Latent to Latent: A Learned Mapper for Identity Preserving Editing of Multiple Face Attributes in StyleGAN-generated Images

Siavash Khodadadeh* khodadad@adobe.com
Shabnam Ghadar* ghadar@adobe.com
Saeid Motiian* motiian@adobe.com
Wei-An Lin* wlin@adobe.com
Ladislau Bölöni† Ladislau.Boloni@ucf.edu
Ratheesh Kalarot* kalarot@adobe.com

* Adobe Inc.
San Jose, CA 95110

† Department of Computer Science
University of Central Florida
Orlando, FL 32816

Abstract

Several recent papers introduced techniques to adjust the attributes of human faces generated by unconditional GANs such as StyleGAN. Despite efforts to disentangle the attributes, a request to change one attribute often triggers unwanted changes to other attributes as well. More importantly, in some cases, a human observer would not recognize the edited face to belong to the same person. We propose an approach where a neural network takes as input the latent encoding of a face and the desired attribute changes and outputs the latent space encoding of the edited image. The network is trained offline using unsupervised data, with training labels generated by an off-the-shelf attribute classifier. The desired attribute changes and conservation laws, such as identity maintenance, are encoded in the training loss. The number of attributes the mapper can simultaneously modify is only limited by the attributes available to the classifier – we trained a network that handles 35 attributes, more than any previous approach. As no optimization is performed at deployment time, the computation time is negligible, allowing real-time attribute editing. Qualitative and quantitative comparisons with the current state-of-the-art show our method is better at conserving the identity of the face and restricting changes to the requested attributes.

1. Introduction

Attribute-based editing of human face images is an active research field with many immediate practical applications. Starting from an image of a person, the objective is to generate a new picture where one or more specific attributes such as age or facial expression are changed. To provide the user with the greatest amount of creative control, this change needs to be implemented in such a way as to keep all the other attributes as close as possible to their original values. It is not desirable, for instance, for the procedure to add eyeglasses to the face when making it older: the presence of the glasses should be an independently controllable attribute. Furthermore, in many applications, it is a prerequisite that the person in the modified image remains recognizably the same person as in the original one, even if many attributes had changed.

Until recently, such functionality was achieved by creating 3D models of faces, and implementing the changes by either modifying the 3D model or by replacing the textures. However, progress towards achieving photorealistic images using these techniques had been slow.

Recent progress in generative adversarial networks (GANs) offers a different approach for the generation of face images. Inspired by the style transfer literature, architectures such as StyleGAN [12] and StyleGAN-v2 [13] generate face images by successively applying different "styles" at the various layers of the network. Although the images show a non-existing person, randomly sampled from the latent space of possible faces, the images are photorealistic to the degree that most observers mistake them for photographs. More recently, GAN inversion techniques made possible to find a position corresponding to a real-world photograph in the latent space, allow the creation of a workflow that can change the attributes of arbitrary pictures. Techniques that rely on autoencoders instead of GANs can use the encoder component for the same purpose [19].

Once we have the latent space representation of a picture, it is a reasonable assumption that variants of the picture with one or more changed attributes will be encoded by nearby latent space points. Several recent papers investigated the possibility of finding directions in the latent space that cor-

Figure 1: Examples of attribute edits with the proposed method. Notice that the background remains usually the same, and the edits are restricted to specified attribute.

respond to specific attribute changes. For instance, we can find the latent space direction corresponding to changing an attribute by relying on samples with positive and negative values for the attribute. Unfortunately, the attributes are entangled and non-linear in the latent space. For instance, the gradual aging of a person does not correspond to movement in the latent space along a linear trajectory. The farther one moves on the trajectory, the more likely that spurious attribute changes appear in the generated images. To alleviate this problem, more recent papers are finding non-linear mappings [1]. A further technique that conserves the initial attributes of the image is to apply the changed latent vector only to some attribute dependent "style" layers of the generator - for instance to layer 6 and 7 of a StyleGAN for "age" [1, 6].

In this paper, instead of finding directions or paths in the latent space, we train a neural network to perform a *latent-to-latent transformation* which finds the latent encoding corresponding to the image with the changed attribute. As the technique is one-shot, it does not rely on a linear or non-linear trajectory of the gradual change of the attributes. By training the network end-to-end over the full generation pipeline, the system can adapt to the latent spaces of off-the-shelf generator architectures. Conservation properties, such as maintaining the identity of the person can be encoded in the form of training losses. Once the latent-to-latent network was trained, it can be reused for arbitrary images without retraining. Applying the attributes is as simple as forward propagating through the latent-to-latent network, and rendering the resulting latent code through the generator. The attribute change adds an insignificant overhead to the image generation.

Training the latent-to-latent network does not require manual image labeling – it can be performed using an external, pre-trained, off-the-shelf attribute recognizer. Thus the number of attributes that our network can independently control is only limited by the capabilities of the recognizer(s) - if one has a recognizer for an attribute, we can add it to arbitrary faces. In our experiments, we trained the latent-to-latent network to allow the adjustment of 35 different facial attributes, more than any previous approach.

Another advantage of the end-to-end training is that it allows us to choose the conservation properties of the generation process beyond the specified attribute change. For instance we consider it important that we only change the specified attributes and keep the other ones constant – for instance, changing the age should not automatically also change the hair color. Another conservation property that is important in almost every conceivable application is that the face with the changed attributes remains recognizably the same as the one in the original image. In our architecture this can be achieved by adding an identity loss by calculating the Euclidean distance between FaceNet features of both the original image and the generated images [22]. Other conservation properties that can be highly relevant in certain applications include maintaining the openness of the eyes, the lighting, the clothes, the background and so on.

Another observation is that our system does not necessarily require the selective application of the changed latent vector to only some of the style layers of the generator. However, if desired, this technique can be added to the approach.

The main contributions of this paper can be summarized as follows:

- We describe a novel technique for editing the attributes of a face using a neural network that performs a transformation in the latent space of a generator.
- We show that latent to latent network can be trained in an end-to-end setting without requiring human label-

ing using off-the-shelf attribute regressors. The training process allows the user to specify the desired conservation properties using custom losses.

- Compared to current state-of-the-art methods, we find that our approach significantly reduces the cases when editing an attribute also changes the value of a different attribute which had not been requested to change, such as a change in gender when adjusting age. Our approach also reduces the cases when an attribute change leads to a loss of the personal identity of the face.

2. Related work

Editing the attributes of a human face is a longstanding goal in computer graphics. In recent years deep generative models brought a new perspective to this approach. While early GANs created images that could be easily distinguished from photographs, recent GANs such as ProgressiveGAN [11], StyleGAN [12] and StyleGAN2 [13] create photorealistic images of faces or cars. Recently, autoencoder-based techniques also achieved comparable results [19].

To edit the attributes of a face corresponding to the user's intent, we need to have a way to control the image generation. One possible approach is based on conditional GANs, a technique first proposed by [17]. The idea is to condition image generation on a label or other information such as images [9]. A related technique, CycleGAN [28] learns to translate an image between two domains with different attributes, such as winter and summer, without the need for paired training data. The StarGAN approach [4] extends this model to multiple domains.

Approaches that train a different GAN for each domain would not scale to large number of attributes - the 35 attributes we are editing would correspond to 2^{35} domains even in the simplest case of binary attributes. Another consideration is that while training a high-resolution GAN demands significant computational resources, once the training is done, generating images from a latent space input is a fast and computationally inexpensive process. This observation led to a different approach for image manipulation, based on manipulating the latent vectors. The roots of this approach go back to models such as word2vec [16]. [20] showed that interpolating the latent vectors in generative models enables interesting edits.

To use latent space edits for face attribute editing, we need to find transformations of latent vectors that correspond to changes in specific attributes. [23] propose finding directions in the latent space along which a specific attribute increases or decreases. [25] propose a method for discovering style channels and identifying the channels that control an attribute. StyleFlow [1] uses continuous normalizing flows and a neural differential equations solver [3] to synthesize images from a vector and set of target attributes. [6] introduced GANSpace, an unsupervised method that uses principal component analysis to find directions for edits. After finding these directions, GANSpace requires the user to observe the outputs and manually select meaningful directions based on the target attribute. All the methods above rely on an approach where only some of the style layers are modified in generator. The choice of the layers to modify had been acquired through trial-and-error. For example, to edit the yaw or pitch, it was found that the best results are obtained when changing only the first 4 layers of StyleGAN. Recently, [8] proposed an approach that uses attention to find the most relevant layers automatically. [5] proposed instance-aware search that leverages attribute classifiers and disentanglement transformation metric to achieve disentangled edits.

[10] proposes a neural network that finds directions in the latent space based on target parameters without conditioning it on input latent vector. StyleRig [24] uses losses defined between the generated image and original image based on pose, illumination and expression. [18] propose a method that combines attributes and face identity features from two different images to generate a new image with losses on identity and attributes.

The approaches based on latent space manipulation are usually relying on a variation of a StyleGAN approach to generate the images, which means that even the original image is GAN-generated. However, many applications require the editing of a real-world image. For this we need to first find a latent space vector that faithfully reconstructs the real image, a step called GAN inversion, recently surveyed by [26].

3. Latent-to-latent

3.1. Defining and measuring face attributes

Before we proceed to modify the attributes of a face, let us discuss how we define and measure such attributes. When asked to describe a face in an image I, humans will use attributes such as gender, age, hair, presence of eyeglasses or pose. We represent these human evaluations as a collection of attributes associated with the image $\boldsymbol{a} = \{a_1, \cdots, a_N\}$ $\forall i\ a_i \in [0, 1]$. With their origins in human language, it is not possible to make all attributes orthogonal; for instance, "mustache" and "beard" imply the "facial hair" attribute. In many cases, however, the attributes are independent, but statistically correlated in the human population: for instance age is independent from hair color, even if high age is often correlated with white hair.

We call an *attribute regressor* a function $\mathcal{H}(I) \rightarrow \widehat{\boldsymbol{a}}$ that takes in an image and returns an estimate of the attribute values. As we shall see below, having a pre-trained network that can *measure* attributes is a key component in our ability

to train a network that can *adjust* those attributes.

A convenient way to build an attribute regressor is to take advantage of the rich set of features extracted by large image recognizer networks. We used a randomly initialized ResNet50 network and processed its feature output through an additional convolution layer and two fully connected layers (see section 4.1 for more details). The final output is a direct representation of the estimated attributes $\widehat{\boldsymbol{a}}$. The number of attributes are only limited by the availability of the labeled training data. In this paper we used a set of $N = 35$ different attributes.

The network was trained with supervised multi-class training, with a training dataset of image and attribute vector pairs $\mathcal{D} = \{(I_i, \boldsymbol{a}_i)\}$. There are several ways in which such a dataset can be created. One possibility is to start from a dataset of faces and manually label them. Another approach, which we took in this work is to gather the training data using existing tools such as the Microsoft Face API [15].

3.2. Adjusting the attributes with a latent-to-latent network

Let us consider an image I with attribute vector $\boldsymbol{a}$. Our objective is to allow the user to specify a different attribute vector $\boldsymbol{a}' = \boldsymbol{a} + \Delta\boldsymbol{a}$ and generate a new image I' with the changed attributes. Decades of work in computer graphics illustrated the difficulty of performing such a transformation in the image domain.

Our approach is based on the idea that both I and I' is generated by a StyleGAN-type generator $\mathcal{G}$ from the corresponding latent vectors $\boldsymbol{w}$ and $\boldsymbol{w}'$ from the latent space W_P. If I originated from a real-world photo, we can find $\boldsymbol{w}$ through a GAN inversion procedure. The attribute edit will be performed by a latent-to-latent transformation using a network representing a parameterized function $f(\boldsymbol{w}, \Delta\boldsymbol{a}, \theta) \rightarrow \boldsymbol{w'}$, such that $\mathcal{H}(\mathcal{G}(\boldsymbol{w}')) = \boldsymbol{a}'$, where θ is a trainable parameter.

We note that the latent-to-latent transformation f finds the new latent point $\boldsymbol{w}'$ for all the attribute changes specified in $\Delta\boldsymbol{a}$ in a *single shot*. In contrast to other approaches, it does not perform the transformation gradually by determining and following directions in the latent space corresponding to certain attributes, nor it relies on the addition of changes along the different attributes. Finally, our approach does not aim to modify the latent representation to make it more amenable to attribute changes. Thus, our approach sidesteps the challenges associated to the nonlinearities of the latent space.

We implement the transformation f through a neural network whose architecture reflects its objectives. The input and the output are determined by the dimensionality of the latent space, which in our case is the 18x512 model of StyleGAN; in addition to this the input is concatenated with the 35-dimensional attribute change vector. With the dimensions of the latent space representing "styles" as opposed to spatial maps, the use of convolutional layers are not justified in the transformation network. Our transformation is single shot, making recurrent architectures not applicable. We thus use a fully connected, multilayer network, with the free parameters being the size and number of layers and the type of non-linearity used. As we aim to preserve all the details in the input images while editing the attributes, we chose to make the hidden layers the same size as the input and output ones. For the number of layers, we performed experiments with 1, 2, 3 and 4 layers; while adding layers increases the expressiveness of the network, it also significantly increases the need for training data and the computational cost of the training (see appendix 10 for ablation studies on the network).

A special consideration involves the nonlinearity used in the network as previous research demonstrated that simply following a linear direction for an attribute does not yield good results. While ReLU is currently the default non-linearily in neural networks, in our network the use of ReLU would inevitably suppress some of the information as it propagates through the network, leading us to a use of a tanh nonlinearity.

3.3. Training the latent-to-latent network

In principle, the latent to latent transformation network f could be trained using a straightforward supervised learning technique using curated and labeled face images. However, such a process is unrealistically expensive.

The training procedure we propose is based instead on embedding the network in a framework where information flows through a number of pre-trained components with frozen weights in addition to the trainable f network (see Figure 2). Applying different flows of information through these components we can define a number of different loss functions that contribute to the training of the latent-to-latent network.

The input to this framework takes the form of triplets $(I, \boldsymbol{w}, \boldsymbol{a})$ of an image, the corresponding latent encoding and its initial attributes. We only need either an initial image I or an initial latent encoding $\boldsymbol{w}$ to create this triplet. If we start with $\boldsymbol{w}$, the image I can be created using the generator $\mathcal{G}$. If we start with I, the latent encoding can be created through GAN inversion. In both cases, the attribute regressor $\mathcal{H}$ can be used to find $\boldsymbol{a}$. Thus, the training process is based on an unsupervised input data.

The training is proceeding through randomly generated tasks for the latent-to-latent transformer, organized in batches. To create a task, we sample a triplet from $\mathcal{U}$ and randomly generate a target attribute vector $\boldsymbol{a}'$ which only changes a single attribute from the attribute vector. As we discussed above, changing only a single attribute might cre-

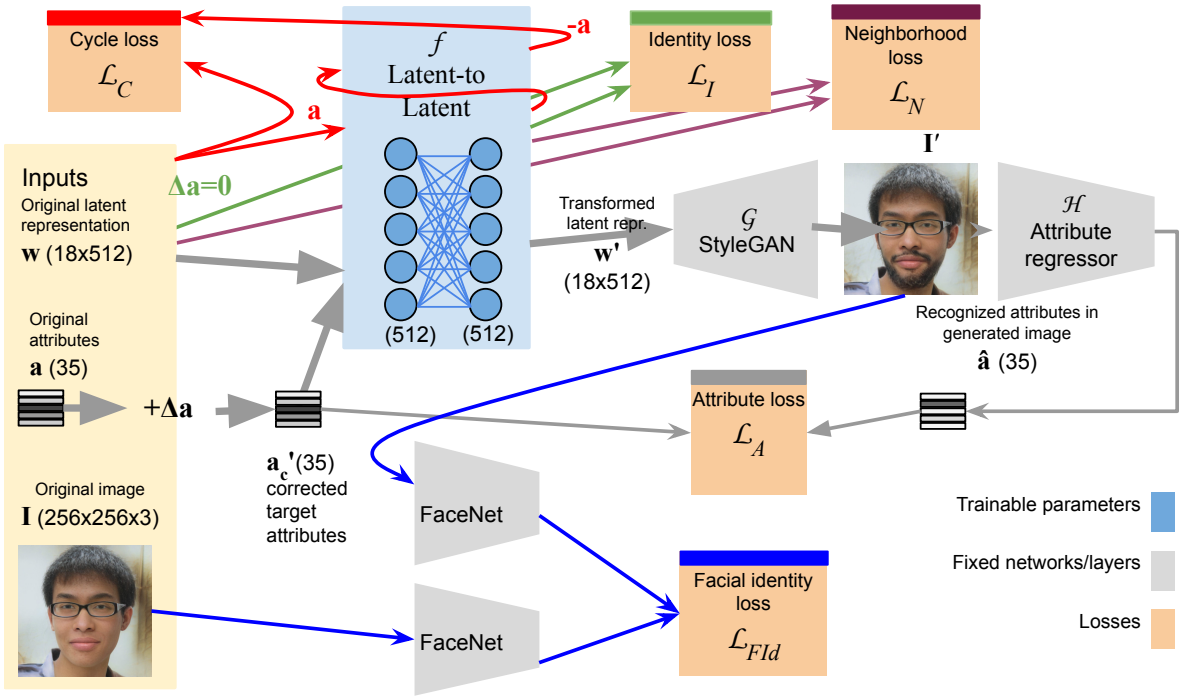

Figure 2: The embedding framework for the training of the latent-to-latent network f. Every network except f (presented in blue) is fixed during the training. The training is based on different information flows leading to the calculation of the attribute loss (gray), cycle loss (red), identity loss (green), neighborhood loss (purple) and facial identity loss (purple). The thicker gray arrows show the flow of information during training time. All the other components are only used during training.

ate unfeasible attribute vectors, such as a bearded man with no facial hair. Thus, we first correct the target attribute by adding a correction term obtained from the mutual information matrix (see appendix 7 for details), leading to a corrected attribute vector $\boldsymbol{a}_c'$. The input of the latent-to-latent network will be thus $\Delta\boldsymbol{a} = \boldsymbol{a}_c' - \boldsymbol{a}$. This input is forward propagated through the full network along several paths, each leading to a different loss as shown in Figure 2. These losses are then back-propagated into updates for f.

Target attribute loss: expresses the primary goal of the network. The attributes of the generated image should be as close as possible to the corrected target attribute. We use the mean square error between the output $\hat{\boldsymbol{a}}$ and the corrected target value $\boldsymbol{a}_c'$, as the behavior of the mean square error to penalize outlier values is actually helpful here.

$$\mathcal{L}_A = ||\hat{\boldsymbol{a}} - \boldsymbol{a}_c'||^2 \tag{1}$$

Note that this loss penalizes not only if the modified attributes are not achieved, but also changes in attributes that were not supposed to be changed.

Cycle loss: Logically, an attribute change for the same person should be reversible: increasing the age with ten years and then decreasing it should result in the same image. This is not automatically enforced by the target attribute loss, as in this process it is possible that other aspects of the picture not covered by a named attribute would change. We implement this objective in a form of a cycle loss, implemented as a distance between the original latent vector $\boldsymbol{w}$ and the latent vector obtained by applying two attribute changes with opposite signs. We will use the L1 distance for measuring distances in the latent space:

$$\mathcal{L}_C = ||\boldsymbol{w} - f(f(\boldsymbol{w}, \Delta\boldsymbol{a}), -\Delta\boldsymbol{a})|| \tag{2}$$

Identity loss: Another desirable property of the latent-to-latent network is that when no attribute change is requested, it will map the latent vector to itself. This is expressed by the following loss:

$$\mathcal{L}_I = ||\boldsymbol{w} - f(\boldsymbol{w}, 0)|| \tag{3}$$

Neighborhood loss: A challenge with latent space transformations is that due to the non-linearity and entangled nature of the latent space, relatively small attribute transformations might put the representation far from the initial location. The fact that the attribute loss penalizes changes in attributes that were not requested still leaves open the possibility of many changes in the image that are not covered

by the attributes - such as background, clothing or illumination. The neighborhood loss penalizes outputs that depart significantly from the original latent vector, encouraging the network to achieve the attribute transformation with the minimum amount of the departure from the original latent vector:

$$\mathcal{L}_N = ||\boldsymbol{w} - f(\boldsymbol{w}, \Delta\boldsymbol{a})||^2 \quad (4)$$

When training an attribute transformer network, the user might want to ensure some specific conservation properties that are not covered by the "keep attributes unchanged unless explicit change is requested" rule enforced by the attribute loss. For instance, the user might want to keep the background the same, or keep the eyes of the person open or closed, and so on. Our architecture allows the user to add additional losses that express these custom conservation properties. In the following, we will discuss a particular example of such a custom loss, based on the identity of the person represented in image I. We will define $s(I, I') \in [0, 1]$ a function that estimates whether the person would recognize the images I and I' as belonging to the same person. Due to evolutionary pressure for recognizing members in their social groups, humans are very good at estimating this function, even if many attributes are different between the images I and I'. On the other hand, you can have two faces that have all the attributes matching, which however clearly belong to two different people. Thus, this conservation property cannot be expressed in terms of attributes. We find, however, that networks trained for face recognition, although they do not have the concept of attributes, match closely the human perception of facial identity.

Face identity loss: To express this loss, we rely on FaceNet [22], a network that had been trained explicitly for facial recognition. If we denote with $\mathcal{F}(I)$ the features extracted by FaceNet from an image, we can express the face identity loss as follows:

$$\mathcal{L}_{FId} = ||\mathcal{F}(\mathcal{G}(\boldsymbol{w})) - \mathcal{F}(\mathcal{G}(f(\boldsymbol{w}, \Delta\boldsymbol{a})))||^2 \quad (5)$$

4. Experimental validation

4.1. Training the attribute regression network

We trained the attribute regression network on three different datasets: CelebAMask-HQ [14], FFHQ [12] and a locally generated dataset. The latter was generated by sampling 400K vectors from the $\mathcal{Z}$ space of StyleGAN-v2 and using these to find corresponding $\boldsymbol{w}$ vectors. We truncated the vectors by a factor of 0.7 following the StyleGAN protocol in order to avoid the generation of out-of-distribution low-quality images. We extracted attributes for the generated images using the Microsoft Face API [15]. We split the data into a 90/10 split that corresponds to 721718 images for training and 72172 images for the test. For more details, including the accuracy of regression network on the test, please refer to appendix 7.

4.2. Qualitative evaluation

In the first series of experiments, we test whether our approach fulfills our objectives for (a) being able to adjust a large number of different attributes and (b) being able to adjust a given attribute without modifying other attributes (except in cases when this is inevitable, such as the pairs of beard and facial hair). Figure 3 shows an example of six different attributes being changed on six different faces. We find that our approach is *mostly* successful on leaving unchanged attributes other than the requested one. We also find that other features of the images, not covered by the attributes, such background, clothes and illumination as well as the personal identity of the face what also been successfully retained in these images. In the case of other images, such as certain examples in the next section, we see that these conservation of attributes and features with our technique is not always fully successful.

A wider range of pictures and adjustments can be seen in our anonymized demo video at https://youtu.be/fptbQi_yIDg.

4.3. Qualitative comparisons with SOTA baselines

In this section, we qualitatively compare our approach with three state-of-the-art methods for face editing: InterFaceGAN [23], GANSpace [6], and StyleFlow [1]. Although our method can edit 35 different attributes, for the comparison, we choose eight attributes that are supported by all these methods: Age, Baldness, Beard, Expression, Gender, Glasses, Pitch, and Yaw. Whenever possible, we extended the publicly available codebases [1]. For GANSpace, we used components provided by Härkönen [6] that best match these attributes. For InterFaceGAN, we trained two new, additional directions for Baldness and Beard since these attributes were not in the repository. As the starting images we used 50 images sampled from the StyleFlow test set which were not used as training data by any of the approaches.

One of the challenges in comparing these methods is that the same request, such as adding 0.2 or subtracting 0.3 from the age is interpreted differently by different methods. To calibrate the ranges of valid values for each technique, we found the smallest and largest attribute value for which the MTCNN face detector still found a valid face in the generated image. Then, we divided this range into 40 intervals and generated the 41 images at the interval boundaries. Thus, we used $8 \times 50 \times 41 = 16400$ generated images in the qualitative comparisons.

[1] StyleFlow: https://github.com/RameenAbdal/StyleFlow, GANSpace: https://github.com/harskish/ganspace , InterFaceGAN: https://github.com/a312863063/generators-with-stylegan2

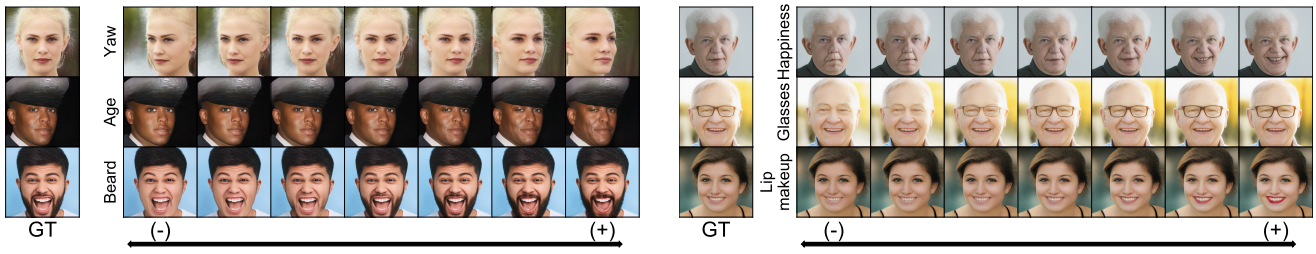

Figure 3: Changing six different attributes on six different faces.

Figure 4: Examples of cases where L2L conserves attributes while other SOTA approaches introduce unrequested attribute changes. **Top Left:** Changing gender or hair color when changing age. **Top Right:** Eyebrows' style is preserved the most by our method due to face features loss. **Bottom Left:** Adding glasses or changing hair color when balding or adding hair. **Bottom Right:** Changing gender when adding or removing facial hair.

Figure 4 (top left) compares the approaches in adjusting the Age attribute. All approaches generated images that correspond to persons of the specified ages. What differentiates our approach, however, is that it largely succeeds in changing *only* the age attribute. For instance, both InterFaceGAN and StyleFlow also turn the hair white when generating an older person. In addition, InterfaceGAN and StyleFlow also changed the gender when making the person older.

Another example of the editing the age attribute is shown in Figure 4 (top right). In this case, the GANSpace and InterFaceGAN approaches changed the eyebrows, illumination as well as the background of the figure. These extraneous changes are reduced with our approach. Nevertheless, our approach also changed the expression of the mouth and, to some degree, the background.

Another illustration of how a change in an attribute might trigger inadvertent changes in other attributes is the change of the Baldness attribute in Figure 4 (bottom left). In this example, both StyleFlow and GANSpace added glasses to the face when the baldness attribute was moved in the negative direction, while InterfaceGAN radically changed the hair color. Our approach had shown a significantly smaller degree of hair color change.

Another example that shows the impact of the neighborhood and personal identity losses in the training of the latent-to-latent network is the adjusting of the attribute

Table 1: Quantitative results. IG, SF, GS, and L2L refer to InterFaceGAN, StyleFlow, GANSpace and our proposed method.

(a) Inference speed (seconds)

METHOD	IG	SF	GS	L2L
TIME	0.1080	0.6783	0.1095	0.1654

(b) Retaining the facial identity: the average cosine distance between the MTCNN features of the original and edited images (the lower the better)

ATTRIBUTE	IG	SF	GS	L2L
AGE	0.77	0.65	**0.46**	0.57
BALDNESS	0.53	0.64	0.46	**0.21**
BEARD	0.57	0.55	0.53	**0.28**
EXPRESSION	0.60	0.21	0.23	**0.14**
GENDER	0.54	0.52	0.58	**0.28**
GLASSES	0.59	0.46	**0.24**	0.25
PITCH	0.54	**0.51**	**0.51**	**0.51**
YAW	**0.39**	0.46	0.41	0.46

(c) Metrics for comparing the quality of edited faces. All metrics are computed based on the Inception features.

METRIC	IG	SF	GS	L2L
FID	43.16	41.64	45.44	39.83
KID	0.0118	0.0086	0.0122	0.0062
ISD	0.0025	0.0014	0.0044	0.0085

Beard in Figure 4 (bottom right). In the original image, the person did not have a beard; all four approaches were successful in adding a beard when moving towards the increased attribute value. However, all three baseline approaches change the person to female when the attribute change was in the negative direction. (Note that our system has an explicit editable attribute for gender). In addition, GANSpace also significantly changed the background of the edited image.

4.4. Quantitative comparisons

A possible way to quantitatively evaluate how well an approach preserves facial identity is to measure whether the images would be recognized by an external facial recognition system. We use the features extracted by MTCNN [27] which are different from FaceNet features that we used for training. In Table 1b we report the cosine distance between the feature vectors of the original vs. edited images averaged over all test subjects (we also report the Euclidean distance in supplemental material). We find that our method outperforms the other methods on 5 out of 8 target attributes, and also scores very close to the winner on two more attributes.

To evaluate the quality of the edited images, we compared the set of images from Section 4.3 with a set of 1000 images generated by FFHQ StyleGAN-v2 with truncation=0.7. The results for the Frechet Inception Distance(FID) [7], Inception Score(IS) [21], and Kernel Inception Distance(KID) [2] are shown in Table 1c. We find that our approach achieves a better result on the FID and KID compared to the baselines. For the ISD metric, all approaches achieve a result smaller that 0.01, making them effectively equally good on this metric.

In addition, we measure the time necessary to generate a new image with the adjusted attribute. This metric is important for interactive applications and when mass generation of images is necessary. The measured average generation time is shown in Table 1a. We find that our generation time is somewhat longer than the InterFaceGAN and GANSpace approaches, but more than four times faster than StyleFlow. In practice, we found that the approach is suitable for real-time interactive editing, as illustrated in the linked video.

Finally, more qualitative and quantitative results and ablation studies could be found in our supplementary material. We also perform a user study, and the details are in supplementary material submitted with the paper.

5. Conclusions

In this paper, we introduced a novel technique for editing face attributes in a StyleGAN-generated image. Our approach is based on a network that maps the latent vector of an image to the latent vector of the desired target image. The latent-to-latent mapper can be learned by embedding it in a larger network and training it with a combination of losses. Some of these losses enforce the new attribute values, while others guide the network to find solutions that retain the personal identity of the person and other characteristics of the original picture. Through qualitative and quantitative experiments we found that, compared to competing state-of-the-art approaches, our approach can handle a larger number of attributes and it is better at maintaining the personal identity of the face and restricting the changes to the requested attributes.

6. Acknowledgements

We would like to thank the anonymous reviewers for their feedback on this paper. We are also thankful to Adobe Sensei and Search department for providing computational resources for this research as well as providing feedback on the paper.

References

[1] Rameen Abdal, Peihao Zhu, Niloy Mitra, and Peter Wonka. StyleFlow: Attribute-conditioned exploration of StyleGAN-generated images using conditional continuous normalizing flows. *arXiv preprint arXiv:2008.02401*, 2020.

[2] Mikołaj Bińkowski, Dougal J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd GANs. *arXiv preprint arXiv:1801.01401*, 2018.

[3] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366*, 2018.

[4] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 8188–8197, 2020.

[5] Yuxuan Han, Jiaolong Yang, and Ying Fu. Disentangled face attribute editing via instance-aware latent space search. *arXiv preprint arXiv:2105.12660*, 2021.

[6] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. GANSpace: Discovering interpretable GAN controls. *arXiv preprint arXiv:2004.02546*, 2020.

[7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *Advances in neural information processing systems (NeurIPS)*, pages 6626–6637, 2017.

[8] Xianxu Hou, Xiaokang Zhang, Linlin Shen, Zhihui Lai, and Jun Wan. GuidedStyle: Attribute knowledge guided style manipulation for semantic face editing. *arXiv preprint arXiv:2012.11856*, 2020.

[9] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proc. of the IEEE Conf. on computer vision and pattern recognition (CVPR)*, pages 1125–1134, 2017.

[10] Ali Jahanian, Lucy Chai, and Phillip Isola. On the "steerability" of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019.

[11] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[12] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4401–4410, 2019.

[13] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 8110–8119, 2020.

[14] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. MaskGAN: Towards diverse and interactive facial image manipulation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5549–5558, 2020.

[15] Microsoft Azure Face. 2021. `https://azure.microsoft.com/en-in/services/cognitive-services/face/`. Online; accessed 19 January 2021.

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[17] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[18] Yotam Nitzan, Amit Bermano, Yangyan Li, and Daniel Cohen-Or. Face identity disentanglement via latent space mapping. *ACM Transactions on Graphics (TOG)*, pages 1–14, 2020.

[19] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. *arXiv preprint arXiv:2007.00653*, 2020.

[20] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[21] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training GANs. *Advances in neural information processing systems (NeurIPS)*, pages 4844–4852, 2016.

[22] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proc. of the IEEE Conf. on computer vision and pattern recognition (CVPR)*, pages 815–823, 2015.

[23] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. InterFaceGAN: Interpreting the disentangled face representation learned by GANs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020.

[24] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhofer, and Christian Theobalt. StyleRig: Rigging StyleGAN for 3d control over portrait images. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 6142–6151, 2020.

[25] Zongze Wu, Dani Lischinski, and Eli Shechtman. StyleSpace analysis: Disentangled controls for StyleGAN image generation. *arXiv preprint arXiv:2011.12799*, 2020.

[26] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. GAN inversion: A survey. *arXiv preprint arXiv:2101.05278*, 2021.

[27] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 2016.

[28] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proc. of the IEEE Int'l Conf. on computer vision (ICCV)*, pages 2223–2232, 2017.