

RADBOUD UNIVERSITY NIJMEGEN
FACULTY OF SOCIAL SCIENCES



Bachelor Thesis in Artificial Intelligence

Exploring the Latent Space of styleGAN

REPRESENTATION OF IDENTITY, GENDER AND EMOTION IN LATENT SPACE

Author:

Karlijn Bok
S4804821

Supervisors:

Tim Kietzmann
Franc Grootjen

February, 2020

Abstract

Although only recently introduced in 2014, Generative Adversarial Networks (GANs) have already undeniably shown their potential. However, there is still a lot of room for improvement regarding the understanding of the latent space. This thesis aims to improve this understanding by exploring the representation of identity, gender and emotion in the structure of the latent space of styleGAN trained on two different face-datasets (i.e., FFHQ and CelebA-HQ). Latent representations of input images from the Radboud Faces Database (RaFD) were found, which were analyzed using Representational Distance Matrices (RDMs) and visualized using t-SNE. The results for the latent spaces of the two styleGAN networks were analyzed both individually and comparatively, in order to determine the effect of the training dataset on the structure of the latent space. The results of this thesis indicate that both networks cluster for identity and gender in their latent space, yet not for emotion. However, for all identities, distinct directions were found for the different emotions when considering the average face of an individual as the origin point. Lastly, the two networks both represent identity, gender and emotion similarly in their latent space, thus the training dataset does not seem to have a substantial effect on the structure of the latent space for the investigated features.

Contents

1	Introduction	3
1.1	Generative Adversarial Networks	4
1.2	The Latent Space	5
1.3	StyleGAN	5
1.3.1	Implementation	7
1.3.2	Training Datasets	7
2	Research Question & Hypothesis	7
3	Methods	8
3.1	Input	8
3.2	Data Collection	9
3.3	Data Processing	11
3.4	Data Analysis	11
3.4.1	RDMs	11
3.4.2	Visualization	13
3.4.3	Significance Testing	13
4	Results	15
4.1	Identity	15
4.2	Gender	16
4.3	Emotion	17
4.3.1	Directions with Neutral Base	18
4.3.2	Directions with Average Base	20
4.4	Comparison	22
5	Discussion	24
6	Conclusion	26
	References	27
A	Significance Values	29
B	Emotion Vectors	30
C	Code Listings	32
C.1	Google Colab	32
C.1.1	Data Processing	32
C.1.2	Sorting Spaces	32
C.1.3	Significance Functions	33
C.1.4	Emotion Directions	34
C.2	Contributions	35

1 Introduction

The Generative Adversarial Network (GAN) was introduced in 2014 by Ian Goodfellow et al. [10] which made it possible for computers to create new faces [8], change the facial expressions of existing faces [4], translate text into images [18] and even create convincing artwork [7]. In contrast to previously introduced generative models, samples can be generated in parallel, there is no need for Markov chains, and there are few limitations for the generator function [9].

A GAN is an architecture consisting of two networks: a discriminator and a generator. These two networks are adversaries, trained simultaneously and in competition with each other. A common analogy is to think of the generator as an art forger and the discriminator as an art expert. These networks force each other to improve by participating in a competitive game, where the generator (forger) tries to make convincing art work while the discriminator (expert) tries to differentiate real from fake art pieces (see Figure 1). The generator has to create more realistic images to keep fooling the discriminator, as the discriminator gets better at deciding whether an image is fake or real. [5] The concept of a GAN is explained in more detail in Section 1.1.

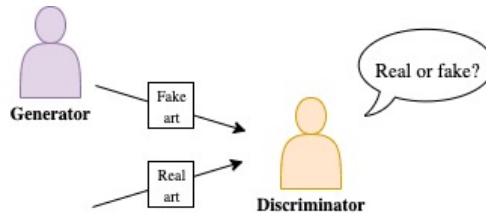


Figure 1: GAN: art forger (generator) tries to fool the art expert (discriminator)

Although GANs have been introduced only recently, the networks have already shown their massive potential, rapidly improving the resolution and quality of generated images [12, 13]. However, since most of the research has been devoted to improving metrics to generate better outputs, there is still a lot of room for improvement regarding the understanding and visualisation of the networks and image synthesis process [2, 13]. Moreover, more quantitative ways are necessary for the comparison of different generators [13].

The purpose of this thesis is to improve the understanding of the “black box” that is the latent space (see Section 1.2). To achieve this, the latent space of styleGAN [13] (see Section 1.3) is explored to gain a better understanding of its structure. The focus of this thesis is on the representation of faces in the latent space specifically. The latent spaces are first investigated individually to determine how faces of a certain identity, gender, and expressing different emotions are represented. Particularly, whether the latent space clusters for identity, gender and/or emotion. Finally, the networks are compared to see whether they make similar distinctions or not, i.e., whether the dataset used to train the network affects the structure of its latent space.

Additionally, Puzer [16] has demonstrated the possibility of learning directions of specific features in the latent space of an original pre-trained styleGAN network [17]. These directions can in turn be used to manipulate images. For example learning a smiling direction and moving images (faces) into that smiling direction. This implies that the latent space of styleGAN might not simply be clustered for emotion, but that it contains

directions for specific emotions instead. This possibility is also explored in this thesis.

1.1 Generative Adversarial Networks

As already briefly explained, the general idea of a GAN is to set up an adversarial game between a generator and a discriminator. As the names indicate, the task of the generator is to generate realistic samples to fool the discriminator, which in turn has to decide whether the sample it gets is either real or fake. The structure of a GAN can be found in Figure 2. The type of output that the generator learns to generate depends fully on which dataset the network is trained. In this thesis, only datasets consisting of images of faces are considered, thus the generator learns to generate faces and the discriminator has to make a verdict on the genuineness of its input image.

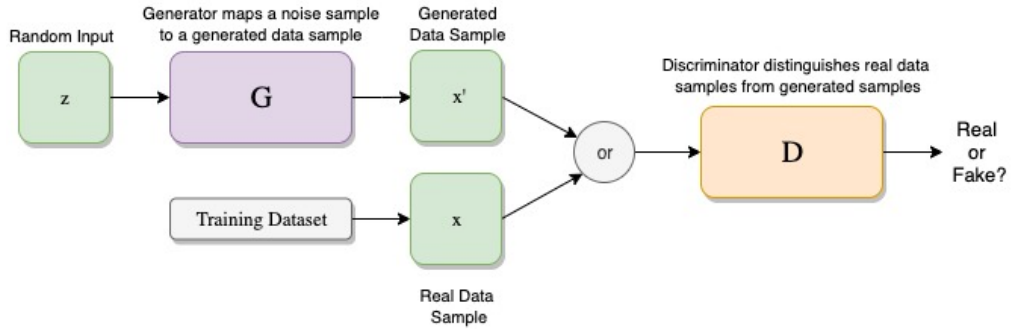


Figure 2: GAN architecture: discriminator D and generator G

More formally, a GAN is a structured probabilistic model. The generator, G , and discriminator, D , are usually multi-layer neural networks made up of convolutional and/or fully connected layers [5]. The task of D in a GAN is simply to classify its input data. It predicts $p(y|x)$, the probability of label y (real/fake) given input sample x . The goal of G in a GAN is to maximize the classification error of D . To achieve this, G receives a random input vector z and outputs a generated data sample x' , which is connected directly to D as an input.

To train a GAN, a competitive (minimax) game is set up between G and D using some training dataset [9] [5]. In this game, the two networks fight over one number: the classification error of D . D wants a low classification error, whereas G wants a high classification error. The competition in this game forces both players to improve.

The game develops as follows:

1. G receives a random input and generates a sample x' ,
2. D alternately receives a sample x from the training dataset or a generated sample x' from G ,
3. D classifies the input sample, which provides a signal that G uses to update its weights,
4. D updates its weights simply by receiving the source of the input (either G or the training dataset) as feedback,
5. Repeat.

For G to succeed against D , G needs to learn how to generate samples which are indistinguishable from real ones. When this has been achieved, the output of D will always be 0.5, having reached the Nash Equilibrium [9]. The end result is a generator that is able to generate very realistic samples, at which point the discriminator can be disregarded.

1.2 The Latent Space

The latent space, also called the hidden space, is the non-observable part of a neural network. The input and output of neural networks can be observed, but everything in between is hidden and operates as a black box. This is called the latent space. It can be viewed as a massive multidimensional distribution, in which the features of your training dataset lie (i.e., the latent space captures the structure of the training data). The training datasets in this thesis consist of images of faces, consequently the latent space is a multidimensional distribution with its data points representing faces. In this distribution, similar data points are closer together. For instance, one can expect two smiling faces to be closer together in the latent space than a smiling face and an angry face.

Since the latent space is hidden, its structure is unknown, which means it is unknown how the network makes its decisions. Exploring and understanding the latent space is relevant, because it can help improve networks by gaining the ability to find causes for strange results, which not only makes it possible to figure out correct and efficient solutions, but also helps to improve future networks.

Exploring the latent space of a GAN corresponds to exploring the latent space of the generator, since the generator has to represent all the information necessary to be able to generate realistic images indistinguishable from the ones in the training dataset [2]. Therefore, in this thesis, representations of the input images are found in the latent space of the generators of the pre-trained styleGAN networks.

There are multiple ways to explore the latent space. It is for example possible to sample from the latent space. Previous work has already shown that representations (i.e., locations) of images can be found in the latent space using gradient descent [15]. Furthermore, many parts of a GAN can be interpreted, with neurons not only correlating with certain objects, but also functioning as variables with causal effect on the object synthesis [2]. It has also been shown that the latent space of GANs in particular seems to linearize the image space, resulting in smooth image transformations when moving through latent space [3]. Moreover, performing linear arithmetic in latent space leads to meaningful results [3]. For example, if you take the latent representation of an image showing a smiling man, subtract the representation of a man, and add the representation of a woman, the result will map to an image of a smiling woman.

1.3 StyleGAN

In March 2019 Karras, Laine and Aila introduced the styleGAN network in their article “A Style-based Architecture for Generative Adversarial Networks” [13]. With this article they managed to develop a GAN that is not only able to generate authentic high-quality images, but also has control over the style of the generated image, allowing for a better understanding of the generated output.

Previous efforts to improve GANs have mainly focused on improving the discriminator model [1, 19] in an attempt to consequently end up with a better generator. With the

development of styleGAN however, the focus was put on improving the generator model, using the architecture of Progressive GAN [12] as a baseline. Progressive GAN was introduced to tackle the problem of generating high-quality high-resolution images. This was solved by training the GAN starting with low resolution images (4x4), while gradually increasing the resolution during training up to high resolution images (1024x1024). This not only improves the quality of the output images, but also speeds up and stabilises the training process. Nevertheless, entangled features still caused for very little control over specific features. StyleGAN enhances the generator of Progressive GAN to solve this by providing a way of controlling the visual features in every level, namely by modifying the inputs of each level separately. This way coarse features up to fine details can be controlled, without affecting any of the other levels.

The structure of styleGAN can be found in Figure 3, which shows the difference between a traditional and styleGAN generator. In contrast to a traditional generator, the styleGAN generator does not simply feed the random input z through the input layer, but instead maps the random input vector z onto an intermediate latent space W using a fully-connected 8-layer mapping network f . The intermediate latent vector w in turn controls the 18-layer (two layers for each resolution $4^2 - 1024^2$) synthesis network g through Adaptive Instance Normalization (AdaIN) at each convolutional layer, after which Gaussian noise is added. 1x1 convolution is then used to convert the output of the last layer to RGB. [13]

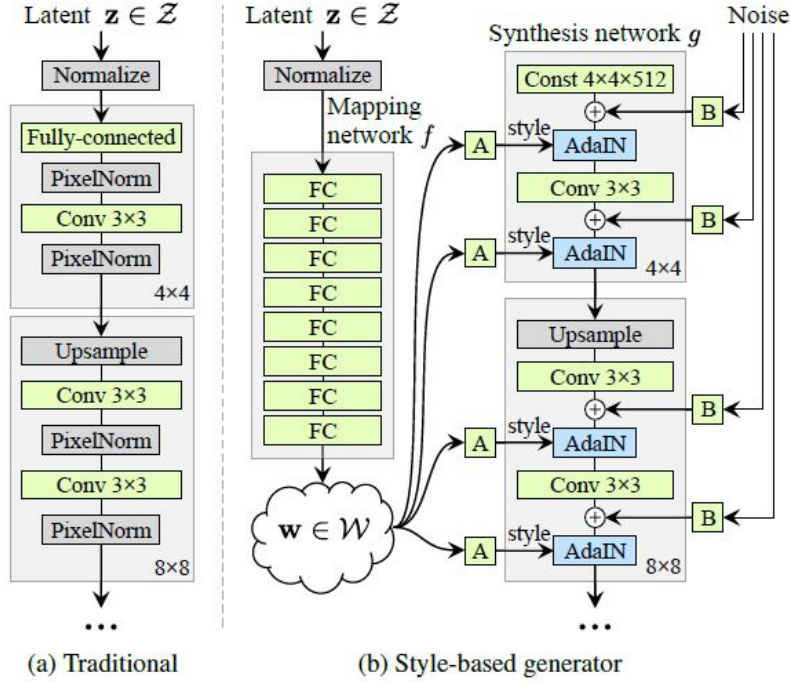


Figure 3: While a traditional generator (a) feeds the latent code through the input layer only, the style-based generator (b) first maps the random input z to an intermediate latent space W , using the 8-layer mapping network f . This then controls the 18-layer synthesis network g through AdaIN at each convolution layer, after which Gaussian noise is added before evaluating the nonlinearity. ‘A’ stands for a learned affine transform, and ‘B’ applies learned per-channel scaling factors to the noise input. ‘FC’ indicates a fully-connected layer. [13]

For more detailed information on styleGAN, refer to the paper “A style-based generator architecture for generative adversarial networks” by Karras et al. [13].

1.3.1 Implementation

The pre-trained versions of styleGAN used in this thesis are from the official TensorFlow implementation [17] of the styleGAN paper by Karras et al. [13]. A styleGAN encoder implemented by Dmitry Nikitko¹ (known as Puzer on Github) [16] is used to find representations (locations) of input images in the latent space of these pre-trained styleGANs. How this code is used exactly is explained in more detail in the Methods section.

1.3.2 Training Datasets

The two StyleGANs were both trained on a different face-dataset: the Flickr-Faces-HQ (FFHQ) dataset and the high-quality version of the CelebFaces Attributes (CelebA) dataset. Exploring the latent space of the same network, but trained on different datasets consisting of similar content, allows for examining the effect of the training dataset on the structure of the latent space.

The CelebA dataset was introduced in 2015, containing more than 200,000 celebrity images with a large variety of poses and background clutter. The high-quality version of this dataset was presented by NVIDIA developers in 2018² in their paper “Progressive Growing of GANs for Improved Quality, Stability, and Variation” [12].

The FFHQ dataset was introduced in the same article as styleGAN [13] specifically designed as a benchmark for GANs, containing 70,000 high-quality images from Flickr, aligned and cropped using dlib. It was created as an improvement upon the existing CelebA(-HQ) dataset, to include more variation with regard to age, ethnicity and image backgrounds.

2 Research Question & Hypothesis

Concluding, the research question for this thesis is:

How are identity, gender and emotion represented in the structure of the latent space of styleGAN trained on different face-datasets?

For this question, there are three main hypotheses (Figure 6a, 6b and 6c respectively):

1. The latent space of styleGAN is clustered for individuals, i.e., images of the same person are clustered together
2. The latent space of styleGAN is clustered for gender, i.e., images of females are clustered together and images of males are clustered together
3. The latent space of styleGAN is clustered for emotion, i.e., images representing the same emotion are clustered together

¹The original code is available at: <https://github.com/Puzer/stylegan-encoder>

²The dataset and full implementation are available at: https://github.com/tkarras/progressive_growing_of_gans

The possibility of emotion directions is summarized in the exploratory hypothesis that the direction from a certain “base” towards an emotion are the same for all individuals. More specifically, that the vectors starting from that origin (base) location, towards the emotion locations have the same direction for all individuals (i.e., that there is a “happy”, “sad”, “angry”, etc. direction for all identities). In this thesis, two base vectors are considered, which concludes the following two exploratory hypotheses (Figure 7):

4. The latent space of styleGAN contains emotion directions considering neutral as the base vector (origin point)
5. The latent space of styleGAN contains emotion directions considering average as the base vector (origin point)

The final hypothesis does not regard the individual latent spaces of the two networks, but concludes the comparison of two latent spaces:

6. The structure of the latent space of styleGAN trained on FFHQ and the latent space of styleGAN trained on CelebA-HQ is similar, i.e., identity, gender and emotion are represented in a similar manner

3 Methods

The first step to investigating the representation of identity, gender and emotion in the structure of the latent space, was to establish and pre-process the input images (Section 3.1). Then the data was collected by finding the representations in latent space of these input images (Section 3.2), after which the data was processed (Section 3.3) to be able to conduct the analysis (Section 3.4).

Data was collected and analysed using Google Colaboratory³ [11]. This is a product of Google Research that offers a hosted Jupyter notebook service for users to run Python code online for free. The sections of the Methods correspond to the sections of the Colab. The most relevant functions of the Colab can also be found in Appendix C.1.

3.1 Input

To be able to determine whether identity, gender and emotion are clustered in latent space, input was required that could be classified into these three categories. For this purpose, images from the Radboud Faces Database (RaFD) [14] were used. The RaFD is an initiative of the Behavioural Science Institute of the Radboud University Nijmegen. It is a database consisting of images of 67 models, including female and male Caucasian adults and children and Moroccan male adults. For all these models eight images are included expressing different emotions (angry, contemptuous, disgusted, fearful, happy, neutral, sad and surprised) according to the Facial Action Coding System. Each emotion is displayed for three different gaze directions and for five different camera angles.

For this thesis, only Caucasian adults were considered with frontal gaze direction and camera angle. This selection was made with practical regards to data availability and concludes a total of 39 identities, of which 19 are female and 20 are male. For each of these individuals there are eight images, one for each of the emotions, giving a total of 312 (39*8) images. Figure 4 shows an example of all images for one identity expressing each of the eight emotions, with frontal gaze and camera angle.

³The Google Colab containing the implementation of all methods mentioned can be found at: https://colab.research.google.com/drive/185vviMJ4m0b0e_z9F09q80pKboaMFhUW

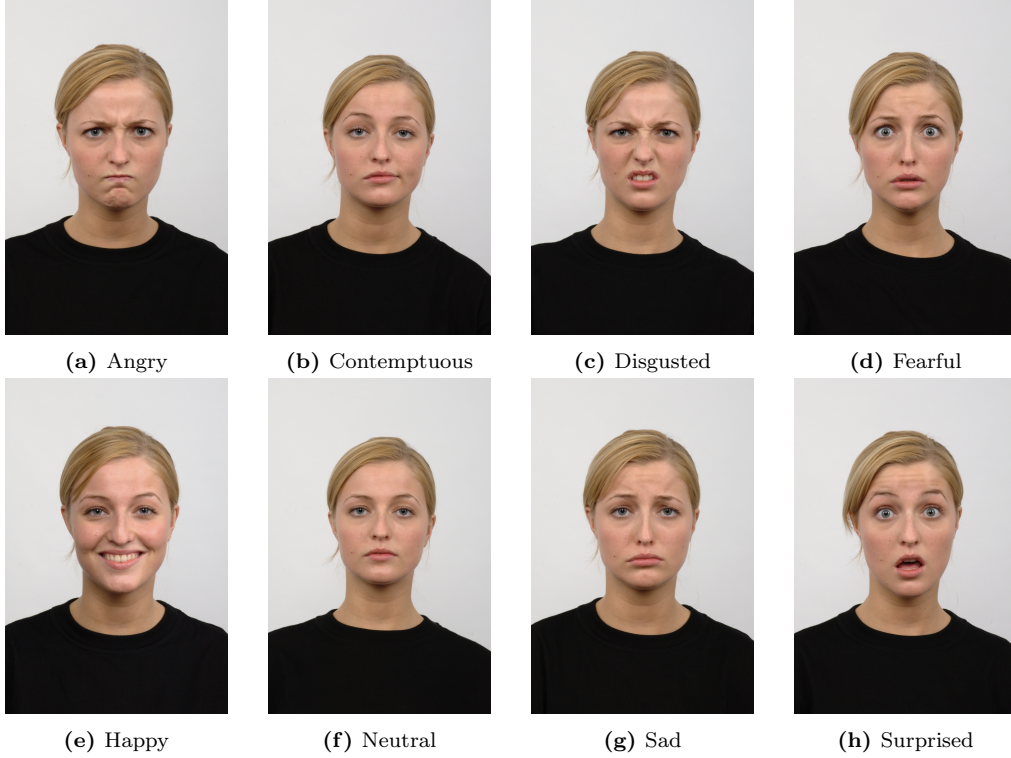


Figure 4: Original images from the RaFD dataset [14] for one of the Caucasian female models, showing the eight different emotions

3.2 Data Collection

To collect all the data (i.e., find representations of input images in latent space), two functions were used from the existing styleGAN-encoder [16]. A few minor alterations were made in order to use the correct pre-trained generator and to change the in- and output folders (Appendix C.2). The first function is `align_images.py`, which was used for the pre-processing of the data, and the second is `encode_images.py`, which was used to find latent representations of the aligned input images (i.e., the latent vector that approximates the input image best). Figure 5 shows the process from original input image to a (generated) latent representation for a single input image. This was done for all 312 input images, for both styleGAN trained on the FFHQ dataset and styleGAN trained on the CelebA-HQ dataset.

Since `encode_images.py` requires a 1024x1024 image, the input data had to be pre-processed using `align_images.py`, by extracting the faces from the original images and aligning them. Step 1 and 2 in Figure 5 show the process of aligning an original image to an input image.

The latent representations were found using `encode_images.py`. This file encodes aligned images (1024x1024) and finds their representative latent vectors (18, 512) (`enc`) and corresponding generated images (`gen`) (Step 2-5 in Figure 5). As shown in Figure 3, the latent space of the generator of styleGAN consists of a random input vector z (`qlatent` in Puzer’s code [16]) of shape (512), and an intermediate latent vector w (`dlatent`) of shape (18, 512). Mapping network f transforms `qlatent` to `dlatent`, and synthesis network g (`g`) transforms `dlatent` into an image.

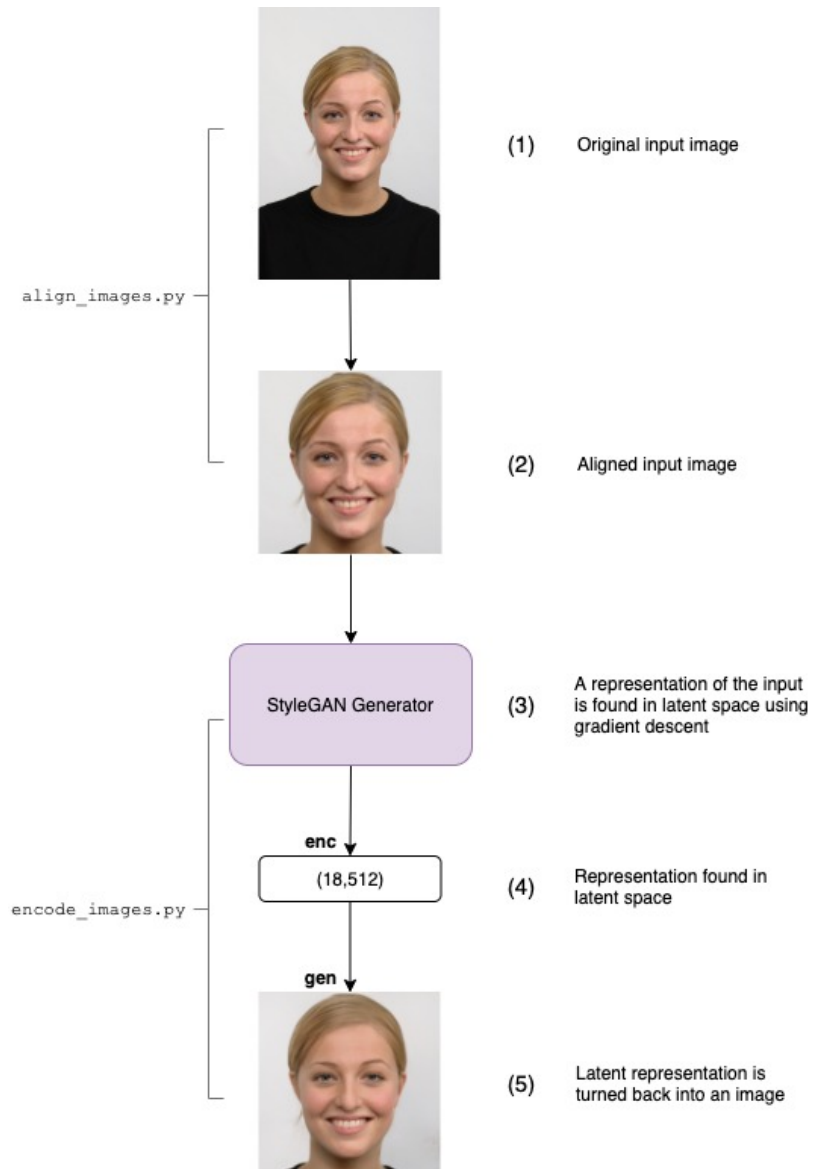


Figure 5: Process of data collection: `align_images.py` extracts and aligns the faces from the original input images, and `encode_images.py` takes an aligned input image and finds its latent representation (`enc`) and corresponding image (`gen`)

To encode an input image \mathbf{x} , the generator and perceptual model weights are frozen completely. The process starts with a random `dlatent`. Using a pre-trained VVG16 model for perceptual loss, gradient descent is then used to minimize the mean squared error of `VVG16(x)` and `VVG16(gen(dlatent))`. Once this process has finished, the optimized `dlatent` (i.e., latent representation) is then also turned into an image by mapping it through the synthesis network `g(dlatent)`.

3.3 Data Processing

Once all encoded latent representations were gathered, these first had to be processed before any analysis could be done. To do this, all of the latent vectors (18, 512) of one network were concatenated and collected into one matrix (312, 18*512), with each row containing a concatenated latent vector (18*512) representing one of the 312 input images (Appendix C.1.1). This original space was then sorted on identity, gender and emotion respectively (Appendix C.1.2), to use for analysis on these features.

The spaces for the emotion directions were constructed using the sorted identity space. As discussed in the hypothesis (Section 2), two possible options for the base vector were examined: (1) the neutral vector of an individual, that is, the latent vector representing the neutral image, and (2) the average vector of an individual, which is the average of all the latent vectors of that individual (angry, contemptuous, disgusted, fearful, happy, neutral, sad and surprised). For both the neutral and average base vector case, the new spaces did not include the representations of the neutral input images. This resulted in a new space of shape (273, 18*512)⁴ with the rows of the matrix containing the vectors calculated by subtracting the base vector from the original vector (see `get_space_without_neutral()` and `get_space_without_average()` in Appendix C.1.4). The matrix was then sorted on the seven different emotions, using the `get_sorted_emotion_space()` function (Appendix C.1.4).

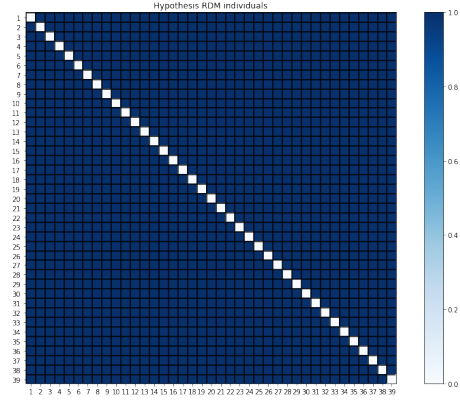
3.4 Data Analysis

The processed data was analysed in a few different manners. Firstly, Representational Distances Matrices (RDMs) were constructed for all of the sorted spaces (Section 3.4.1). These RDMs were then visualized using dimensionality reduction techniques, specifically, t-Distributed Stochastic Neighbour Embedding (t-SNE) (Section 3.4.2). Lastly, the significance was tested by comparing the RDMs to hypothesis RDMs using Spearman's rank correlation coefficient (Section 3.4.3). Significance testing was also performed to compare the latent spaces of styleGAN trained on FFHQ to the latent space of styleGAN trained on CelebA-HQ.

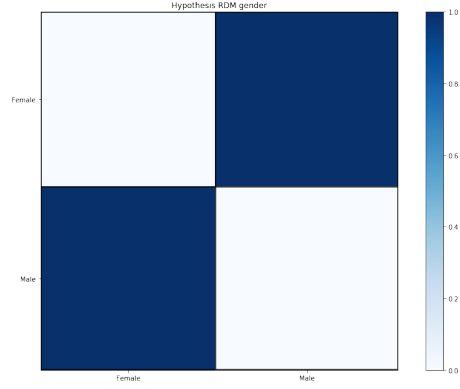
3.4.1 RDMs

For all of the sorted spaces an RDM was computed using cosine as the distance measure. An RDM captures the structure of the latent space as distances between pairs of data points, where in this case a data point amounts to a representation (location) of an input image in latent space. This technique was used to maintain as much information as possible that is captured in the highly dimensional latent space. The RDMs were computed using the Python functions `pdist` and `squareform` from the `scipy.spatial.distance` library.

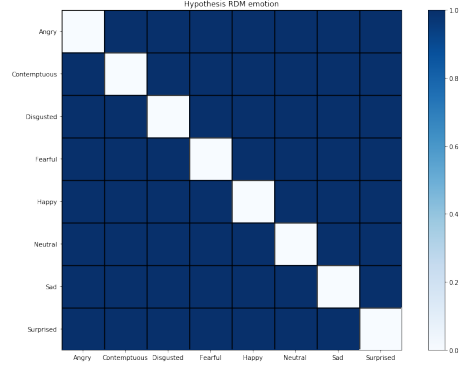
⁴312 - 39 (all neutrals) = 273



(a) Identity



(b) Gender



(c) Emotion

Figure 6: Hypothesis RDMs for the identity, gender and emotion hypotheses respectively

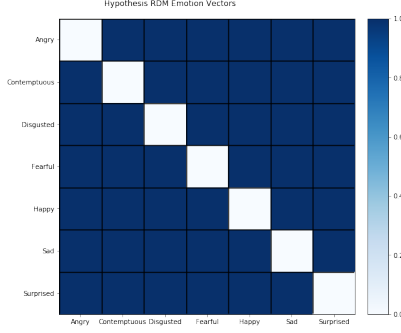


Figure 7: Hypothesis RDM for the emotion-directions hypotheses

Each value in an RDM represents the distance between its indices. Therefore, in for example a latent space that clusters perfectly for identity, the pairwise distances between all images of one identity are zero, and the pairwise distances between one identity and another are one. The same way of thinking was used to compose the other RDMs. This resulted in the three hypothesis matrices shown in Figure 6 and the hypothesis matrix for the emotion directions shown in Figure 7. Section 3.4.3 explains how these matrices are used for significance testing.

3.4.2 Visualization

Since the highly dimensional latent space cannot be plotted, t-SNE was used to map it onto a 2-dimensional space, while trying to keep the distance between points the same. The information obtained from t-SNE was then visualized in scatter plots and coloured for different features, depending on the space (identity, gender or emotion). The emotion directions, however, were visualized as vectors and not scatter plots. The `scatter` and `quiver` functions from the `matplotlib.pyplot` library were used to plot the results for the main hypotheses and exploratory hypotheses, respectively.

3.4.3 Significance Testing

The previously explained hypothesis RDMs (Figure 6, 7) were used to do the significance testing. By comparing these “perfect” RDMs to the RDMs representing the latent spaces, it was possible to make conclusions about the clustering of the latent spaces for particular features (depending on the feature the hypothesis RDM represents).

The RDMs found were compared to the hypothesis RDMs using Spearman’s rank correlation coefficient (Spearman’s ρ). The p -value, however, could not be used since the input images did not satisfy the independence requirement. Therefore, to still be able to make conclusions regarding significance, a different approach was taken, which is explained in the following example.

Consider testing the significance of the RDM sorted on identity for styleGAN trained on FFHQ. First, the `make_significance_vector()` (Appendix C.1.3) function is used to put the top right half of the RDM into an array, excluding the diagonal since it is trivially zero (thus adds no additional information). This is also done for the identity hypothesis RDM. The correlation coefficient (ρ) between these two arrays is then computed using the `spearmanr` function from the `scipy.stats` library. Thereafter, the correlation values are interpreted as described by Dancey and Reidy [6] and shown in Table 1. After this, one of the RDMs (e.g., the one sorted on identity for styleGAN trained on FFHQ) is

shuffled using the `shuffle_RDMS()` function (Appendix C.1.3) and turned into a new array. This is repeated many times, where each time the new random array is compared to the identity hypothesis array, eventually resulting in a distribution of random ρ values (`get_random_rhos()` in Appendix C.1.3). If the previously found original ρ value (of the non-shuffled and hypothesis array) is outside of the 95-percent confidence interval, it means that the correlation is significant. This 95th percentile boundary value is computed using the python `numpy.percentile` function.

Table 1: Interpretation of Spearman’s Correlation Coefficient according to Dancy & Reidy [6]

ρ		Interpretation
+1	-1	Perfect
+0.9	-0.9	Strong
+0.8	-0.8	Strong
+0.7	-0.7	Strong
+0.6	-0.6	Moderate
+0.5	-0.5	Moderate
+0.4	-0.4	Moderate
+0.3	-0.3	Weak
+0.2	-0.2	Weak
+0.1	-0.1	Weak

4 Results

All (unrounded) significance values can be found in Table 3 in Appendix A.

4.1 Identity

As can be seen in Figure 8, for both styleGAN trained on FFHQ and styleGAN trained on CelebA-HQ, the latent spaces are clustered for identity, i.e., clustering the eight images of each individual together.

The latent space (RDM in Figure 8a) of styleGAN trained on FFHQ has a weak correlation ($\rho = 0.2239$) with the identity hypothesis RDM (Figure 6a), which is significant ($0.2239 > 0.0048$). The latent space (RDM in Figure 8b) of styleGAN trained on CelebA-HQ also has a weak correlation ($\rho = 0.2064$) with the identity hypothesis RDM (Figure 6a), which is significant ($0.2064 > 0.0053$).

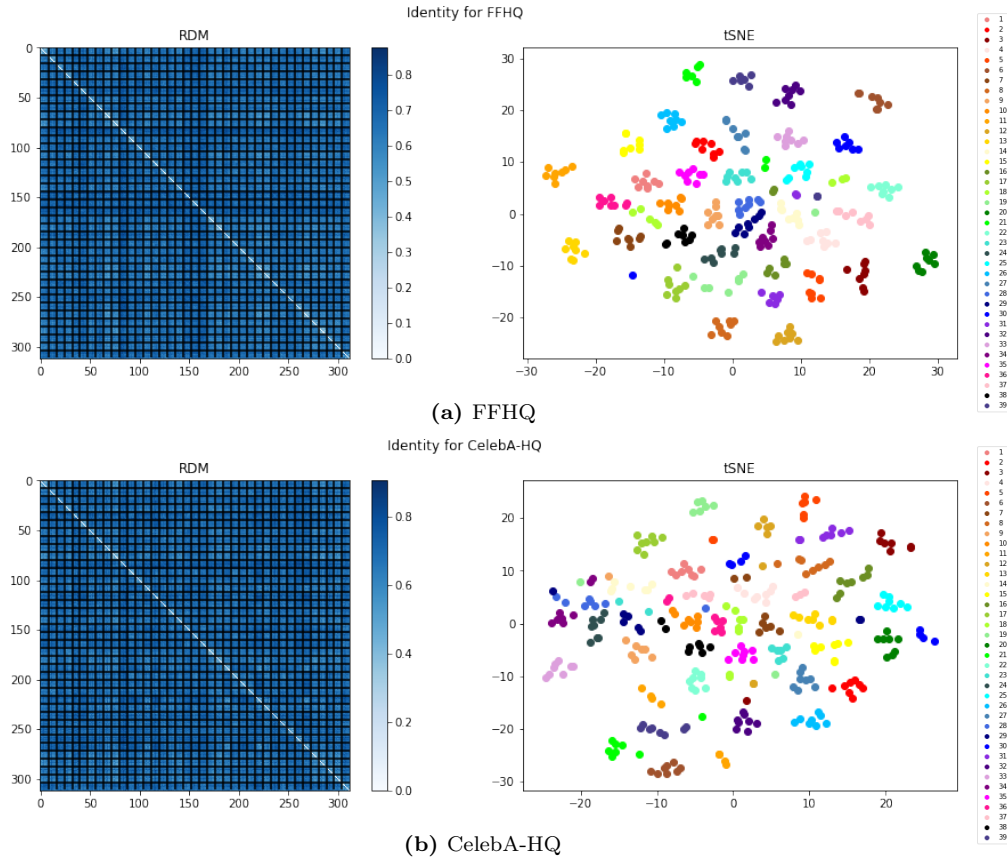


Figure 8: Results for latent space sorted on identity for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs are visualised in the corresponding t-SNE scatter plots, which are coloured for identity.

4.2 Gender

As can be seen in the Figure 9, for both styleGAN trained on FFHQ and styleGAN trained on CelebA-HQ, the latent spaces are clustered for gender, i.e., clustering the data points representing male together and the data points representing females together.

The latent space (RDM in Figure 9a) of styleGAN trained on FFHQ has a weak correlation ($\rho = 0.1703$) with the gender hypothesis RDM (Figure 6b), which is significant ($0.1703 > 0.0062$). The latent space (RDM in Figure 9b) of styleGAN trained on CelebA-HQ also has a weak correlation ($\rho = 0.1617$) with the gender hypothesis RDM (Figure 6b), which is significant ($0.1617 > 0.0064$).

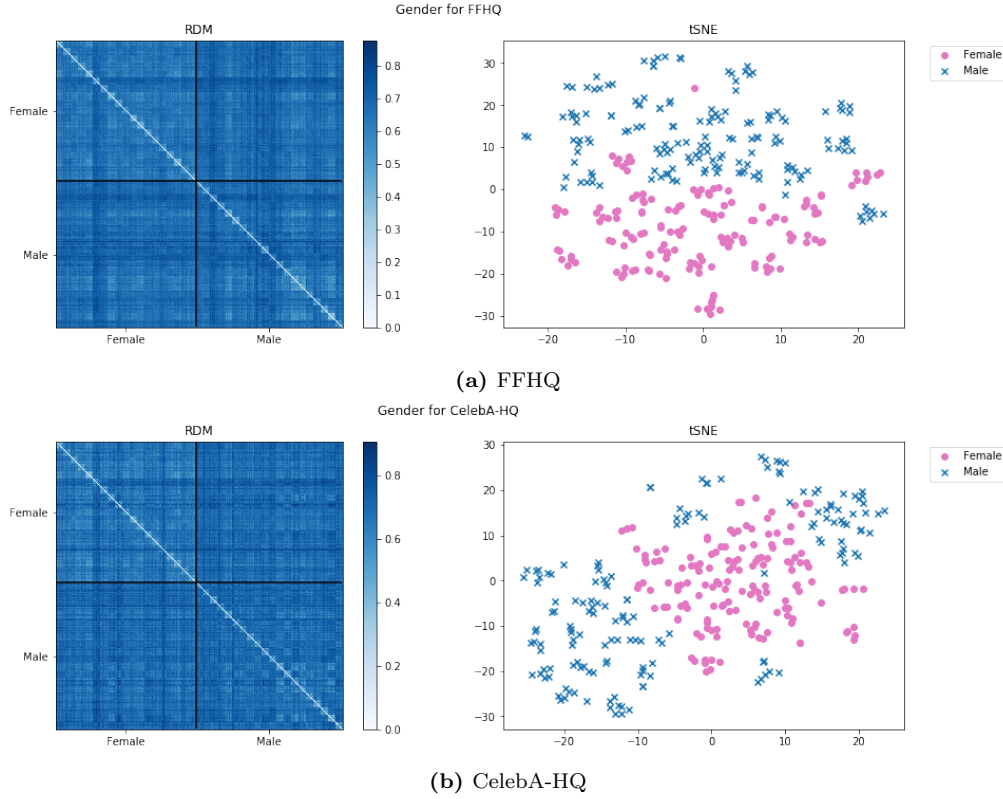


Figure 9: Results for latent space sorted on gender for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs are visualised in the corresponding t-SNE scatter plots, which are coloured for gender.

4.3 Emotion

As can be seen in the Figure 10, for neither styleGAN trained on FFHQ nor styleGAN trained on CelebA-HQ, the latent space seems to cluster for emotion, i.e., the data points representing the same emotion are not clustered together in the latent space.

However, the latent space (RDM in Figure 10a) of styleGAN trained on FFHQ does have a weak correlation ($\rho = 0.1431$) with the emotion hypothesis RDM (Figure 6c), which is significant ($0.1431 > 0.0048$). The latent space (RDM in Figure 10b) of styleGAN trained on CelebA-HQ also has a weak correlation ($\rho = 0.1621$) with the emotion hypothesis RDM (Figure 6c), which is significant ($0.1621 > 0.0055$).

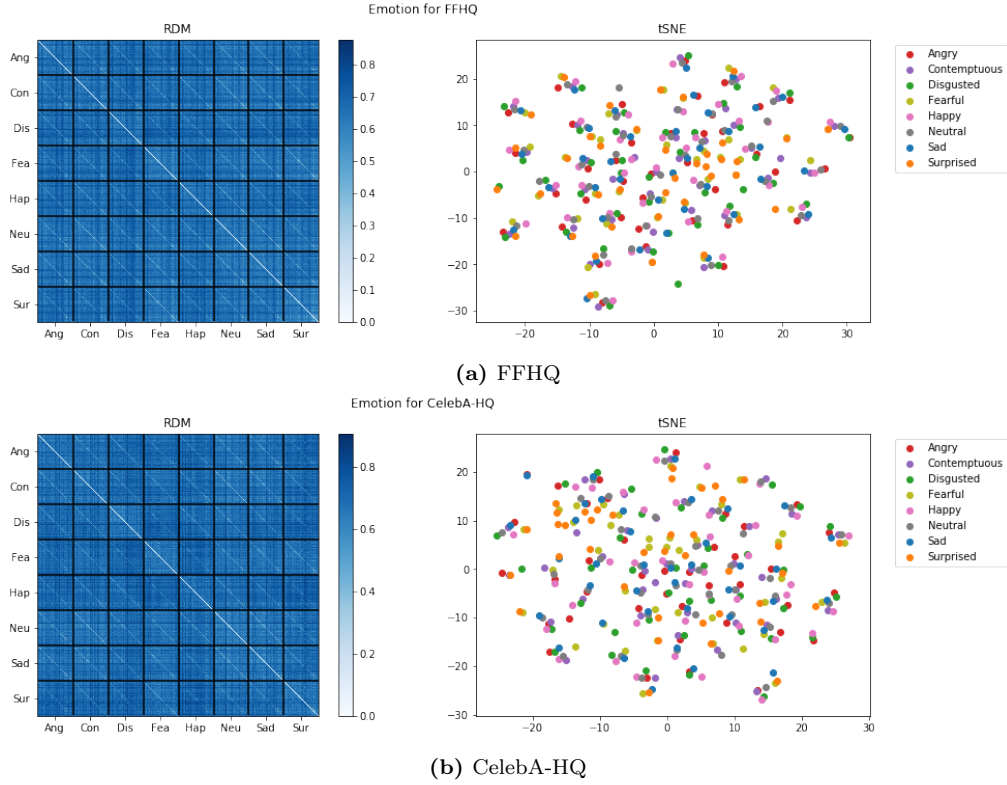


Figure 10: Results for latent space sorted on emotion for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs are visualised in the corresponding t-SNE scatter plots, which are coloured for emotion.

4.3.1 Directions with Neutral Base

As can be seen in Figure 11, for neither styleGAN trained on FFHQ nor styleGAN trained on CelebA-HQ, the latent space seems to have any specific emotion direction when considering neutral as the base vector. Figure 12, which contains the emotion vectors plotted separately for every identity, also confirms the lack of specific emotion directions, since all emotion directions for one identity are very similar, i.e., the happy, sad, etc. vectors for one individual have essentially the same direction.

However, the latent emotion-vector space with neutral as base vector (RDM in Figure 11a) of styleGAN trained on FFHQ does have a weak correlation ($\rho = 0.2508$) with the emotion direction hypothesis RDM (Figure 7), which is significant ($0.2508 > 0.0083$). The latent emotion-vector space with neutral as base vector (RDM in Figure 11b) of styleGAN trained on CelebA-HQ also has a weak correlation ($\rho = 0.2486$) with the emotion hypothesis RDM (Figure 7), which is significant ($0.2486 > 0.0080$).

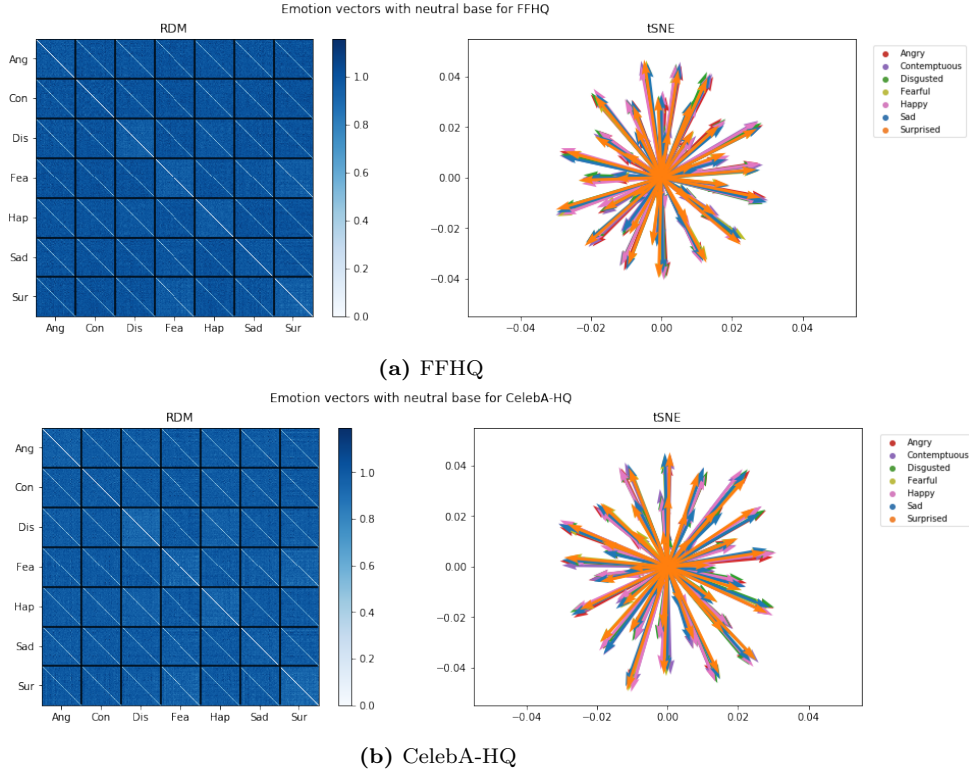


Figure 11: Results for emotion vectors with neutral as base vector (origin) for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs are visualised in the corresponding t-SNE quiver plots, which are coloured for emotion (all except neutral).



Figure 12: Results for emotion vectors with neutral as base vector (origin) for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs from Figure 11 are visualised in these t-SNE quiver plots, in which the vectors are plotted separately for each identity and coloured for emotion.

4.3.2 Directions with Average Base

As can be seen in Figure 13, for both styleGAN trained on FFHQ and styleGAN trained on CelebA-HQ, the latent space does have specific emotion directions when considering the average vector as the base vector. For both networks, the vectors for the Disgusted, Fearful, Happy and Surprised emotion all have a rather distinct direction, whereas the directions are less clear for Angry, Contemptuous and Sad. Moreover, the Fearful and Surprised direction are nearly identical, and the Disgusted and Happy direction are also fairly similar.

Figure 14, which contains the emotion vectors plotted separately for each identity, also supports the idea of distinct emotion directions using an average base, by showing that the emotion directions for one identity are indeed different (i.e., that the happy, sad, etc. vectors for one individual all have a different direction).

The latent emotion-vector space with average as base vector (RDM in Figure 13a) of styleGAN trained on FFHQ has a moderate correlation ($\rho = 0.4307$) with the emotion direction hypothesis RDM (Figure 7), which is significant ($0.4307 > 0.0090$). The latent emotion-vector space with average as base vector (RDM in Figure 13b) of styleGAN trained on CelebA-HQ also has a moderate correlation ($\rho = 0.4254$) with the emotion direction hypothesis RDM (Figure 7), which is significant ($0.4254 > 0.0088$).

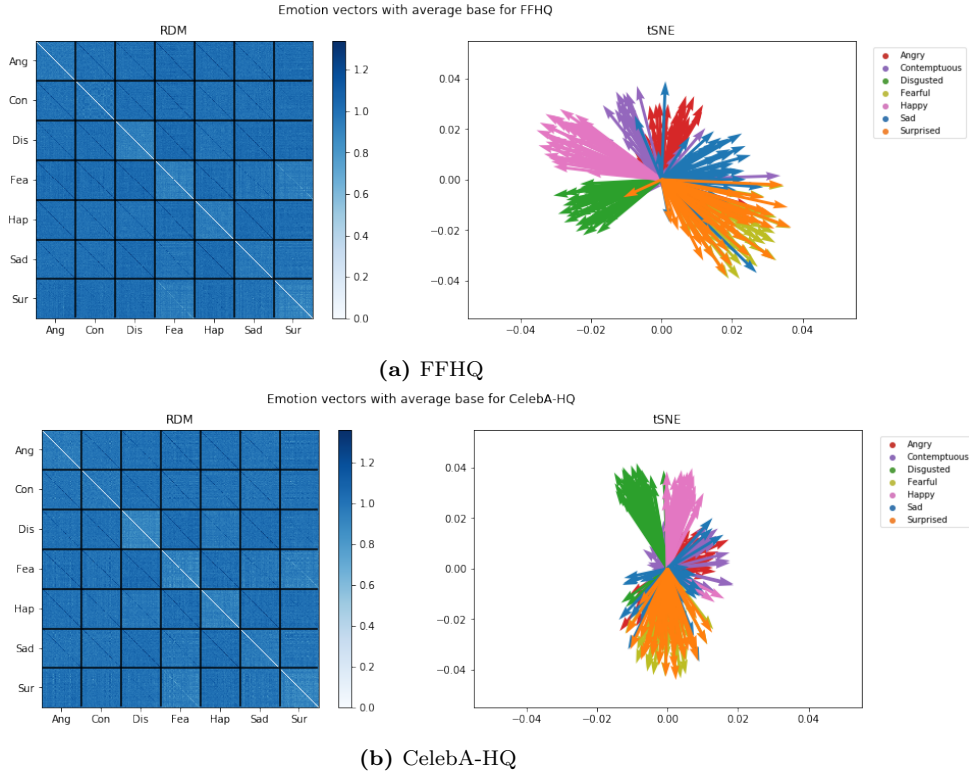


Figure 13: Results for emotion vectors with average as base vector (origin) for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs are visualised in the corresponding t-SNE quiver plots, which are coloured for emotion (all except neutral).



Figure 14: Results for emotion vectors with average as base vector (origin) for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs from Figure 13 are visualised in these t-SNE quiver plots, in which the vectors are plotted separately for each identity and coloured for emotion.

4.4 Comparison

The previously discussed results have shown that styleGAN trained on FFHQ and styleGAN trained on CelebA-HQ both represent identity, gender and emotion similarly in the structure of their latent space.

As an additional comparison, for both networks the latent representations of eight aligned example input images were found (i.e., for one identity) using the `encode_images.py` function as described in the Methods (Section 3.2). Table 2 contains the loss values for the found latent representations, of which the corresponding generated images can be found in Figure 15. As can be seen in Table 2, the average loss value for styleGAN trained on CelebA-HQ (0.35) is quite a bit higher than the average loss value for styleGAN trained on FFHQ (0.17). As can be seen in Figure 15, the representation in latent space of the face itself is very similar to the aligned input for both networks, yet for styleGAN trained on CelebA-HQ, the background colour and hair colour do not match the input image for most representations (dark-blue/brown instead of white/grey background colour, and brown instead of blonde hair).

The latent spaces of styleGAN trained on FFHQ and styleGAN trained on CelebA-HQ have a moderate correlation ($\rho = 0.4856$), which is significant ($0.4856 > 0.0539$).

Table 2: Loss for found representations of example input images in the latent space of styleGAN trained on FFHQ and CelebA-HQ. The corresponding generated images can be found in Figure 15.

Input	Loss	
angry	0.21	0.35
contemptuous	0.15	0.35
disgusted	0.17	0.37
fearful	0.17	0.37
happy	0.18	0.25
neutral	0.13	0.35
sad	0.19	0.36
surprised	0.19	0.38
average	0.17375	0.3475
	FFHQ	CelebA-HQ

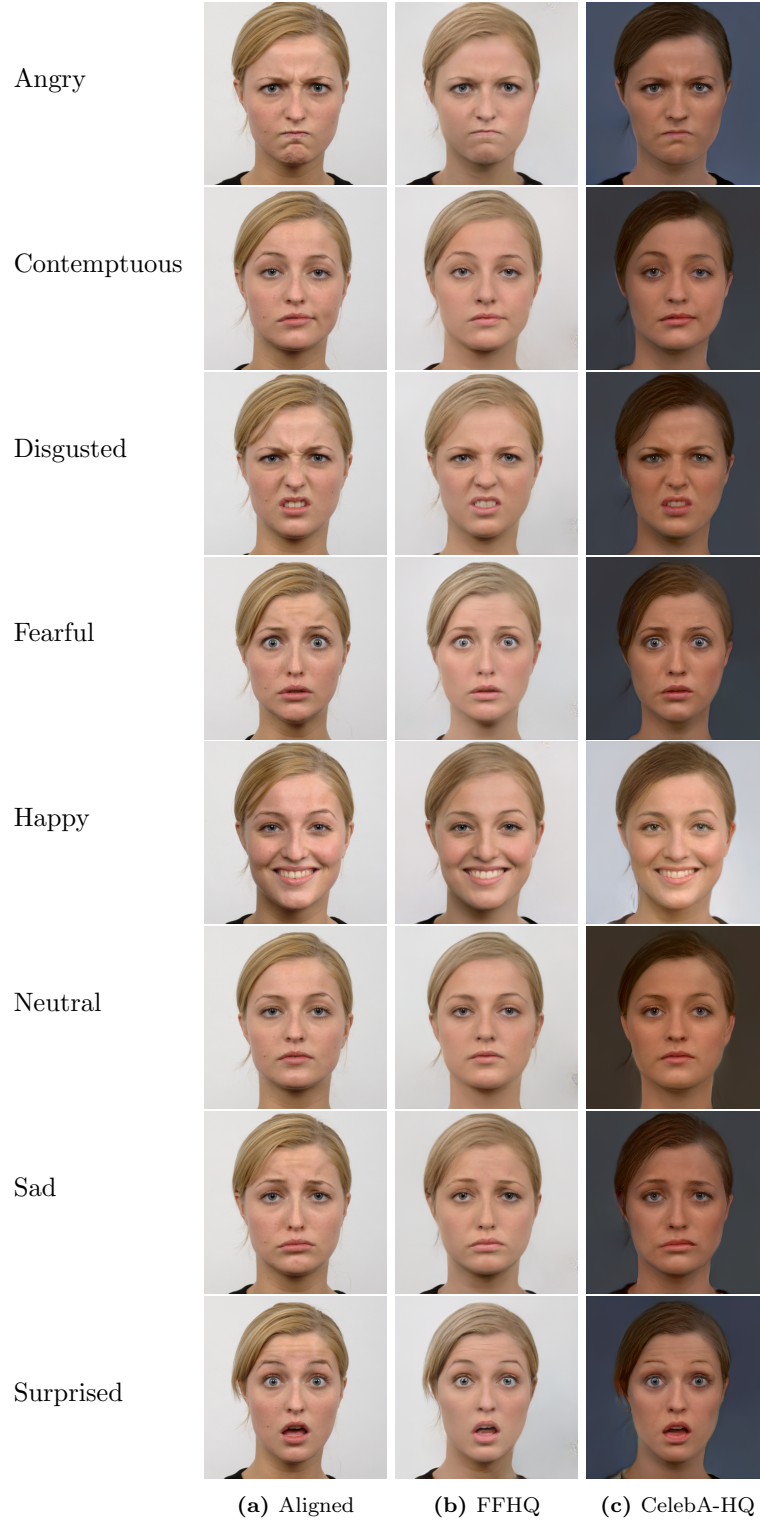


Figure 15: Generated images for latent representations found of aligned example images (a) for styleGAN trained on FFHQ (b) versus CelebA-HQ (c). The loss of these images can be found in Table 2.

5 Discussion

The purpose of this thesis was to advance the understanding of the latent space, of GANs in particular. This was accomplished by investigating the representation of identity, gender and emotion in the structure of the latent space of styleGAN trained on different face-datasets (i.e., FFHQ and CelebA-HQ) using input images from the RaFD database. The structure of the latent space of the networks was first explored individually, after which they were also compared.

The results show that, as hypothesized, the latent spaces of both styleGANs are clearly clustered for identity as well as gender. However, the latent space of both styleGANs does not seem to cluster for the eight emotions examined in this thesis. This could simply be explained by the fact that the latent space is already clustered for identity, thus clustering together all faces of one identity regardless of the emotion the face is expressing. This also confirms the possibility of emotion directions in latent space, i.e., that for one identity there is a specific direction from its “base” towards an emotion.

The results for the exploratory hypotheses regarding emotion directions show that, in contrast to the first hypothesis, the latent vector of the neutral face of an identity is not a good origin point for the emotion directions of that identity. This interpretation is reinforced when separating the data (Figure 11) into different plots for each emotion (Figure 16 in Appendix B), which does not show distinct directions for any of the emotions. A possible explanation might be that the representation (location) of the neutral face of an individual in latent space is not centrally located between the representations of the other emotions of that individual. In other words, that the neutral location relative to the other emotion locations may be different for all individuals. The plots for each identity in Figure 12 support this idea by showing that the emotion vectors for one individual all have nearly the same direction (e.g., the direction from the base to happy is virtually the same as the direction from the base to angry).

An additional base considered in this thesis was the average face of an identity, which was computed by taking the average of all eight latent vectors for that individual. The results for this base indicate that, in correspondence with the hypothesis, the average face works remarkably well as an origin point for the emotion vectors, as multiple distinct directions were found for the emotions (Figure 13). This interpretation is reinforced by looking at the vectors for each individual separately (Figure 14), which show distinct directions for each of the emotions. In fact, this is confirmed even more when plotting the vectors separately for each emotion (Figure 17 in Appendix B) by also showing distinct directions for the different emotions.

However, a clear direction was not visible for all of the emotions, which is probably since not all emotions considered in this thesis are very distinctly visible in the expression of a face. A contemptuous face for example is more similar to an average face than a happy face and an average face. In addition, the expression of the contemptuous emotion might vary more between different identities than for instance a happy face. Moreover, some different emotions are expressed in the face very similarly, such as fearful and surprised, which both involve an open mouth and widely opened eyes. This corresponds to the similar directions found for fearful and surprised in the plots shown in Figure 17 (Appendix B).

The final hypothesis of this thesis concerned the comparison of the two latent spaces (i.e., the latent space of styleGAN trained on FFHQ and the latent space of styleGAN

trained on CelebA-HQ) to investigate the effect of the training dataset on structure of the latent space. The analysis performed in this thesis confirmed the similarity of the representation of identity, gender and emotion in the latent space of the networks. This implies that the training dataset does not affect the structure of the latent space regarding these features. However, to be able to draw any definite conclusions more features and datasets need to be examined.

Although the training datasets explored do not seem to affect the structure of the latent space for identity, gender and emotion, some differences were found. The main difference between the two latent spaces is that the loss of the encoded vectors found in the latent space of styleGAN trained on CelebA-HQ is much higher than the loss of the encoded vectors found in the latent space of styleGAN trained on FFHQ (Table 2). This difference between loss values can be explained by looking at the generated images of the found latent vectors (Figure 15). Although both networks appear to represent faces very well in their latent space, the hair and background colour of the generated images from styleGAN trained on CelebA-HQ do not match those of the input images. For styleGAN trained on CelebA-HQ, only the generated image for “happy” has a similar background colour to the input image, consequently the loss value is much lower (i.e., 0.25 instead of ± 0.36). Hence, the higher loss values can likely be explained by the difference in hair and background colour. The difference between the two networks in the resemblance of the generated results and the input images may be explained by the fact that the FFHQ dataset was created as an improvement upon the CelebA dataset, thus containing more variation and better quality images. Since the latent space relies completely on the training dataset, more variation in the dataset also entails more variation represented within the latent space, also with respect to for instance background colour. The FFHQ dataset might contain more uniform light-coloured backgrounds than the CelebA dataset, allowing for better results for these specific input images.

However, this might raise the question of the importance of background colour when computing the loss of found representations, i.e., in determining how well a network has learned to represent faces. The results indicate that faces are in fact represented very well in both of the networks (Figure 15), which is what the networks were supposed to learn. Yet, the loss for styleGAN trained on CelebA-HQ is still very high. Thus it is up to consideration whether the loss as it is now is completely representative of the quality of the found representations. However, since hair colour is a relevant feature in identifying individuals, this does seem like an important feature to take into account when representing faces, thus should in fact be included when calculating the loss.

There are two major things that need to be taken into account when considering the findings of this thesis. First of all, the highly dimensional latent space (i.e., $18 \times 512 = 9216$) is approximated using RDMs and t-SNE, reducing the dimensions into a value (distance) and 2D plot, respectively. Consequently, information is lost. This reduction of dimensionality on top of shifting of the data makes for complex interpretation. Additionally, the latent space might be better explained in more than just two dimensions, but this would need additional inspection.

Secondly, the significant correlation values merely indicate that the latent space makes some sort of distinction for a specific feature. However, most correlation values are quite low, thus do not indicate a strong (or even a moderate) correlation. This makes sense since they are calculated using hypothesis RDMs that represent a “perfect” space, containing only distances of either zero or one, which will never correlate perfectly with the true latent space. Additionally, the hypothesis RDMs suggest the clustering of a only

a single feature (and no others). Higher correlation values might be found for instance by comparing the computed RDMs to hypothesis RDMs suggesting the clustering for multiple features (e.g., identity as well as gender). Therefore, the significant correlation values indicate that the latent space does distinguish somewhat for the features examined, but to what extent exactly warrants further investigation.

To conclude, the results found in this thesis should be considered carefully and in combination with each other, due to the loss of information captured in the high-dimensional space and the RDMs representing a perfect space for one feature at a time.

This thesis has attempted to improve the understanding of the latent space of styleGAN, but additional research is necessary and possible to further improve this comprehension. First of all, the research done in this thesis could be compared to many more other GAN architectures trained on face-databases. Moreover, the research can be expanded by investigating the representation of more features, like ethnicity, age, hair colour, etc. Eventually it might even be possible to compare the representation of faces in the latent space of GANs to the representation of faces in the human brain. Lastly, the effect of training datasets on the structure of the latent space can be examined for more datasets, including ones not consisting of faces.

The research into emotion directions could be expanded by computing the actual direction vectors for the emotions, for example by using logistic regression as implemented by Puzer [16]. These directions can then be used to manipulate images, for instance by moving a neutral face into the angry direction, which in turn can be compared to the original angry face to assess the quality of the found direction. An additional interesting idea would be to, for example, examine possible differences between male and female regarding these emotion directions. Finally, future research could explore the possibility of a better base vector for the emotion directions.

6 Conclusion

In summary, this thesis explored the representation of identity, gender and emotion in the latent space of styleGAN trained on two different face-datasets. The key findings, emerged from the results, are that both networks cluster for identity and gender in their latent space, but no considerable clustering was found for emotion. However, the results of this thesis confirmed previous findings about the possibility of emotion directions in latent space, for which the average face of an individual seems like a remarkably good estimate, but the neutral face of an individual does not. Moreover, the results for both networks regarding the three features examined in this thesis were very similar, indicating that the training dataset does not have substantial effect on the representation of faces in the structure of the latent space.

Although this thesis may have only scratched the surface, it highlights the realm of possibilities yet to be explored on this exciting subject.

References

- [1] S. Azadi, C. Olsson, T. Darrell, I. Goodfellow, and A. Odena. Discriminator rejection sampling. *arXiv preprint arXiv:1810.06758*, 2018.
- [2] D. Bau, J. Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba. Gan dissection: Visualizing and understanding generative adversarial networks. *arXiv preprint arXiv:1811.10597*, 2018.
- [3] P. Bojanowski, A. Joulin, D. Lopez-Paz, and A. Szlam. Optimizing the latent space of generative networks. *arXiv preprint arXiv:1707.05776*, 2017.
- [4] Y. Choi, M. Choi, M. Kim, J. W. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.
- [5] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.
- [6] C. P. Dancey and J. Reidy. Statistics without maths for psychology. *Pearson Education*, 2007.
- [7] A. Elgammal, B. Liu, M. Elhoseiny, and M. Mazzone. Can: Creative adversarial networks, generating ‘art’ by learning about styles and deviating from style norms. *arXiv preprint arXiv:1706.07068*, 2017.
- [8] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. *Class Project for Stanford CS231N: Convolutional Neural Networks for Visual Recognition, Winter semester 2014.5:2*, 2014.
- [9] I. Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.
- [10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, ..., and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [11] Google. Colaboratory. <https://colab.research.google.com/>, 2020.
- [12] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.
- [13] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.
- [14] O. Langner, R. Dotsch, G. Bijlstra, D. Wigboldus, S. Hawk, and A. van Knippenberg. Presentation and validation of the radboud faces database. *Cognition & Emotion*, 24(8), page 1377–1388, 2010.
- [15] Z. C. Lipton and S. Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017.
- [16] N. R. Projects. Stylegan — official tensorflow implementation. <https://github.com/NVlabs/stylegan>, 2019.

- [17] Puzer. Stylegan - encoder for official tensorflow implementation. <https://github.com/Puzer/stylegan-encoder>, 2019.
- [18] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.
- [19] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016.

A Significance Values

Table 3: Significance results for comparing the found RDMs to the hypothesis RDMs. Correlation is significant if rho is bigger than the 95th percentile. Interpretations of the rho values can be found in Table 1.

		rho	95th percentile	significant
Identity	FFHQ	0.2239372705691981	0.004781192671666683	True
	CelebA-HQ	0.2063895019389554	0.005280024072033491	True
Gender	FFHQ	0.17026036333740513	0.006179444378317391	True
	CelebA-HQ	0.1617119774914968	0.006377531902659089	True
Emotion	FFHQ	0.14308099217756795	0.004778783037530968	True
	CelebA-HQ	0.1620805507342279	0.0055042964187941535	True
Neutral base	FFHQ	0.2508150354331845	0.00830191970508079	True
	CelebA-HQ	0.24863765916274677	0.00797638352660076	True
Average base	FFHQ	0.4307068903733777	0.009048542219578062	True
	CelebA-HQ	0.4254103216944323	0.008802474360076614	True
FFHQ vs CelebA-HQ		0.4856444687406472	0.053893917137200424	True

B Emotion Vectors

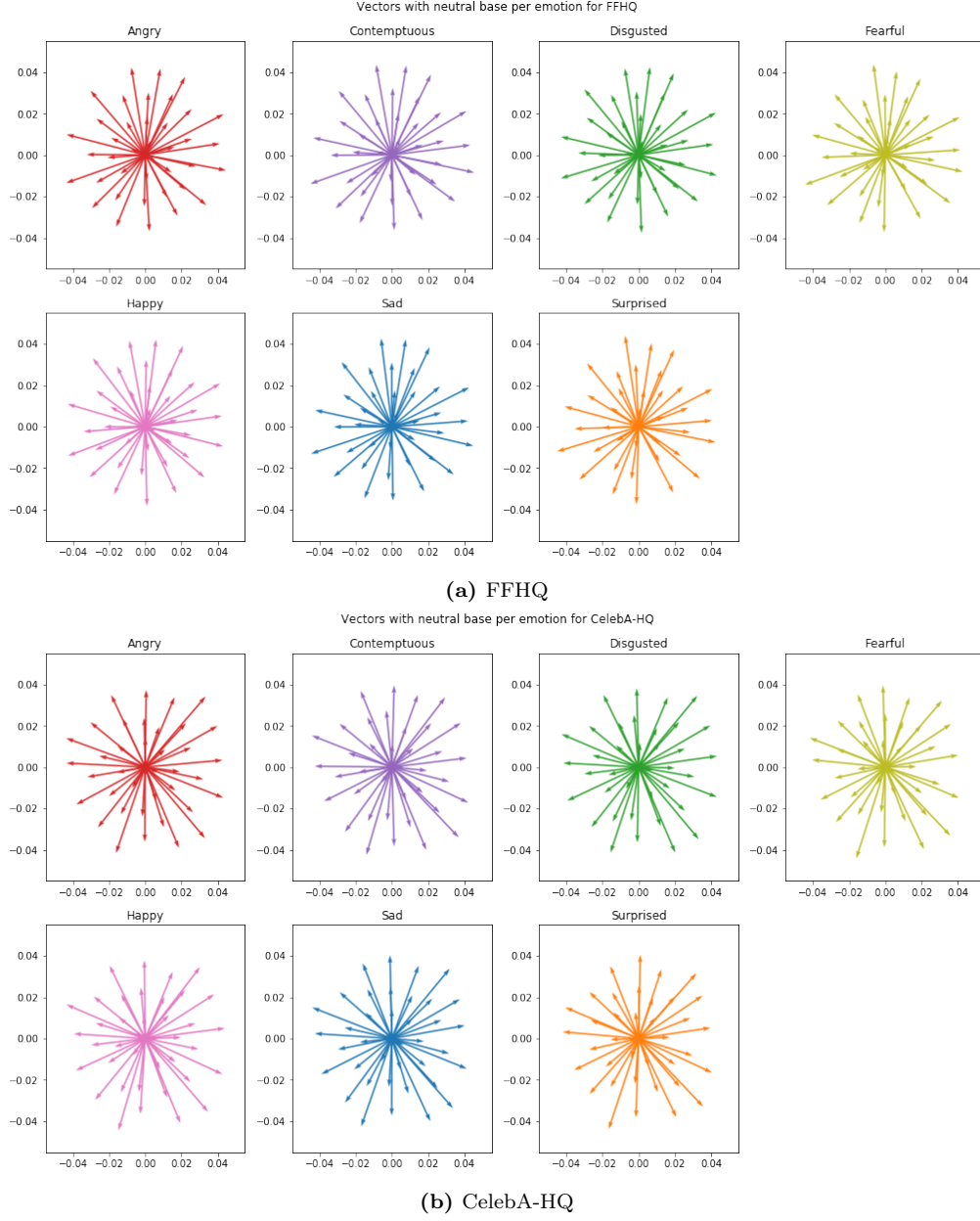


Figure 16: Results for emotion vectors with neutral as base vector (origin) for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs from Figure 11 are visualised in these t-SNE quiver plots, coloured and plotted separately for each emotion.

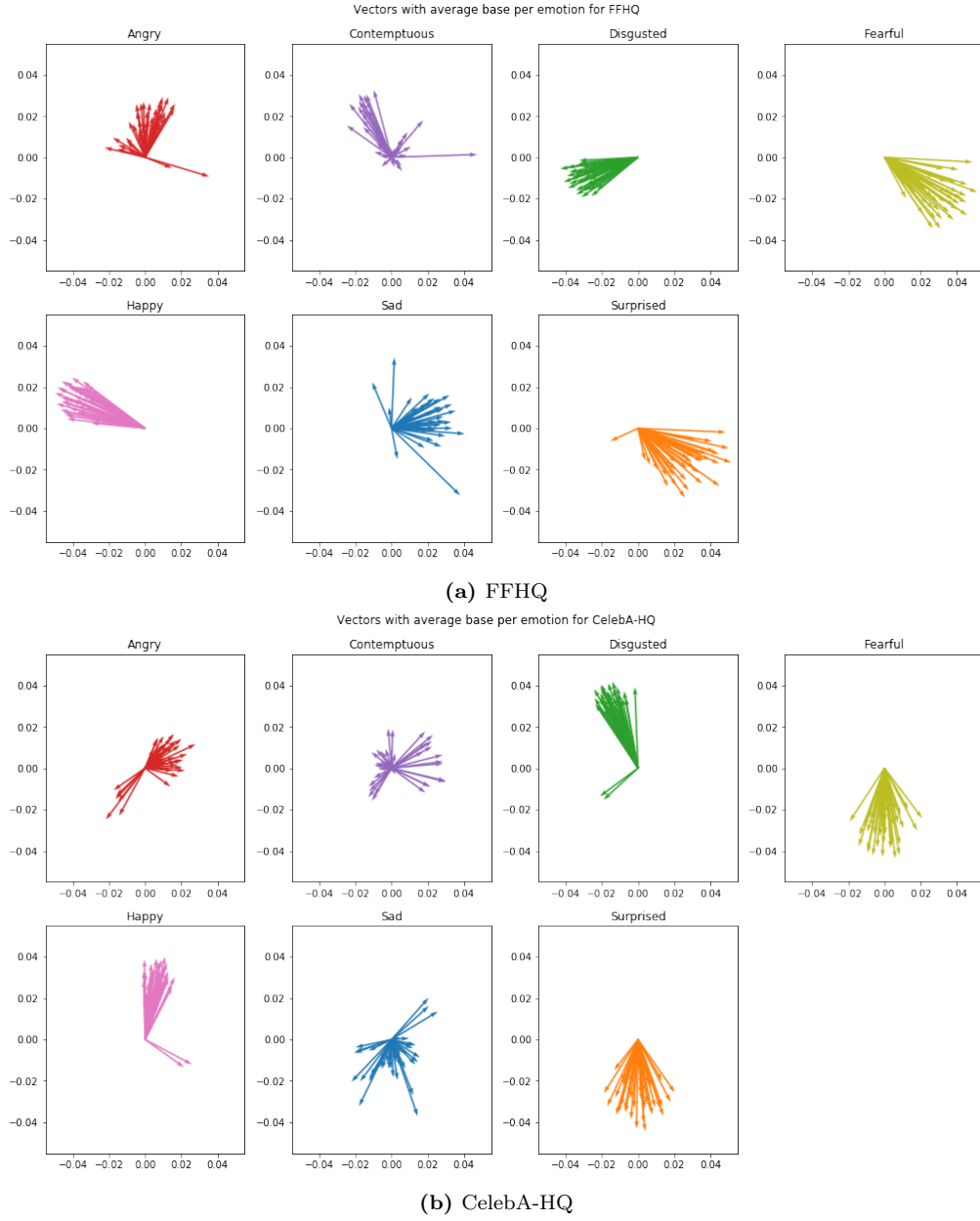


Figure 17: Results for emotion vectors with average as base vector (origin) for styleGAN trained on FFHQ (a) versus CelebA-HQ (b). The RDMs from Figure 13 are visualised in these t-SNE quiver plots, coloured and plotted separately for each emotion.

C Code Listings

C.1 Google Colab

For the complete Google Colab notebook, containing all implemented methods and results, refer to: https://colab.research.google.com/drive/185vviMJ4m0b0e_z9F09q80pKboaMFhUW. This Appendix only contains the most relevant functions.

C.1.1 Data Processing

```
import os
import numpy as np
```

For latent representations found in styleGAN trained on FFHQ:

```
assert len(os.listdir('enc_ffhq')) == 312
identity_space_ffhq = np.zeros((312, 18*512)) # (312, 9216)

i = 0
for enc in sorted(os.listdir('enc_ffhq')):
    array = np.load('enc_ffhq/' + enc)
    identity_space_ffhq[i] = np.concatenate(array, axis=0)
    i+=1

np.save('identity_space_ffhq.npy', identity_space_ffhq)
```

For latent representations found in styleGAN trained on CelebA-HQ:

```
assert len(os.listdir('enc_clba')) == 312
identity_space_clba = np.zeros((312, 18*512)) # (312, 9216)

i = 0
for enc in sorted(os.listdir('enc_clba')):
    array = np.load('enc_clba/' + enc)
    identity_space_clba[i] = np.concatenate(array, axis=0)
    i+=1

np.save('identity_space_clba.npy', identity_space_clba)
```

C.1.2 Sorting Spaces

Spaces sorted on identity (1 - 39):

```
# Since the original results were sorted for identity already,
# this space does not need to be sorted
identity_space_ffhq = np.load('identity_space_ffhq.npy')
identity_space_clba = np.load('identity_space_clba.npy')
```

Spaces sorted on gender (Female, Male):

```
gender_space_ffhq = np.zeros((312, 18*512)) # (312, 9216)
for i in range(19): # Add females
    for j in range(8):
        gender_space_ffhq[8*i + j] = identity_space_ffhq[8*female_index[i] + j]
for i in range(20): # Add males
    for j in range(8):
```

```

gender_space_ffhq[152 + 8*i + j] = identity_space_ffhq[8*male_index[i] + j]

gender_space_clba = np.zeros((312, 18*512)) # (312, 9216)
for i in range(19): # Add females
    for j in range(8):
        gender_space_clba[8*i + j] = identity_space_clba[8*female_index[i] + j]
for i in range(20): # Add males
    for j in range(8):
        gender_space_clba[152 + 8*i + j] = identity_space_clba[8*male_index[i] + j]

```

Space sorted on emotion (Angry, Comtemptuous, Disgusted, Fearful, Happy, Neutral, Sad, Surprised):

```

emotion_space_ffhq = np.zeros((312, 18*512)) # (312, 9216)
count = 0
for i in range(8):
    for j in range(39):
        emotion_space_ffhq[count] = gender_space_ffhq[i + 8*j]
        count+=1

emotion_space_clba = np.zeros((312, 18*512)) # (312, 9216)
count = 0
for i in range(8):
    for j in range(39):
        emotion_space_clba[count] = gender_space_clba[i + 8*j]
        count+=1

```

C.1.3 Significance Functions

```

# Take right top half of rdm (excluding diagonal) and turn into vector
def make_significance_vector( rdm ):
    n = rdm.shape[0]
    l = n-1
    vector_length = int( (np.square(n)-n)/2 )
    vector = np.zeros(vector_length)
    start, end = 0, l
    # Loop through rows
    for i in range(l):
        length = l - i
        vector[start:end] = rdm[i][-length:]
        start = end
        end += length-1
    return vector

# Shuffle hypothesis rdm randomly
def shuffle_rdm( rdm ):
    n = rdm.shape[0]
    idx = np.random.permutation(n)
    new_rdm = np.zeros((n,n))
    for i in range(n):
        for j in range(n):
            if(j > i):
                break

```

```

        else:
            if(i == j):
                new_rdm[i][j] = 0
            if(j < i):
                x = rdm[idx[i]][idx[j]]
                new_rdm[i][j] = x
                new_rdm[j][i] = x
    return new_rdm

def get_random_rhos( hyp_vector, rdm, n ):
    rhos = list()
    for i in range(n):
        vector = make_significance_vector(shuffle_rdm(rdm))
        rho, pval = spearmanr(hyp_vector, vector)
        rhos.append(rho)
    return rhos

```

C.1.4 Emotion Directions

```

def get_space_without_neutral( original_space ):
    # 312 - 39 (all neutrals) = 273
    direction_space = np.zeros((273, 18*512)) # (273, 9216)
    for i in identities:
        x = i*7
        y = i*8
        neutral = original_space[y+5] # 5th index is neutral vector
        direction_space[x] = original_space[y] - neutral # angry
        direction_space[x+1] = original_space[y+1] - neutral # contemptuous
        direction_space[x+2] = original_space[y+2] - neutral # disgusted
        direction_space[x+3] = original_space[y+3] - neutral # fearful
        direction_space[x+4] = original_space[y+4] - neutral # happy
        direction_space[x+5] = original_space[y+6] - neutral # sad
        direction_space[x+6] = original_space[y+7] - neutral # surprised
    return direction_space

def get_space_without_average( original_space ):
    direction_space = np.zeros((273, 18*512)) # (273, 9216)
    for i in range(39):
        x = i*7
        y = i*8
        average = np.average( original_space[y:y+7], axis=0 )
        direction_space[x] = original_space[y] - average # angry
        direction_space[x+1] = original_space[y+1] - average # contemptuous
        direction_space[x+2] = original_space[y+2] - average # disgusted
        direction_space[x+3] = original_space[y+3] - average # fearful
        direction_space[x+4] = original_space[y+4] - average # happy
        direction_space[x+5] = original_space[y+6] - average # sad
        direction_space[x+6] = original_space[y+7] - average # surprised
    return direction_space

def get_sorted_emotion_space( direction_space ):
    direction_space_sorted = np.zeros((273, 18*512)) # (273, 9216)

```

```

count = 0
for i in range(7):
    for j in identities:
        direction_space_sorted[count] = direction_space[i + 7*j]
        count+=1
return direction_space_sorted

```

C.2 Contributions

The contributions to the original Puzer code [16] are indicated between "# -- KB: --" and "# -- end KB --". `encode_images.py` was turned into `encode_images_ffhq.py` and `encode_images_clba.py` to find representations in the latent spaces of styleGAN trained on FFHQ and CelebA-HQ respectively.

`encode_images_ffhq.py`:

```

import os
import argparse
import pickle
from tqdm import tqdm
import PIL.Image
import numpy as np
import dnnlib
import dnnlib.tflib as tflib
import config
from encoder.generator_model import Generator
from encoder.perceptual_model import PerceptualModel

# -- KB: add --
import shutil
# -- end KB --

def split_to_batches(l, n):
    for i in range(0, len(l), n):
        yield l[i:i + n]

def main():
    parser = argparse.ArgumentParser(description=
        'Find latent representation of reference images using perceptual loss')

    # -- KB: add default locations --
    parser.add_argument('--src_dir',
                        help='Directory with images for encoding',
                        default='aligned')
    parser.add_argument('--generated_images_dir',
                        help='Directory for storing generated images',
                        default='gen_ffhq')
    parser.add_argument('--dlatent_dir',
                        help='Directory for storing dlatent representations',
                        default='enc_ffhq')

    # -- end KB --

```



```

# for now it's unclear if larger batch leads to better performance/quality
parser.add_argument('--batch_size', default=1,
                    help='Batch size for generator and perceptual model', type=int)

# Perceptual model params
parser.add_argument('--image_size', default=256,
                    help='Size of images for perceptual model', type=int)
parser.add_argument('--lr', default=1.,
                    help='Learning rate for perceptual model', type=float)
parser.add_argument('--iterations', default=1000,
                    help='Number of optimization steps for each batch', type=int)

# Generator params
parser.add_argument('--randomize_noise', default=False,
                    help='Add noise to dlatents during optimization', type=bool)
args, other_args = parser.parse_known_args()

ref_images = [os.path.join(args.src_dir, x) for x in os.listdir(args.src_dir)]
ref_images = list(filter(os.path.isfile, ref_images))

if len(ref_images) == 0:
    raise Exception('%s is empty' % args.src_dir)

# -- KB: add --
shutil.rmtree(args.generated_images_dir, ignore_errors=True)
shutil.rmtree(args.dlatent_dir, ignore_errors=True)
# -- end KB --

os.makedirs(args.generated_images_dir, exist_ok=True)
os.makedirs(args.dlatent_dir, exist_ok=True)

# Initialize generator and perceptual model
tflib.init_tf()
# -- KB: put in comments --
# with dnnlib.util.open_url(URL_FFHQ, cache_dir=config.cache_dir) as f:
#     generator_network, discriminator_network, Gs_network = pickle.load(f)

# -- KB: add --
# FFHQ
with open('karras2019stylegan-ffhq-1024x1024.pkl', 'rb') as f:
    generator_network, discriminator_network, Gs_network = pickle.load(f)

# CelebA-HQ
# with open('karras2019stylegan-celebahq-1024x1024.pkl', 'rb') as f:
#     generator_network, discriminator_network, Gs_network = pickle.load(f)
# -- end KB --

generator = Generator(Gs_network, args.batch_size,
                    randomize_noise=args.randomize_noise)
perceptual_model = PerceptualModel(args.image_size, layer=9,
                                   batch_size=args.batch_size)
perceptual_model.build_perceptual_model(generator.generated_image)

```

```

# Optimize (only) dlatents by minimizing perceptual loss between
# reference and generated images in feature space
for images_batch in tqdm(split_to_batches(ref_images, args.batch_size),
                           total=len(ref_images)//args.batch_size):
    names = [os.path.splitext(os.path.basename(x))[0] for x in images_batch]

    perceptual_model.set_reference_images(images_batch)
    op = perceptual_model.optimize(generator.dlatent_variable,
                                   iterations=args.iterations,
                                   learning_rate=args.lr)

    pbar = tqdm(op, leave=False, total=args.iterations)
    for loss in pbar:
        pbar.set_description(' '.join(names)+' Loss: %.2f' % loss)
    print(' '.join(names), ' loss:', loss)

# Generate images from found dlatents and save them
generated_images = generator.generate_images()
generated_dlatents = generator.get_dlatents()
for img_array, dlatent, img_name in
    zip(generated_images, generated_dlatents, names):
    img = PIL.Image.fromarray(img_array, 'RGB')
    img.save(os.path.join(args.generated_images_dir, f'{img_name}.png'), 'PNG')
    np.save(os.path.join(args.dlatent_dir, f'{img_name}.npy'), dlatent)

generator.reset_dlatents()

if __name__ == "__main__":
    main()

encode_images_clba.py:

import os
import argparse
import pickle
from tqdm import tqdm
import PIL.Image
import numpy as np
import dnnlib
import dnnlib.tflib as tflib
import config
from encoder.generator_model import Generator
from encoder.perceptual_model import PerceptualModel

# -- KB: add --
import shutil
# -- end KB --

def split_to_batches(l, n):
    for i in range(0, len(l), n):
        yield l[i:i + n]

```

```

def main():
    parser = argparse.ArgumentParser(description=
        'Find latent representation of reference images using perceptual loss')

    # -- KB: add default locations --
    parser.add_argument('--src_dir',
        help='Directory with images for encoding',
        default='aligned')
    parser.add_argument('--generated_images_dir',
        help='Directory for storing generated images',
        default='gen_clba')
    parser.add_argument('--dlatent_dir',
        help='Directory for storing dlatent representations',
        default='enc_clba')

    # -- end KB --

    # for now it's unclear if larger batch leads to better performance/quality
    parser.add_argument('--batch_size', default=1,
        help='Batch size for generator and perceptual model', type=int)

    # Perceptual model params
    parser.add_argument('--image_size', default=256,
        help='Size of images for perceptual model', type=int)
    parser.add_argument('--lr', default=1.,
        help='Learning rate for perceptual model', type=float)
    parser.add_argument('--iterations', default=1000,
        help='Number of optimization steps for each batch', type=int)

    # Generator params
    parser.add_argument('--randomize_noise', default=False,
        help='Add noise to dlatents during optimization', type=bool)
    args, other_args = parser.parse_known_args()

    ref_images = [os.path.join(args.src_dir, x) for x in os.listdir(args.src_dir)]
    ref_images = list(filter(os.path.isfile, ref_images))

    if len(ref_images) == 0:
        raise Exception('%s is empty' % args.src_dir)

    # -- KB: add --
    shutil.rmtree(args.generated_images_dir, ignore_errors=True)
    shutil.rmtree(args.dlatent_dir, ignore_errors=True)
    # -- end KB --

    os.makedirs(args.generated_images_dir, exist_ok=True)
    os.makedirs(args.dlatent_dir, exist_ok=True)

    # Initialize generator and perceptual model
    tflib.init_tf()
    # -- KB: put in comments --
    # with dnnlib.util.open_url(URL_FFHQ, cache_dir=config.cache_dir) as f:

```

```

#     generator_network, discriminator_network, Gs_network = pickle.load(f)

# -- KB: add --
# FFHQ
# with open('karras2019stylegan-ffhq-1024x1024.pkl', 'rb') as f:
#     generator_network, discriminator_network, Gs_network = pickle.load(f)

# CelebA-HQ
with open('karras2019stylegan-celebahq-1024x1024.pkl', 'rb') as f:
    generator_network, discriminator_network, Gs_network = pickle.load(f)
# -- end KB --

generator = Generator(Gs_network, args.batch_size,
                     randomize_noise=args.randomize_noise)
perceptual_model = PerceptualModel(args.image_size, layer=9,
                                   batch_size=args.batch_size)
perceptual_model.build_perceptual_model(generator.generated_image)

# Optimize (only) dlatents by minimizing perceptual loss between
# reference and generated images in feature space
for images_batch in tqdm(split_to_batches(ref_images, args.batch_size),
                        total=len(ref_images)//args.batch_size):
    names = [os.path.splitext(os.path.basename(x))[0] for x in images_batch]

    perceptual_model.set_reference_images(images_batch)
    op = perceptual_model.optimize(generator.dlatent_variable,
                                  iterations=args.iterations,
                                  learning_rate=args.lr)
    pbar = tqdm(op, leave=False, total=args.iterations)
    for loss in pbar:
        pbar.set_description(' '.join(names)+' Loss: %.2f' % loss)
    print(' '.join(names), ' loss:', loss)

# Generate images from found dlatents and save them
generated_images = generator.generate_images()
generated_dlatents = generator.get_dlatents()
for img_array, dlatent, img_name in
    zip(generated_images, generated_dlatents, names):
    img = PIL.Image.fromarray(img_array, 'RGB')
    img.save(os.path.join(args.generated_images_dir, f'{img_name}.png'), 'PNG')
    np.save(os.path.join(args.dlatent_dir, f'{img_name}.npy'), dlatent)

generator.reset_dlatents()

if __name__ == "__main__":
    main()

```