High Resolution Neural Frontal Face Synthesis from Face Encodings using Adversarial Loss

by

Andy Wang

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2019

Author .....
Department of Electrical Engineering and Computer Science
May 24, 2019

Certified by .....
Wojciech Matusik
Associate Professor
Thesis Supervisor

Accepted by .....
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

High Resolution Neural Frontal Face Synthesis from Face Encodings using Adversarial Loss

by

Andy Wang

Submitted to the Department of Electrical Engineering and Computer Science
on May 24, 2019, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In this thesis, we present a novel neural network method to synthesize a person's face imagery with frontal face and neutral expression, given a single unconstrained face photograph. We achieve this by a data-driven approach to train neural networks with a large-scale in-the-wild dataset of face images. The most common way to tackle this is supervised learning, which requires many ground-truth input-output pairs. Moreover, in our problem context, finding clean frontal and neutral expression faces without occlusions leads to other challenging problems.

To avoid this, we take a neural knowledge transfer approach, where we first train modular networks for each well-defined sub-task and exploit them to instill semantic senses to train the face decoder, i.e., neutral face synthesizer. For sub-tasks, we utilize face landmark detection and recognition modules, where curated datasets exist. In particular, the face recognition sub-task learns features strongly invariant to lighting, pose, and facial expression variations. Given the recognition feature, we leverage this invariance to train our face decoder to produce consistent frontal and neutral expression faces, while constraining each generated face: 1) to be a forward facing pose using the network trained for the landmark detection, and 2) to preserve the same identity as the input face using the network trained for face recognition. Furthermore, we attempt to boost the realism of the output faces using adversarial loss, in which a discriminator competes with the generator network and guides the generation of higher quality faces. In test time, only the face recognition network and face decoder are used to synthesize neutral faces.

Our approach does not require supervised data and further minimizes sensitive data pre-processing pipelines. Compared to competing fully-supervised methods, our method produces comparable or often even favorable face appearances.

Thesis Supervisor: Wojciech Matusik
Title: Associate Professor

Contents

List of Figures

Acknowledgements

First and foremost, I would like to thank Wojciech Matusik for accepting me into the Computational Fabrication Group as an MEng student and putting me in contact with Tae-hyun Oh and Changil Kim. He gave me the opportunity to work on this fantastic project and meet my incredible lab partners while teaching the advanced graphics and fabrication class at MIT. I am also grateful for Wojciech's help and comments during my thesis work, and I hope the lab continues to thrive and continue fabricating new inventions.

Tae-Hyun Oh has given me an incredible amount of patience and guidance during the project and I could not have made it this far without him. To facetiously use neural network terminology, he took a neural knowledge transfer approach and instilled knowledge of neural network design for my own personal training and growth. The Tensorflow and training advice I received from Tae-Hyun helped me greatly during my thesis. Additionally, he helped set me up for attending lab meetings, viewing the group calendar, and working at my own desk in the lab space at the beginning of the year. These actions may seem small, but especially at a time of high anxiety for a new Master's student like myself, I am incredibly grateful that he could take such kind measures and more that proved invaluable for my own confidence. I wish the best for Tae-Hyun at his current job at Facebook AI Research and his upcoming time in Korea.

Changil Kim has provided me invaluable guidance especially at time of need. I want to thank Changil for helping me get started on my thesis work and providing me with interesting papers to read on face encoders and past work on face frontalization. These formed a basis for my understanding of the scope of my work, and for that I am very grateful. He has provided invaluable advice on my thesis work that I will take to heart for future scientific work. I wish the best for Changil in his future endeavors, and I hope to try his recommendation for

Porter Square pies soon.

I would also like to thank Allan Zhao for being my lab partner for the advanced fabrication class and also helping me in understanding more technical aspects of Tensorflow and machine learning. In a chance encounter where we discovered we were in the same classes, I would never have imagined that he would end up joining the same lab as myself and that we would continue to converse in the same lab-space. All the best for Allan and his newly begun PhD program and life in Boston.

I also thank the other members of the Computational Fabrication Group who were here at the same time as myself: Beichen Li, Harrison Wang, Liane Makatura, Andrew Spielberg, Liang Shi, Jie Xu, Yiyue Luo, Timothy Erps, Mike Foshey, Tao Du, James Minor, Wan Shou, Petr Kellnhofer, and Alex Kaspar. I have learned so much from listening to all their presentations at lab meetings, and I have felt so welcomed by these lab-mates. It is a rare and special event in my opinion for all the lab-mates to go and enjoy lobster on the off-hand recommendation of an MEng student. It has been a pleasure.

Finally, I want to thank my family and my mom especially for supporting me throughout my undergrad and MEng program. Their love helped me get through my five years at MIT, and we have grown so much together despite living on opposite coasts.

Chapter 1

Introduction

Figure 1-1: Examples of face frontalization achieved by our method on a set of 4 input faces. The left two columns image are our input images. The right two columns are our outputs.

There are many face based applications that people are interested in, such as face detection, verification, recognition, and sentiment analysis, each of which can be used in various fields such as criminal justice, phone technology, avatar creation, digital art, etc. The reason is that out of the five human senses: sight, sound, touch, smell, and taste, visual signals by facial appearance form a dominant part of human social perception. For instance, existing research shows that deep learning can strongly predict subjective measures of trustworthiness solely from facial profiles [12].

In most face-based applications, the technical challenges come from variations in head

pose and facial expression, as well as real world perturbations such as accessories like makeup, varying light conditions, blurry image quality, and so on. Consequently a valuable and fundamental technique that can benefit many subsequent face applications is face neutralization, or the ability to take an arbitrary face image, and produce the corresponding front-facing face with neutral expression.

In this thesis, we accomplish this task with a neural knowledge transfer approach through the application of sub-modules pretrained with labeled data. We demonstrate that despite lacking labels required to form a superior dataset as in the method of Cole et al. in [5], this is sufficient for the task of frontalization and face decoding. We additionally compare our results to the method of Cole et al. and demonstrate that information used by their method ie. identity labels, are not needed at all. Even without this information it suffices to produce comparable results.

The intuition behind our encoder-decoder architecture is that the classical face encoder can be leveraged to guide the learning scheme due to its invariant property. For instance, the face encoder termed "FaceNet" produces a 128-dimensional vector encoding of any face image that is invariant to changes in pose and illumination [17]. Face decoders on the other hand, attempt to generate a face from the output encoding of face encoders. Several face decoders have been proposed in the last couple years to varying degrees of success such as the 3D face decoder of [20] and the separated landmark-texture learning approach of [5].

That said, the biggest challenge to face frontalization and neutral expression generation is the absence of supervised data. Whereas there are numerous datasets of unconstrained faces, they do not provide the corresponding frontal and neutral-expression face for each identity. Without this direct supervision, we split the larger problem into smaller sub-tasks for which supervised data exists. These components are: 1) a face encoder that maps a cropped face into a latent space containing identity info, provided by Facenet [16] and 2) a 3D facial landmark detector, provided by PRNet [8]. Our strategy given these two components is to use a CNN that learns a decoding of the identity latent space to frontal-facing neutral-expression images. Although we lack a ground truth face image to perform direct pixel correspondence, we are able to warp the input face image to an approximately frontal-facing and neutral-expression face for to artificially create such correspondence. Finally, to further guide the

learning of the network, we also exploit the 3D landmark detection to enforce similar facial geometries of the input and output. To prevent the freely varying background of the image from negatively impacting the network training, we additionally learn an attention mask that effectively forms a white background.

We note that our approach is similar in design to the approach of the recent Cole et al. [5], except for several differences such as the attention mask as mentioned above, adversarial loss, and major simplification of the data preprocessing step. Our results are insensitive to ad-hoc preprocessing while showing comparable performance. On the other hand, Cole's method requires preprepared identity labels with sufficient faces per identity along with very carefully designed landmark detector and warping module to augment data.

To evaluate our model performance, we visually inspect our results on a small set of images during training, and show a sampling of the results as compared with related works that also deal with face neutralization such as that of Cole et al. [5] and Yin et al. [22]. We prepare the results of these methods side-by-side with our own and visually inspect the differences and provide reasoning as to differences in results. These results demonstrate that our model is capable of results on par with supervised approaches, but also show some limitations. As an example, a high texture loss results in a possible over-dependence on superficial color which results in the undesired leaking of dark shadows onto the output frontalized face. Furthermore, the output faces are somewhat lacking in high frequency details such as fine wrinkles. Although the facial geometry in general reflects that of the input quite well, there are some cases when the appearance is different, notably around the jawline. Even despite these limitations, we find that a novel use of the landmark detection submodule to control output facial geometry and major simplification of previously sensitive data preprocessing pipelines result in comparable results to supervised approaches.

Chapter 2

Related Work

In this chapter, we describe related work to our own, mostly concerning or indirectly related to the goal of face frontalization. The structure of this chapter is as four sections. In the first, we describe face encoders and their applications. In the second, we talk about the method of Cole et al. which also utilizes pose-invariant features calculated by face encoders to create frontal neutral faces. In the third, we discuss a technique for frontalization using an underlying model to represent face texture and geometry. In the fourth, we talk about the potential of face encoders to be generalized to define audio-visual latent spaces and their ability to reconstruct face images.

2.1 Face Encoders

Face encoders learn essential information behind distinctive face features that can discriminate identity while remaining invariant to head pose, facial expression, accessories, environmental light conditions, etc. This invariant property is the key to mapping diverse perturbed faces of a person to a canonical face. Some well known examples of face encoders include FaceNet by [17] and VGG-Face [14]. In our work, we primarily use FaceNet encodings, so we provide an overview of the work and benefits of using this particular encoder, but give a high level explanation of both.

As described in its original paper, FaceNet is a unified system designed with the following goals in mind: face verification, recognition, and clustering. To accomplish these goals, the

deep convolutional network learns a 128 dimensional Euclidean embedding such that the squared L_2 norm of the embedding is a measure of similarity between faces. Tasks such as verification can then be mapped to simple operations on the embedding vectors. In addition, FaceNet demonstrates invariance in pose, illumination, occlusion by other objects, and age; the only requirement is that the input face image is cropped to the face. These two properties together mean that multiple pictures of the same person, under many different environmental conditions, will map to a cluster of 128-dimensional FaceNet embeddings. Because the network is trained using triplet loss which maximizes embedding distance between non-matching images, the embeddings of a separate facial identity are also distant from that of the original.

As discussed, the benefits of FaceNet are many. However, the VGG-Face encoder as described in [14] also performs well on face verification tasks as measured on both the LFW (Labeled Faces in the Wild) [10] and YTF (YouTube Faces) [21] datasets. VGG-Face, which is based on the popularized VGG networks as described in [18], outputs a 1024-dimensional embedding and is trained using classification loss. FaceNet [17] on the other hand is trained using triplet loss, so its output manifold in latent space is smoother than that of VGG-Face. We consequently choose to work with the Facenet encoder during experiments.

2.1.1 Face Encoder Applications

Face encoders are used for a variety of applications; One particularly effective use case is in multi-modal domains. The work of Ephrat et al [7] for instance uses face embeddings as a vital part of their pipeline to recognize speakers and isolate a single person's audio from audio-visual video clips. In their example video clip consisting of two comedians Rory and Jon simultaneously carrying out separate standup routines, they are able to isolate either comedian's speech to play during the video. Face encoders play a vital role in capturing essential visual signals from the video: By concatenating the output of their dilated CNN face encoder with their processed audio signals, they can achieve better audio isolation results than with state of the art audio-only processing techniques.

Another example of face encoders used in multi-modal applications is the work of Nagrani et al [13], which designs a pipeline for determining which face is more likely to have spoken

a particular speech clip. In their base pipeline involving exactly two faces, both faces are forwarded through the same VGG-M architecture [3] to obtain a 1024-dimensional feature vector. They are then concatenated with a feature vector produced by a voice sub-network, then passed through three fully connected layers to produce a binary output that selects from the two faces. Their model can match human performance in easy scenarios with differing gender and even exceed human performance when gender, age, and nationality match.

Face encoders are an important tool in face image analysis. As demonstrated by these examples of multi-modal applications that heavily rely on visual signals, we can easily incorporate visual signals by running images through well-known face encoders like FaceNet and then simply concatenating the output encoding with other signals. Utilized this way, the face encoder can be considered a plug-and-play component of any neural network architecture. In addition, the FaceNet encoder's invariant properties prove particularly valuable in terms of mapping a single identity under different environmental conditions to nearby embeddings.

2.2 Frontalization through Pose-Invariant Features

Cole et al. recently published a paper called "Synthesizing Normalized Faces From Facial Identity Features" [5] that utilizes an encoder-decoder architecture where the encoder is a pretrained FaceNet model. As a quick summary, the decoder network takes the corresponding latent space learned by the FaceNet model and proceeds to learn geometry and texture information separately in the form of landmarks and texture map. The geometry and texture are then combined into the final output using a differentiable warping method that is now written into the Tensorflow contrib library under the function name `sparse_image_warp()`.

The dataset preparation involves non-trivial filters and operations on existing datasets. Starting with the entire set of 2.6M face photos from VGGFace2 [2], they use the Google Cloud Vision API to filter for monochrome, blurry, high emotion score, eyeglasses, and high tilt and pan angles; additionally obtaining highly reliable ground truth landmarks. They are strict with the pan and expression filters, so that they can guarantee that the filtered set of images is frontal. Prelabeled identity information is used to warp and average each identity's set of photos into a single photo with relatively soft colored background. These

photos' landmark labels further enable Cole et al. to combine faces into new identities and avoid the issue of insufficient training data.

Unlike our own, the method of Cole et al. uses a decoder that separately decodes geometry and texture from the FaceNet embedding. Additionally, the preprocessing step is intricate and requires highly reliable landmark detection. We show that even without such costly operations and implementation, with a decoder that outputs the decoded face directly, we can achieve comparable results.

2.3 Frontalization with Model-based Techniques

FF-GAN [22] is an example of model-based face frontalization. It relies on a neural network to learn parameters of 3D Morphable Models [1] which aim to represent faces as a combination of a PCA combination of a predetermined set of textures and geometries. By leveraging the underlying model and also applying an adversarial loss, they aim to and succeed in accomplishing large pose frontalization in highly difficult scenarios where large portions of the face are not visible. They also succeed in the difficult task of reproducing high frequency features of the original face.

3DMM or 3D morphable model represents faces as a combination of texture T and geometry S in PCA space.

$$\begin{aligned} S &= \bar{S} + A_{id}\alpha_{id} + A_{exp}\alpha_{exp} \\ T &= \bar{T} + A_{tex}\alpha_{tex}, \end{aligned} \tag{2.1}$$

where $\bar{S}$ is the mean face shape, A_{id} is a basis of face shapes, A_{exp} is a basis of expressions, $\bar{T}$ is the average face texture, and A_{tex} is a basis of face textures. The coefficients $\{\alpha_{id}, \alpha_{exp}, \alpha_{tex}\}$ determine a linear combination of the bases.

These coefficients along with a projection matrix m representing pose of the face are sufficient for FF-GAN to generate a face image. Therefore the goal of the neural network in FF-GAN is to learn these particular parameters.

The FF-GAN reconstruction module solves the problem of heavily occluded faces by

applying a symmetry loss. By projecting the silhouette of the frontalized 3D Morphable Model, they recover a binary mask $\mathcal{M}$ indicating the visible portions of the face. The symmetry loss then demands that the generated frontal images and its flipped version should be similar, but only within the visible portion of the face as defined by $\mathcal{M}$. In the case where a face is turned 90 degrees and only one side of the face can be seen, the symmetry loss is able to reconstruct the non-visible side of the face based on the visible side alone. In cases where the mask and its horizontally flipped version do not cover the entire face, FF-GAN rely on adversarial loss to generate the hidden regions.

An obvious difference between FF-GAN and our technique is that FF-GAN relies on the underlying representation of the face as a 3D Morphable Model whereas our technique generates the face directly pixel-wise. Although using an underlying model provides the benefit that the solution space is easier to navigate, it also entails that the solution space is smaller and thus entails less flexibility. We provide comparisons in section 4.3.

2.4 Audio-Visual Representations

Several recent works concern the closely tied problem learning of audio-visual latent spaces [4, 23, 11].

For example, Chung et al. developed a method termed "Speech2Vid" [4] to generate a frame-by-frame video of a talking face given a single still face image and a speech audio segment not necessarily spoken by the given identity. Their neural network takes a single still image and a 0.3-second audio clip as input, passes them through an identity encoder and an audio encoder to create an audio-visual representation, and decodes a single frontal facing image with changing expression. By stitching these together for consecutive sliding windows over the entire audio segment, they can reconstruct a complete frame-by-frame video.

In another example, Zhou et al. [23] developed a generalized method that again creates a video, but with either audio or video. By creating a disentangled audio-visual latent space and using a temporal GAN to link frames, they can construct a smooth video with better preservation of identity and facial detail. The more intricately designed latent space allows the retrieval of speech information from either the separated audio clip or the separated

video clip. Consequently providing a still face image with either external speech audio or external video can make for high quality talking face video generation.

In both of these cases, the learned representation explicitly captures both identity and expression information which facilitates the construction of face along with the pose. The case of [23] further demonstrates that either visual and audio cues can provide sufficient information to capture expression information. Although these methods are less directly tied to our main goal of face frontalization, they nevertheless provide insight to the degree in which the specific face encoding method captures relevant identity and pertaining information.

Chapter 3

Approach

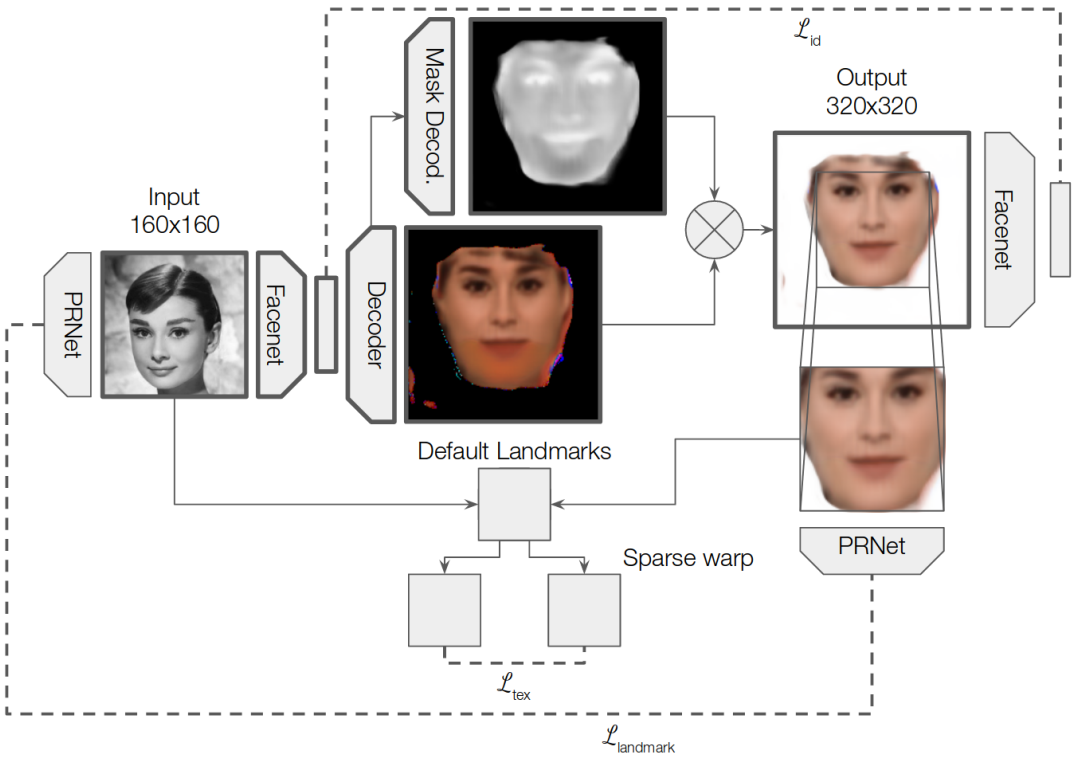

Figure 3-1: Architecture of our entire network, except for the discriminator module (more details in figure 3-7.) FaceNet and PRNet are pretrained components while the decoder and mask decoder are learned during network training. The input is a 160×160 loose-crop of a face on the left of the diagram, and the output is a 320×320 frontal-face image found on the right side of the diagram. Each dotted line connects two tensors to compute a separate loss, where $\mathcal{L}_{tex}$ is texture loss, $\mathcal{L}_{landmark}$ is landmark loss, and $\mathcal{L}_{id}$ is identity loss. See Section 3.5 for more details.

Figure 3-1 describes the overall encoder-decoder architecture of our network. The input is the 160×160 image on the left. It is a loose crop where the face is centered in the image. The input image is passed through a pretrained FaceNet module (see section 3.1) to produce an identity embedding. The identity embedding is then decoded into a 320×320 "noisy" face image through the decoder module (see section 3.3.) An intermediate tensor of the decoder module is upsampled through a mask decoder to produce an attention mask that dictates the background of the image. The attention mask is then merged with the "noisy" face to produce our final output.

The rest of the components are only used to specify loss during training. The texture loss is computed as an L_1 loss after warping both the input image and the cropped face to a canonical set of landmarks. The landmark loss utilizes landmarks determined by a pretrained landmark detector module (see section 3.2) to enforce various requirements of the output face geometry such as symmetry of the face, and similarity to the original input geometry. The identity loss is computed by comparing the FaceNet embeddings of both the input and output images. Additionally, we calculate an adversarial loss that is not shown in our figure.

This chapter detailing our approach begins with this summary of our overall architecture, then detailing different modules including FaceNet, PRNet, and the decoder, then ends with descriptions and definitions of our loss functions.

3.1 Face Encoder Module

In this section, we describe the particular face encoder that we used. In practice however, we may use any face encoder that takes a face image as input and outputs a meaningful embedding containing identity information.

FaceNet as introduced by Schroff et al in [17], is a state of the art neural network model for producing embeddings for face recognition and clustering. The authors at Google have not released a public pretrained model, so in practice we rely on the open source repository by Sandberg [16] which contains FaceNet models trained on the VGGFace2 dataset [2] with evaluation accuracy scores of 0.9965 on the LFW dataset [10].

Notably, the FaceNet network has been demonstrated to produce embeddings that are pose, lighting, and expression-invariant. This is essential to capturing identity information, since faces from the same identity under different environments and poses are guaranteed to discard unrelated information in the embedding space. By structuring the embedding space as a compact Euclidean space through triplet losses, operations such as face recognition and clustering can be performed directly using the embeddings themselves.

The original FaceNet paper experiments with several CNN architectures and discover that it performs best when using an Inception based architecture, similar to that described in [19]. Their best performing Inception based architecture is denoted by NN2 and differs slightly than the Inception Resnet V1 architecture. Specifically, it replaces some of the max pooling with L_2 pooling and also uses 3×3 pooling, other than the final pooling.

In more technical detail, FaceNet trains directly on the embedding itself, rather than use an intermediate bottleneck layer as prior techniques have done. They use a triplet loss, but instead of using hard-negative mining to select triplets, they use the self-termed "semi-hard mining" to avoid bad local minima in the early training process. Intuitively, the negative sample faces are chosen so that the L_2 distance between the negative sample and the anchor image is larger than that between the anchor and the positive samples, but the negative sample is still close to the anchor.

3.2 Differentiable Landmark Detection

The particular differentiable landmark detector module we use is termed Position Map Regression Network or "PRNet" by the authors Feng et al. in [8].

PRNet takes a 2D face image and can perform dense 3D face reconstruction and face alignment simultaneously. In our use case, PRNet is able to regress the widely used 68 Dlib facial landmarks in a cheap, differentiable and reliable manner due to its dense face reconstruction.

We start with a brief introduction of the PRNet technique: in order to regress the 3D facial geometry along with dense correspondence, PRNet introduces a representation called UV position map. This is a $256 \times 256 \times 3$ RGB image where the R,G,B values are the x,y,z

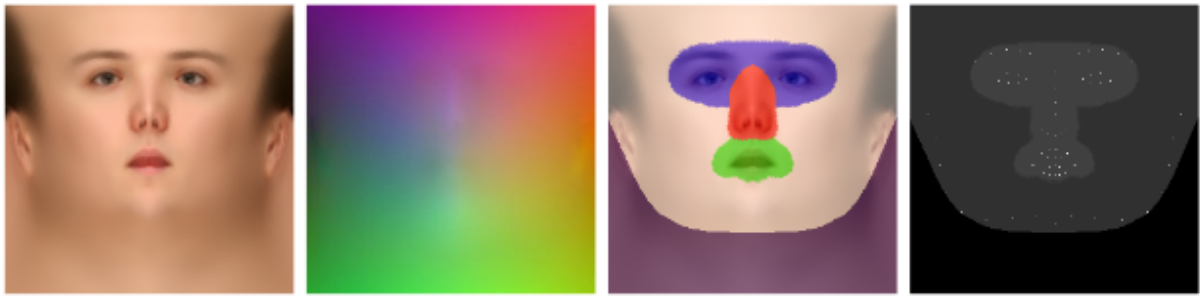

Figure 3-2: The leftmost is the UV texture map. The second from the left is the UV position map, where RGB colors dictated 3D position of each point in the texture map. The second from the right is the colored texture map, where each color dictates different weights / priorities during training ie. eyes are important so the purple region gets higher priority. The rightmost is the actual weight mask, in which weights / priorities are coded into grayscale. This is taken from figure 4 in the PRNet paper by Feng et al.[8].

coordinates of the corresponding points in the UV texture map (see Figure 3-2.) Intuitively, the position map is used to store 3D coordinates of points from the 3D face model and can be used to form a dense correspondence from the texture map to the 3D point cloud.

This method depends on the 3DMM model [1] to provide semantic meaning to these points and construct a Tutte embedding for the UV maps. For this purpose Feng et. al use the 300W-LP dataset [24] which contains 60K images annotated with the fitted 3DMM parameters. These annotated faces provide a ground truth UV position map which can be compared with the network output during training.

Furthermore, since 3DMM face meshes use 53,490 vertices, PRNet regresses a position map of resolution $256 \times 256 = 65,536$ to get a high precision point cloud. Due to the embedding of the face in UV space, PRNet uses a weight mask as seen in Figure 3-2 to give higher weight to facial regions with more discriminating features. There are a total of five predefined regions for eyes, nose, mouth, general face, and background. This way, the network places a higher priority around the eye, nose, and mouth region without noise from the neck and clothes.

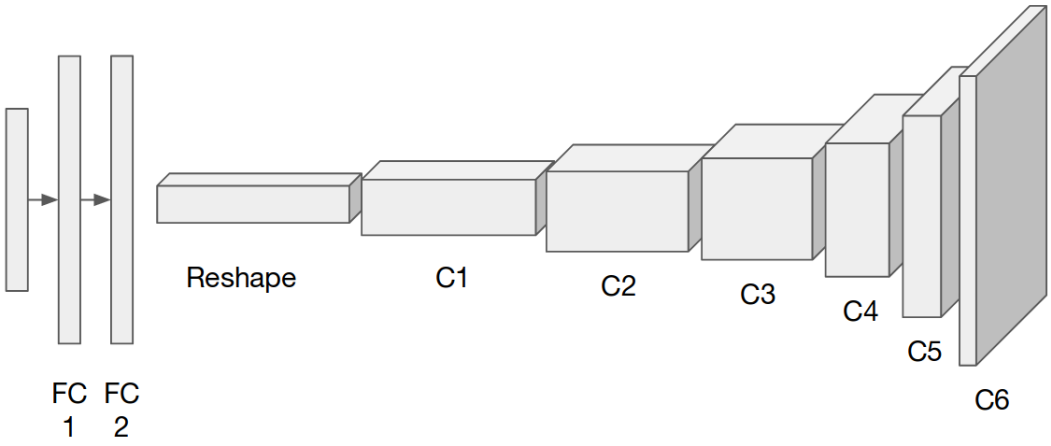

Figure 3-3: Architecture of face decoder. The face embedding is first passed through two fully connected layers, reshaped into a volumetric tensor and then passed through six convolutional layers. Here we shorten fully convolutional layer to FC and convolutional layer to C. See table 3.1 for details.

3.3 Face Decoder Module

The face decoder generates a "noisy" output face image as well as an attention mask. The attention mask simultaneously guides the learning of color, and masks the background of the noisy face image with white. The input of the reconstruction module is the output of the lowest non-spatially varying layer, the "avgpool" layer of David Sandberg's Facenet implementation, based on the Inception Resnet-v1 architecture [19] and trained on the VGGFace2 dataset [2]. By relying on the lowest such layer, we can capture the most information when reconstructing an image of the face.

We add a fully connected layer to change our latent space to one of size W^2K where $W = 5$ and $K = 192$. This 1D tensor is then reshaped into a volumetric tensor of size $W \times W \times K$. These W, K are specifically chosen such that by upsampling image dimensions by a factor of 2 and dividing channels by a factor of 2, we may arrive at a $320 \times 320 \times 3$ tensor that defines a color image. We repeatedly transform our volumetric tensor using a repeated sequence of two residual blocks followed by a singular upsample convolution. The sequence repeats until we reach a tensor of size $320 \times 320 \times 3$.

In summary, the tensor sizes are as listed in table 3.2:

layer	tensor shape
FC1	4800
FC1	4800
Reshape	5x5x192
C1	10x10x96
C2	20x20x48
C3	40x40x24
C4	80x80x12
C5	160x160x6
C6	320x320x3

Table 3.1: Shapes of intermediate tensors in face decoder.

We then map the output range to [-1,1] using tanh activation. Output images in the [0,1] or [0,255] range can be obtained using a linear mapping.

The attention mask, also of size $320 \times 320 \times 1$, is created using the same architecture except it stems from the intermediate $80 \times 80 \times 12$ tensor. In addition, we apply a final convolutional layer to reduce the $320 \times 320 \times 3$ tensor to shape $320 \times 320 \times 1$.

The final output face image F_O is defined as $F_O = WF_N + (1 - W)B$ where W is our attention mask, broadcasted to the shape of F_N and B is a tensor of ones, corresponding to a white background.

3.4 Generative Adversarial Networks

Generative adversarial networks are an important framework for capturing data distributions from Goodfellow et al [9]. They are composed of two networks, namely a generator G that learns to generate samples from the input training distribution, and an adversary D that learns to discriminate samples from the input distribution and the generated outputs. The generator seeks to better mimic samples from the input distribution based on feedback from the discriminator, which also seeks to improve its performance. Based on this concept, GANs have produced images with high level of realism [15].

GANs have been used for a variety of different tasks including image generation, 3D object generation, etc [6]. The widely popular Deep Convolutional GAN (DC-GAN) [15] for

instance extends the original multi-perceptron architecture to use convolutional layers and provides a very successful architecture for many neural network designs seeking to employ an adversarial training technique.

Our system uses the GAN architecture, where our generator G is the encoder-decoder architecture that maps input unconstrained face images to the frontal facing and neutral expression image. The discriminator D is a separate convolutional network that discriminates the output of G. On one hand, our generative model is capable of producing a decent output even when training without the discriminator. On the other hand, our goal in introducing D as an adversary to the system is to generate faces with as much realism as the input dataset. Our conjecture is that this is possible, considering that our target distribution of frontal, neutral faces is an existing subset of the input distribution of unconstrained faces. We discuss the results in a later section.

3.5 Loss functions

3.5.1 Texture Loss

Texture loss penalizes pixel level differences between the output face image and the input face. To set up pixel correspondences, we first warp the images according to a preselected, reliable set of landmarks. Those landmarks refer to the chin, nose, inner eye corners, outer eye corners, four outer mouth corners, and four inner mouth corners. Their 0-based DLib indices are [8, 33, 36, 39, 42, 45, 48, 51, 54, 57, 60, 62, 64, 66]. They are the red highlighted points in figure 3-4.

We choose these landmarks based on our own subjective experience on how reliably PRNet is able to calculate these landmarks. An example of unreliable landmarks is the set of landmarks denoting the back of the jaw. Due to the 3D nature of PRNet, these landmarks tend to be located in a range of locations when projected from their 3D coordinates to the 2D image, and may experience some incorrect detections as well. In figure 3-5 we see that sometimes the back of the jaw is occluded by hair and PRNet has some trouble making a guess as to where those occluded landmarks are. Sometimes, the jaw is of a particularly

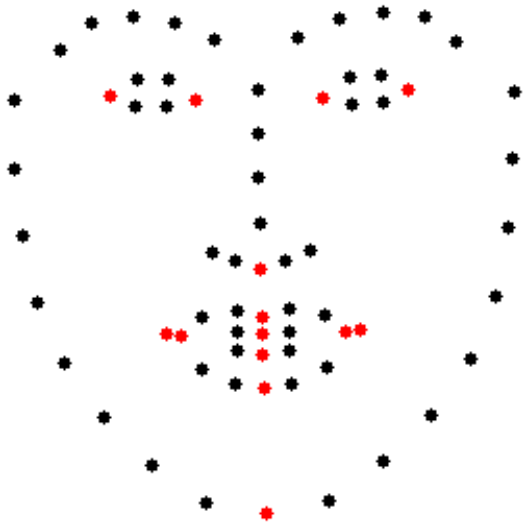

Figure 3-4: Subjectively most reliable set of landmarks detected by PRNet landmark detection submodule. These are marked by the fourteen red colored landmarks on the diagram.

square shape and PRNet produces a more rounded jaw structure. In addition, PRNet is often unable to exactly locate the eyebrows, although it can make a decent guess.

To warp the images, we use the differentiable `sparse_image_warp()` function from the tensorflow contributions library. This function is in fact based on the technique described in Section 4.1 of [5], and as such we use the radial basis function $\phi_1(r) = r$ such that the spline interpolant is less prone to overshooting and artifacts.

Our final loss is

$$\mathcal{L}_{tex} = \sum_{i,j}^{H,W} \left[\mathcal{M}_{i,j} \odot \left| I^*_{i,j} - O^*_{i,j} \right| \right], \tag{3.1}$$

where $\mathcal{M}$ is the learned mask, I^* is the warped input, O^* is the warped output and $\odot$ is element-wise multiplication

3.5.2 Landmark Loss

Now that we have a way to enforce texture, we additionally require a method of enforcing geometry. Our landmark detection module facilitates very flexible definitions according to

Figure 3-5: Landmarks detected by PRNet on 4 different photos of identities n000002 and n000003 of the VGGFace2 training dataset [2] to varying degrees of success.

our needs. Landmark loss is composed of four separate components that govern different aspects of the face shape we wish to control.

To enforce the geometry to be as close to that of the input face as possible, we use a geometry loss that enforces similarity between the ratio of 3D distances between landmarks. We introduce the concept of a landmark distance matrix $\mathbf{D}$ which is defined as

$$\{\mathbf{D}\}_{i,j} = |L_i - L_j|_1 , \tag{3.2}$$

where L_i, L_j are the 3D coordinates of the i'th and j'th landmarks. We use L_1 norm in these calculations to avoid discontinuities of square root function during gradient calculation for

Figure 3-6: Example faces from identities n000002 and n000003 of VGGFace2 dataset [2] warped according to a predetermined sparse set of landmarks.

L_2 norms.

Our geometry loss is then the cosine distance between the landmark distance matrices for the input and output face images.

$$\mathcal{L}_{geo} = \frac{\mathbf{D}_{inp} \cdot \mathbf{D}_{out}}{|\mathbf{D}_{inp}|\,|\mathbf{D}_{out}|} \tag{3.3}$$

Because we are normalizing the distances between landmarks using cosine distance, we effectively fix the geometry of the face in the output image while leaving extra degrees of freedom such as rotation, location, and size of the face to the other loss functions. Most notably, the scale of the output face is not in any way impacted by this particular loss.

We further employ a symmetry loss that enforces that all left-right pairs of landmarks are centered around the center vertical axis.

$$\mathcal{L}_{sym} = \mathbb{E}_i \left[\frac{L_{li}[x] + L_{ri}[x] - m}{2} \right], \tag{3.4}$$

where $L_{li}[x], L_{ri}[x]$ refer to the x coordinates of the i'th pair of left and right landmarks and m refers to the middle x coordinate.

In addition, we employ a tilt loss that enforces all left-right pairs of landmarks have the same height in the output image. Our tilt loss is defined as the mean absolute difference in height for each pair of landmarks:

$$\mathcal{L}_{tilt} = \mathbb{E}_i \left[|L_{li}.y - L_{ri}.y| \right] \tag{3.5}$$

Finally, we use an expression loss that enforces the expression to be neutral. Here, we have simply defined neutral expression to be when the lips are shut together.

$$\begin{aligned} \mathcal{L}_{expr} = & |L[61].y - L[67].y| \\ & + |L[62].y - L[66].y| \\ & + |L[63].y - L[65].y| \,, \end{aligned} \tag{3.6}$$

where these particular pairs of landmarks ie. $\{(61, 67), (62, 66), (63, 65)\}$ refer to the three vertically paired landmarks on the inner lips.

Our final landmark loss is then

$$\mathcal{L}_{landmark} = \alpha_{geo}\mathcal{L}_{geo} + \alpha_{sym}\mathcal{L}_{sym} + \alpha_{tilt}\mathcal{L}_{tilt} + \alpha_{expr}\mathcal{L}_{expr}, \tag{3.7}$$

where the hyperparameters we use are $\alpha_{geo} = 160$, $\alpha_{sym} = 1$, $\alpha_{tilt} = 1$, $\alpha_{expr} = 1$. These values are approximately chosen so that the magnitude of the losses are approximately equal and have not been tuned very much.

3.5.3 Mask Loss

The generated attention mask tends towards values of 1 everywhere, so we penalize the mask using an exponential penalty term.

This exponential penalty is defined as:

$$\mathop{\mathbb{E}}_{i,j} \left[A_{i,j}^{p} \right], \tag{3.8}$$

where A is our attention mask and p is the penalty exponent for which we use $p = 1$ in practice.

In addition, we penalize variation in the mask to enforce smoothness. The total variation penalty is defined as the average sum of L_2 differences between each pixel and the neighboring

pixels to the right and below.

$$\mathop{\mathbb{E}}_{i,j}\left[(A_{i+1,j} - A_{i,j})^2 + (A_{i,j+1} - A_{i,j})^2\right] \tag{3.9}$$

Our total mask loss $\mathcal{L}_{mask}$ is defined as the sum of the above two penalties from equations 3.8 and 3.9.

3.5.4 Adversarial Loss

We construct our discriminator as another convolutional network that learns to differentiate the output images from the input images. Because our final output incorporates the white background of our attention mask, we employ an earlier output as the input to the discriminator, namely the "noisy" face image F_N.

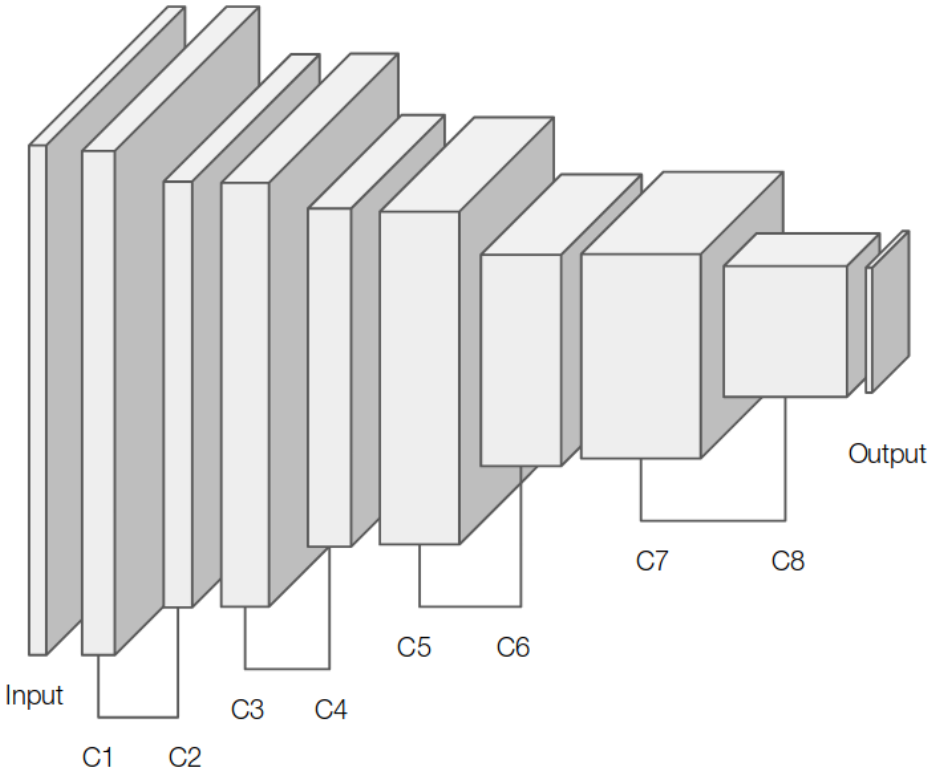

Figure 3-7: Discriminator architecture consisting of four pairs of convolutional layers with an ultimate layer to convert to the output, each followed by leaky ReLU activations. Here we shorten the term convolutional layer to C. See table 3.2 for more details

layer	kernel / stride	tensor shape
input		$320 \times 320 \times 3$
C1	$3 \times 3/1$	$320 \times 320 \times 32$
C2	$3 \times 3/2$	$160 \times 160 \times 32$
C3	$3 \times 3/1$	$160 \times 160 \times 64$
C4	$3 \times 3/2$	$80 \times 80 \times 64$
C5	$3 \times 3/1$	$80 \times 80 \times 128$
C6	$3 \times 3/2$	$40 \times 40 \times 128$
C7	$3 \times 3/1$	$40 \times 40 \times 256$
C8	$3 \times 3/1$	$20 \times 20 \times 256$
C9	$3 \times 3/1$	$20 \times 20 \times 1$

Table 3.2: Kernel, stride, and output tensor shape for all convolutional layers in the discriminator module.

Our discriminator takes as input a tensor of shape $b \times 320 \times 320 \times 3$. It is followed by a sequence of convolutional layers with leaky ReLU activations. We follow popular practice with $\alpha = 0.2$. Our final output is a $20 \times 20 \times 1$ tensor whose elements' receptive fields correspond to patches of the original image.

Our generator loss $\mathcal{L}_G$ is then determined by Equation 3.10.

$$\mathcal{L}_G = \mathbf{E}_{z \sim p_z(z)} \left[D(G(z)) - 1 \right]^2 \tag{3.10}$$

Our discriminator loss $\mathcal{L}_D$ is determined by Equation 3.11.

$$\mathcal{L}_D = \mathbf{E}_{x \sim p_{data}(x)} \left[D(x) - 1 \right]^2 + \mathbf{E}_{z \sim p_z(z)} \left[D(G(z)) + 1 \right]^2 \tag{3.11}$$

During training, we update generator 4 times for every update of discriminator. We schedule the training in this manner according to the intuition that the discriminator learns more easily than the generator. Additionally, when updating G, we sum the generator loss $\mathcal{L}_G$ with the original $\mathcal{L}_{tex}, \mathcal{L}_{landmark}, \mathcal{L}_{id}$.

Chapter 4

Experiment Results

In this chapter, we detail the results of testing our model. First, we include pictures of some of the example output as well as failure cases, providing some analysis as to limitations of our model. We follow with a section on comparisons to closely related techniques of Cole et al. [5] and FF-GAN [22]. Finally, we perform an ablation study in which we remove certain portions of the network to inspect their contribution.

4.1 Output Samples

Figures 4-1 and 4-2 contain samples of our output for various demographics and face shapes and skin colors. All of these photos are taken from the test set of VGGFace2 and were not used to train the network.

As displayed by these photos, we perform well across multiple ethnicities, genders, types of facial hair, and expressions. Furthermore, even given extreme face pose variations ie. left and right in first row of figure 4-1 and left in first and second rows of figure 4-2, our model is capable of generating reasonable face frontalizations that well-reflect the input identity's facial geometry, identity, and skin tone. Naturally, our model is thus also capable of generating frontalized output for input faces where the eyes are closed ie. left second row in figure 4-1.

Curiously, in some cases where the input photo contains a second person on the side ie. left second row and both images in the third row in figure 4-2, our model frontalizes the face

of the center-most person. In another case where the input photo has irregular hues ie. left third row in figure 4-1, we see that the our model can reflect the same hue. For relatively blurry photos ie. right third row in figure 4-1, we see that our method is able to construct a face that is higher resolution than the original.

Figure 4-1: Results of our method on images from test set of VGGFace2: Page 1. The first and third columns contain input photos. The second and fourth columns contain outputs of our model for the corresponding input photos directly to the left.

One comment we have on the limitations is that our model tends towards slight smiles,

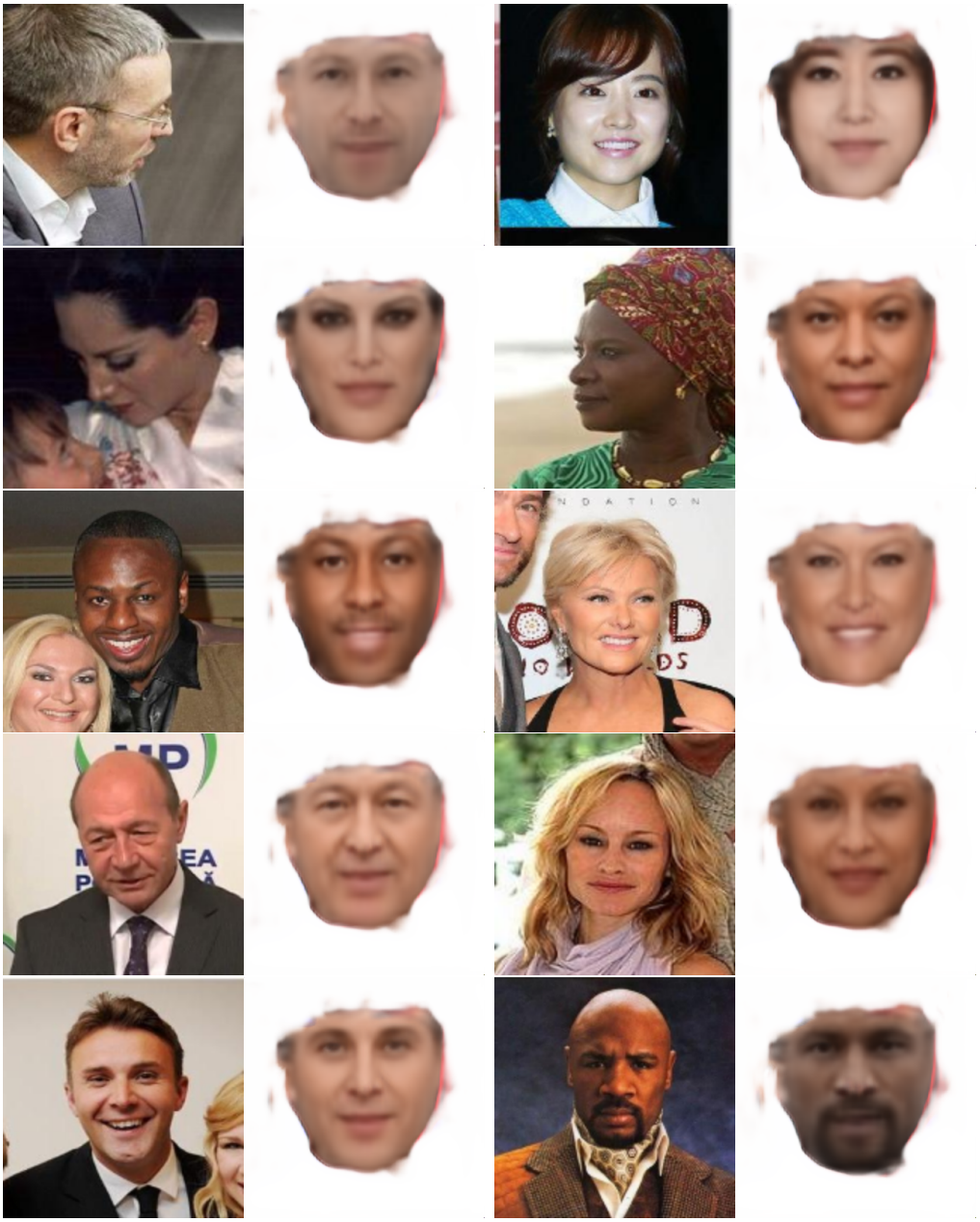

Figure 4-2: Results of our method on images from test set of VGGFace2: Page 2. The first and third columns contain input photos. The second and fourth columns contain outputs of our model for the corresponding input photos directly to the left.

with the mouth corners tilted upwards. We remark that upon examining the average landmarks of our dataset, the majority of photos have a smiling tendency, which is likely learned by the network. To alleviate this effect, one possible solution is increasing the weight on the expression loss $\mathcal{L}_{expr}$.

During training, we perform validation by evaluating our network on a preselected set of

16 images. Figure 4-3 contains the outputs of our model after 280k iterations. These faces are chosen from various sources that are not part of the VGGFace2, so we include these results for the sake of completeness and examine the results below.

Figure 4-3: Results of evaluating our model on validation set after 280k training steps. The figure is split into a left three columns and right three columns. The first column contains the input photos. The second column contains the final output. The third column contains the intermediate attention mask.

As seen from this collection of sampled outputs, our model performs exceedingly well especially on the task of capturing facial geometry. The texture is a little more difficult to capture in that our model still demonstrates some variance due to lighting changes in the input face, as seen in the top-left corner identity. Otherwise, we see that details such as eyebrow pose, nose shape, and eye shadow are well captured. Higher frequency details such as wrinkles are less ably captured, but still decently reflected in the final output.

Interestingly, we observed that the learned attention mask helps learn a shading / highlighting of the face better than the model without the mask module. On the other hand, we also see that the attention mask design can be improved to somehow capture a wider region of interest that includes peripheral characteristics such as hair. Our current network, according to our results, automatically decided that hair is not important to achieve lower loss value. On the other hand, we argue that this is a fairly interesting region to keep in the output image and leave an attention mask design update for future work.

4.1.1 Issues

In this section we summarize and raise a couple additional concerns that are worth mentioning and not simply products of training randomness. We also attempt to provide some explanation or speculation regarding these issues and also try to propose possible ways to alleviate them.

One such concern is the tendency of the output to smile or grin. There also remains a very slight tendency of the network to produce an asymmetrically higher right lip corner than the left lip corner. As we argue that this is due to a tendency of the data towards a smiling facial pose, we may further increase the weight of our expression loss $\mathcal{L}_{expr}$ to further enforce neutral expression.

Another issue is the matter of texture learning that fails in some cases, notably when the input face is wearing glasses, a cap, or simply has long hair that overshadows the eyes. A likely cause is an over-dependence on texture during training. We propose that a higher weighting towards identity can potentially alleviate this issue, especially since the identity info of the embedding space is robust to lighting.

Another issue that we had not mentioned yet is the smoothness of the attention mask.

Despite achieving a high degree of smoothness in the background and foreground, it would make the model output more visually appealing to either have 1) a smoother gradient from the foreground to the background or 2) a sharp gradient that also includes hair in the foreground.

4.2 Failure Cases

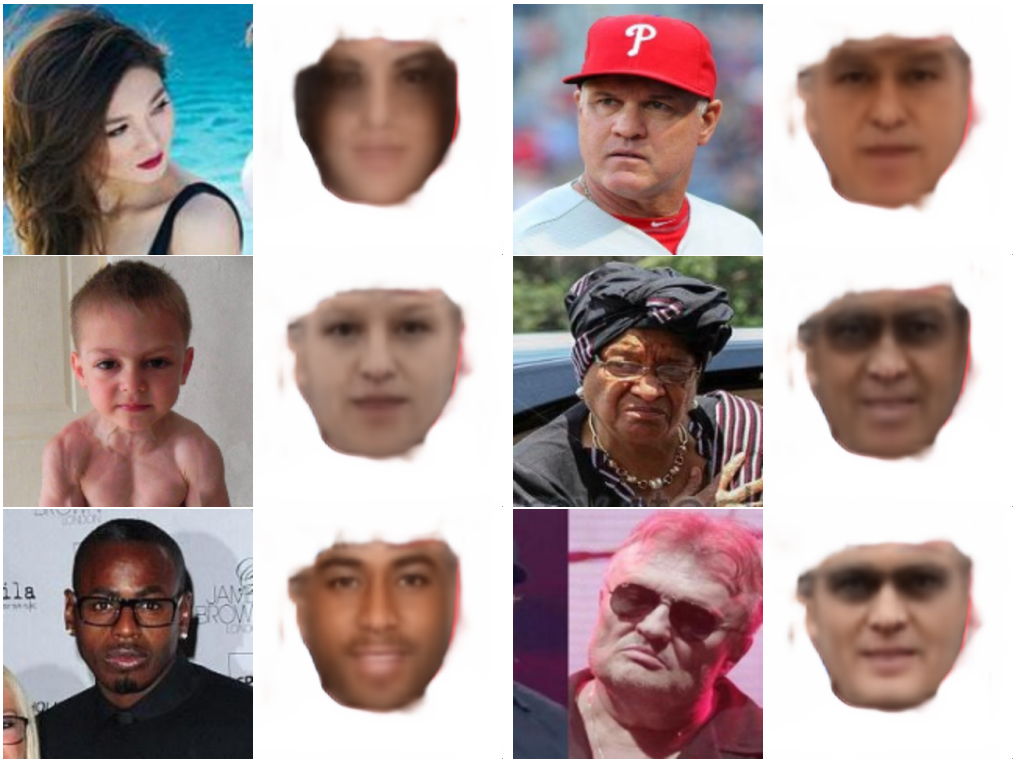

Figure 4-4: Moderate to extreme failure cases of our method, in no particular order.

Figure 4-4 contains several failure cases of our model. Although these cases are relatively scarce, we notice that there is a recurring theme that headwear and even sometimes long hair partially covering the eyes causes our model to create either blurring effects or dark tones around the eye region. We also note that our model tends to construct more aged appearance in the output, even if the input contains a baby's face ie. left second row. Another interesting case is the left third row, in which glare reflecting off the forehead may cause the network to produce a sort of averaged skin tone that is considerably lighter than the original skin tone.

4.3 Comparisons

In this section, we compare our method to previous work by Cole et al. in [5] and Yin et al. in their FF-GAN paper [22]. In our comparisons, we remark that we are able to produce very plausible results even given unfavorable conditions and lack of supervision in comparison to the technique of Cole et al.

4.3.1 Comparison to Method of Cole et al.

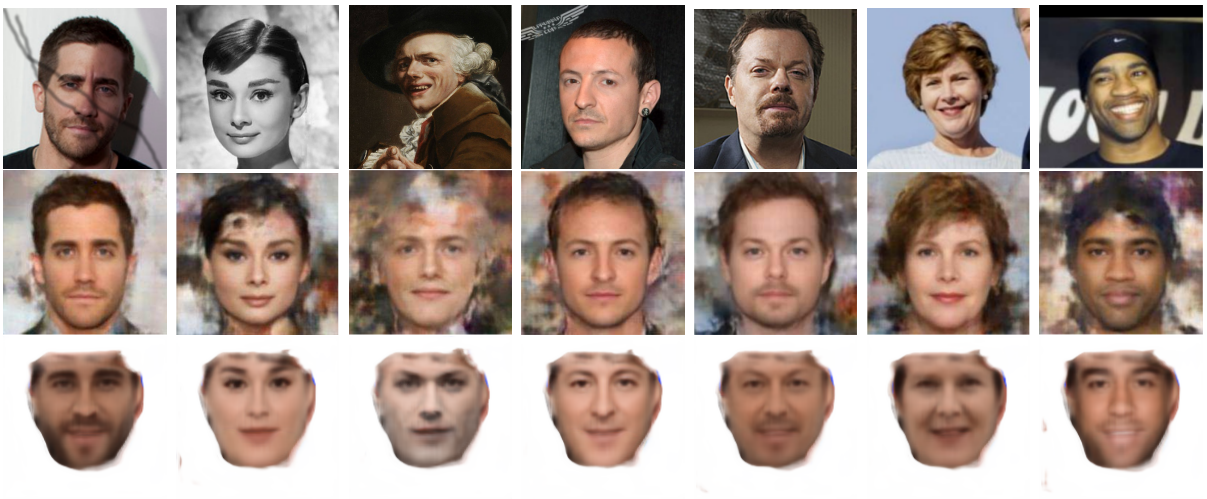

Figure 4-5: Top row: Input photos. Middle row: Output of method of Cole et al. [5]'s. Bottom row: Output of our method. For convenience, we number them from 1-7 starting from the left.

Observe the following results by Cole et al. as well as comparison with our own results in figure 4-5. In general, we observe that our model performs comparably and in some cases favorably in terms of facial geometry. For example, our model performs better in the third identity containing a painted portrait, more accurately capturing the narrowed eyebrows in his expression. For these samples, the nose shape hooks and widens as appropriate, and cheeks are puffy when expected. There are still some incorrect geometries however, such as when our model evaluates a slimmer jawline than expected in the fourth identity.

In terms of texture, we can see more high frequency features from Cole's network such as smile lines in the fourth identity. From the first identity in the first column, we see that our model sometimes causes dark shadows in the input to leak into the output face's skin color.

We speculate that this phenomenon may be due to our model's relatively strong dependence on texture, which is not necessarily a limitation but a correctable issue in the loss design.

Even with well-supervised data, their model has a tendency to produce green colored irises, particularly in identities 1,3,5,6. These are likely artifacts considering it occurs even when the original identity has dark eye color.

Overall, considering that Cole's method has advantageous access to well-supervised data and ours does not, our method's performance is quite decent.

4.3.2 Comparison to FF-GAN

Figure 4-6: Top row: Input photos. Middle row: Output of FF-GAN [22]. Bottom row: Output of our method.

Observe the following results by FF-GAN [22] as well as comparison with our own results in figure 4-6. It is immediately clear that a central goal of FF-GAN is to produce a frontalization that reflects the original as closely as possible, even if it does not satisfy constraints such as symmetry about the middle vertical axis. Notably, the skin color in the output of FF-GAN is fairly constant, whereas our method tends to output a darker skin tone as well as thicker beard when the input face is turned and displaying more facial hair.

We observe that in this experiment, our model demonstrates consistency across varying pan angles as slight changes in our decoded face manifest themselves as a darkening beard as the input face turns towards its left. Even though our model produces widely varying output in the first and last columns, we note this is reasonable behavior as it is subjectively difficult

even to humans ie. ourselves to realize that the leftmost input photo and the rightmost input photo are indeed the same person. Furthermore, we remark that our model demonstrates much greater consistency in identity in comparison with FF-GAN as the results of FF-GAN contain far more facial geometry artifacts ie. changes in jaw structure and relative eye size and positioning. On a deeper look, despite changes in skin tone that are relatively more apparent, the identity stays fairly consistent.

4.4 Ablation Study

In our ablation study we compare the results of our complete model with that when trained without adversarial loss, and with that when trained without either adversarial loss or attention mask.

In figures 4-7 and 4-8, we observe that the results of training without adversarial loss closely reflect the results of training with adversarial loss, with alterations readily attributable to randomness of training. This emphasizes that at the moment of this thesis, we have progress to make in terms of tuning our GAN training parameters. In other words, our adversarial loss does not improve our results. The results of training our model without either adversarial loss or attention mask are markedly worse, and in general the output faces reflect the identity of the input face much less. Curiously, each output image has a soft background with varying intensities and colors, which is likely due to the network incorrectly learning how to generate a background.

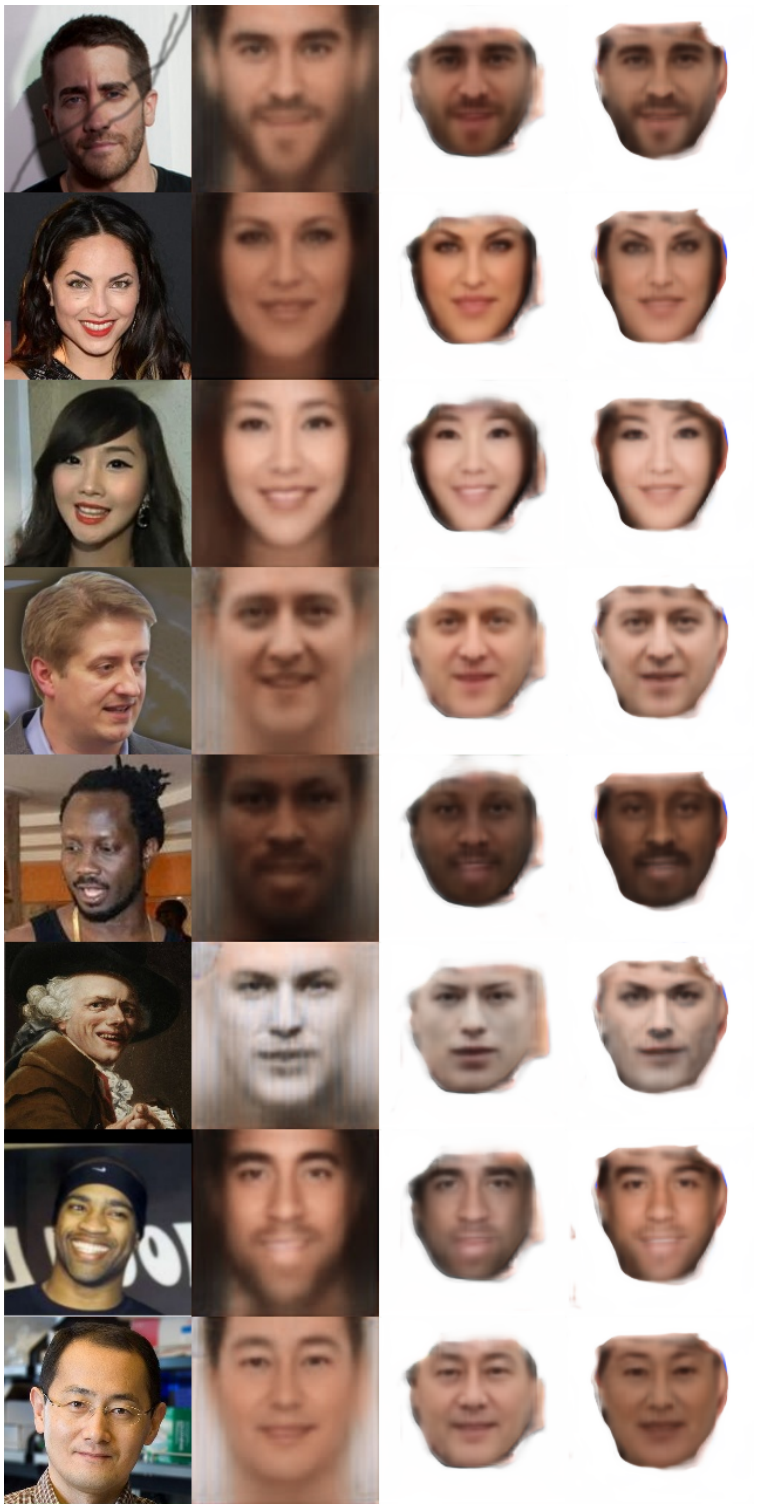

Figure 4-7: Ablation study results for first 8 preselected identities. The leftmost column contains the input photos. The second column contains the results of training without attention mask or adversarial loss. The third column contains the results of training without adversarial loss. The last column contains the results of the full model.

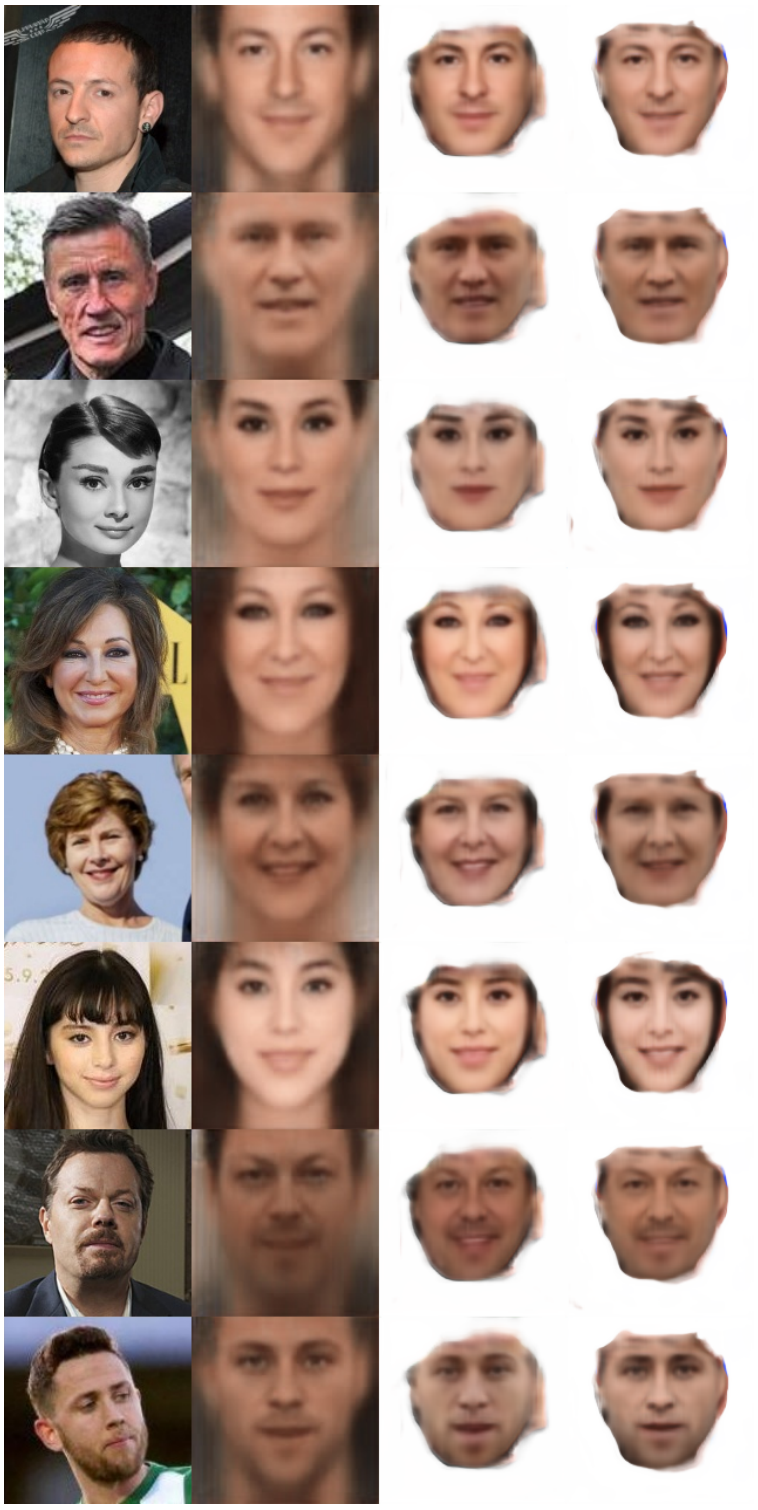

Figure 4-8: Ablation study results for second 8 preselected identities. The leftmost column contains the input photos. The second column contains the results of training without attention mask or adversarial loss. The third column contains the results of training without adversarial loss. The last column contains the results of the full model.

Chapter 5

Implementation Details

Our neural network is implemented in Python 3 using Tensorflow version 1.13. In this chapter, we discuss some of the details and some locations with potential room for improvement.

5.1 Dataset Preparation

An important aspect of our method is that we perform relatively inexpensive dataset preparation. In our case, we use the VGGFace2 dataset [2] which contains face images that come with tight facial bounding boxes and annotated with 5 basic facial landmarks.

Our method is simply filtering all the images for frontal faces images. We then crop the images to a square box around the face. Because the annotated bounding box information makes a tight crop around the face, we use a face ratio of 0.625, meaning that the square crop around the face is 0.625 the length of the square image.

To filter the images, we first pass the images through PRNet [8] and filter out faces with a pan angle > 5 degrees and a pitch outside the range of [-60, 0] degrees. This leaves a little over 500 thousand images of the original 2.6 million.

We may further filter all the images using Google Cloud Vision API [1] for frontal facing images, although this is not a necessary step in our case. To do so, we define a limit of 5 degrees for facial pan angle and reject all faces outside this range. Using all the available annotations, we are also able to filter out faces with headwear (although in practice we

[1] http://cloud.google.com/vision/

observe that GCloud API has only mediocre performance in detecting glasses in images.) We may additionally filter out blurry and monochrome images, although relatively few of the images are filtered out by these steps in comparison with the pan filter.

The cost of running our 500k images through the Vision API is around 500 dollars.

5.2 Potential factors for improvement

One clear location for improvement is the attention mask design. Our network is learning to discard pieces of peripheral information such as hair which happens because our current mask loss definition makes it more optimal to discard these attributes. These attributes may be very helpful to show in our final output, which implies some room for improvement in the attention mask design. It may also be the case however that the tendency of our mask to discard peripherals is reflective of a larger problem that we faced which was the domination of our texture loss during training. As a quick summary of this larger problem, we often observe that our texture loss is an order of magnitude or sometimes two above the other losses, causing other parts of the network to experience much larger gradients than is to be expected.

Additionally, we remark that by the time of writing of this thesis, we had not yet done enough tuning for the adversarial loss. All our attempts have resulted in the discriminator loss reaching zero, but given that the desired output set of frontalized faces is a subset of the input set of face images from VGGFace2, we expect that more tuning and proper scheduling of updates on the generator versus discriminator should turn out significantly better results.

Chapter 6

Discussion and Conclusion

In this thesis, we have presented a neural method for synthesizing frontal faces from face encodings, and list our contributions again here. We demonstrated that the ability of neural knowledge transfer through the application of sub-modules pretrained with labeled data, is sufficient for the task of frontalization and face decoding. We additionally compare our results to the method of Cole et al. and demonstrate that information used by their method ie. identity labels, are not needed at all and it suffices without this information to produce comparable results. Finally, we also analyze some of the limitations of our method, for example our method's failures on faces with glasses or caps.

As a summary, our method is based on the idea that the identity information captured by an encoder such as FaceNet [17] should suffice to construct the fully frontalized face of the original identity. To achieve this, we utilize an unsupervised approach, considering that there is a lack of ground truth frontal-facing neutral expression faces. As part of our design, we have first trained modular networks for the separate subtasks of facial landmark detection and facial recognition. Well-labeled data exist for these subtasks, allowing us to plug these components into our network as is. In particular, we have adopted the publicly available PRNet [8] as a landmark detection module and the publicly available FaceNet encoder as written by Sandberg [16] for use in our own network. The particular FaceNet recognition module learns features that are robust to lighting pose and facial expression variations, making it very attractive for our goal of face frontalization.

We leverage this invariance to train our decoder to produce consistent frontal and neutral

expression faces. Overall, our approach permits far less sensitive data pre-processing timelines. Compared to competing supervised approaches, our method performs comparably, and at times favorably.

6.1 Future Work

The network we designed has shown excellent promise, however there is still more to be done to tune our learned decoder towards better results. Our adversarial loss in particular requires more tuning so that we can achieve more realistic faces.

Our attention mask design can be improved to capture peripheral information such as hair. On another note, our current attention mask has significantly improved quality of the output face. This may even suggest that an additional deblurring module, similar to as proposed by Chung et al. in [4], can be tacked on for significant further performance increase.

This thesis has explored only a small portion of the tools and applications for face analysis. Originally, this thesis work was envisioned as itself a module of a "speech2face" network in which speech audio is converted into face images using a shared identity latent space. One possibility in this context is capturing various features of the face by removing the restriction of using a pretrained face encoder. By training our own face encoder alongside the face decoder in an end-to-end fashion, we can additionally capture expression information.

6.2 Lessons Learned

Throughout the course of this thesis, I have learned several important lessons regarding the design, implementation, and testing of the main network architecture.

First, in the case of handling unsupervised tasks, it is essential to break up the main goal into smaller sub-tasks where supervised data exist. Modularity of the network permits the potential training of sub-modules and reduction of moving parts.

Second, a valuable technique for background handling is the use of attention masks. Especially in face-based neural networks and in similar scenarios where background is un-

helpful, it is wise to mask the background to reduce its noise during training of the neural network. For this reason, adding on an attention mask module can help greatly, including for the performance of the original task.

Third, during the training of the network, it is helpful to record the performance of the model given various sets of hyper-parameters. This recording becomes crucial when, as is the case here, there are many hyper-parameters to be tuned. Additionally, it is helpful to tune the hyper-parameters by increasing or decreasing them one at a time to isolate effects, and also by factors of 10 for easy management.

Additionally, it is wise to prioritize writing the network architecture and setting up the minimum testable pipeline. This is widely known in software engineering practice as the minimum viable product, but likewise applies to writing code in machine learning. The case here, as is the case in many neural network projects, is to implement these core parts: dataset preparation, data loaders, network architecture, and training pipeline.

Lastly, visualization is also crucial for efficiently debugging and testing networks, especially so in these face analysis applications. Tracking losses in Tensorboard, plotting detected landmark locations, and additionally writing intermediate tensors have exposed bugs in my code where originally unexpected. I have found that it is difficult for me to write separable chunks of code that are individually testable, and consequently monitoring intermediate results has become even more crucial to the development process.

Bibliography

[1] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[2] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face and Gesture Recognition*, 2018.

[3] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *British Machine Vision Conference*, 2014.

[4] Joon Son Chung, Amir Jamaludin, and Andrew Zisserman. You said that? *CoRR*, abs/1705.02966, 2017.

[5] Forrester Cole, David Belanger, Dilip Krishnan, Aaron Sarna, Inbar Mosseri, and William T. Freeman. Face synthesis from facial identity features. *CoRR*, abs/1701.04851, 2017.

[6] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A. Bharath. Generative adversarial networks: An overview. *CoRR*, abs/1710.07035, 2017.

[7] Ariel Ephrat, Inbar Mosseri, Oran Lang, Tali Dekel, Kevin Wilson, Avinatan Hassidim, William T. Freeman, and Michael Rubinstein. Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation. *CoRR*, abs/1804.03619, 2018.

[8] Yao Feng, Fan Wu, Xiaohu Shao, Yanfeng Wang, and Xi Zhou. Joint 3d face reconstruction and dense alignment with position map regression network. *CoRR*, abs/1803.07835, 2018.

[9] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *arXiv e-prints*, page arXiv:1406.2661, June 2014.

[10] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[11] Changil Kim, Hijung Valentina Shin, Tae-Hyun Oh, Alexandre Kaspar, Mohamed A. Elgharib, and Wojciech Matusik. On learning associations of faces and voices. *CoRR*, abs/1805.05553, 2018.

[12] Mel McCurrie, Fernando Beletti, Lucas Parzianello, Allen Westendorp, Samuel E. Anthony, and Walter J. Scheirer. Predicting first impressions with deep learning. *CoRR*, abs/1610.08119, 2016.

[13] Arsha Nagrani, Samuel Albanie, and Andrew Zisserman. Seeing voices and hearing faces: Cross-modal biometric matching. *CoRR*, abs/1804.00326, 2018.

[14] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *British Machine Vision Conference*, 2015.

[15] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2016.

[16] David Sandberg. Facenet. `https://github.com/davidsandberg/facenet`, 2018.

[17] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[18] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[19] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[20] Ayush Tewari, Michael Zollhöfer, Hyeongwoo Kim, Pablo Garrido, Florian Bernard, Patrick Pérez, and Christian Theobalt. Mofa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. *CoRR*, abs/1703.10580, 2017.

[21] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. *CVPR 2011*, pages 529–534, 2011.

[22] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Towards large-pose face frontalization in the wild. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[23] Hang Zhou, Yu Liu, Ziwei Liu, Ping Luo, and Xiaogang Wang. Talking face generation by adversarially disentangled audio-visual representation. *CoRR*, abs/1807.07860, 2018.

[24] Xiangyu Zhu, Zhen Lei, Xiaoming Liu, Hailin Shi, and Stan Z. Li. Face alignment across large poses: A 3d solution. *CoRR*, abs/1511.07212, 2015.