Computational Photography, CS413
Project Report
Master Semester 2 - Spring 2022

CLIP-supervised GAN-based Image Editing

Students: Albert Aillet
Chady Moukel
Xinran Tao

Supervised by: Ehsan Pajouheshgar

October 15, 2022

Image and Visual Representation Lab

EPFL

Abstract

The latent space of Generative Adversarial Networks (GANs) is disentangled and can be traversed to produce distinguishable output images with distinguishable semantics. Therefore, we explore a combination of three different existing loss functions, which are $CLIP_{Loss}$, ID_{Loss}, and $Segmentation_{Loss}$, to optimise CLIP supervised Latent Space Directions (CLIPLSD). Experiments are conducted to verify the performance of our methods. Our results show that locally trained directions can generalise to other images and that images can successfully be edited using the trained directions.

Contents

1 Introduction

Generative Adversarial Networks (GANs) can generate synthetic images. GANs have the potential to be of great use for content generation, the production of facial composites (police sketches), synthetic data generation and image editing which is the focus of this project. Therefore, numerous extensions and improvements have been made to the original GANs architecture. As a continuous work based on previous literature, this project aims to understand the latent space structures of StyleGAN models, and find ways to manipulate the input latent code or weights of a pretrained GAN to achieve the desired change in the generated image using the vision language-model CLIP.

2 Background

Goodfellow et al. [5] first introduced GANs, a model architecture containing two distinct parts: a generator and a discriminator. The role of the generator is to learn to generate data similar to the original data distribution. The role of the discriminator is to distinguish the samples generated by the generator and the samples generated from the real dataset. In the original definition, [5] both the generator and the discriminator are arbitrary functions, which practically are implemented as multi-layer perceptrons. However, the original model of GANs is difficult to train and does not produce high-quality results. Moreover, there is a lack of representations for GAN's intermediate layers, and hence it is also difficult to understand what GANs have learned. The article [5] lays the foundation for a new research direction within maximum likelihood problems with implicit densities, where the networks directly learn the densities.

Radford et al. [13] replace the deterministic spatial pooling functions in the original architecture with strided convolutions for the discriminator and fractional-strided convolutions for the generator, which allowed the networks to learn their spatial downsampling. This model is called *deep convolutional generative adversarial networks (DCGANs)*. Furthermore, batch normalisation is also applied after each convolutional layer for both networks to stabilise them during training. This paper has also investigated the networks' latent space, aiming to understand the correlations between directions in the latent space and the resulting semantic changes in the images.

Huang and Belongie [6] extend *instance normalization (IN)* to allow arbitrary style transfer, namely *adaptive instance normalization (AdaIN)*. AdaIN specifically adjusts each content image to have the target style. The idea is to align the channel wise mean of the content image to that of the style image. The style input adaptively computes the affine parameters for normalisation. The author argues that these affine parameters normalise the feature statistics and allow the styles to be represented by different values. AdaIN adopts an encoder-decoder architecture. The AdaIN layer is added after the content and style images have been encoded in the feature space. Additionally, the decoder maps the stylised image back to its original style space.

StyleGAN [8] has made a step further in disentangling the semantics of the generated images. The author modifies the traditional GANs generator architecture so that before each convolutional layer, AdaIN is applied with the affine transform learned from latent space W. On top of that, a single-channel image with uncorrelated Gaussian noise is used as the input to learn per-channel scaling factors before each AdaIN to allow stochastic features in the generated images. This architecture is useful for the disentanglement because AdaIN normalises each channel for the current layer and removes the dependence of the subsequent layers on the current statistics. Although StyleGAN has greatly succeeded in image generation with GANs, it still contains flaws such as having droplet-like artefacts in some generated images.

Therefore, *StyleGAN2* [9] modify the previous generator architecture to not only fix the artefact issue but also improve other aspects of image generation. The major change was that the noise addition has moved from within a style block to outside. Furthermore, AdaIN is a

modulation of means and standard deviations in previous literature. With the new architecture, AdaIN is replaced with modulation as scaling on the weights, followed by demodulation with standard deviations represented as weights. This technique alters the dependence of styles on actual contents of the feature maps to statistical assumptions about the signal.

Collins et al. [2] discover StyleGAN's ability to spatially disentangle semantic objects and parts in its latent space by running spherical k-means clustering on the activation vectors of several generative models. The author uses this property to conduct local editing of an image given only a reference image. Essentially, the linear interpolation between a target image and a reference image generates the resulting images.

Härkönen et al. [7] apply the *principle component analysis (PCA)* on the latent space of *StyleGAN* and *BigGAN* [1] aiming to find directions that correspond to semantics. For *StyleGAN*, the author samples N random vectors in the z space and computes their corresponding values in w space. The transformed vectors provide a basis for w, and thus the author edits a new image given in w by altering the image's PCA coordinates before feeding it into the networks. For *BigGAN*, the author first inputs the sampled random vectors from z into the networks to produce feature tensors at a layer. Then the PCA from these tensors produces a low-rank basis matrix which is transferred back to the latent space with linear regression. The results show that changes in images are only encoded in certain principal components. Furthermore, *StyleGAN2* 's latent space has a relatively simple structure while *BigGAN* 's components are class-independent.

Wu et al. [14] explore the level of disentanglement and completeness of the w, $w+$, and s space with DCI (disentanglement / completeness / informativeness) metrics [4], and find out that s has the highest DCI scores in both aspects. The author further confirms this discovery by investigating channels in the w, $w+$, and s space that control the visual appearances. The results show that the s space is the least entangled as each channel of the other spaces affects multiple attributes of the images. To put the disentanglement property of the s space into practical use, the author also proposes a method to identify its channels which control a specific target attribute, specified by a set of examples. Specifically, the deviation of the population mean of the style vectors of the dataset from the exemplar mean indicates how relevant the channels are for the target attributes.

Pajouheshgar et al. [10] propose an objective function that evaluates the locality of an image edit given a binary segmentation mask. The author designs a localisation score that measures the proportion of feature-map change inside the semantic segmentation mask for an intermediate layer in the generator. The objective function is then the sum of all localisation scores. Furthermore, a regularisation term is used for finding more than two distinct directions for editing the same semantic in an image.

Radford et al. [12] use natural language supervision on image classifications. The author first creates a new large-scale dataset containing 400 million (image, text) pairs from the internet. Then the author develops a new training method called *Contrastive Language-Image Pre-training (CLIP)* to learn from natural language supervision efficiently. The idea is to train a text and an image encoder jointly to make correct text-image pairings. At test time, the trained text encoder combines the class labels of the target dataset to produce a zero-shot linear classifier.

Patashnik et al. [11] develop three methods of for text-driven image manipulation, namely *optimizer*, *mapper*, and *global direction*. The first two approaches are based on the $w+$ space, while the last approach is based on the s space. The *optimiser* approach solves an objective function which contains a CLIP loss term, a L_2 loss in the latent space, and an identity loss. Although this approach is versatile, it takes several minutes to optimise an image for editing, and it is sensitive to its parameters. Therefore, the *mapper* approach solves the issue with a new architecture which consists of three groups of layers: coarse, medium, and fine. The mapper function is then defined by three sub-mapping that take the latent code of each group as input separately. On top of that, both the CLIP loss and the L_2 loss adapt to this mapper function.

The *mapper* approach nevertheless still suffers from entanglement issues, and thus the *global direction* approach defines a manipulation direction in the s space with the manifold of image embeddings in CLIP's joint embedding space and the corresponding text embeddings. In this case, the direction is simply the channel-wise projection of the CLIP embeddings.

3 Methods

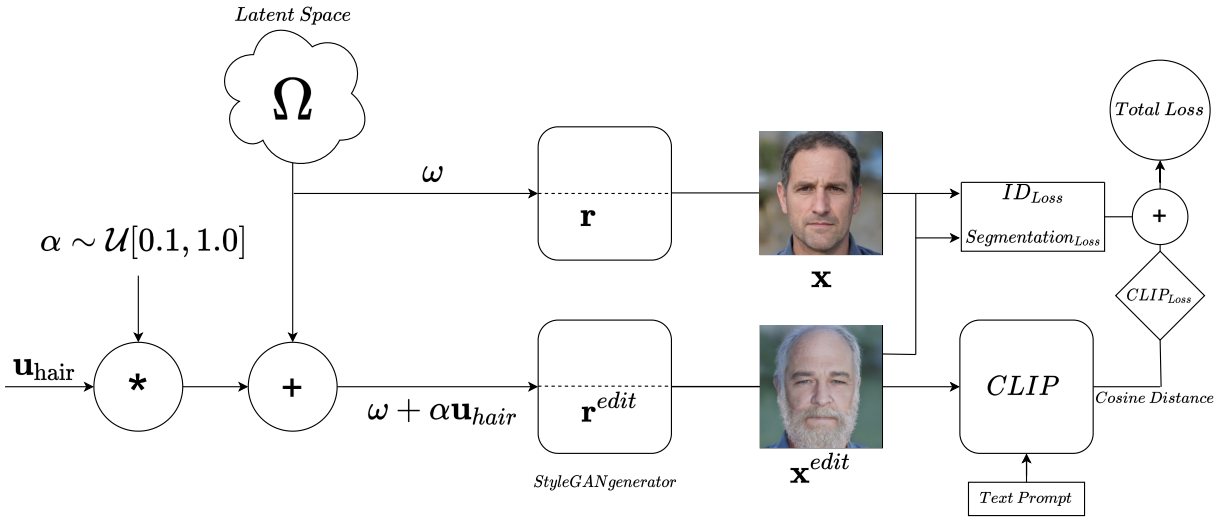

Figure 1: Chart of the method

$$\arg\min_{\mathbf{u}\in\mathcal{W}+} D_{\text{CLIP}}(G(w+\alpha\mathbf{u}),t) + \lambda_{\text{ID}}\mathcal{L}_{\text{ID}}(\mathbf{u}) + \lambda_{\text{S}}LS(\mathbf{u}) \tag{1}$$

The general implementation of the method presented in this paper is based on a combination of the techniques presented in Pajouheshgar et al. [10] and Radford et al. [11]. The general skeleton for training and visualization are modified versions of the LELSD framework and the loss functions are inspired from StyleCLIP [11]. Figure 3.1 presents a chart of the general method used for this project. Our generator $G(.)$ is a pretrained StyleGAN [8] network, it generates an image using a sampled latent code $\omega \in \Omega$, the image is therefore $x = G(\omega) = f(h(\omega))$ where $r = h(\omega)$ is a tensor representing the activation of an intermediate layer in the network. Latent space in StyleGAN can be any of $\mathcal{Z}, \mathcal{W}, \mathcal{W}+, \mathcal{S}$ but for our implementation we only use $\mathcal{W}+$. To edit the image we need to move the latent code across a specific direction u which will result in different semantics depending on its value with a factor α that defines the strength of the change.

$$\mathbf{x}^{edit}(\mathbf{u}) = f\left(\mathbf{r}^{edit}(\mathbf{u})\right) = G(\omega+\alpha\mathbf{u}) \tag{2}$$

The goal is to find a specific direction u_x which will minimize the various losses used in our technique, specifically $CLIP_{Loss}$, ID_{Loss}, and $Segmentation_{Loss}$. The most important loss we want to minimize is the $CLIP_{Loss}$, all the other loss functions act as regularizers, their weights can be tuned via hyper-parameter.

3.1 CLIP Loss

Similarly as in StyleCLIP [11], our method also the optimization of latent space directions based on CLIP Loss which is defined in Patashnik et al. [11] as :

$$\arg\min_{\mathbf{u}\in\mathcal{W}+} D_{\text{CLIP}}(G(w+\alpha\mathbf{u}),t) \tag{3}$$

where $w \in \mathcal{W}+$ is the latent code to be optimized, t is a text prompt in natural language, G is a pretrained StyleGAN [8] generator and D_{CLIP} is the cosine distance between the CLIP[12] embeddings of the two arguments. The cosine distance is defined as 1 minus the cosine similarity

$$D(a,b) = 1 - S(a,b) = 1 - \frac{a \cdot b}{||a|| \cdot ||b||} \tag{4}$$

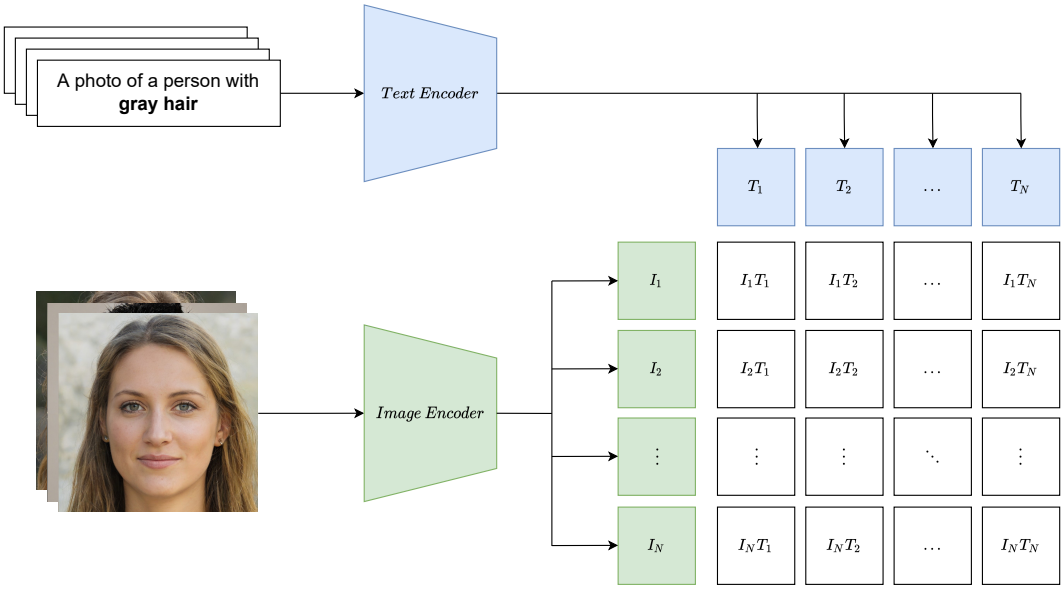

Figure 2: CLIP Loss scheme. A natural language text prompt is encoded into the CLIP[12] latent space (T_i), the image x^{edit} is also encoded into the CLIP[12] latent space (I_i); cosine distance between embeddings is then computed.

3.2 ID Loss

In the previous part we defined CLIP[12] Loss which is computed directly between the edited image and the text prompt and thus does not depend on the original image sampled from $\mathcal{W}+$. We use two additional losses which are, this time, computed using both x and x^{edit}. First the identity loss :

$$\arg\min_{\mathbf{u} \in \mathcal{W}+} \lambda_{\text{ID}} \mathcal{L}_{\text{ID}}(\mathbf{u}) \tag{5}$$

Where λ_{ID} is defined as follows :

$$\mathcal{L}_{\text{ID}}(\mathbf{u}) = 1 - \langle R\left(G\left(w\right)\right), R(G(w + \alpha \mathbf{u}))\rangle \tag{6}$$

R is a face recognition network, in our case, a pretrained Arcface[3] model. Our goal is to maximize the cosine distance between $R(x)$ and $R(x^{edit})$. By minimizing this loss as a regularization term added to the already present $CLIP_{Loss}$ we switch the focus of the optimization to parts which will not affect the identity of the subject.

3.3 Segmentation Loss

The second loss function we have implemented is the Segmentation Loss, defined as follows :

$$LS(\mathbf{u}) = \frac{\sum_{i,j} \mathbf{s}_c(\mathbf{x}, \mathbf{x}^{edit}) \odot |\mathbf{x} - \mathbf{x}^{edit}(\mathbf{u})|^2}{\sum_{i,j} |\mathbf{x} - \mathbf{x}^{edit}(\mathbf{u})|^2} \tag{7}$$

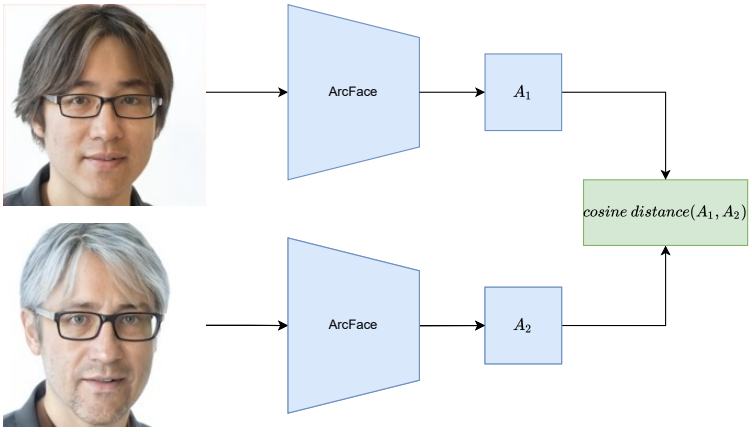

Figure 3: Identification Loss scheme. A_1 and A_2 are the ArcFace embeddings of x and x^{edit} respectively, the cosine distance between both embeddings is computed.

where $\mathbf{s}_c$ refers to the logical operation (union, intersection or average) applied to the segmentation masks of $\mathbf{x}$ and $\mathbf{x}^{\mathbf{edit}}$. The subscript c is the type segmentation applied by the model (e.g. hair, eyes etc.). By adding this loss as a regularization term to the $CLIP_{Loss}$ we focus the optimization of the latent to directions that will change specific parts indicated by the mask.

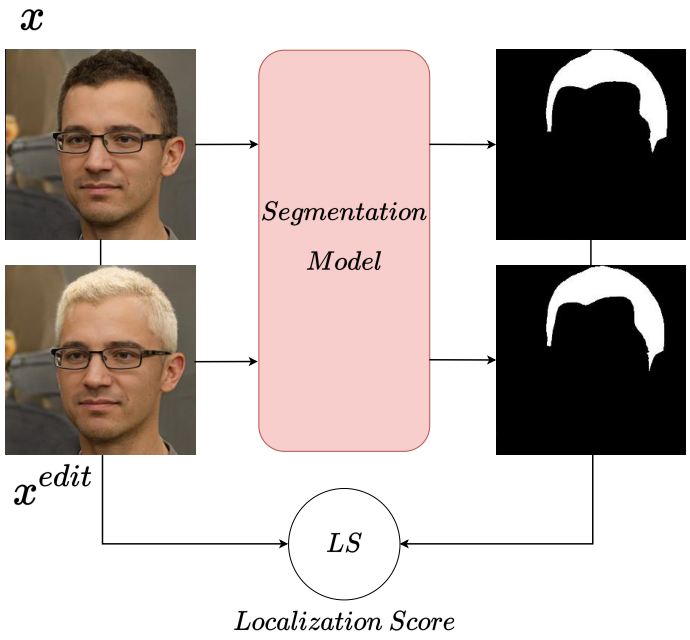

Figure 4: Segmentation Loss scheme. The Hadamard product is computed between the resulting mask $\mathbf{s}_c$ and the MSE of x and x^{edit}; the result is then normalized by the latter.

4 Experiments

4.1 CLIP Similarity

Before starting to implement our loss, we engaged in a preliminary testing of various text prompts coupled with random StyleGAN [8] generated images and selected the ones that were most likely to be identified with a high score by D_{CLIP} .

In figure 5 it is apparent that a high clip similarity indicates a matching semantic between the image and the text prompt. An example of this is the text prompt **a photo of a person with**

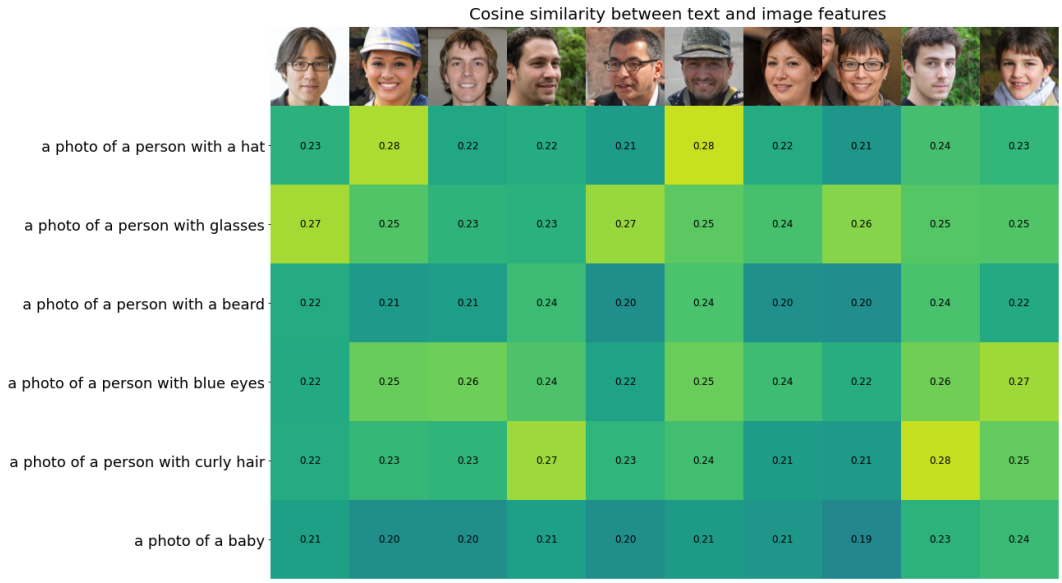

Figure 5: Cosine similarity of text embedding and image embedding using 6 text prompts and 10 randomly generated images from StyleGAN 2 [9].

Figure 6: Series of images showing the edits in using the direction optimized on this image for the text prompt "A photo a a person with blonde hair".

a hat that has a high similarity with the images a columns 2 and 6, both containing generated photo of people with hats.

4.2 First trained directions

After implementing the clip loss, we then ran trained our first directions using the semantic text **a photo of a person with a hat**. The resulting image edits can be seen in the figure 6 and in figure 7.

The two edits shown in figure 6 and 7 are using the same direction that is optimized only on the image from 6. It is apparent that the edit has had the desired effect on the image in figure 6 as the person now seems to have very blonde hair. Similar edits are also present on the image in figure 7 even thought the direction was not explicitly trained on that image. It is therefore of interest to us to investigate how the trained directions generalize from unseen images.

4.3 Generalization of locally optimized directions

To measure the generalization of locally optimized directions, the following experiment was conducted. 20 different directions were locally trained on 20 different images. These directions

Figure 7: Series of images showing the edits in using the direction optimized on the image from figure 6 for the text prompt "A photo a a person with blonde hair".

were then used to edit 50 other random images and the change in CLIP similarity was recorded for different values of α. A high difference in CLIP similarity would indicate that the trained directions correctly manipulate the semantic of interest. We also calculate the change in CLIP similarity of the direction on the images they were originally trained on.

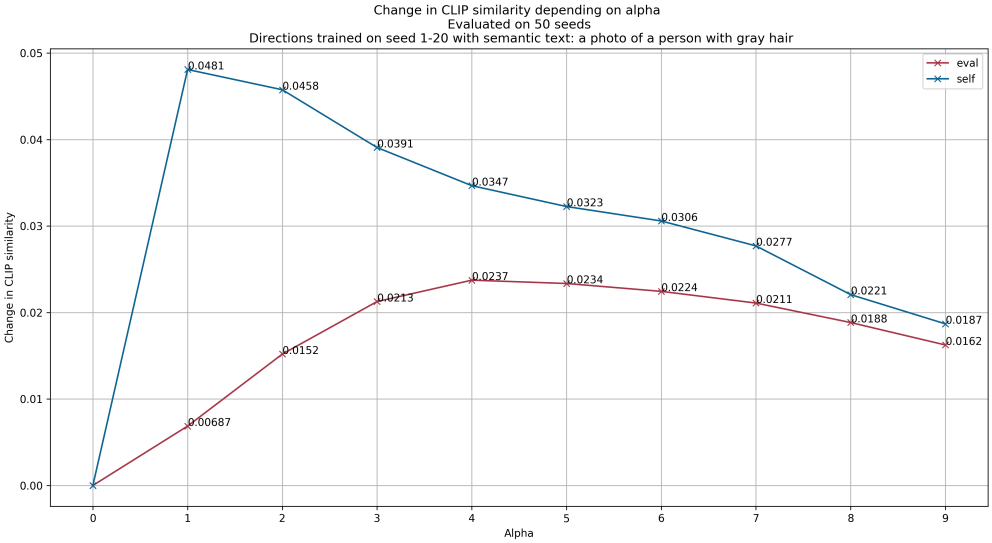

Figure 8: Change in CLIP similarity for generalization of locally optimized directions.

In the figure 8 the change in CLIP similarity is as expected higher when editing the images that the directions where trained on. It is however worth noting that the CLIP similarity also increases for other images, meaning that the locally optimized directions are generalizable to other images.

4.4 Use of ID and Segmentation loss as regularizers

It should however be noted that the edits in figure 6 are neither localized nor preserve the identity of the person. This is explained by the fact that when the direction is trained, the notion of localization or identity preservation is disregarded as the sole metric to optimize for is the CLIP loss. The remedy this, we use the aforementioned ID loss and Segmentation as regularizers when training to keep the found directions to preserve identity and/or edit on certain parts of the image.

In figure 9 directions trained using the same text prompt and the same image, but with different loss parameters are shown. The first row only uses CLIP loss and as expected, the edit maximises the semantic of gray hair but does not keep the identity very well and is not localized only to the hair. The second row that uses the ID loss seems to better retain the identity of the edited image. The third row that uses Segmentation loss localizes the edit to only the hair. Finally when using all three loss terms, it seems to produce the best result for this example.

For other examples see appendix 6

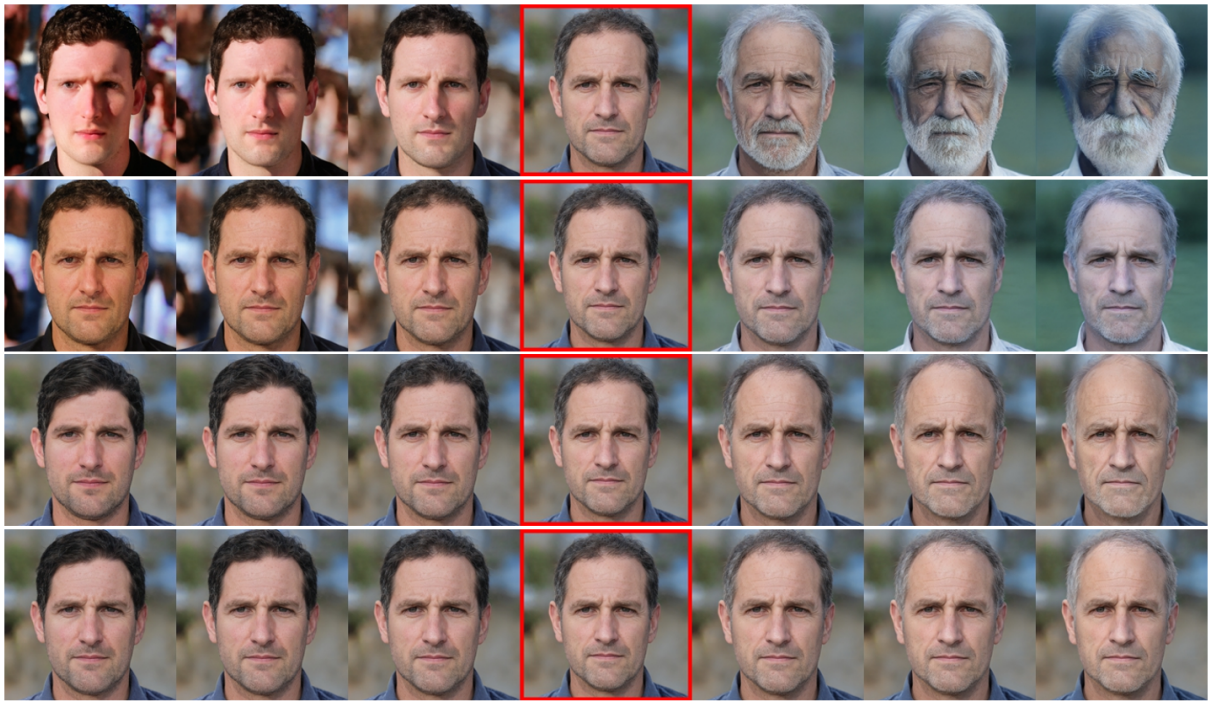

Figure 9: Series of images showing the edits in using the direction optimized on the middle image for the text prompt "A photo a a person with gray hair". Each row shows edit from a different direction. The direction used in the first row only optimized for the CLIP loss. The direction used in the second row optimizes for CLIP and ID loss ($\lambda_{\text{ID}} = 1.0$). The direction used in the third row optimizes for CLIP and segmentation loss ($\lambda_{\text{S}} = 0.1$). The last row uses a direction with CLIP, ID and segmentation loss ($\lambda_{\text{ID}} = 1.0$, $\lambda_{\text{S}} = 0.1$).

5 Conclusion

In conclusion, this project provided us with a good opportunity to learn about the state of the art of GAN models as well as implement techniques for text-based supervised GANs. We were able to edit images using learnt latent space directions supervised by CLIP as well as using identity and segmentation losses as regularizers to maintain the identity of the generated faces. We performed numerous experiments, compared different types of optimization methods and assessed their performances. We also showed data on how directions for specific text prompts that were obtained by training on a specific set of images could potentially generalize to different unseen images. Nevertheless there is still room for improvement, for example being able to implement the same methods on less straightforward text prompts as our methods still fails on specific prompts such as **a photo of a person wearing glasses**, but for that to work, it would probably be necessary to use more targeted training on specific seeds, or use different types of regularizers.

Bibliography

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018.

[2] Edo Collins, Raja Bala, Bob Price, and Sabine Süsstrunk. Editing in style: Uncovering the local semantics of gans, 2020.

[3] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition, 2018.

[4] Cian Eastwood and Christopher K. I. Williams. A framework for the quantitative evaluation of disentangled representations. In *ICLR*, 2018.

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[6] Xun Huang and Serge J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *CoRR*, abs/1703.06868, 2017.

[7] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls, 2020.

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

[9] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. *CoRR*, abs/1912.04958, 2019.

[10] Ehsan Pajouheshgar, Tong Zhang, and Sabine Süsstrunk. Optimizing latent space directions for gan-based local image editing, 2021.

[11] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery, 2021.

[12] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[13] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015.

[14] Zongze Wu, Dani Lischinski, and Eli Shechtman. Stylespace analysis: Disentangled controls for stylegan image generation, 2020.

6 Appendix

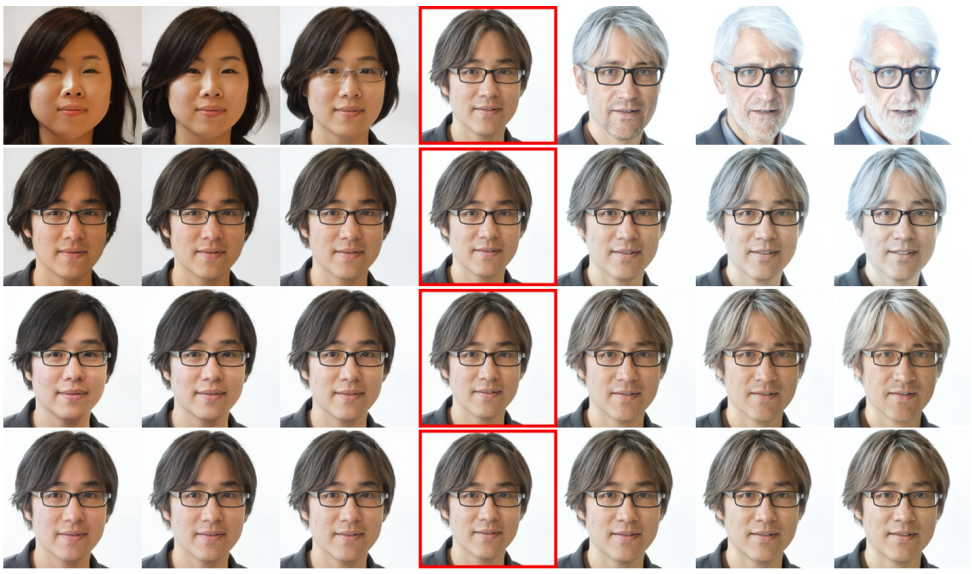

Figure 10: Series of images showing the edits in using the direction optimized on the middle image for the text prompt "A photo a a person with gray hair". Each row shows edit using a different direction. Same parameters as in figure 9

Figure 11: Series of images showing the edits in using the direction optimized on the middle image for the text prompt "A photo a a person with gray hair". Each row shows edit using a different direction. Same parameters as in figure 9