changes. Examples of such cases are given in Fig. 5.1. We believe that a better way of training would also consider geometric consistency, which is not explicitly modeled with StyleRes when applying the edits. We leave this as future work.

Figure 5.1: High rate inversion models struggle with pose edits. Our model is also affected by this limitation.

Chapter 6

Conclusion

We present a novel image inversion framework and a training pipeline to achieve high-fidelity image inversion with high-quality attribute editing. In this work, to achieve high-fidelity inversion, we learn residual features in higher latent codes that lower latent codes were not able to encode. This enables preserving image details in reconstruction. To achieve high-quality editing, we learn how to transform the residual features for adapting to manipulations in latent codes. We show state-of-the-art results on a wide range of edits explored with different methods both quantitatively and qualitatively while achieving faster run-time than competing methods.

Bibliography

[1] 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.

[2] 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.

[3] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *International conference on machine learning*, pp. 7354–7363, PMLR, 2019.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, "Encoding in style: a stylegan encoder for image-to-image translation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2287–2296, 2021.

[16] R. Abdal, Y. Qin, and P. Wonka, "Image2stylegan++: How to edit the embedded images?," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8296–8305, 2020.

[17] H. Pehlivan, Y. Dalva, and A. Dundar, "Styleres: Transforming the residuals for real image editing with stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1828–1837, June 2023.

[18] A. Creswell and A. A. Bharath, "Inverting the generator of a generative adversarial network," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 7, pp. 1967–1974, 2018.

[19] D. Roich, R. Mokady, A. H. Bermano, and D. Cohen-Or, "Pivotal tuning for latent-based editing of real images," *ACM Transactions on Graphics (TOG)*, vol. 42, no. 1, pp. 1–13, 2022.

[20] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *European conference on computer vision*, pp. 592–608, Springer, 2020.

[21] A. B. Yildirim, H. Pehlivan, B. B. Bilecen, and A. Dundar, "Diverse inpainting and editing with gan inversion," *arXiv preprint arXiv:2307.15033*, 2023.

[22] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8798–8807, 2018.

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] R. Abdal, P. Zhu, N. J. Mitra, and P. Wonka, "Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows," *ACM Transactions on Graphics (ToG)*, vol. 40, no. 3, pp. 1–21, 2021.

[34] Y. Dalva, S. F. Altındiş, and A. Dundar, "Vecgan: Image-to-image translation with interpretable latent directions," in *European Conference on Computer Vision*, pp. 153–169, Springer, 2022.

[35] Y. Dalva, H. Pehlivan, O. I. Hatipoglu, C. Moran, and A. Dundar, "Image-to-image translation with disentangled latent vectors for face editing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[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] 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.

[39] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[40] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *Proceedings of European Conference on Computer Vision (ECCV)*, 2020.

[41] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4690–4699, 2019.

[42] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017.

[43] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *Proceedings of the IEEE international conference on computer vision workshops*, pp. 554–561, 2013.

[44] 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.

[45] Z. Chen, R. Jiang, B. Duke, H. Zhao, and P. Aarabi, "Exploring gradient-based multi-directional controls in gans," in *European Conference on Computer Vision*, pp. 104–119, Springer, 2022.

[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] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

[48] Y. Alaluf, O. Patashnik, and D. Cohen-Or, "Restyle: A residual-based stylegan encoder via iterative refinement," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6711–6720, 2021.

[49] 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.

[50] X. Yao, A. Newson, Y. Gousseau, and P. Hellier, "A style-based gan encoder for high fidelity reconstruction of images and videos," *European conference on computer vision*, 2022.