InterFaceGAN: Interpreting the Disentangled Face Representation Learned by GANs

Yujun Shen, Ceyuan Yang, Xiaoou Tang, *Fellow, IEEE,* and Bolei Zhou, *Member, IEEE*

Abstract—Although Generative Adversarial Networks (GANs) have made significant progress in face synthesis, there lacks enough understanding of what GANs have learned in the latent representation to map a random code to a photo-realistic image. In this work, we propose a framework called InterFaceGAN to interpret the disentangled face representation learned by the state-of-the-art GAN models and study the properties of the facial semantics encoded in the latent space. We first find that GANs learn various semantics in some linear subspaces of the latent space. After identifying these subspaces, we can realistically manipulate the corresponding facial attributes without retraining the model. We then conduct a detailed study on the correlation between different semantics and manage to better disentangle them via subspace projection, resulting in more precise control of the attribute manipulation. Besides manipulating the gender, age, expression, and presence of eyeglasses, we can even alter the face pose and fix the artifacts accidentally made by GANs. Furthermore, we perform an in-depth face identity analysis and a layer-wise analysis to evaluate the editing results quantitatively. Finally, we apply our approach to real face editing by employing GAN inversion approaches and explicitly training feed-forward models based on the synthetic data established by InterFaceGAN. Extensive experimental results suggest that learning to synthesize faces spontaneously brings a disentangled and controllable face representation.

Index Terms—Generative adversarial network, face editing, interpretability, explainable artificial intelligence, disentanglement.

1 Introduction

Recent years have witnessed the great success of Generative Adversarial Networks (GANs) [2] in high-fidelity face synthesis [1], [3], [4]. Based on adversarial training, GANs learn to map a random distribution to the real data observation and then produce photo-realistic images from randomly sampled latent codes.

Despite the appealing synthesis quality, it remains much less explored about what knowledge GANs learn in the latent representation and how we can reuse such knowledge to control the generation process. For example, given a latent code, how does GAN determine the attributes of the output face, *e.g.*, an elder man or a young woman? How are different attributes organized in the latent space? Can we manipulate the attributes of the synthesized face as we want? How does the attribute manipulation affect the face identity? Can we apply a well-trained GAN model for real image editing?

To answer these questions, we propose a novel framework, termed as *InterFaceGAN*, to *Inter*pret the *Face* representation learned by *GAN*s. We employ some off-the-shelf classifiers to predict semantic scores for synthesized images. In this way, we can bridge the latent space and the semantic space and further utilize such a connection for representation analysis. In particular, we analyze how an individual semantic is encoded in the latent space both theoretically and empirically. It turns out that a true-or-false attribute aligns with a linear subspace of the latent space. Based on this discovery, we study the entanglement between different semantics emerging in the latent representation and manage to disentangle them via subspace projection.

After identifying the latent semantics, InterFaceGAN proposes a simple yet effective pipeline for face editing. By linearly varying the latent code, we can manipulate the gender, age, expression, presence of eyeglasses, and even face pose of the synthesized image, as shown in Figure 1 (a). Moreover, thanks to our disentanglement analysis, we propose conditional manipulation to alter one attribute without affecting others, as shown in Figure 1 (b). More importantly, InterFaceGAN controls the face semantics by reusing the GAN knowledge *without* retraining the model.

To better interpret the representation learned by GANs, we conduct a thorough analysis of the editing results made by InterFaceGAN. First, we compare the semantic scores before and after the manipulation to quantitatively evaluate the identified semantics. Then, we make a layer-wise analysis on StyleGAN, whose generator is with per-layer stochasticity [3], to explore how semantics originate from the latent representation layer by layer. Finally, considering the importance of the identity information for faces, we make in-depth identity analysis to see how identity is preserved in the manipulation process as well as how identity is sensitive to different facial attributes.

Applying a GAN model to real image editing is another important issue since GANs commonly lack the inference ability. In this work, we apply two approaches to extend InterFaceGAN for real face manipulation. One works with GAN inversion, which can invert a target image back to a latent code. The other uses InterFaceGAN to build a dataset that contains the pairs of synthetic images before and after manipulation and then train a pixel-to-pixel model [5] on this synthetic dataset. We compare these approaches and evaluate their strengths and weaknesses.

• *Y. Shen, C. Yang, X. Tang, and B. Zhou are with the Department of Information Engineering, the Chinese University of Hong Kong, Hong Kong SAR, China.*
E-mail: {sy116, yc019, xtang, bzhou}@ie.cuhk.edu.hk

Code and models are available at https://genforce.github.io/interfacegan/.

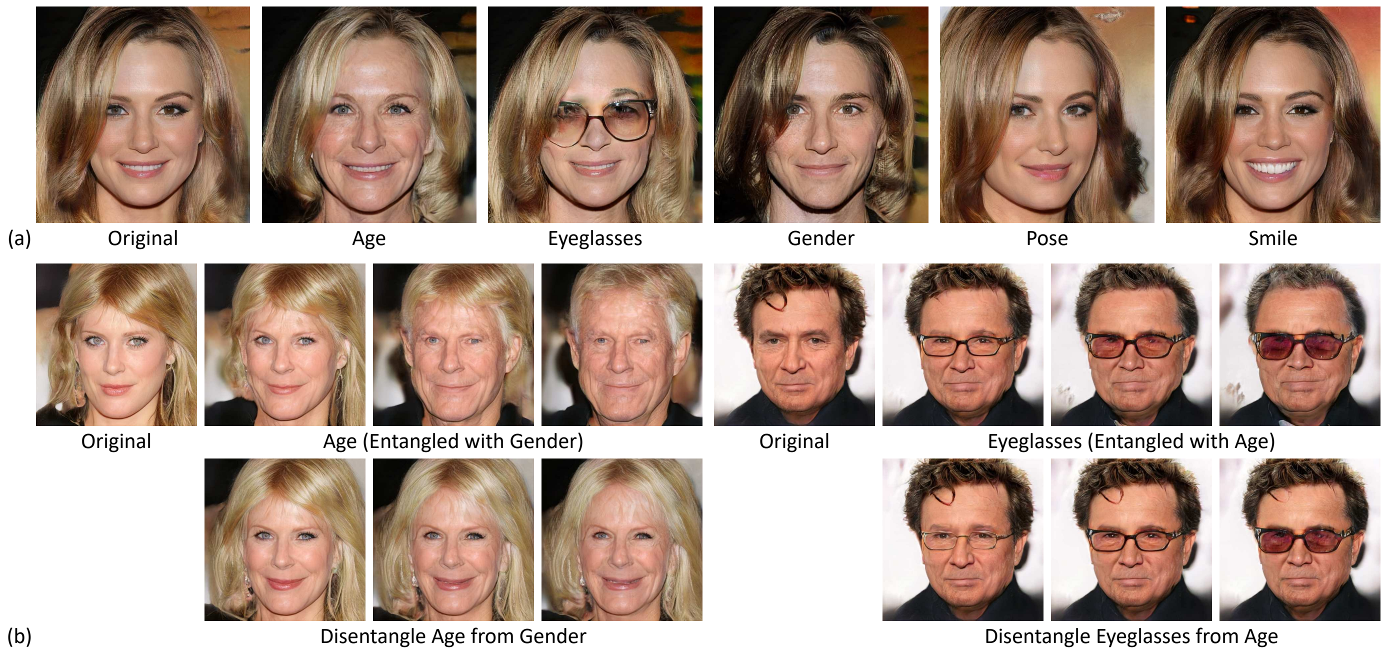
(a)
Original
Age
Eyeglasses
Gender
Pose
Smile
Original
Age (Entangled with Gender)
Original
Eyeglasses (Entangled with Age)
(b)
Disentangle Age from Gender
Disentangle Eyeglasses from Age

Fig. 1. (a) **Manipulating various facial attributes** through varying the latent codes of a well-trained GAN model. (b) **Conditional manipulation** results using InterFaceGAN, where we can better disentangle the correlated attributes (top row) and achieve more precise control of the facial attributes (bottom row). All results are synthesized by PGGAN [1].

The preliminary result of this work is published at [6]. Compared to the conference paper, we include the following new contents: (i) a detailed analysis of the face representation learned by StyleGAN [3] and its comparison to PGGAN [1]; (ii) a comparison between the entanglement of identified latent semantics and the attribute distribution of training data, which sheds light on how GANs learn to encode various semantics in the training process; (iii) quantitative evaluation of the editing results achieved by InterFaceGAN; (iv) layer-wise analysis of the per-layer representation learned by StyleGAN [3]; (v) identity analysis of the manipulated images; (vi) a new method for real face editing, which is to train feed-forward models on the synthetic data collected by InterFaceGAN.

2 Related Work

Generative Adversarial Networks. Due to the great potential of GAN [2] in producing photo-realistic images, it has been widely applied to image editing [7], [8], super-resolution [9], [10], image inpainting [11], [12], video synthesis [13], [14], *etc*. Many attempts have been made to improve the synthesis quality and training stability of GANs [1], [3], [4], [15], [16], [17], [18], [19], [20]. Despite this tremendous success, little work has been done on understanding how GANs learn to connect the latent representation with the semantics in the real visual world.

Study on Latent Space of GANs. Latent space of GANs is generally treated as Riemannian manifold [21], [22], [23]. Prior work focused on exploring how to make the output image vary smoothly from one synthesis to another through interpolation in the latent space [24], [25]. Bojanowski *et al.* [26] optimized the generator and latent code simultaneously to learn a better representation. However, the studies on how a well-trained GAN can encode different semantics in the latent space and reuse this semantic knowledge to control the generation process are still missing. Bau *et al.* [27] found that some units from intermediate layers of the GAN generator are specialized to synthesize certain visual concepts, such as sofa and TV for living room synthesis. Some work [28], [29] observed the vector arithmetic property in the latent space. This work provides a detailed analysis of how semantics are encoded in the face representation learned by GANs from the perspective of a single semantic and the disentanglement of multiple semantics. Some concurrent work also explores the latent semantics in GANs: Goetschalckx *et al.* [30] improves the memorability of the output image. Jahanian *et al.* [31] studies the steerability of GANs concerning camera motion and image color tone. Yang *et al.* [32] observes the semantic hierarchy emerging in the scene synthesis models. Differently, we focus on interpreting the face representation by theoretically and empirically studying how various semantics originate from and are organized in the latent space. We further extend our method to real image editing.

Semantic Face Editing with GANs. Semantic face editing aims at manipulating the facial attributes of a given image. Compared to unconditional GANs, which can generate images arbitrarily, semantic editing expects the model to change the target attribute yet maintain other information of the input face. To achieve this goal, existing methods require carefully designed loss functions [33], [34], [35], introduction of additional attribute labels [7], [36], [37], [38], [39], or special architectures [40], [41] to train new models. Unlike previous learning-based methods, this work explores the interpretable semantics inside the latent space of *fixed* GAN models. By reusing the semantic knowledge spontaneously learned by GANs, we can unleash its manipulation capability and *turn unconstrained GANs to controllable GANs* through varying the latent code.

GAN Inversion. The generator in GANs typically takes a latent code as the input and outputs a synthesized image. Hence, it leaves no space for the inference on real images. To solve this problem, a common practice is to get the reverse mapping from the image space to the latent space, which is also known as GAN Inversion [8], [42], [43], [44], [45]. Prior work either performed instance-level optimization [46], [47], [48] or explicitly learned an encoder to reverse the generator [49], [50], [51]. Some methods combined these two ideas by using the encoder to produce a good starting point for optimization [52], [53], [54]. There are also some works changing the optimization objects by using multiple codes to reconstruct a single image [55] or optimizing the model weight together with the latent code [56]. Being orthogonal to those approaches, our work interprets the latent representation and utilizes GAN inversion as a tool to reuse GAN knowledge for real image editing.

Image-to-Image Translation. Image-to-Image translation learns a deterministic model to transfer images from one domain to another. It is widely used for real image editing considering its fast inference speed [5], [57], [58], [59], [60], [61]. Some attempts increased the diversity of the translated images by introducing stochasticity [62], [63] or translating images among multiple domains [64], [65]. However, all these models rely on paired data or domain labels, which are not easy to obtain. In this work, we manage to leverage the semantics learned by GANs to create unlimited synthetic data pairs. We can then apply the knowledge encoded in the latent representation to feed-forward real image editing by training image-to-image translation networks with such synthetic data.

3 Framework of InterFaceGAN

In this section, we first provide a rigorous analysis of the properties of the semantics emerging in the latent representation learned by GANs and then construct a pipeline of utilizing the identified semantics for face editing.

3.1 Semantics in Latent Space

Given a well-trained GAN model, the generator can be viewed as a deterministic function $g : \mathcal{Z} \rightarrow \mathcal{X}$. Here, $\mathcal{Z} \subseteq \mathbb{R}^d$ denotes the d-dimensional latent space, for which Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ is commonly used [1], [3], [18], [20]. $\mathcal{X}$ stands for the image space, where each sample $\mathbf{x}$ possesses certain semantic information, like gender and age for face model. Suppose we have a semantic scoring function $f_S : \mathcal{X} \rightarrow \mathcal{S}$, where $\mathcal{S} \subseteq \mathbb{R}^m$ represents the semantic space with m semantics. We can bridge the latent space $\mathcal{Z}$ and the semantic space $\mathcal{S}$ with $\mathbf{s} = f_S(g(\mathbf{z}))$, where $\mathbf{s}$ and $\mathbf{z}$ denote semantic scores and the sampled latent code respectively.

Single Semantic. It has been widely observed that when linearly interpolating two latent codes, $\mathbf{z}_1$ and $\mathbf{z}_2$, the appearance of the synthesis changes continuously [3], [20], [28]. It implicitly means that the semantics contained in the image also change gradually. According to ***Property 1***, the interpolation from $\mathbf{z}_1$ to $\mathbf{z}_2$ defines a direction in $\mathcal{Z}$, which further defines a hyperplane. We therefore make an assumption[1] that for any binary semantic (*e.g.*, male *v.s.* female), there exists a hyperplane in the latent space serving as the separation boundary. Semantic remains the same when the latent code walks within one side of the hyperplane yet turns into the opposite when across the boundary.

Given a hyperplane with unit normal vector $\mathbf{n} \in \mathbb{R}^d$, we define the "distance" from a sample $\mathbf{z}$ to this hyperplane as

$$\mathrm{d}(\mathbf{n}, \mathbf{z}) = \mathbf{n}^T \mathbf{z}. \tag{1}$$

Here, $\mathrm{d}(\cdot, \cdot)$ is not a strictly defined distance since it can be negative. When $\mathbf{z}$ lies near the boundary and is moved toward and across the hyperplane, both the "distance" and the semantic score vary accordingly. Moreover, it is just when the "distance" changes its numerical sign that the semantic attribute reverses. We therefore expect these two items to be linearly dependent with

$$f(g(\mathbf{z})) = \lambda \mathrm{d}(\mathbf{n}, \mathbf{z}), \tag{2}$$

where $f(\cdot)$ is the scoring function for a particular semantic, and $\lambda > 0$ is a scalar to measure how fast the semantic varies along with the change of "distance". According to ***Property 2***, random samples drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ are very likely to locate close enough to a given hyperplane. Therefore, the corresponding semantic can be modeled by the linear subspace that is defined by $\mathbf{n}$.

Property 1 *Given $\mathbf{n} \in \mathbb{R}^d$ with $\mathbf{n} \neq \mathbf{0}$, the set $\{\mathbf{z} \in \mathbb{R}^d : \mathbf{n}^T \mathbf{z} = 0\}$ defines a hyperplane in $\mathbb{R}^d$, and $\mathbf{n}$ is called the normal vector. All vectors $\mathbf{z} \in \mathbb{R}^d$ satisfying $\mathbf{n}^T \mathbf{z} > 0$ locate from the same side of the hyperplane.*

Property 2 *Given $\mathbf{n} \in \mathbb{R}^d$ with $\mathbf{n}^T \mathbf{n} = 1$, which defines a hyperplane, and a multivariate random variable $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, we have $\mathrm{P}(|\mathbf{n}^T \mathbf{z}| \leq 2\alpha\sqrt{\frac{d}{d-2}}) \geq (1 - 3e^{-cd})(1 - \frac{2}{\alpha}e^{-\alpha^2/2})$ for any $\alpha \geq 1$ and $d \geq 4$. Here, $\mathrm{P}(\cdot)$ stands for probability and c is a fixed positive constant.*[2]

Multiple Semantics. When the case comes to m different semantics, we have

$$\mathbf{s} \equiv f_S(g(\mathbf{z})) = \Lambda \mathbf{N}^T \mathbf{z}, \tag{3}$$

where $\mathbf{s} = [s_1, \ldots, s_m]^T$ denotes the semantic scores, $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_m)$ is a diagonal matrix containing the linear coefficients, and $\mathbf{N} = [\mathbf{n}_1, \ldots, \mathbf{n}_m]$ indicates the separation boundaries. Aware of the distribution of random sample $\mathbf{z}$, which is $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$, we can compute the mean and covariance matrix of the semantic scores $\mathbf{s}$ as

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{s}} &= \mathbb{E}(\Lambda \mathbf{N}^T \mathbf{z}) = \Lambda \mathbf{N}^T \mathbb{E}(\mathbf{z}) = \mathbf{0}, \qquad (4) \\ \boldsymbol{\Sigma}_{\mathbf{s}} &= \mathbb{E}(\Lambda \mathbf{N}^T \mathbf{z}\mathbf{z}^T \mathbf{N} \Lambda^T) = \Lambda \mathbf{N}^T \mathbb{E}(\mathbf{z}\mathbf{z}^T) \mathbf{N} \Lambda^T \\ &= \Lambda \mathbf{N}^T \mathbf{N} \Lambda. \qquad (5) \end{aligned}$$

We therefore have $\mathbf{s} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{s}})$, which is a multivariate normal distribution. Different entries of $\mathbf{s}$ are disentangled if and only if $\boldsymbol{\Sigma}_{\mathbf{s}}$ is a diagonal matrix, which requires $\{\mathbf{n}_1, \ldots, \mathbf{n}_m\}$ to be orthogonal with each other. If this condition does not hold, some semantics will entangle with each other. $\mathbf{n}_i^T \mathbf{n}_j$ can be used to measure the entanglement between the i-th and j-th semantics to some extent.

1. This assumption is empirically demonstrated in Section 4.

2. When $d = 512$, we have $P(|\mathbf{n}^T \mathbf{z}| > 5.0) < 1e^{-6}$. It suggests that almost all sampled latent codes are expected to locate within 5 unit-length to the boundary. Proof can be found in **Appendix**.

3.2 Manipulation in Latent Space

In this part, we introduce how to use the semantics found in the latent space for image editing.

Single Attribute Manipulation. According to Eq. (2), to manipulate the attribute of a synthesized image, we can easily edit the original latent code $\mathbf{z}$ with $\mathbf{z}_{edit} = \mathbf{z} + \alpha\mathbf{n}$. It will make the synthesis look more positive on such semantic with $\alpha > 0$ since the score becomes $f(g(\mathbf{z}_{edit})) = f(g(\mathbf{z})) + \lambda\alpha$ after editing. Similarly, $\alpha < 0$ will make the synthesis look more negative.

Conditional Manipulation. When there is more than one attribute, editing one may affect another since some semantics can be entangled. To achieve more precise control, we propose *conditional manipulation* by manually forcing $\mathbf{N}^T\mathbf{N}$ in Eq. (5) to be diagonal. In particular, we use projection to make different vectors orthogonal. As shown in Figure 2, given two hyperplanes with normal vectors $\mathbf{n}_1$ and $\mathbf{n}_2$, we find a projected direction $\mathbf{n}_1 - (\mathbf{n}_1^T\mathbf{n}_2)\mathbf{n}_2$ such that moving samples along this new direction can change "attribute 1" without affecting "attribute 2". If there are multiple attributes to be conditioned on, we subtract the projection from the primal direction onto the plane constructed by all conditioned directions. Note that the proposed conditional manipulation requires each semantic subspace to be independent of others. In other words, if two semantics share the same subspace, it is hard to isolate one from the other via subspace projection, but it rarely happens in a high-dimensional space.

Real Image Manipulation. InterFaceGAN enables semantic editing from the latent space of a *fixed* GAN model. To manipulate real images, we should infer the best latent code that can reconstruct the target image, *i.e.*, GAN inversion. For this purpose, both optimization-based [54] and learning-based [51] approaches can be used. We thoroughly evaluate their strengths and weaknesses in Section 6. We also use InterFaceGAN to prepare synthetic data pairs and then train image-to-image translation models [5] to achieve real face editing, with the results shown in Section 6.

3.3 Implementation Details

We choose five key facial attributes for analysis, including pose, smile (expression), age, gender, and eyeglasses. The corresponding positive directions are defined as turning right, laughing, getting old, changing to male, and wearing eyeglasses. Note that we can always easily plug in more attributes as long as the attribute classifier is available.

To better predict these attributes from synthesized images, we train an auxiliary attribute prediction model using the annotations from the CelebA dataset [66] with ResNet-50 network [67]. This model is trained with multi-task losses to simultaneously predict smile, age, gender, eyeglasses, as well as the 5-point facial landmarks (*i.e.*, left eye, right eye, nose, left corner of mouth, right corner of mouth). Here, the facial landmarks are used to compute yaw pose, which is also treated as a binary attribute (*i.e.*, left *v.s.* right). Besides the landmarks, all other attributes are learned as a bi-classification problem with softmax cross-entropy loss, while landmarks are optimized with l_2 regression loss. As images produced by PGGAN and StyleGAN are with 1024×1024 resolution, we resize them to 224×224 before feeding them to the attribute model.

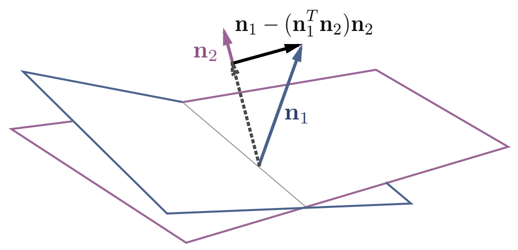

Fig. 2. Illustration of the **conditional manipulation via subspace projection**. The projection of $\mathbf{n}_1$ onto $\mathbf{n}_2$ is subtracted from $\mathbf{n}_1$, resulting in a new direction $\mathbf{n}_1 - (\mathbf{n}_1^T\mathbf{n}_2)\mathbf{n}_2$.

Given the pre-trained GAN model, we synthesize $500K$ images by randomly sampling from the latent space. There are two main reasons for preparing such a large-scale dataset: (i) to eliminate the randomness caused by sampling and make sure the distribution of the sampled code is as expected, and (ii) to get enough wearing-glasses samples, which are rare in CelebA-HQ dataset [1].

To find the semantic boundaries in the latent space, we use the pre-trained attribute prediction model to assign attribute scores for all $500K$ synthesized images. We sort the corresponding scores for each attribute and choose $10K$ samples with the highest scores and $10K$ with the lowest ones as candidates. The reason in doing so is that the prediction model is not perfect and may produce wrong predictions for ambiguous samples, *e.g.*, middle-aged person for age attribute. We then randomly choose 70% samples from the candidates as the training set to learn a linear SVM, resulting in a decision boundary. Recall that normal directions of all boundaries are normalized to unit vectors. The remaining 30% samples are used for verifying how the linear classifier behaves. Here, for SVM training, the inputs are the $512d$ latent codes and the predicted attribute scores are treated as binary labels. Here, note that learning a "bias" term in the SVM classifier or not barely affects the learned direction. That is because we only choose the samples with the highest confidence level for training and they can be easily separated. We have tried to add the "bias" term and the learned bias turns out to be small.

4 Interpreting Face Representation

In this section, we apply InterFaceGAN to interpret the face representation learned by state-of-the-art GAN models, *i.e.*, PGGAN [1] and StyleGAN [3], both of which can produce high-quality faces with 1024×1024 resolution. PGGAN is a representative of the traditional generator where the latent code is only fed into the very first convolutional layer. By contrast, StyleGAN proposed a style-based generator, which first maps the latent code from latent space $\mathcal{Z}$ to a disentangled latent space $\mathcal{W}$ before applying it for the generation. In addition, the disentangled latent code is fed to all convolutional layers.[3]

3. When feeding $\mathbf{w}$ to each convolutional layer, StyleGAN supports truncation to ensure the synthesis quality. We use truncation 0.7 (1 means no truncation) for the first eight layers in all experiments.

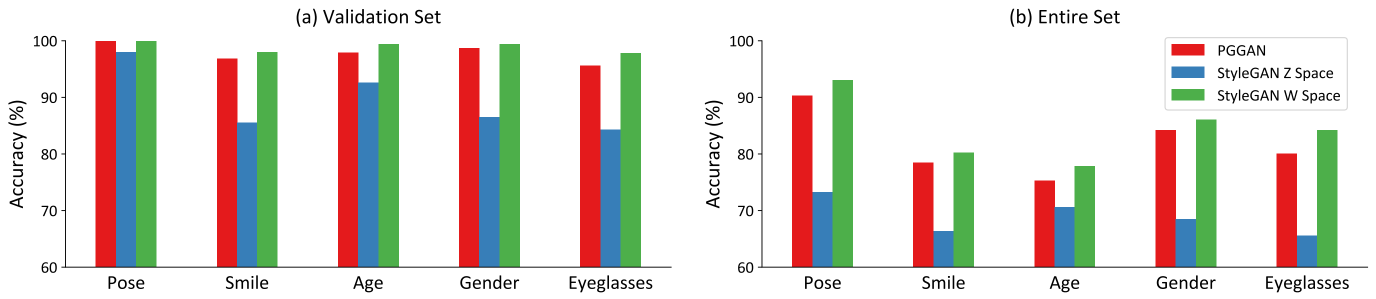

Fig. 3. **Classification accuracy** (%) on the latent separation boundaries of PGGAN [1] and StyleGAN [3] with respect to different attributes.

4.1 Separability of Latent Space

Section 3.1 introduces our assumption that for any binary attribute, there exists a hyperplane in the latent space such that all samples from the same side are with the same attribute. In this part, we first evaluate this assumption.

4.1.1 PGGAN

For each attribute, we will get a latent boundary after the training of the linear SVM classifier. We evaluate the classification performance on the validation set ($3K$ positive testing samples and $3K$ negative testing samples) and the entire set (remaining $480K$ samples besides the $20K$ candidates with high confidence level). Figure 3 shows the results. We find that all linear boundaries of PGGAN achieve over 95% accuracy on the validation set and over 75% on the entire set, suggesting that for a binary attribute, there indeed exists a linear hyperplane in the latent space that can well separate the data into two groups.

We also visualize some samples in Figure 4 by ranking them by the "distance" to the decision boundary. Those extreme cases (top and bottom rows) are very unlikely to be directly sampled, instead constructed by moving a latent code towards the normal direction "infinitely". Here, to achieve "zero" manipulation step, we randomly sample latent codes within the separation hyperplane, while to achieve "infinite" manipulation step, we directly use the normal vector $\mathbf{n}$ (or $-\mathbf{n}$) as the sampled code. From Figure 4, we can tell that the positive samples and negative samples are distinguishable to each other with respect to the corresponding attribute. This further demonstrates that the latent space is linearly separable and InterFaceGAN can find the separation hyperplane successfully.

4.1.2 StyleGAN

Compared to PGGAN, StyleGAN employs two latent spaces, which are the native latent space $\mathcal{Z}$ and the mapped latent space $\mathcal{W}$. We analyze the separability of both spaces, and the results are shown in Figure 3. Note that $\mathcal{W}$ space is not subject to normal distribution like $\mathcal{Z}$ space, but we can conduct a similar analysis on $\mathcal{W}$ space, demonstrating the generalization ability of InterFaceGAN.

Figure 3 shows three observations: (i) Boundaries from both $\mathcal{Z}$ space and $\mathcal{W}$ space separate the validation set well. Even though the performances of $\mathcal{Z}$ boundaries on the entire set are not satisfying, it may be caused by the inaccurate attribute prediction on the ambiguous samples. (ii) $\mathcal{W}$ space shows much stronger separability than $\mathcal{Z}$ space. That is because $\mathbf{w} \in \mathcal{W}$, instead of $\mathbf{z} \in \mathcal{Z}$, is the code finally fed into the generator. Accordingly, the generator tends to learn various semantics based on $\mathcal{W}$ space. (iii) The accuracy of StyleGAN $\mathcal{W}$ space is higher than the PGGAN $\mathcal{Z}$ space. The reason is that the semantic attributes may not be normally distributed in real data. Compared to $\mathcal{Z}$ space, which is subject to the normal distribution, $\mathcal{W}$ space has no constraints and hence is able to better fit the underlying real distribution. This is consistent with the design of $\mathcal{W}$ space in StyleGAN [3].

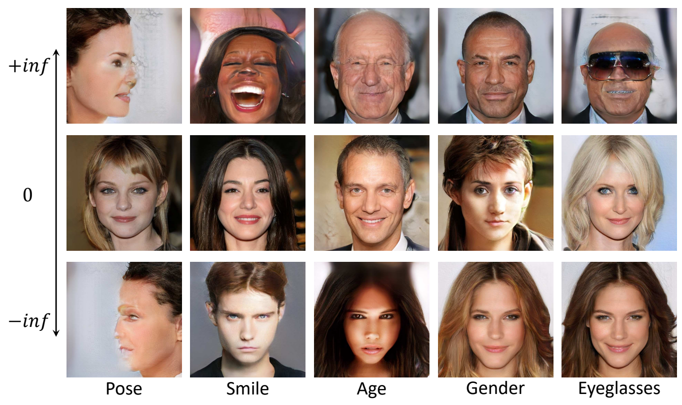

Fig. 4. Synthesized samples by PGGAN [1] with the distance near to (middle row) and extremely far away from (top and bottom rows) the separation boundary. Each column corresponds to a particular attribute.

4.2 Semantics in Latent Space for Face Manipulation

In this part, we verify whether the semantics found by InterFaceGAN are manipulable.

4.2.1 PGGAN

Manipulating Single Attribute. Figure 5 plots the manipulation results on five different attributes. It suggests that our manipulation approach performs well on all attributes in both positive and negative directions. Particularly on "pose" attribute, we observe that even the boundary is searched by solving a bi-classification problem, moving the latent code can produce continuous change. Furthermore, although there lacks enough data with extreme poses in the training set, GAN can hallucinate how profile faces should look. The same situation also appears on the eyeglasses attribute. We can manually create many faces wearing eyeglasses despite the inadequate data in the training set. These two observations provide strong evidence that GAN does not produce images randomly, but learns some interpretable semantics in the latent space.

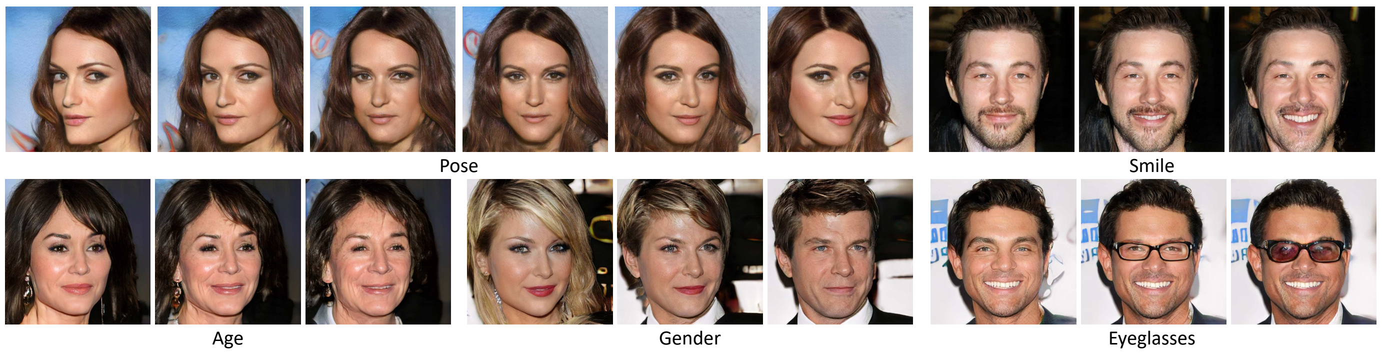

Fig. 5. **Single attribute manipulation** results with PGGAN [1]. The top left shows the same person under gradually changed poses. The remaining samples correspond to the results of manipulating four different attributes. The central one is the original synthesis for each triplet, while the left and right stand for the results by moving the latent code towards the negative and positive directions respectively.

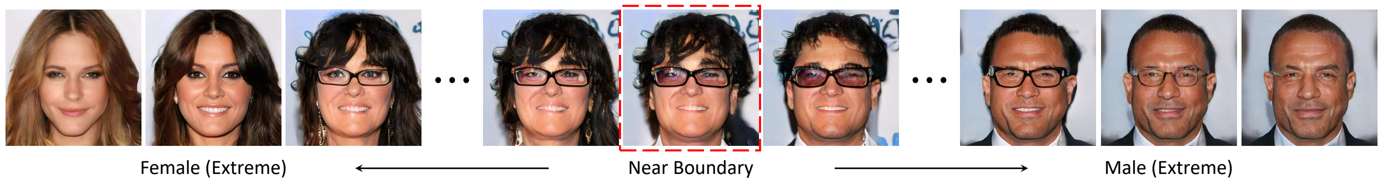

Fig. 6. Illustration of the **distance effect** by taking gender manipulation with PGGAN [1] as an example. The image in the red dashed box stands for the original synthesis. Our approach performs well when the latent code locates close to the boundary. However, when the distance keeps increasing, the synthesized images are no longer like the same person.

Distance Effect of Semantic Subspace. When manipulating the latent code, we observe a noticeable distance effect: the samples will suffer from severe changes in appearance if being moved too far from the boundary, and finally tend to become the extreme cases shown in Figure 4. Figure 6 illustrates this phenomenon by taking gender editing as an instance. Near-boundary manipulation works well. When samples go beyond a certain region, however, the editing results are no longer like the original face anymore. However, this effect does not affect our understanding of the disentangled semantics in the latent space. That is because such extreme samples are unlikely to be directly drawn from a standard normal distribution, which is pointed out in ***Property* 2** in Section 3.1. Instead, they are constructed manually by keeping moving a normally sampled latent code along a certain direction.

Fixing Artifacts. We further apply our approach to fix the artifacts that sometimes occur in the synthesis. We manually labelled $4K$ deficient synthesis and then trained a linear SVM to find the separation hyperplane. We surprisingly find that GAN also encodes such quality information in the latent space. Based on this discovery, we manage to correct some mistakes GAN has made in the generation process by moving the latent code towards the positive "quality" direction, as shown in Figure 7.

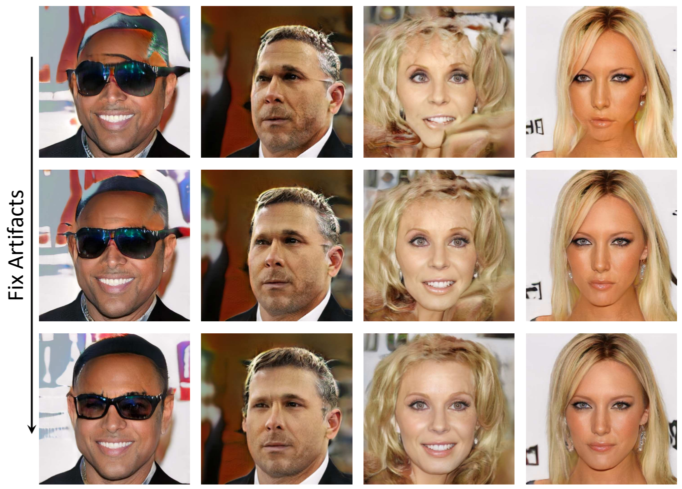

Fig. 7. Examples on **fixing the artifacts** that PGGAN [1] made. The first row shows several bad synthesis results, while the following two rows present the gradually corrected synthesis by moving the latent codes towards the positive "quality" direction.

4.2.2 StyleGAN

We further apply InterFaceGAN to StyleGAN by manipulating the latent codes in both $\mathcal{Z}$ space and $\mathcal{W}$ space. Figure 8 shows the following observations: (i) InterFaceGAN works well on the style-based generator. We manage to edit the attributes by altering the latent code in either $\mathcal{Z}$ space or $\mathcal{W}$ space. (ii) By learning from a more diverse dataset, FF-HQ [3], StyleGAN learns various semantics more thoroughly. For example, StyleGAN can produce high-quality profile faces (first example) and generate children when making people younger (second example). This is beyond the ability of PGGAN, which is trained on CelebA-HQ dataset [1]. (iii) Some attributes correlate with each other. For example, people are wearing eyeglasses when turning old (second example). A more detailed analysis of this phenomenon will be discussed in Section 4.3. (iv) $\mathcal{W}$ space shows better performance than $\mathcal{Z}$ space, especially for long-distance manipulation. In other words, when the latent code locates near the separation boundary, manipulations in $\mathcal{Z}$ space and $\mathcal{W}$ space have a similar effect. However, when the latent code goes further from the boundary, manipulating one attribute in $\mathcal{Z}$ space might affect another. Taking pose editing (first example) as an instance, the skin color of the target person varies when moving the latent code along the pose direction. By contrast, $\mathcal{W}$ space shows stronger robustness.

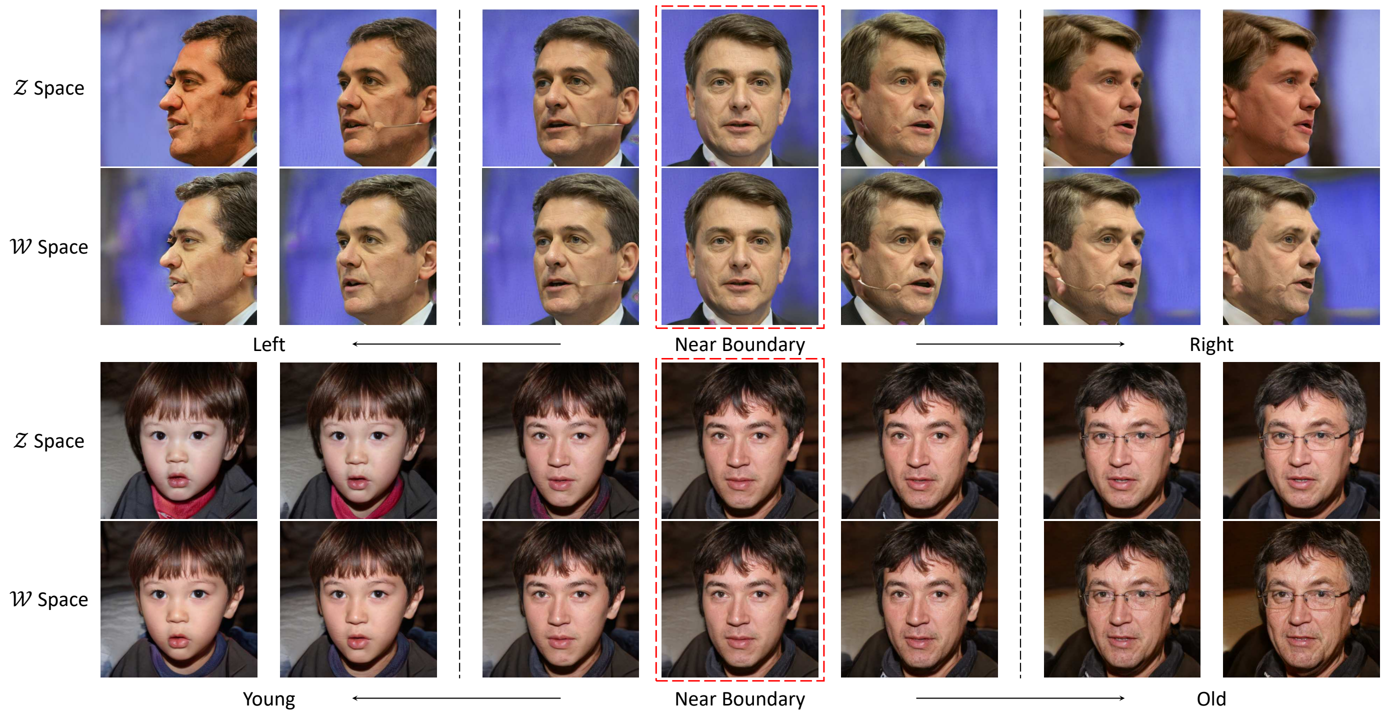

Fig. 8. **Attribute editing** results on StyleGAN [3]. For two attributes pose and age, the top row shows the manipulation results with respect to $\mathcal{Z}$ space, whilst the bottom row corresponds to $\mathcal{W}$ space. Images in red dashed boxes represent the original synthesis. Images between two black dashed lines stand for near-boundary manipulation, and the other images stand for long-distance manipulation.

4.3 Disentanglement Analysis and Conditional Manipulation

This section discusses the disentanglement between different semantics encoded in the latent representation and evaluates the conditional manipulation approach.

4.3.1 Disentanglement Measurement

Besides how one semantic can be well encoded in the latent space, we also study the correlation between multiple semantics and examine the way to decouple correlated attributes. In particular, we use the following metrics for analysis.

1) *Attribute correlation of real data*. We use the prepared predictors to predict the attribute scores of real data on which the GAN model is trained. Then we compute the correlation coefficient ρ between any two attributes with $\rho_{A_1,A_2} = \frac{Cov(A_1,A_2)}{\sigma_{A_1}\sigma_{A_2}}$. Here, A_1 and A_2 represent two random variables with respect to these two attributes. $Cov(\cdot,\cdot)$ and σ denote covariance and standard deviation respectively.
2) *Attribute correlation of synthesized data*. We also compute the attribute correlation score on the $500K$ synthesized data. By comparing this score to that of the real data, we can get clues on how GANs learn to encode such semantic knowledge in the latent representation.
3) *Semantic boundary correlation*. Given any two semantics, with the latent boundaries $\mathbf{n}_1$ and $\mathbf{n}_2$, we compute the cosine similarity between these two directions with $\cos(\mathbf{n}_1, \mathbf{n}_2) = \mathbf{n}_1^T\mathbf{n}_2$. Here, $\mathbf{n}_1$ and $\mathbf{n}_2$ are both unit vectors. This metric is used to evaluate how the above attribute correlation is reflected in our InterFaceGAN framework.

4.3.2 PGGAN

Disentanglement Analysis. Table 1 reports the correlation metrics of PGGAN model trained on CelebA-HQ dataset [1]. By comparing the attribution correlation of real data (Table 1 (a)) and that of synthesized data (Table 1 (b)), we can tell that they are very close to each other. For example, "pose" and "smile" are almost independent to other attributes, while "gender", "age", and "eyeglasses" are highly correlated with each other from both real data and synthesized data. For example, elder men are more likely to wear eyeglasses. This implies that GANs learn the underlying semantic distribution from real observation when trained to synthesize images. Then, by comparing such attribution correlation with the boundary correlation (Table 1 (c)), we also find that they behave similarly. This suggests that InterFaceGAN is able to not only accurately identify the semantics encoded in that latent representation but also capture the entanglement among them.

Conditional Manipulation. As shown in Table 1 (c), if two boundaries are not orthogonal to each other, modulating the latent code along one direction will affect the other. Hence, we propose conditional manipulation via subspace projection to reduce this entanglement as much as possible. Details are described in Section 3.2. Figure 9 shows the discrepancies between unconditional manipulation and conditional manipulation. Taking the top-left example in Figure 9 as an instance, the results tend to become male when being edited to get old (top row). We address this issue by subtracting its projection onto the gender direction from the age direction, resulting in a new direction. By moving latent codes along this projected direction, we can make sure the gender component is barely affected in the editing process (bottom row). Figure 10 shows a more