DD2424 Project - Fader Networks

Mauricio Byrd Victorica **Hugo Bellem Westin** **Fredrik Diffner** **Valter Lundegårdh**

Abstract

In this project we aim to replicate the paper "Fader Networks: Manipulating Images by Sliding Attributes", where the authors propose an encoder-decoder architecture with an adversarial component. This model learns to encode pictures into a latent representation invariant to certain attributes. These attributes can then be added during the decoding phase, allowing a user to modify them. Thus the model is able to manipulate the attributes of a given picture, seemingly like *faders* on a mixing table. We achieved a satisfactory downscaled reproduction of the original work, and our results for flipping image attributes under different setups provide meaningful insights into the workings of the model.

1 Introduction

In the original paper, further referred to as "Fader Networks" in the rest of this report, the authors studied an auto-encoder's ability to reconstruct a realistic image, while modifying it with respect to given visual attributes. To solve this task the auto-encoder is coupled with a discriminator in an adversarial fashion: the discriminator aims to predict the attributes of an image given its latent representation, and the auto-encoder attempts to reconstruct images accurately while "fooling" the discriminator. Their results show the proposed model to successfully achieve this objective. Moreover, their method could potentially be extended to other areas such as speech and or text generation. [1]

For our project we decided to replicate the Fader Networks paper to get a better understanding of the components involved, mainly convolutional layers and adversarial loss, while also gaining experience with implementing deep learning models. Overall, both the application and the approach are very interesting. Even with our time and computing power constraints, forcing us to significantly downscale the original setup, our implementation succeeded in capturing the essence of the model.

Although the ability to generate a realistic looking version of an image is perhaps not by itself a world changing phenomenon our work is still important. Firstly, a significant portion of other works tackling the same problem rely on adversarial training in pixel space, but this approach offers a much simpler training pipeline and the authors of Fader Networks claim that their approach scales well to multiple attributes. Secondly, with the current misuse of deepfakes, further study of the creation of deepfakes can also be beneficial in order to counter them [2]. Finally, attribute manipulation is useful for counterfactual tests, which are specially relevant in the topic of fairness within machine learning [3].

2 Related Work

In the Fader Networks paper they divided the related work into three groups depending on the required level of supervision. One group consisted of fully supervised approaches, which during training demands data of both possible outcomes, for example a picture of a person with and without a glasses. This limits the applicability of such approaches, since the required data sets are few and costly to gather. Even with these limitations such approaches have successfully been applied to tasks such as 3D rotation of objects [4], light and pose variation [5], as well as 2D animations [6].

On the opposite side of the spectrum lie fully unsupervised approaches: An example of this is InfoGan [7] which aims to learn disentangled representations of the data. The third group consists of the approaches placed in between unsupervised and fully supervised approaches, and includes the Fader Networks approach. As a baseline for comparison, Fader Networks uses the Invertible Conditional GAN (IcGAN) model proposed by Perarnau et al. in [8]. Perarnau et al. successfully uses an encoder to map images to a high feature space and a cGAN to, from that space, generate images with modified attributes such as gender, hair color and smiling.

The authors of [3] first retrieves pictures of people from web searches, and then uses an architecture based on the Fader Networks to create synthetic versions of these images with variations in the *gender* and *race* attributes. These images are then used to examine how commercial computer vision classifiers are affected by counterfactual changes to these attributes.

Worth mentioning is that the authors of [9] criticise the approach of Fader Networks. They claim that a network with a latent representation invariant to the attributes will have constraint capacity and suffer from information loss since "the relation between the attributes and the face latent representation is highly complex and closely dependent". Instead they apply an attribute classification constraint while the reconstruction learning is encourage to preserve attribute-excluding details.

3 Data

Due to our previously mentioned constraints we were unable to use as much data as the Fader Networks paper. Whereas we extracted 10,000 of the images from the CelebA training set, with their corresponding attributes, the original authors used roughly 160,000. [1, 10]

Regarding the attributes we considered *age* and *gender* to be of extra importance since these showed some of the most promising results in the original paper [1]. To avoid biased data towards one

of the attributes we then made sure our subset of images contained 2500 images of each possible combination of these two attributes. That is, the first quarter contained *young* and *male* images, the second *old* and *male* images, the third *young* and *female* images and the fourth *old* and *female* images. We selected the images by linearly going through all of the images from the start for one of the quarters at a time until we had 2500 images. This new subset was then shuffled according to good machine learning practices to avoid learning a biased representation. After this initial data extraction we then checked the distribution of the remaining attributes in the data set. We saw that the split for the attribute *mouth slightly open*, with 0.494/0.506 between being present and not present, showed great promise so we decided to use that in our training as well.

The collected images were then pre-processed by cropping the original 178x218 images to 178x178 followed by an up-scaling to 256x256 using the OpenCV python library[1]. Finally the RGB image values were scaled to [-1, 1]. Something worth mentioning is that some of the images in the CelebA data set seem to contain a strange machine learning artefact. This could potentially affect learning as the data set contains unrealistic pictures with respect to some attributes. For an example we refer you to Figure 6 in the appendix.

4 Methods

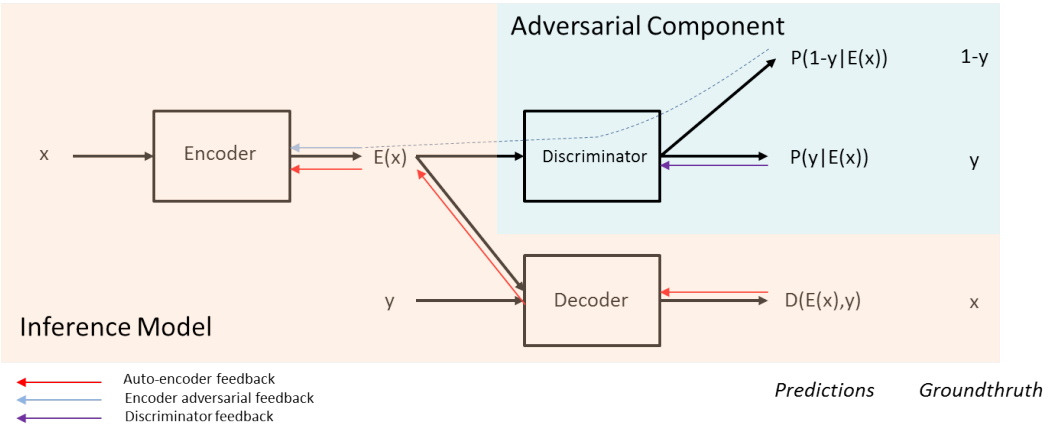


Figure 1: The model architecture from Fader Networks [1]

4.1 General description

For our project we decided to use Tensorflow instead of PyTorch as the authors do [1]. We made this decision to ensure we had a clear conceptual grasp of the architecture we implemented and its components, instead of just copying the original code available online[2]. As a bonus we also wanted to get more acquainted with this machine platform.

A high-level description of the architecture can be seen in Figure 1. There are, fundamentally speaking, three components: the encoder, the decoder and the discriminator, each of which has its own parameters θ_{enc}, θ_{dec} and θ_{dis} respectively. The encoder transforms a given image x at its input into a latent representation $E(x)$. The decoder reconstructs the original image x as $D(E(x), y)$ using $E(x)$ and x's attributes y. Finally, the discriminator attempts to predict the attributes of an image x given its latent representation $E(x)$ as $P(y|E(x))$.

The aim of this model is to learn how to create an encoding $E(x)$ such that it is invariant to the image x's attributes y. Achieving this would mean that the decoder could, subsequently to the encoding, provided with new attributes y', generate a modified image with attributes y' instead of y. Consequently, the discriminator would also be unable to accurately predict the true attributes y from $E(x)$.

[1] https://docs.opencv.org/master/

[2] https://github.com/facebookresearch/FaderNetworks.

However, learning a suitable latent representation which is invariant to the attributes is not enough. One must also ensure that $E(x)$ contains enough information about x so that the decoder can reconstruct the image convincingly, with the only difference being the attributes.

With this knowledge, using the decoder for the reconstruction task and the discriminator for the invariance task the complete adversarial loss for the auto-encoder that should be minimised, given m (image x, attribute y) pairs in a data set $\mathcal{D}$, can be described as:

$$\mathcal{L}(\theta_{enc}, \theta_{dec}|\theta_{dis}) = \underbrace{\frac{1}{m} \sum_{(x,y)\in\mathcal{D}} \parallel D_{\theta_{dec}}(E_{\theta_{enc}}(x), y) - x \parallel_2^2}_{\text{reconstruction error}} - \lambda_E \underbrace{logP_{\theta_{dis}}(1 - y|E_{\theta_{enc}}(x))}_{\text{attribute misclassification}} \quad (1)$$

Here the reconstruction error is the mean squared error (MSE) and the attribute misclassification is the probability of the discriminator predicting the wrong attributes (note it is negative as "fooling" the discriminator lowers the loss from the auto-encoder's perspective). This separates this approach from many other proposed solutions, where the adversarial training is happening in pixel space instead of the latent space [1]. The Fader Networks paper also mentions that MSE is in no way a necessity and other alternatives exist. For the purpose of replication we chose to go with the MSE in our implementation. Furthermore, the variable λ_E refers to the discriminator loss coefficient and controls how much the attribute invariance should be considered compared to the reconstruction error. Initially, $\lambda_E = 0$, but it is linearly increased to 0.0001 over the first 500,000 iterations. This allows the network to primarily focus on learning to encode/decode in the beginning and then slowly learn to trick the discriminator[3].

4.2 Network structure

The encoder contains convolutional layers and the decoder transposed convolutional layers, all with a kernel size of 4x4, a stride of 2, and a padding of 1. This means that for the convolutions the size of the input is divided by two while for the transposed convolutions it is multiplied by 2. The encoder consists of 7 convolutional layers, denoted with C below

$$C_{16} \rightarrow C_{32} \rightarrow C_{64} \rightarrow C_{128} \rightarrow C_{256} \rightarrow C_{512} \rightarrow C_{512}$$

where the subscript specifies the number of filters in a layer. Consequently the original x, a $256 \times 256 \times 3$ image, is transformed into a $2 \times 2 \times 512$ latent representation $E(x)$.

The decoder is a mirrored version of the encoder, with the exception that the attributes y are added as additional constant input channels to each layer. Each attribute is encoded as the concatenation of the two one-hot encoding vectors representing the attributes presence and absence respectively in an image, hence the $2n$ below, where n is the number of attributes. A basic visualisation of the decoder, where a transposed convolutional layer is denoted by the letter D and the concatenation of $2n$ attributes to the transposed convoluted input is denoted by A_{2n}, can therefore be as represented as

$$D_{512} + A_{2n} \rightarrow D_{256} + A_{2n} \rightarrow D_{128} + A_{2n} \rightarrow D_{64} + A_{2n} \rightarrow D_{32} + A_{2n} \rightarrow D_{16} + A_{2n} \rightarrow D_3$$

For more information about the concatenation we refer you to section 7.1 in the appendix.

One thing we considered for the decoding layers was to use convolutional networks for the transposed convolutions. These can after all, as we were taught in lecture 7 as well as described in [11], be seen as normal convolutions provided that the input is modified first. Due to Tensorflow's in-built functions for transposed convolutions we did however decide to not choose this option to avoid any unnecessary hassles.

The last component, the discriminator, is composed of one convolutional layer with 512 filters followed by two fully connected feed forward layers of size 512 and n. Lastly, the activation function in the encoder layers and first discriminator layer is a Leaky-ReLU with a slope of 0.2, while in the decoder layers it is a simple ReLU.

4.3 Training and generation

During training the discriminator attempts to minimise $-P(y|E(x))$, the decoder the MSE using the true image attributes, and the encoder the full adversarial loss specified by expression 1. The

[3] At some point we attempted to accelerate the λ_E schedule aiming to adapt it to our downscaled implementation, but the same schedule used by the authors worked best despite the big differences in our setups. More details are available in the appendix.

discriminator loss and the "attribute misclassification" component for the full adversarial loss are implemented as a categorical cross-entropy each. Despite using binary attributes, a redundant categorical interpretation (i.e. 0 is [1, 0] and 1 is [0, 1]) used in Fader Networks simplifies the implementation of the full architecture. Finally, throughout the training process we use the Adam optimizer and a batch size of 32 for both losses.

Dropout with a probability of 0.3 was also applied to the first convolutional layer in the discriminator, as that improved performance by quite a bit in the Fader Networks paper [1]. For data augmentation the authors also flipped the images horizontally with a 50% probability during training. However, from our initial tests we noticed that this flipping decreased our performance by quite a bit, and thus we dropped it in our implementation.

After training one can disregard the discriminator and only use the auto-encoder architecture to generate a modified image by providing the encoder with an image and the decoder with different attributes (which may be the true ones or a modified version of these).

5 Experiments

Our goal was to reproduce the Fader Networks paper in a downscaled setup, aiming for reasonable results despite our limitations. In the original study they trained with approximately 160,000 pictures for 1000 epochs but to match our computational resources and time frame we opted to use 10,000 images in our final model, which we managed to train for 977 epochs.

First we performed preliminary experiments, where we trained our model using different amounts of data, schedules for the parameter λ_E, attributes and combinations of them to decide upon the setup for our main model. More details can be found in the appendix in section 7.2.

We decided on a model using gender as its sole attribute, which we trained as described in the beginning of this section. We inputted images from the train and test set to see how well the model reconstructs the pictures and modifies their attributes. After training it is possible to reconstruct images with arbitrary attributes using the trained model, allowing us to subjectively evaluate model performance based on how they look, especially in terms of attribute manipulation.

Based on our visual evaluation, our setting does not yield the full expressibility in comparisons with the Fader Networks paper as can be seen Figure 7 in the appendix. But even with less data and training, our trained model for the *gender* attribute managed to reconstruct images and also change key components in the image indicating the gender of the person, even though the quality varies among both training and test samples.

When evaluating the training checkpoints of our definitive model, we observed that its ability to change the gender attribute for reconstructed images seemed to improve for some 700 epochs before starting to overfit with respect to the training data. Results after 760 epochs, covering "gender fades" from female to male and male to female in training and test images are shown in Figures 3 and 4, respectively. The original input image for each case can be found in Figure 2.

(a) **Original Training Images.** (b) **Original Test Images.**

Figure 2: Original training and test images for the "gender fades" presented in Figures 3 and 4.

Our results show that our model was successful in achieving the desired fader-like behaviour with respect to the gender attribute, even on images unseen during training. Note however, that the performance of the model for the samples in Figures 3 and 4 are not the norm, and for our trained

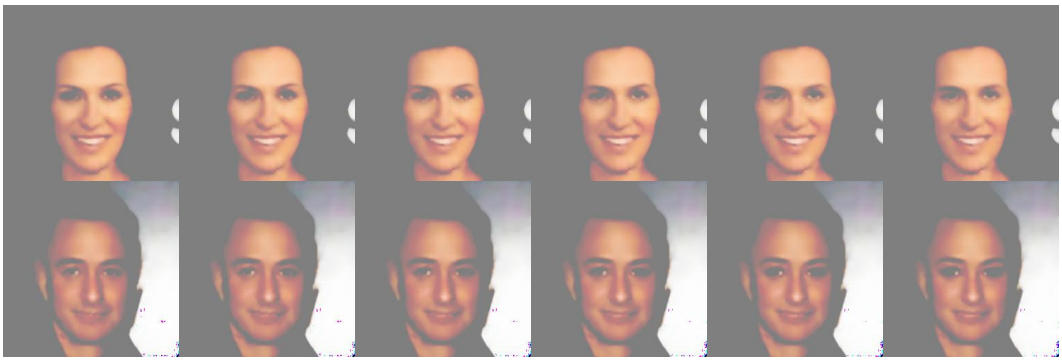


Figure 3: Output of a model trained with respect to the *gender* attribute for 760 epochs. The two original images are from the training set, and the figure shows a linear change in their respective *gender* attribute.

Figure 4: Output of a model trained with respect to the *gender* attribute for 760 epochs. The two original images are from the test set, and the figure shows a linear change in their respective *gender* attribute.

model we could find multiple training and test images with poor reconstructions or slighter gender-fading effects.

Additionally, despite showing a nice "gender fade" for the reconstructed images, these may vary in accuracy. For example, the originally male training image in the right side of Figure 2(a) is not very similar to its reconstruction using original attributes in Figure 3.

The adversarial loss for the auto-encoder and the loss for the discriminator across epochs are shown in Figure 5. We concluded that the evolution of the losses reasonably reflects the relationship between auto-encoder and discriminator: The auto-encoder performs increasingly better with some oscillations, while the discriminator's performance fluctuates constantly as the auto-encoder continuously modifies the latent representation fed into the discriminator.

Under our simple setup it is hard to make strong claims, but in different points in the plot, specially in later epochs, peaks in the auto-encoder adversarial loss seem to occur during valleys in the discriminator loss. Also, it should be noted that the auto-encoder loss is strongly dominated by the reconstruction error, since the error related to the discriminator is weighted by λ_E, which increases linearly from 0 to 0.0001 over 500,000 update steps. For our setup and batch size this implies λ_E never reaches its final value, as we only have a bit more than 300,000 update steps; more details about this schedule are available in the appendix.

6 Conclusion

In this work, we have, to some extent, managed to reproduce the results from the original Fader Networks paper [1]. Despite heavily downscaling on data and training time, our model managed to reconstruct pictures and change visual components linked to gender in a fader-like manner. Hence, despite some of the short-comings of our implementation mentioned in the previous section, we consider that we achieved the objective set for this project. We believe that to further improve the

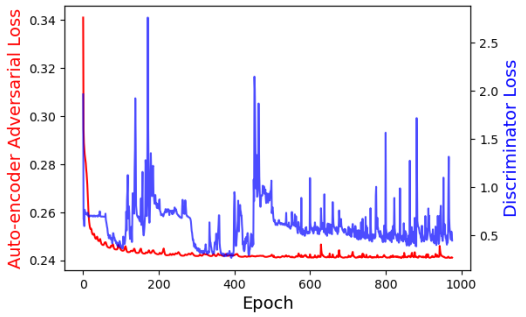


Figure 5: Adversarial auto-encoder loss and the discriminator loss (notice the different scales) for a model trained with respect to the *gender* attribute for 977 epochs. The plots show how the auto-encoder loss decreases and flattens out, while the discriminator loss fluctuates throughout training as the latent representation at its input is continuously modified by the auto-encoder.

model and increase its expressibility more training data is needed, and consequently longer training times.

Moreover, our results from the preliminary experiments (covered in the appendix) confirmed the big impact of the λ-scheme used during training, as mentioned in the Fader Networks paper, and a λ-scheme properly adjusted to our setup would probably have a positive impact on the results.

We also found that in our setup, it was significantly harder to change attributes (even single ones) when incorporating multiple attributes during training, i.e. building models able to flip more than one attribute at once, which contradicts the original paper's claim that the approach scales well for multiple attributes or at the very least suggests this claim may not hold in lower-data regimes.

6.1 Future work

As mentioned in our conclusions, we found that under our lower-data setting, extending the architecture to a multi-attribute case was challenging. It would be interesting to explore what adjustments (if any) could be applied to this architecture to get better performances in the multi-attribute case for smaller data sets.

The schedule for increasing λ_E is another crucial aspect of this approach which we briefly explored (see the appendix for details) and which is not covered in depth in the original paper. It could be useful to make a more rigorous study to determine how one can find an adequate schedule for a given setting or whether a single schedule performs equally well across different settings, and also determine the relevant exemptions (if there are any).

Finally, there is plenty of potential future work in terms of applications. In Fader Networks it is mentioned that this approach could also be applied in different areas such as text or speech [1]. We envisage dialect translation as an interesting potential application: One could have their speech transformed into another dialect within the same language using a similar model architecture as used in our study. Adapting the Fader Network to that setting could be interesting both from a linguistics and from a natural language processing (NLP) standpoint.

Acknowledgements

We would like to thank our supervisor Lennart Alexander Van der Goten who recommended us the Fader Networks project and guided us during the initial parts of the project. We wish you the best of luck with the rest of your PhD.

References

[1] Guillaume Lample et al. *Fader Networks: Manipulating Images by Sliding Attributes*. 2018. arXiv: 1706.00409 [cs.CV].

[2] *What are deepfakes – and how can you spot them?* Accessed: 2021-05-19. 2020. URL: https://www.theguardian.com/technology/2020/jan/13/what-are-deepfakes-and-how-can-you-spot-them.

[3] Jungseock Joo and Kimmo Kärkkäinen. "Gender Slopes: Counterfactual Fairness for Computer Vision Models by Attribute Manipulation". In: *CoRR* abs/2005.10430 (2020). arXiv: 2005.10430. URL: https://arxiv.org/abs/2005.10430.

[4] Jimei Yang et al. “Weakly-supervised Disentangling with Recurrent Transformations for 3D View Synthesis”. eng. In: (2016).

[5] Tejas D. Kulkarni et al. “Deep Convolutional Inverse Graphics Network”. In: *CoRR* abs/1503.03167 (2015). arXiv: 1503.03167. URL: http://arxiv.org/abs/1503.03167.

[6] Scott E Reed et al. “Deep Visual Analogy-Making”. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/e07413354875be01a996dc560274708e-Paper.pdf.

[7] Xi Chen et al. “InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets”. eng. In: (2016).

[8] Guim Perarnau et al. “Invertible Conditional GANs for image editing”. eng. In: (2016).

[9] Zhenliang He et al. “AttGAN: Facial Attribute Editing by Only Changing What You Want”. In: *IEEE Transactions on Image Processing* 28.11 (2019), pp. 5464–5478. DOI: 10.1109/TIP.2019.2916751.

[10] Ziwei Liu et al. “Deep Learning Face Attributes in the Wild”. In: *Proceedings of International Conference on Computer Vision (ICCV)*. Dec. 2015.

[11] Vincent Dumoulin and Francesco Visin. *A guide to convolution arithmetic for deep learning*. 2018. arXiv: 1603.07285 [stat.ML].

7 Appendix

7.1 Concatenation

The A_{2n} does for each n contain two matrices with the same number of rows and columns as the transposed convoluted data with the first one containing only ones and the second only zeroes or the opposite depending on the attribute value. That is, if attribute vector y is true for an attribute n then the first matrix for that attribute will contain ones and the second one zeroes while if it is false it will be the opposite. This makes it so that the input to the second layer will have size 4x4x514.

Figure 6: A distorted image of Anderson Cooper from the CelebA data set

Figure 7: The swap quality achieved by the original paper.[1]

7.2 Preliminary Experiments

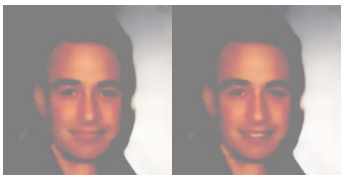
7.2.1 Choosing between single and multiple attributes

Our initial experiments indicated that for our setup, we obtain better results when training a model with only one changeable attribute versus multiple changeable attributes. This can be seen by comparing Figure 8 and Figure 9. An interpretation of this is that our model loses expressibility when we increase the number of changeable attributes. This contradicted the Fader Networks paper's claim that this approach scales well with the number of attributes, and strengthens the claim made by [9] that a latent representation invariant to one or more attributes reduces the model's expressibility.

However, our aim was merely determining the definitive model we would train, and more rigorous experiments must be performed to draw any conclusions regarding this.

Figure 8: Output from a model trained with respect to the attributes *gender*, *age*, *smiling*, *mouth open/close*, *lipstick*. The leftmost picture is the reconstructed picture without any changes of the attributes, the following pictures have one attribute swapped corresponding to the order in which they were just mentioned, and the rightmost picture has all the attributes swapped simultaneously. Here it is hard to distinguish any differences between the pictures.

(a) **Gender.** The biggest difference could probably be seen around the eyes, where the rightmost picture seems to have more makeup.



(b) **Mouth.** The model works well since the mouth clearly opens.

(c) **Age.** In this model we could not see any difference between the two outputs.

Figure 9: Each picture is an output of a model trained with respect to that attribute for 200 epochs. Each picture comes from the training data and the left image in each case is the reconstructed picture without changing the attribute, while the right image is a reconstructed version but with the corresponding attribute shifted.

7.2.2 Choosing among single attributes

Figure 9(a) shows the promising output of a model trained for 200 epochs when shifting the *gender* attribute of an image from the training data. We also saw successful results when training a model with respect to the attribute *mouth opening*, which can be seen in Figure 9(b).

However, when training a model with respect to the *age* attribute for the same number of epochs and on the same data, we did not see any significant changes when swapping this attribute. This is probably due to the low reconstruction quality in the result, and the outcome would probably be different with a bigger training set and more training. Figure 9(c) shows the result from this experiment. We chose gender and not mouth opening because good results for shifting the latter were extremely rare, while noticeable gender shifts we not uncommon in the data set.

7.2.3 Choosing lambda schedule

In Fader Networks, the authors claim that the λ-schedule has a big impact on the results, which also seems to be strengthened by our experiments. Since we train on a subset of the data and for a fewer number of epochs, the λ-schedule used in the Fader Networks paper does probably have a different impact on our model compared to the model in the Fader Networks paper.

When adjusting the λ-scheme based on the size of our training set, the model seemed to produce a sharper reconstruction. Unfortunately, it did not manage to swap attributes as well as when we used the same λ-schedule as the Fader Networks paper. This could be seen when comparing Figure 9(a) and 10. An additional interesting detail in Figure 10, is that the reconstructed image seems to have some unintentional gender shift, which suggests that the faster increase to λ_E caused this effect by constraining the auto-encoder to fool the discriminator more sharply.

Due to restrictions in time and computing resources, we did not perform any exhaustive search for a λ-schedule beneficial for our model and decided to use the same used in Fader Networks for our experiments.

7.2.4 Choosing the amount of training data

To determine whether using more training data was feasible and beneficial enough, we trained a 5 attribute and a 2 attribute model using both 10,000 and 20,000 training images in each case. The 2 attribute model was for age and gender, while the 5 attribute one included also "smiling", "mouth slightly open" and "lipstick" on top of the first two. Our aim was to determine if using more training data could help us tame the difficulties we found in the multi-attribute case. The results are shown in Figures 12 and 11.

As shown in the figures there are no relevant changes in terms of performance even though the 20,000 image cases have a considerably larger amount of steps. There is noticeable change between the rows of Figure 11, but not in the quality of attribute manipulation, these are just due to our use of Leaky ReLU activations instead of regular ReLUs for our tests with the two attribute model with 10,000 images, and this is the case. We thus concluded that the best course of action would be to train on the 10,000 data set, also taking into account our time constraints for the project. Had we seen a significant improvement, we would have considered employing more data and hence a longer training time.

Figure 10: Output of a model trained with respect to the *gender* attribute for 200 epochs, and with a λ-scheme adjusted to the size of our training data. When comparing to Figure 9(a) it seems that the model produce a sharper reconstructions, but does not manage to shift components in the picture related to the *gender* attribute as well.

(a) **10,000 training images.**.

(b) **20,000 training images.**

Figure 11: Outputs from models trained with respect to the *gender* and *age* attributes using 10,000 and 20,000 training images. Training was performed for 200 and 153 epochs, respectively. The leftmost picture is the reconstructed picture without any changes of the attributes, while the following pictures have one attribute swapped, corresponding to the order in which they were just mentioned, and the rightmost picture has both attributes swapped.

(a) **10,000 training images.**.

(b) **20,000 training images.**

Figure 12: Outputs from models trained with respect to the attributes *gender*, *age*, *smiling*, *mouth open/close*, *lipstick* for 200 epochs using 10,000 and 20,000 training images. The leftmost picture is the reconstructed picture without any changes of the attributes, while the following pictures have one attribute swapped, corresponding to the order in which they were just mentioned, and the rightmost picture has all of them swapped.