A Re-Implementation of "VOGUE: Try-on by StyleGAN Interpolation Optimization"

Project Report for the Course 'Implementing ANNs with Tensorflow'
(Winter Semester 2020/21)
University of Osnabrück

Hermann Singer, Sarah Neuhoff and Sascha Senko

April 30, 2021

1 Task Description

We decided to re-implement the paper "VOGUE: Try-On by StyleGAN Interpolation Optimization" by K.M. Lewis et al. which was published in Januray 2021 [1]. In this paper, they use linear interpolation on the latent space of a StyleGAN2 [2] to fit garments onto people. To be more specific they are given an image of a target person, I_p, and another image of a person wearing a garment, I_g, and want to generate an image I_t that displays the person from I_p wearing the garment from I_g.

2 Related Work

To understand how StyleGAN2 works it is important to first explain its precursor StyleGAN [3]. StyleGAN is a GAN architecture that learns to encode the style of an image, that means the high-level features of the image like posture, identity, color etc., automatically and in an unsupervised way that does not need any human control, which was still necessary in some style transfer techniques before StyleGAN. The network also enables the user to have some control over the style.

The main reason for the success of StyleGAN lies in the generator. StyleGAN does not generate an image directly from the latent code, z, which is sampled from a probability distribution like the normal distribution. Instead, z first gets mapped to an intermediate latent code w, which should be less disentangled, as w doesn't have to follow the constraint of coming from a probability distribution as z anymore. A code is called disentangled if the change in the value of only one dimension of the code leads to the change of only one semantic unit (like nose, eye, etc.). In contrast, if a code is entangled, changing one dimension of the vector leads to changes in multiple semantic units.

For each style layer l, w again gets fed into a simple network A_l which outputs σ_l. The style layers consist of an affine transformation and an adaptive instance normalization

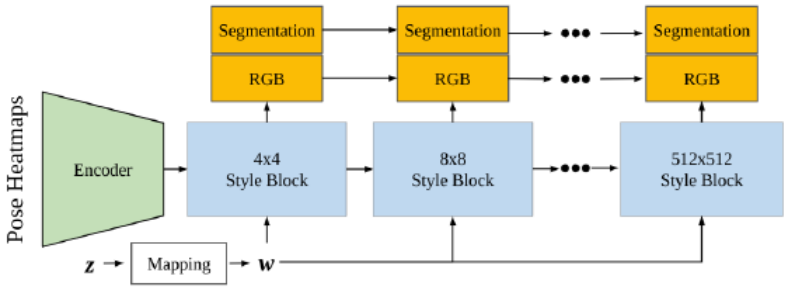

Figure 1: Overview of the StyleGAN2 that the authors modified and trained for VOGUE

AdaIN. Another difference to vanilla GANs is that noise here is added throughout the whole network and not just at the beginning. Also, the network gets a learned constant as the very initial input instead of a random one.

StyleGAN2 is basically just an improved version of the StyleGAN architecture. One improvement consists of fixing artifacts that were present in the images of StyleGAN.

Another important paper for this project was "Editing in Style: Uncovering the Local Semantics of GANs" [4]. The researchers propose a method for making local edits to the output image that consider semantic units and by this transferring localized styles. This relies on the emergent disentanglement that is learned by StyleGAN and has the advantage of not requiring spatial reasoning. The semantic objects are disentangled in a deep feature space of the generator's layers. A batch of GAN images provides many samples from this space. On the activation tensors of these samples they then perform k-means clustering. We assume that with activation tensors they mean the tensors after feeding them through an activation function like ReLU. The resulting clusters align well with the semantic objects. Therefore only features that overlap well with the cluster membership mask are transferred from the reference to the target.

3 Model

In this section we describe the original model of the VOGUE paper that the authors proposed. VOGUE wants to perform a virtual try-on with different clothes. The idea of the VOGUE model is to take an input pair of images which consists of an image of a person, I_p, and an image with another person with a different garment on, I_g. The goal is to project the garment on a new image with the first person, I_t.

The algorithm used in the VOGUE paper is a kind of upgraded version of the one used in [4].

In terms of StyleGAN the try-on image is projected somewhere in the latent space of the generator. More precisely, this is an interpolation:

$$\sigma_t = \sigma_p + Q(\sigma_g - \sigma_p) \tag{1}$$

σ_t, σ_p and σ_g are scaling coefficients per layer of the try-on image, the person image and the garment image, respectively.

The goal then is to learn the diagonal matrix Q, where the diagonal entries are given by a vector q, where every entry of q is in the range $[0, 1]$ by using a sigmoid function. [1] then borrows from [4] a greedy algorithm to train Q such that we have optimal changes in the garment segment of the image but not in the other, neutral parts of the image.

The VOGUE model is based on StyleGAN2. However, some modifications are applied in the generator. Because the model uses two input images, we need to map two latent codes z_p^+ and z_g^+ on the intermediate latent codes w_g^+ and w_p^+. The σ_p, σ_g that we get from them are fed into the interpolation block before every style block like in equation (1). Another difference to StyleGAN is that instead of using a constant input they use a pose heatmap as input such that in the output image the pose of the target person is adopted as well.

The whole StyleGAN inspired generator is trained with the following loss function consisting of three seperate loss functions, each of them scaled with a coefficient α:

$$L = \alpha_1 L_{localization} + \alpha_2 L_{garment} + \alpha_3 L_{identity} \tag{2}$$

In the following we will explain the different parts of the loss function:

The Editing-localization loss: Since we want the model to only try-on a garment and therefore change the image only at the region of interest where the garment is, the loss function includes a localization loss. In this loss term the spatial overlap between the segmentations of the image and the activation tensors is measured.

The Garment Loss is used to minimize the perceptual distance between the garment part of the original garment image and the try-on image. To achieve this the segmentation labels help to create binary masks. The masked images then are blurred with a gaussian filter and downsampled to 256x256. The perceptual distance between the two images then is computed via VGG-16 features.

The Identity Loss is computed similarly as the garment loss, but instead makes sure that the identity of the person is retained.

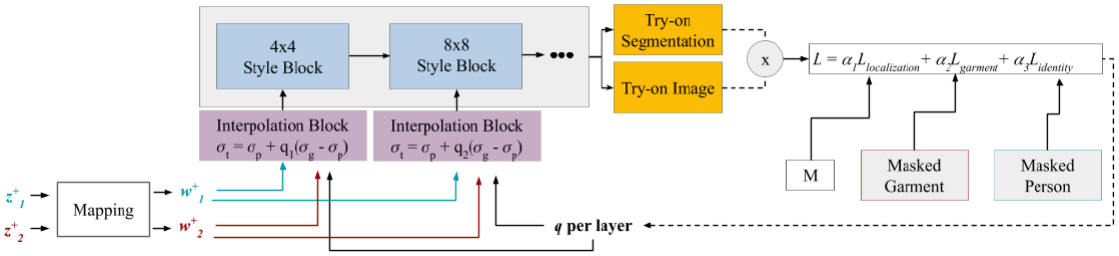

Figure 2: The overview of the generator of VOGUE

4 Re-Implementation

4.1 Dataset

We asked the authors of the VOGUE paper for their dataset by messaging them, as their dataset was not to be found online. As we did not receive an answer, we at first we searched the internet for some other clothing dataset and a StyleGAN2 pre-trained on clothing, because training one ourselves would have taken too long. Unfortunately we did not find such an algorithm and furthermore we would have needed a dataset that included semantic segmentation, which we also did not find.

So instead we used the FFHQ aging dataset [5], where the training data are faces of people of different ages. It was actually designed for benchmarking age-transformation algorithms, but it contains semantic segmentation and there also exists a pretrained StyleGAN2 for it. Our goal was now to train the StyleGAN2 in a way such that it does something similar to the one in the VOGUE paper. Just instead of clothes the network in our case fits a body part of one person to another.

In our approach to re-implement VOGUE the first step was to find a suitable version of StyleGAN2. The original authors of StyleGAN2 published their code in TensorFlow version 1.14. Since our education mainly focused on TensorFlow 2, we picked a TensorFlow 2 reimplementation. First, we wanted to use the version from [6], which had the highest ranking on Github. However, this turned out to be difficult since TensorFlow 2.0 supports a different CUDA version. Thus, we used [7].

As explained earlier, StyleGAN2 does not sample directly from a latent space but uses an encoder for intermediate latent codes. This requires the training of the encoder. Initially, we wanted to use the DenseNet from sample solution of homework 6 [8]. However, we ran out of GPU memory on Google Colab, we decided for a simpler model (see [9] under "Instantiate Encoder" for the architecture).

The training process of this simplified encoder still took a long time: 3500 images in 45 minutes. Google Colab's limited computation capacities led to a mode collapse of the encoder. For details see the outputs below "Check Encoder outputs" of [9] - the first row shows the original images, the second row the reconstructed one).

In [1], the loss function computes the perceptual distance with VGG-16 layers. However, as mentioned before we ran out of GPU memory on Google Colab. Thus, we picked the smaller but still well-performing EfficientNet from Keras [10]. To calculate the perceptual loss, we used this equation from [11, p. 6]:

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \|w_l \odot (\hat{y}^l_{hw} - \hat{y}^l_{0hw})\|_2^2 \tag{3}$$

As an optimizer we used Adam with the default learning rate of 0.001, a batch_size of 2 and prefetching.

5 Project workflow

5.1 Understanding relevant papers

We started working on the project on the 25.02.. The plan was to understand the papers [1]–[3] until 05.03.. Sadly, none of us were able to understand the papers in sufficient detail by then and we also realized that the paper [4] is also of relevance. On 09.03., we sent an e-mail to all three authors of [1] asking if we could get access to the dataset, but sadly never got a response.

5.2 Writing the paper

Sarah and Sascha wrote their parts for the paper between 26.03. and 05.04.. Hermann wrote for the paper between 26.04. and 30.04..

5.3 Writing the code

Writing the code was done entirely by Hermann. On 10.03., we realized that [12] is actually implemented in Tensorflow 1.14 and 1.15, so not compatible with Tensorflow 2. As such, we needed to look for another implementation. We first tried using [6] instead, but as Tensorflow 2.0 also isn't compatible with Tensorflow 2.4, we finally settled on [6]. Actually writing code for real started around 18.03. (before then, only code to understand [6], [7], [12] and [1]–[4] were written). On 20.03., the encoder that was supposed to map images to the latent space such that StyleGAN2 would reconstruct the original image got started in [9]. [9] got finished around 29.03., albeit with an Encoder that suffers from mode collapse. A little bit before 29.03., [13] also got started, but sadly never finished.

6 Reason for failure

The reason for why we failed to complete our project has multiple reasons: We all were busy with other course(s) during the break besides IANNwTF, and Sascha and Sarah were also overwhelmed by the difficulty of both the papers and the code writing. Next to a seminar, Hermann also had to deal with bureaucratic complications between RISE and a scholarship from Vienna that started at the end of February and finally successfully got resolved on April the 30th, the date of the deadline for it (which may have very well resulted in a total loss of over 4000€ and having to do a 90 day internship if not resolved). There also were two deaths of family members during the project time, apartment hunting and furniture purchasing for moving in together with his girlfriend and two weeks of lab rotation work. With regards to the project itself, there were also some problems, though probably most groups had to deal with problems similar to these: Trying to resolve in a clean way OOM errors for the models, which occured if the batch size exceeded ~ 2 even without shuffling; Trouble finding a pre-trained face segmentation model that is compatible with our tensorflow version (Hermann still hasn't found one and training one ourselves would take too much time and also go over what is reasonable

for this project.); Switching the code base for the StyleGAN2 always cost additional time since Hermann needed to familiarize himself with each of them anew.

Bibliography

[1] Kathleen M Lewis, Srivatsan Varadharajan, and Ira Kemelmacher-Shlizerman. "VOGUE: Try-On by StyleGAN Interpolation Optimization" (2021). arXiv: 2101.02285v1. Retrieved from https://arxiv.org/abs/2101.02285.

[2] Tero Karras, Samuli Laine, Miika Aittala, et al. "Analyzing and Improving the Image Quality of StyleGAN". *arXiv:1912.04958 [cs, eess, stat]* (Mar. 2020). Retrieved from http://arxiv.org/abs/1912.04958. Last retrieved: 03/29/2021.

[3] Tero Karras, Samuli Laine, and Timo Aila. "A Style-Based Generator Architecture for Generative Adversarial Networks". *arXiv:1812.04948 [cs, stat]* (Mar. 2019). Retrieved from http://arxiv.org/abs/1812.04948. Last retrieved: 03/29/2021.

[4] Edo Collins, Raja Bala, Bob Price, and Sabine Süsstrunk. "Editing in Style: Uncovering the Local Semantics of GANs". *arXiv:2004.14367 [cs]* (May 2020). Retrieved from http://arxiv.org/abs/2004.14367. Last retrieved: 03/29/2021.

[5] Roy Or-El, Soumyadip Sengupta, Ohad Fried, Eli Shechtman, and Ira Kemelmacher-Shlizerman. *FFHQ-Aging Dataset*. 2020. Retrieved from https://github.com/royorel/FFHQ-Aging-Dataset.

[6] manicman1999. *StyleGAN2-Tensorflow-2.0 by manicman1999*. 2019. Retrieved from https://github.com/manicman1999/StyleGAN2-Tensorflow-2.0.

[7] cryu854. *StyleGAN2 by cryu854*. 2020. Retrieved from https://github.com/cryu854/StyleGAN2.

[8] Gerrit Bartels (Group 65). *Sample Solution of homework 6 of the course 'Implementing ANNs with Tensorflow' at University of Osnabrück*. Winter semester 2020/21. 2020. Retrieved from https://github.com/GerritBartels/IANNWTF/tree/main/Homework06.

[9] Hermann Singer, Sarah Neuhoff, and Sascha Senko. *Encoder notebook*. 2021. Retrieved from https://github.com/HSinger04/VOGUE-Reimplementation/blob/main/cryu854/Image2Latent.ipynb.

[10] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". *arXiv:1905.11946 [cs, stat]* (Sept. 2020). Retrieved from http://arxiv.org/abs/1905.11946. Last retrieved: 04/04/2021.

[11] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric". *arXiv:1801.03924 [cs]* (Apr. 2018). Retrieved from http://arxiv.org/abs/1801.03924. Last retrieved: 04/04/2021.

[12] Tero Karras, Samuli Laine, Miika Aittala, et al. *Official StyleGAN2 Implementation*. 2019. Retrieved from https://github.com/NVlabs/stylegan2.

[13] Hermann Singer, Sarah Neuhoff, and Sascha Senko. *TryOn notebook*. 2021. Retrieved from https://github.com/HSinger04/VOGUE-Reimplementation/blob/main/cryu854/TryOn.ipynb.