Exploring the latent space of StyleGAN-type models

LE TRAN Ngoc Tran *

Université de Paris
Instructor: Alasdair Newson

April 13, 2021

In this report, we explore the interesting characteristic of the latent space of Style Generative Adversial Network (StyleGAN) in [1] which is linear and disengtangled. First, we present the overview of GANs and the architecture of StyleGANs. Next, the section 3 is a key of report. We do several experiments to illustrate the linearity and disentanglement of latent spaces. Specifically, the interpolation between two vectors is given to support for the linearity; also we see how images change when the latent vectors are navigated. From that, we have a conclusion about the disentanglement of latent spaces. Lastly, the discussion and link to github (more detail of experiments) are given.

1 Overview of GAN

GANs are generative models that try to learn the model to generate the input distribution as realistic as possible. GANs end goal is to predict features given a label, instead of predicting a label given features.

Example: if we take cat images being x , then the GAN's goal is to learn a model that can produce the realistic or believable cat images from the training data x.

A general adversarial network(GAN) consists of 2 neural networks.

- A neural network called "**Generator**" which generates new data points from some random uniform distribution. The goal is to produce the similar type of fake results from inputs.
- While another neural network called "**Discriminator**" which identifies the fake data produced by **Generator** from the real data.

In class, we implemented a simple GAN with LMP architecture. The figure 2 shows us the result when sampling a vector in latent space.

*tran.le-tran-ngoc@etu.u-paris.fr

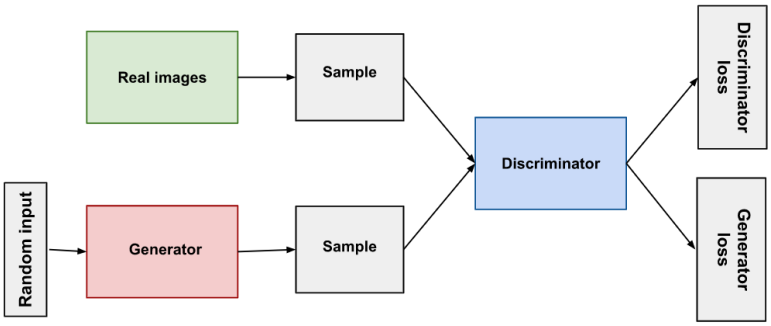
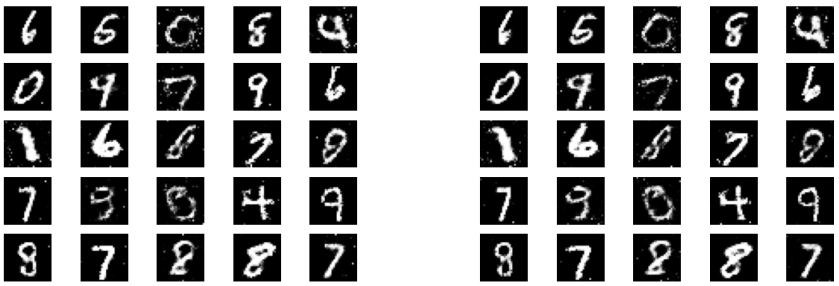


Figure 1: Generative adversial network (GAN) model.



(a) 30.000 epoch (b) 45.000 epoch

Figure 2: A GAN was trained on MNIST dataset.

In a nutshell, GANs learn to generate entirely new images that mimic the appearance of real photos. However, they offer very limited control over the generated images. The authors in [2] proposed a new generator that automatically learns to separate different aspects of the images without any human supervision. It is called StyleGAN.

2 StyleGAN - StyleGAN2 architecture

2.1 StyleGAN

The Style Generative Adversarial Network, or in a common GAN generator, the sampled input latent space vector z is projected and reshaped, so it will be further processed by transpose convolutional or upsample with or without convolutions. Here, the latent vector is transformed by a series of fully connected layers, the so-called mapping network f. The major reason for this choice is that the intermediate latent space $\mathcal{W}$ does not have to support sampling according to any fixed distribution. With continuous mapping, its sampling density is induced. This mapping f "unwrap" the space of $\mathcal{W}$, so it implicitly enforces a sort of disentangled representation. This means that the factors of variation become more linear. The authors

argue that it will be easier to generate realistic images based on a disentangled representation compared to entangled ones. With this totally unsupervised trick, we at least expect $\mathcal{W}$ to be less entangled than $\mathcal{Z}$ space.

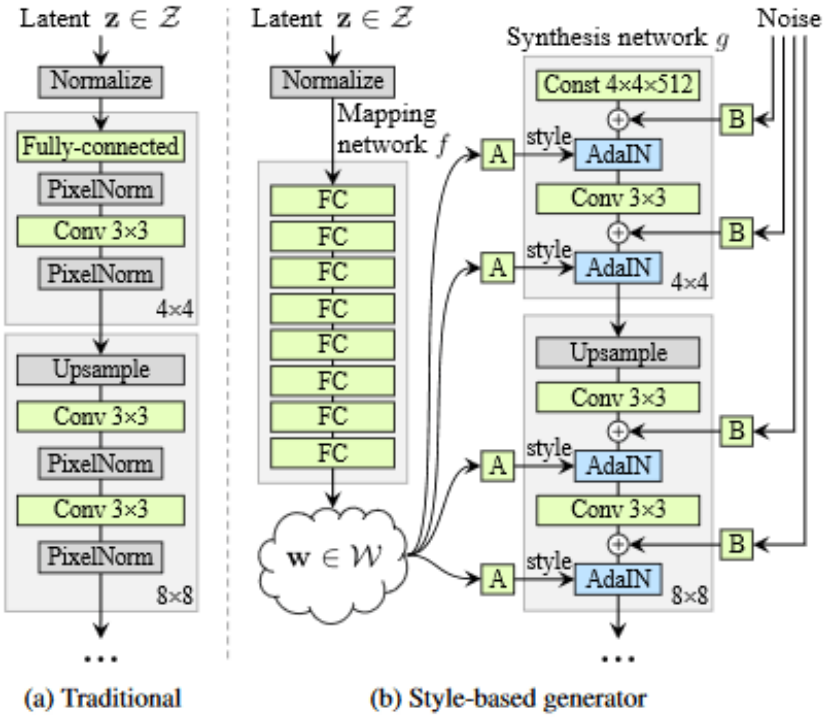


Figure 3: The traditional and style-based generator

More precisely, given a latent code $\mathbf{z}$ in the input latent space $\mathcal{Z}$, a non-linear mapping network $f : \mathcal{Z} \longrightarrow \mathcal{W}$ first produces $\mathbf{w} \in \mathcal{W}$ (Figure 3b, left). For simplicity, we set the dimensionality of both spaces to 512, and the mapping f is implemented using an 8-layer MLP. Learned affine transformations then specialize $\mathbf{w}$ to *styles* $\mathbf{y} = (\mathbf{y}_s, \mathbf{y}_b)$ that control adaptive instance normalization (AdaIN) operations after each convolution layer of the synthesis network g. The AdaIn operation is defined as

$$\text{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i} \tag{1}$$

Finally, the generator with a direct means to generate stochastic detail by introducing explicit *noise inputs* is given. These are single-channel images consisting of uncorrelated Gaussian noise, and we feed a dedicated noise image to each layer of the synthesis network.

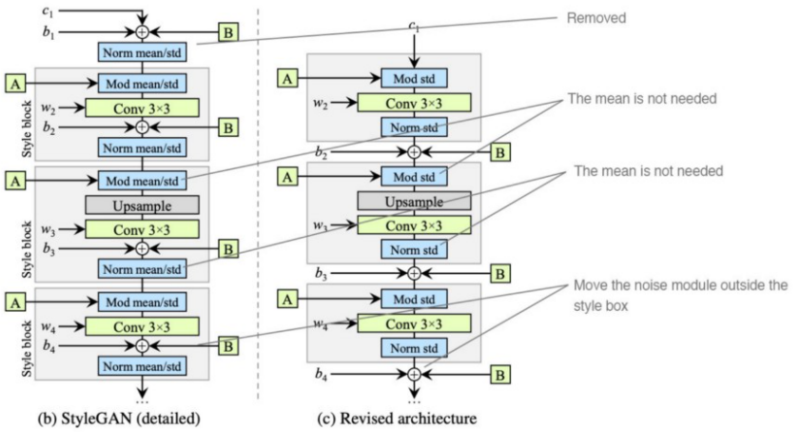
2.2 StyleGAN2

Blob-like artifacts (resembling water droplets) are observed in StyleGAN generated images which are more obvious at examining the intermediate feature maps inside the generator network. This issue seems to occur at 64×64 resolution in all feature maps and get worse in

higher resolution. In [1], StyleGAN2 proposes an alternative design in solving issues that occurred in using progressive growing which its purpose is to stabilize high-resolution training.

Figure 4: Instance normalization causes water droplet like artifacts in StyleGAN images.

The changes in StyleGan2 made inculde: remove how the constant is processed at the beginning; the mean is not needed in normalizing the features, move the noise module outside the style module.



Next, StyleGAN2 simplifies the design as below with weight demodulation. It revisits the instant normalization design (Norm std) with the purpose of replacing it with another normalization method that does not have the blob-like artifacts. The right diagram below is the new design with weight demodulation.

Weight demodulation adds the following changes:

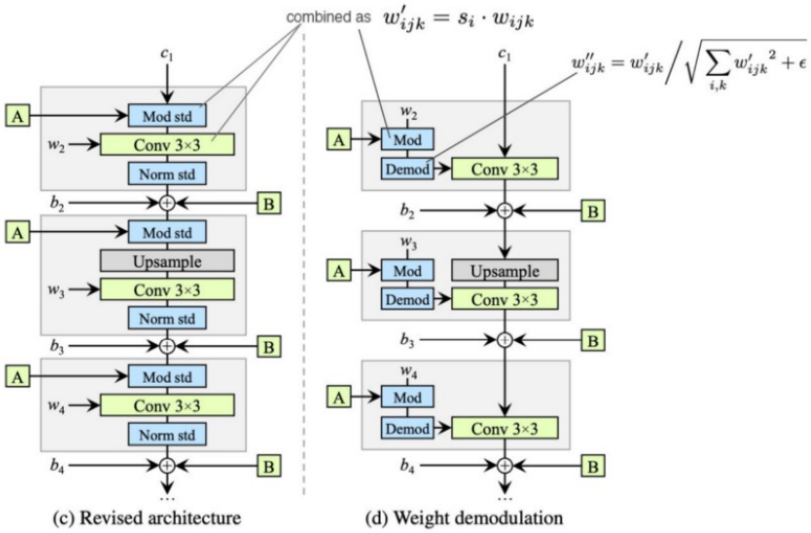
1. The modulation (mod std) followed by the convolution (Conv 3×3) can be combined as scaling the convolution weights and implement as Mod above. (This does not change the model design.)

$$w'_{ijk} = s_i \cdot w_{ijk}$$

where i is the input feature map i.

2. The new weight is normalized as

$$\sigma_j = \sqrt{\sum_{i,k} {w'_{ijk}}^2}$$



The new weight is normalized as

$$w''_{ijk} = \frac{w'_{ijk}}{\sqrt{\sum_{i,k} w'^{2}_{ijk} + \varepsilon}}$$

with a small ε added to avoid numerical instability. Even though it is not mathematically the same as the instant normalization, it normalizes the output feature map to have a standard unit standard deviation and achieve similar goals as other normalization methods (in particular, making the training more stable).

The figure 5 show us some generated images from StyleGAN2

Figure 5: Some generated images from StyleGAN2

3 Experiments

In this section, we give some experiments to illustrate the linearity and disentanglement of the latent space. We begin with the simple approach: make the interpolation between 2 latent vectors $\mathbf{z}$ or $\mathbf{w}$, we see the series generated figures from first to second vector. Next, we mention 2 ways to clarify the disengtanglement. First, we change each layer at the entrance of latent vector $\mathbf{w}$ and see how it affect the attribute of the generated images. The other method is introduced in [3], the hyperplane is constructed after training, then we are able to edit the attributes in face images realistically. We used pre-trained model of StyleGAN2 along with data FFHQ supplied by NVIDA in [2].

3.1 The linearity of the latent space

In this part, we find some interesting experiments on the linearity of both latent spaces $\mathbf{z}$ and $\mathbf{w}$. We can realize significant differences between two results on $\mathcal{Z}$ and $\mathcal{W}$ space. The interpolation is defined as

$$t \cdot \mathbf{z}_1 + (1 - t) \cdot \mathbf{z}_2$$

for $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{Z}$ and $t \in [0, 1]$, analogous with vector $\mathbf{w} \in \mathcal{W}$.

(a) The interpolation on $\mathcal{Z}$ space

(b) The interpolation on $\mathcal{W}$ space

Figure 6: Experiments on the linearity of the latent spaces

3.2 The disentanglement of the latent space

In [2], the authors mentioned “If a latent space is sufficiently disentangled, it should be possible to find direction vectors that consistently correspond to individual factors of variation". In this part, we present two ways to explore the disentanglement of the latent spaces: first, we change the direction corresponding to the attribute of faces, the other we use the framework termed as InterfaceGAN in [3].

3.2.1 Change the direction of vectors w

For this part, we consider some attributes: gender, age, glasses, smile. By changing the coefficients of each direction, we will see how the images change, when we change any attribute, the others are preserved. However, there are some attributes are entanglement, that means the attribute 2 will modify if the attribute 1 changes.

(a) Gender direction

(b) Age direction

(c) Glasses direction

(d) Smile direction

Figure 7: The results when changing the direction of vector $\mathbf{w}$

Via these experiments, we realize when we alter the coefficients of each direction, we receive the new results adapting to the tendency of coefficient variation. The latent space $\mathcal{W}$ is disentangled with some attributes and why it does not happen with all? Figure 7c show us the changing of glasses affect the variation of ages and emotions.

3.2.2 InterfaceGAN

The second approach to study the disentanglement of the latent space is the framework InterfaceGAN in [3]. We assume that for any binary semantic (for example, male versus female), there exist 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. We have the following property:

Property 1. *Given* $\mathbf{n} \in \mathbb{R}^d$ *with* $\mathbf{n} \neq 0$*, 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.*

Attribute	$\mathcal{Z}$ space	Input	$\mathcal{W}$ space
Glasses			
Gender			
Smile			

Table 1: The variation of some attributes is implemented on both $\mathcal{Z}$ and $\mathcal{W}$ space

The table 1 give us the results when we make manipulation in the latent codes in both $\mathcal{Z}$ and $\mathcal{W}$ spaces. In this part, we consider three attributes: glasses, gender, smile to see how the images change when the direction of vectors is varied. At the first row, when we vary the input images (the generated images from StyleGAN2) tend to have no glasses, the result on $\mathcal{Z}$ space also has the difference about color hair and ages; meanwhile, for the $\mathcal{W}$ space, almost the other attributes does not change. Analogously, when we vary the attribute gender or smile, in the $\mathcal{Z}$ space, we realize the glasses turn to be a sunglasses; while there is no big difference in the $\mathcal{W}$ space. We can conclude that the $\mathcal{Z}$ space is less disentangled than $\mathcal{W}$ space. The figure 8 also illustrates the $\mathcal{W}$ space is more disentangled than $\mathcal{Z}$ space.

Figure 8: The age manipulation in $\mathcal{W}$ and $\mathcal{Z}$ spaces, first and second rows, respectively.

Figure 8 also gives an example about the entanglement between "age" and "glasses". In figure 8, manipulating from $\mathcal{Z}$ space and $\mathcal{W}$ space produces similar results when the latent code still locates near the boundary (the images between two red lines). For long-distance manipulation, $\mathcal{W}$ space (first row) shows superiority over $\mathcal{Z}$ space (second row), for example, hair length and face shape do not change in the first row.

3.2.3 The correlation between layers and attributes

Unlike the traditional generator, StyleGAN feeds the latent code to all convolutional layers. This enables us to study the per-layer representation. In pratice, we divide the 18 layers into 5 groups which is 00-01, 02-03, 04-05, 06-07 and 08-17. Then we will see how each layer directly affects each attribute.

How to implement it in practice: In fact , the dimension of latent code is $(1, 512)$, then when we feed it to the synthesis network g which consists of 18 layers, the latent code now is a matrix whose dimension is $(18, 512)$. Mathematically, we call $\mathbf{w}_g$ is a latent code, we express the new vector after making modification:

$$\mathbf{w}_{g(\text{new})} = \mathbf{w}_{g(\text{old})} + d_g \odot a\vec{e}$$

where d_g is the latent direction which have same size with matrix $\mathbf{w}_g$, in this report, we used the latent direction from pretranied model (https://github.com/a312863063/generators-with-stylegan2.git); $\odot$ is element-wise product, a is the coefficients that we want to change; $\vec{e}$ is the $(18, 1)$ vector where $e_i = 1$ if we want change the input of the ith-layer, otherwise $e_i = 0$.

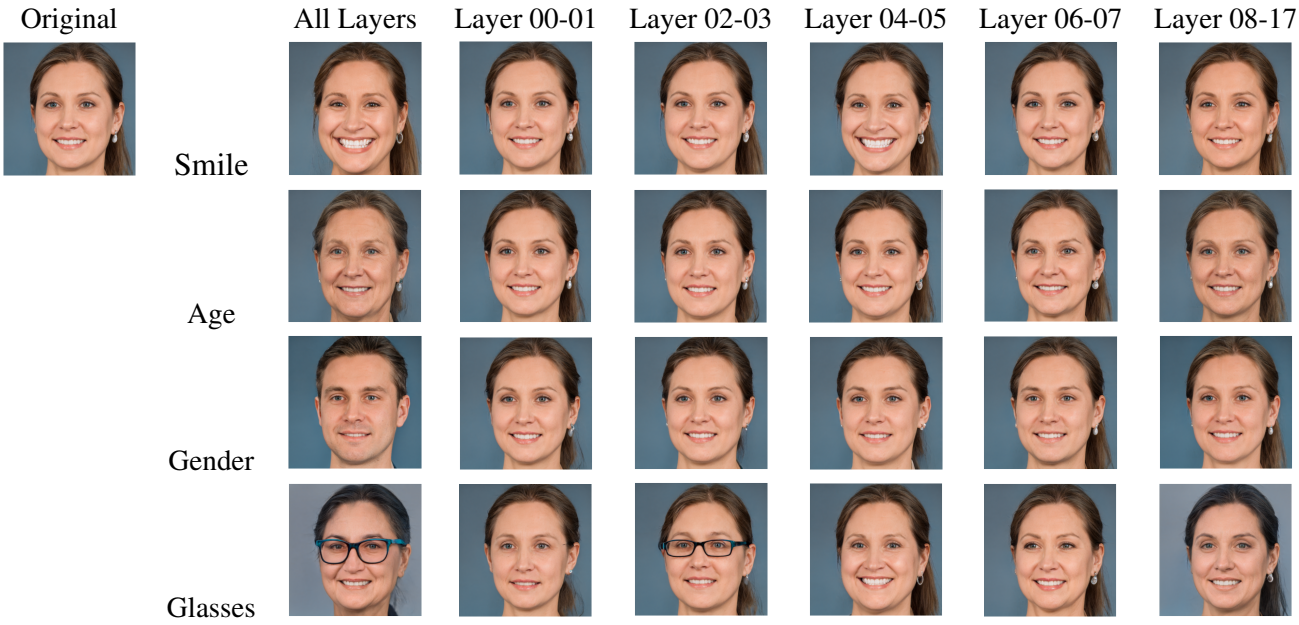


Table 2: The matrix whose the rows show the changes of attributes and the columns is about the layers are varied.

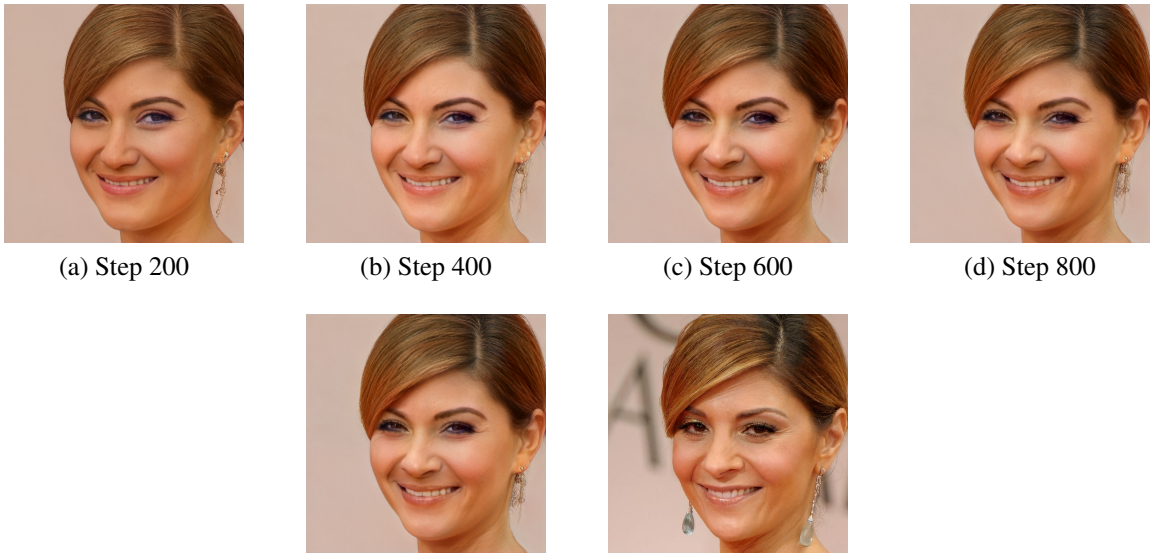
By observing table 2, we can see that "smile" is controlled at layer 02-05, "age" is controlled at layer 02-07, "gender" is controlled at layer 02-03 and "glasses" is controlled at layer 02-03. All attributes are barely affected by editing layer 08-17. That is because layer 08-17 mainly correspond to texture, such as skin color and background.

3.3 Projecting images to the latent space

StyleGAN2 comes with a projector that finds the closest generable image based on any input image. It is evidence to get a feeling for the diversity of the portrait manifold.

4 Conclusion and Discussion

Currently, GANs have played an important role in machine learning, which makes the data is more diverse. Time by time, the generative model is improved to provide better-generated images, that is the reason why PCGAN, StyleGAN was born. With the architecture is more complex, the generator of StyleGAN allows us to modify with many different "style". StyleGAN employs two latent spaces, which are the native latent space $\mathcal{Z}$ and the mapped latent space $\mathcal{W}$ whose each vector $\mathbf{w}$ is the code fed into the generator. By using the pre-trained model with FFHQ dataset, we do some experiments to illustrate the linearity and disentanglement of both $\mathcal{Z}$ and $\mathcal{W}$ space (more detail can be found at https://github.com/



(a) Step 200 (b) Step 400 (c) Step 600 (d) Step 800

(e) Step 1000 (f) Target

Figure 9: The target images and some generated images following steps by the projection

ltngoctran/StyleGAN). With two interesting properties of the latent space, we can easily edit images that training model is not necessary. Further, we can think about implementing StyleGAN encoder, which converts real images to latent space, then we can use generative model of StyleGAN to edit images.

References

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

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

[3] Y. Shen, C. Yang, X. Tang, B. Zhou, Interfacegan: Interpreting the disentangled face representation learned by gans, IEEE Transactions on Pattern Analysis and Machine Intelligence (2020).