Project 1: *Exploring the latent space of StyleGAN-type models*

Project report: *Deep Learning for Image Restoration and Synthesis (MVA 2022)*

Léo Tronchon, Antoine Cadiou
MVA Master's program
École Normale Supérieure Paris-Saclay
firstname.lastname@ens-paris-saclay.fr

1. Introduction

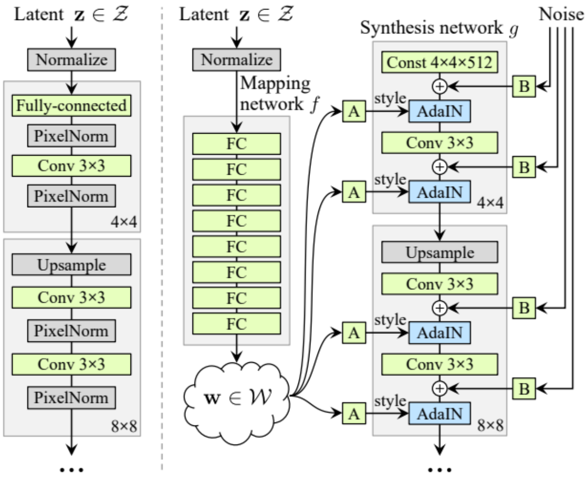
Generative models have been the focus of much research in the past years. Mainly, the advent of Generative Adverserial Networks, which have only been around since 2014 [2], and have had a tremendous impact has allowed for more precise image generation, and particularly regarding natural images. Those types of images are the most useful for practical applications. However standard GANs were still far from perfect in terms of generating high resolution and varied natural images. The advances made incrementally on GANs with Progressive GANs [3], and later StyleGAN [4] have allowed for much more realistic and varied images. StyleGAN, which was built using some of the Progressive GAN tricks and architecture designs, has been one of the most successful and influential generative models in the past few years.

The reason for the success of StyleGAN is two-fold. First the architecture has been cleverly designed, so that the content is separated from the style of the image by design. Therefore the generated images can have a few specific charcteristics changed without changing the nature of the image. Second, the latent space of StyleGAN, allows us to navigate through it to gradually change the images. It means it is very well constructed and linear.

This project aims at exploring the latent space properties and qualities when it is constructed through a StyleGAN model. More specifically our goal will be to experiment on these two major strengths that characterize the StyleGAN-based models and evaluate how much they hold up in practice.

2. StyleGAN

The first StyleGAN model 1 (b) has been built upon the design of its predecessor 2, Progressive Growing GAN 1 (a).



(a) Traditional

(b) Style-based generator

Figure 1. (a) Standard ProGAN [3](b)StyleGAN architecture. While a traditional generator [3] feeds the latent code though the input layer only, styleGAN first maps the input to an intermediate latent space W, which then controls the generator through adaptive instance normalization (AdaIN) at each convolution layer. Gaussian noise is added after each convolution, before evaluating the nonlinearity. Here "A" stands for a learned affine transform, and "B" applies learned per-channel scaling factors to the noise input. The mapping network f consists of 8 layers and the synthesis network g consists of 18 layers— two for each resolution $(4^2 1024^2)$.)

Therefore, we can observe that the synthesis network is built rather similarly, with multiple convolutional blocks which grow the size of the generated image throughout the training. This growing network is a peculiarity of ProGAN and StyleGAN although the more advanced versions of styleGAN have managed to do without. At the time it was created, however, the network's growth throughout training was an important feature to generate realistic high quality

	Method	CelebA-HQ	FFHQ
A	Baseline Progressive GAN [30]	7.79	8.04
B	+ Tuning (incl. bilinear up/down)	6.11	5.25
C	+ Add mapping and styles	5.34	4.85
D	+ Remove traditional input	5.07	4.88
E	+ Add noise inputs	**5.06**	4.42
F	+ Mixing regularization	5.17	**4.40**

Figure 2. Frechet inception distance (FID) for various generator designs (lower is better) starting with ProGAN.

images.

StyleGAN most notable achievement is in its restructuring of latent space and style incorporation in the generator. Usually, the latent code $z \in Z$, with Z the latent space, is directly fed to an input layer of the generator as can be seen in 1 (a). However in StyleGAN, the latent code z is not provided directly to the generator. First it is fed to an encoder neural network meant to map the latent codes from space Z to a different space W

The w latent codes are then used to control styles $y = (y_s, y_b)$ which define parameters in the Adaptative Instance Normalization or AdaIN in each convolution blocks, after each convolution layers. The AdaIN operations are charactterised by the following formula:

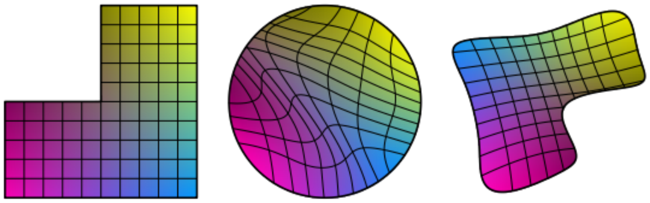
$$\operatorname{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i}$$

with $\mathbf{x}_i$'s, the output feature maps of the preceding convolution. Therefore, after each feature map $\mathbf{x}_i$ is normalized separately, their scale and bias are determined by the corresponding scalar components from style y. This allows different w latent codes to inject different styles into the generated image.

The last important trick used is the noise input, which is also injected after each convolution operation in the generator. More specifically, these noise inputs are single-channel images consisting of uncorrelated Gaussian noise. This noise injection is important because in human faces, and natural images in general, there are significant aspects of the image that can be considered stochastic. For example, in the case of human faces, hair placement and skin pores can be randomized without changing our perception of the images as long as they stay within the correct distribution. The same goes for the placement of grass or orientation of leaves in the more general case of a natural image. As shown in 2, this noise improves the quality of the image (FID score).

StyleGAN combines those tricks and parts to form the following architecture. First, an encoder which generate a style latent code w. Then the Generator made up of multiple sophisticated blocks. In each of those blocks, the input is either a constant (if it is the first block), or the output of the previous block passing through both an upsampling layer and a convolutionnal layer. This input is injected with single-channel images of uncorrelated Gaussian noise. They then pass through an AdaIN layer, modifying the bias and scale of the input according to the desired style y. Finally, they pass through a convolution layer. The number of blocks increases through the training, generating images with higher and higher quality as the number of pixels of the output grows. Finally, the discriminator is the same as in ProGAN [3] and just like the generator, it grows during trianing, each time by a factor of 2.

3. Latent Space in StyleGAN



(a) Distribution of features in training set

(b) Mapping from $\mathcal{Z}$ to features

(c) Mapping from $\mathcal{W}$ to features

Figure 3.

The structure of the latent space in StyleGAN and more generally any generative model is essential. The more linear and disentangled, the better it is when trying to generate specific images.

The specificity of the styleGAN architecture that plays a major role in the latent space's structure is the intermediate latent space W. This latent space, constructed with inputs $z \in Z$ with an encoder does not have to support sampling according to any fixed distribution. Instead, its sampling density is induced by the learned piecewise continuous mapping $f(z)$. This mapping leading to the W space is also believed to be create a latent space with more linear factors of variation as can be seen in 3. The intuition behind this is that the generator's job of making realistic images should be way easier if it is based on a disentangled latent space. Therefore there is a pressure during training for the latent space to be disentangled since it improves the quality of images. Hence, the space W learned is supposed to be much less entangled than the Z latent space.

This property is very desirable and we most importantly, it can be verified through some experiments that we will attempt in the next sections. Indeed, when a latent space is properly disentangled, we should be able to find direction vectors that consistently correspond to individual factors of variation. These vectors can then be used to change a particular attribute and observe whether this affects other attributes or not. A disentangled latent space should not have other attributes affected.

Finally, another endeavor that gets facilitated by a disentangled latent space is that of steering images towards having desired properties. It is possible to think of a GAN latent space as a Riemanian manifold [1] through which you can manipulate the charcteristics of the image. If the latent space is well disentangled, but more specifically, if the different semantics of the image are well disentangled and the space is linear, you get very interesting properties. Mainly, you can steer the latent space towards having a specific semantic attribute, and even define the strength of this attribute. This feature is extremely usefull and has been explored extensively in the paper InterFaceGAN [7]

4. InterfaceGAN and latent space exploration

During GAN training, the generator picks a latent code z from the latent space Z before generating images that aim to be as realistic as possible. However, until InterFaceGAN came out, the latent space structure, properties, and the extent of its disentanglement of StyleGAN were still unclear, despite a few observations made in the original paper. Therefore, InterFaceGAN focused on the latent space organization, and more specifically, how semantics originate from it. The idea was to figure out how the latent code was capable of determining and steering some semantic attributes such as age or smiling on faces. Moreover, they wanted to verify the entanglement and linearity properties, to see how this would affect the attributes on faces generated. InterFaceGAN answered these question by creating a framework capable of identifying the latent space of face synthesis models, and use it to edit faces with respect to clear semantic attributes. This framework was essential in our work to understand and show the extend of linearity and disentanglement in StyleGAN.

4.1. Semantics in the Latent Space

First, to understand how InterFaceGAN works, we need to understand how the semantics work in the GAN's latent space. In a GAN, the generator can be seen as a function g which , when applied to a latent code $z \in Z$ generates an image $x \in X$ with X the image space and Z a d-dimensional space. Then by applying a classifier f on $g(z)$, we create a link between the image created and the semantic space $S \in R^m$ for m possible categories in the classifier. Therefore the semantics of the image $s = f(g(z))$.

This equation can be rewritten: $s = f_S(g(z)) = \Lambda N^T z$. Here N^T is the transpose of a matrix of hyperplanes $n_1, n_2, .., n_m$ that correspond to the boundaries of each semantics, and $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_m)$, with λ_i the linear coefficients associated to each semantic s_i. Using this representation of s, it is possible to compute the mean and variance matrices of s:

$$\begin{aligned}\boldsymbol{\mu}_{\mathbf{s}} &= \mathbb{E}\left(\Lambda \mathbf{N}^T \mathbf{z}\right) = \Lambda \mathbf{N}^T \mathbb{E}(\mathbf{z}) = \mathbf{0} \\ \boldsymbol{\Sigma}_{\mathbf{s}} &= \mathbb{E}\left(\Lambda \mathbf{N}^T \mathbf{z}\mathbf{z}^{\mathbf{T}} \mathbf{N} \Lambda^T\right) = \Lambda \mathbf{N}^T \mathbb{E}\left(\mathbf{z}\mathbf{z}^T\right) \mathbf{N} \Lambda^T \\ &= \Lambda \mathbf{N}^T \mathbf{N} \Lambda\end{aligned}$$

So we have $s \sim N(0, \boldsymbol{\Sigma}_{\mathbf{s}})$ with the different semantic categories s_i being more disentangled from each other in the latent space the closer their corresponding hyperplanes n_i are to being orthogonal. If those hyperplanes are orthogonal from each other, then varying the linear coefficient λ_i corresponding to a single one of them should not affect the other semantics. Therefore we should be able to make images vary according to very specific attributes.

4.2. Finding the Hyperplanes

As has been described above, finding the hyperplanes n_i is essential if one wants to influence some specific semantics meaningfully while keeping the rest of the image untouched. In InterFAce GAN, separate SVMs are trained on separate attributes s_i to produce those hyperplanes necessary for latent space manipulations. In our case, the hyperplanes used are smile, age,gender and eyeglasses.

4.3. Manipulating Latent Space

Using those hyperplanes, for vector direction, it is then quite straightforward to manipulate the different attributes. Basically, usually, we generate an image with a random latent code z. using this z we can generate an image x with semantics $s = f(x)$. Now that we have those direction vectors n corresponding to hyperplanes linked to specific attributes, we can engineer the initial z to become $z_{engineered} = z + \alpha n_i$, with n_i the attribute of interest, and α a coefficient chosen. If $\alpha < 0$, then the image will have less and less of this attribute, or the opposite of the attribute. If $\alpha > 0$, the attribute will become increasingly important in the image. As an example, if the attribute is age, a very negative α will generate images of a younger version of the person, while a very positive value will make a much older version of the person.

Now the idea in our work is to observe whether changes in the α value affect solely the attribute of interest or if it creates collateral damage on other semantics. If the effect of a change in α has an impact on supposedly unrelated attributes, then it means the space is not completely disentangled. We will also compare the 2 latent spaces Z and W

that are part of the StyleGAN model and see if they have different linearity and entanglement qualities.

5. Materials & Methods

6. Experiences, Results & Observations

We will conduct some experiments in order to validate or not the disentanglement property of the latent spaces Z and W (we will be able to compare the behavior of these two different spaces). Then we will try to verify if they are linear. To do this, we will conduct a comparative study based on qualitative and quantitative comparisons.

In order to conduct these studies we will first create a dataset. We will generate 20 very diverse faces (with InterfaceGAN + StyleGAN, pretrained on CelebA dataset [6]), and then we will bring modifications directly to their latent representation (in Z or W) space depending on the attributes we want to change. For each of the attributes in $\{smile, age, eyeglasses, gender\}$, we will vary the importance of this attribute (with a factor between -3 and 3 by steps of 0.5), then we will vary the 20 faces previously generated according to the variations of these attributes. Our final dataset is thus made of 20 images $*$ 4 attributes $*$ 13 variations $=$ 1040 images of dimensions $1024 * 1024$.

When we visualize the manifolds for each attribute in the Z space in Figure 4, we quickly realize from a qualitative point of view that this space does not seem to be totally disentangled. It is true that the attributes are identified, so that if we want to age a face then the resulting face will be older, however there seem to be very strong correlations between getting older, wearing glasses and having white hair. Other examples that seem to appear are that when one wears glasses then the person is older, or that people who smile are younger.
If we pay attention to the manifolds obtained by modifying the W space this time in Figure 5, we can very easily realize that the attributes are better separated compared to Z. Of course the disentangled property of this space is not completely verified: we can again take the example of age: if a person gets older then it is correlated with grey hair and glasses, however each attribute seems a little more decorrelated from the others, there are less coarse correlations.

After having some conclusions based on the visualization of the generated images, we will try to bring a more quantitative comparison. To do this, we will need to extract a feature vector for each image. In fact, we will use the following repository: https://github.com/Hawaii0821/FaceAttr-Analysis which is a multi-label classification model. It is a resnet18 that has been trained on pictures from the CelebA dataset and that returns a vector of 40 activations corresponding to the 40 existing labels in the CelebA dataset (for example: Young, Bald, Male, . . .).
By inferring our 1040 images generated in this classification model, we obtain a vector of 40 predictions for each image, that is to say a matrix of size $1040 * 40$.

We can then study the correlation matrix between these different attributes (both for the images obtained by modifying z: Figure 6 or w: Figure 7). The observed correlations highlight all our previous remarks: there are many positively correlated attributes such as 'Young' and 'Attractive' or others that are negatively correlated such as 'Bald' and 'Young'. Ideally, we would like to see a cluster of correlations with very low values around 0, which would indicate that variations on each attribute do not lead to changes in the others (that they are considered independent of each other). But we can also ask ourselves if the StyleGAN has not learned biases in the learning data, or if on the contrary some correlations would be desired in the context of the generation of human faces. Indeed, it does not seem aberrant to us that age remains correlated with grey hair for example.

In conclusion, the two latent spaces Z and W both seem to be quite good for encoding the attributes of the images, both spaces allow to identify the attributes we want to modify, but the disentangle property is more or less verified: there are less unwanted correlations in W than in Z, even if there are still many 'desired' correlations.

6.1. Linearity

The linear property of latent spaces is an important indicator to study. To do so, we will use the classification model discussed in the previous section : 'refdisentanglement'. This time, we will select the images of our dataset for which we have varied only one attribute, for example 'age'. So we have 20 images $*$ 1 attribute $*$ 13 variations $=$ 260 images. For each of these images, we are interested in the prediction of the 'Young' class of our classification model. By averaging over the 20 images in each value, we obtain the first graph in figure 8. We note that the linear regression obtained on the averages of the 20 images in each value of 'age' is very close to the data, which can allow us to say that the latent space Z is well linear for the Young attribute.
The Eyeglasses parameter is very interesting, indeed, this parameter is very binary in the idea: either we have glasses or we don't. This explains the behavior of the graph: when $value < 0$, then glasses are removed from the face, so no glasses should be detected (which explains why we have a very low probability). Finally, when $value > 0$, there is a linear increase of the prediction which is explained by 2 elements:

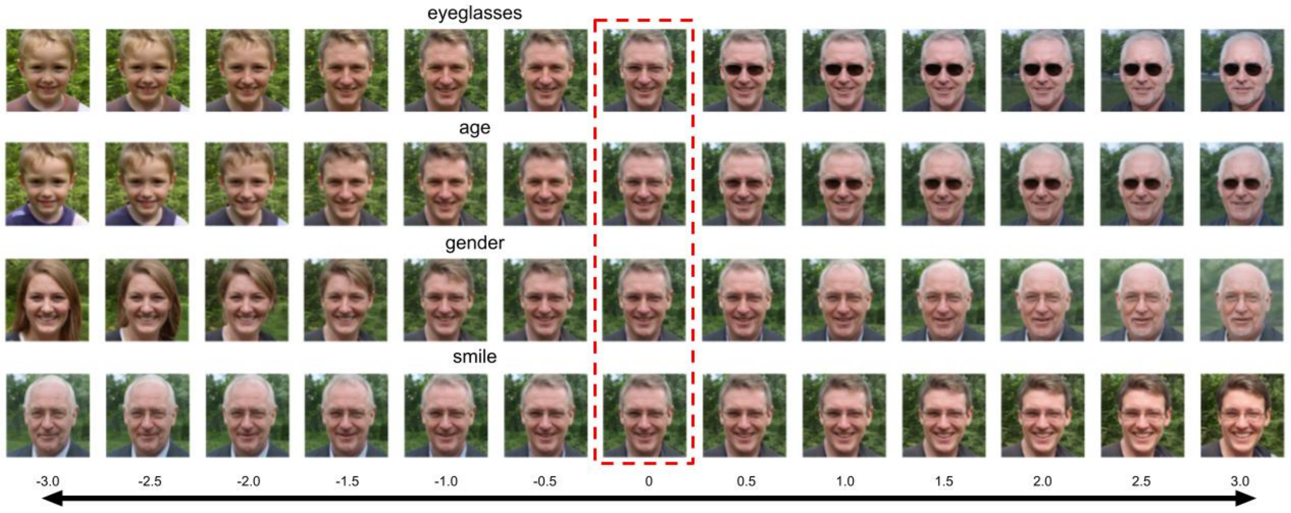


Figure 4. Applying the attribute variations to Z

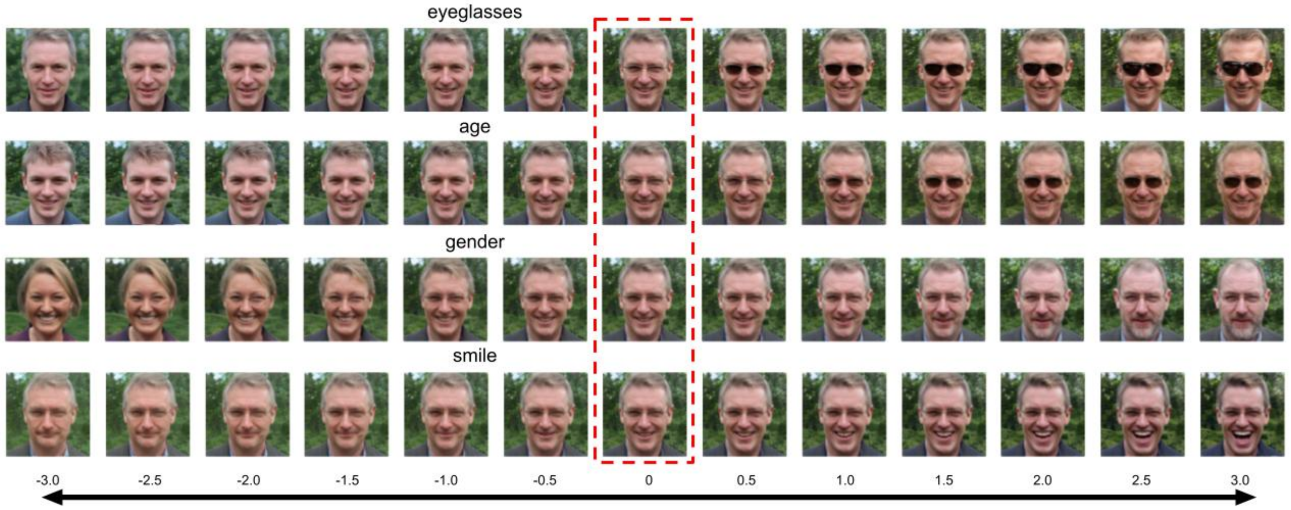


Figure 5. Applying the attribute variations to W

- The fact that in the faces of the database, some have glasses by default (so when $value = 0$)
- some faces of the dataset are children (but, as we mentioned in the previous section, since there are correlations between age and glasses wearing for example, a child will be at a greater distance in the manifold to have glasses because it must be aged too).

When we compare the predictions between the images generated with a modified z: Figure 8 or a modified w: Figure 9, we realize that both seem to be linear with an almost identical behavior, however a slightly smaller standard deviation seems to appear in w (notably on the Smiling attribute). This allows us to conclude on the linearity of the latent spaces Z and W of StyleGAN.

6.2. Mapping images to latent space

We also needed to find a manner to invert the image outputs to get the latent codes associated with them so that we could then compare how the latent codes associated with real images compare with those from synthetic images. In order to achieve that, we needed to map the images from the image space back to their latent space. To do so, we made use of the work of [8] which engineered a very accurate encoder. The use of such an encoder is quite straightforward. We take any image and pass it through the encoder. The output is a latent code that can then be passed through a GAN synthesizer to generate the image.

We used this encoder on 3 types of images to compare

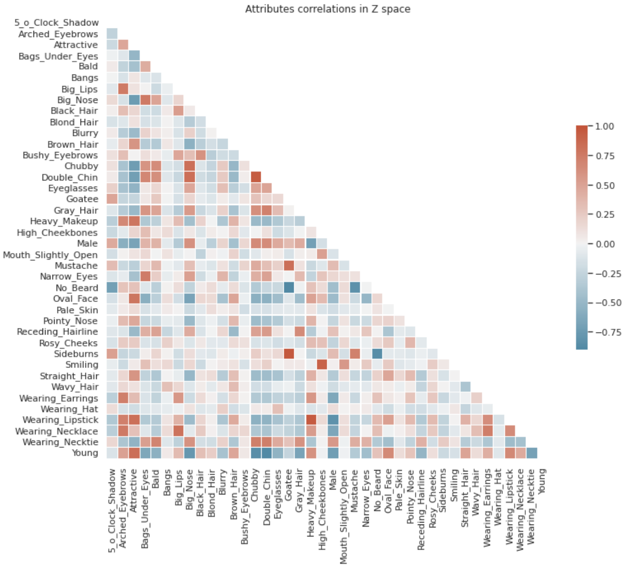


Figure 6. Correlation map by applying attributes' changes to Z

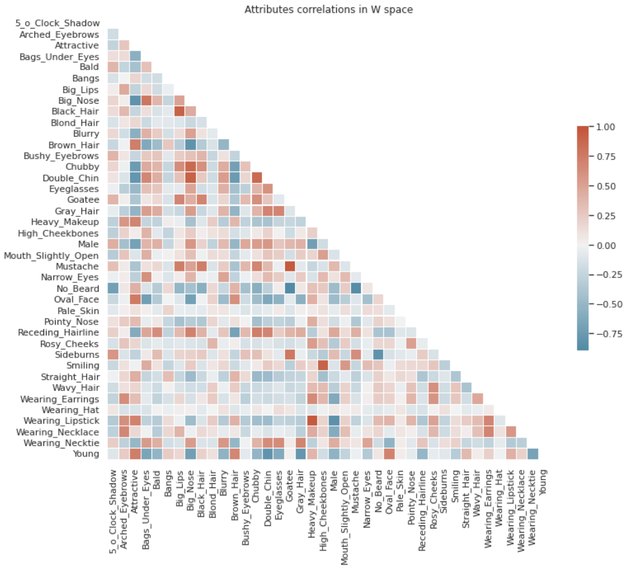


Figure 7. Correlation map by applying attributes' changes to W

the results. First, we used it on synthetic images that we previously generated with StyleGAN as can be seen in 10. Second, we used it on real image faces (ours) to see if the results would be different 12. Finally, to observe whether the result difference could be due to domain change, we used the encoder on real images from the ffhq dataset [5] 10 so we could compare them to the synthetic dataset results.

By comparing multiple images, we observed that synthetic images were generally harder to distinguish when an encoder mapping them back to the latent space. This is expected because they are well defined in such latent spaces. The images from ffhq were also very similar to each other after the process, however there were some notable differences like the hand disappearing or slightly different facial expressions. Finally, the images that differed most were those from profile pictures, with one of them featuring a much more overweight-looking individual than before passing through the encoder. However, the second image from

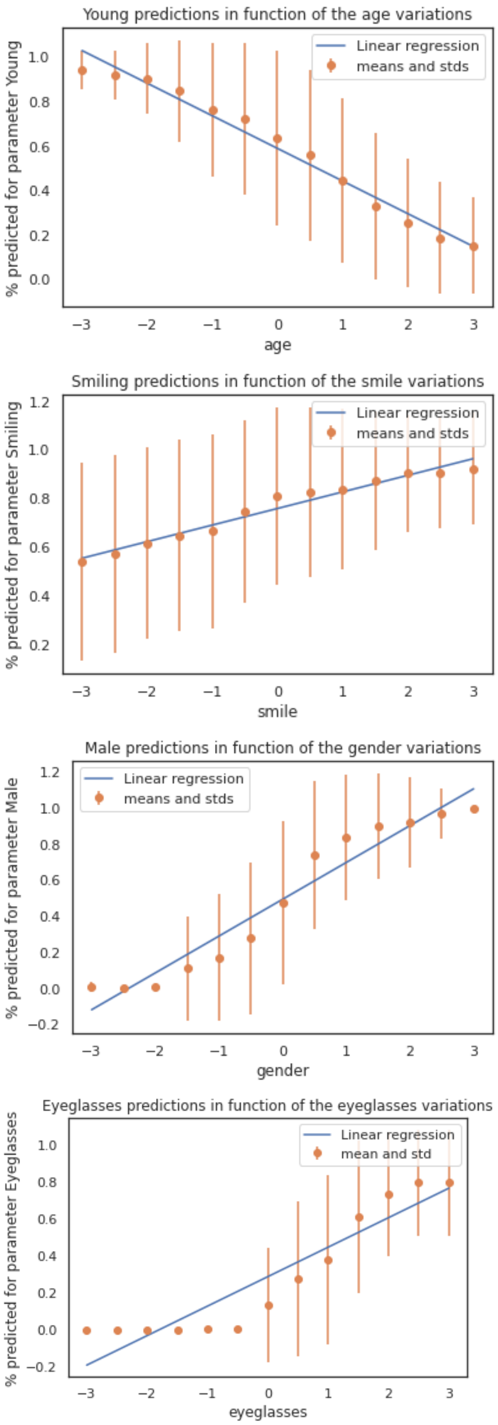


Figure 8. Percentage of the predicted attribute given a modified z representation

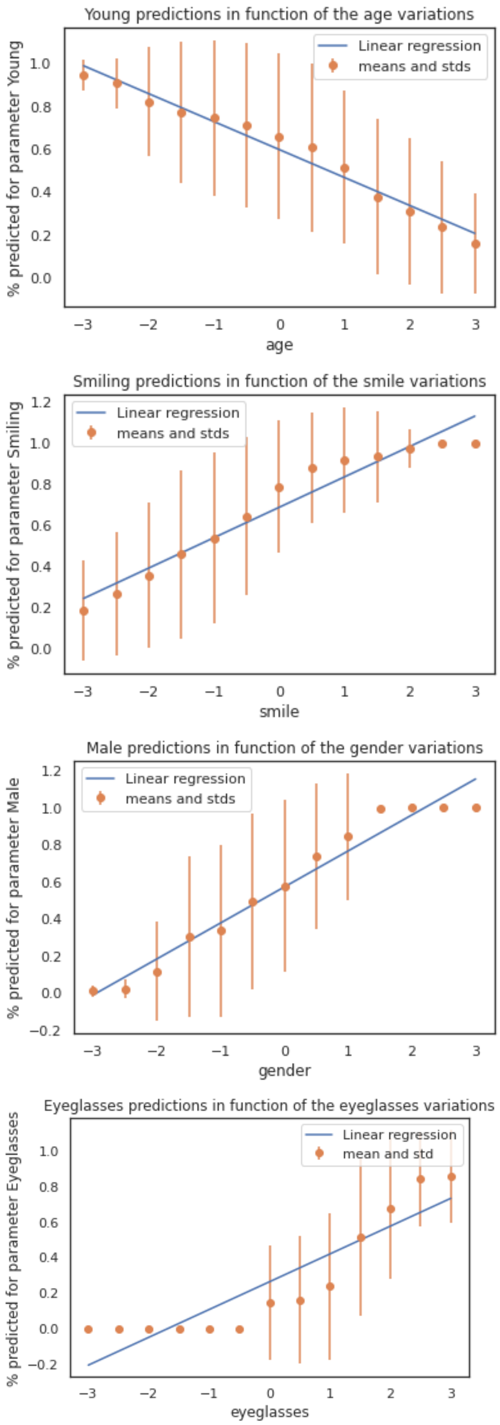


Figure 9. Percentage of the predicted attribute given a modified w representation

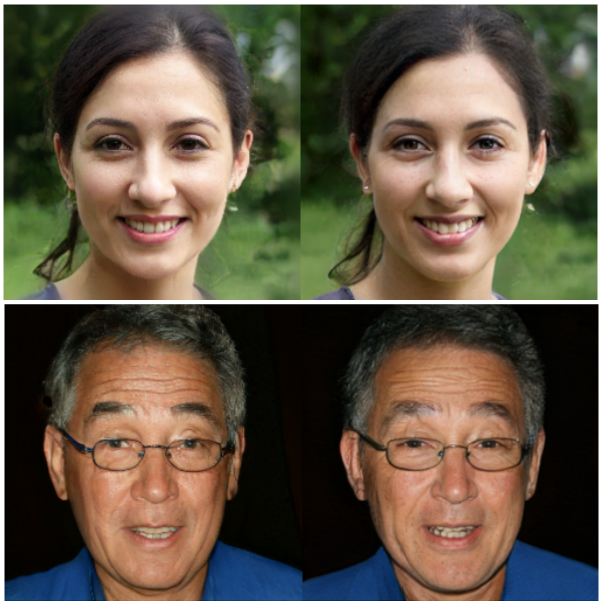


Figure 10. The left images were generated directly from a synthesizer. The right images were generated from the latent code obtained from the left image after going through the encoder

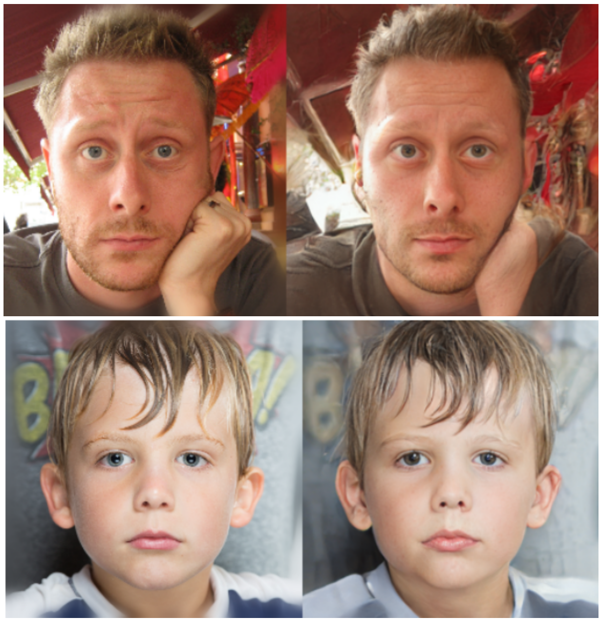


Figure 11. The left images were obtained directly from ffhq. The right images were generated from the latent code obtained from the left image after going through the encoder

a profile picture seem extremely similar. This is an issue when comparing qualitatively as we are forced to do here. However, we have no alternative because we cannot compare the differences in the latent space for those images since the real images do not have an original latent code to compare to. only the sythetic ones do.

7. Conclusions

In order to explore the Z and W latent spaces of StyleGAN, we used a first work called InterfaceGAN which allowed us to vary the importance of different attributes on

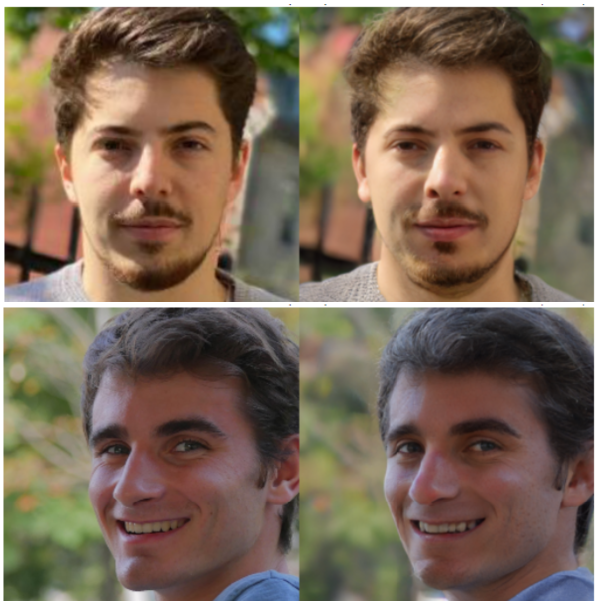


Figure 12. The left images were obtained directly from real profile pictures. The right images were generated from the latent code obtained from the left image after going through the encoder

generated images.

We were able to make these modifications on Z or W and, based on the resulting images, conducted a qualitative and quantitative study on the disentangled and linear properties of these latent spaces. Thanks to an attribute classifier, trained on CelebA, we conducted this study, and we concluded that both spaces were relatively well disentangled, but W was more disentangled than Z because there were less undesired correlations. As for linearity, both spaces seemed to be well linear, although our study on linearity was very strongly biased by the quality of the classifier mentioned above.

Then, we wanted to test StyleGAN on real images, and for that we had to access an encoder, which has for role to encode an image towards a latent representation ($z \in Z$ or $w \in W$). It worked quite well, however it is necessary that the images to encode are 'GAN friendly' i.e. that the face must be relatively centered, with a blurred background, otherwise the encoder will not be able to restore the information that StyleGAN requires in the latent space.

References

[1] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. *arXiv preprint arXiv:1710.11379*, 2017. 3

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1

[3] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017. 1, 2

[4] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CoRR*, abs/1812.04948, 2018. 1

[5] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 6

[6] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *CoRR*, abs/1411.7766, 2014. 4

[7] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9243–9252, 2020. 3

[8] Tengfei Wang, Yong Zhang, Yanbo Fan, Jue Wang, and Qifeng Chen. High-fidelity gan inversion for image attribute editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5