GAN-based face image synthesis and its application to face recognition

Xiangnan Yin

sous le n°d'ordre NNT : 2022LYSEC021

THESE

pour obtenir le grade de
DOCTEUR DE L'ECOLE CENTRALE DE LYON
Spécialisté: Informatique

GAN-based Face Image Synthesis and its Application to Face Recognition

---

dans le cadre de l'Ecole Doctorale InfoMaths
présentée et soutenue publiquement par

Xiangnan Yin

Octobre 2021

Directeur de thèse: Prof. Liming Chen
Co-Directeur de thèse: Di Huang

JURY

Prof. Liming Chen	Ecole Centrale de Lyon	Directeur
Prof. Di Huang	Université de Beihang	Co-directeur
Prof. B. Ben Amor	IIAI à Abu Dhabi, EAU	Rapporteur
Dr. Sébastien Marcel	Institut de Recherche Idiap	Rapporteur
Prof. Alice Caplier	Grenoble-INP	Examinateur
Dr. Antitza Dantcheva	INRIA Sophia Antipolis - Méditerranée	Examinateur
Dr. Stéphane Gentric	LCTI- Télécom ParisTech	Examinateur

Abstract

Recently, with the development of deep Convolutional Neural Networks (CNNs) and large-scale datasets, face recognition (FR) has made remarkable progress. However, recognizing faces in large poses and under heavy occlusion remains a vital challenge due to the unbalanced training data. Thanks to Generative Adversarial Neural networks (GANs), synthesizing photorealistic multi-view faces and unveiling heavily occluded face images becomes feasible, which significantly facilitates FR and has a wide range of applications in entertainment and art fields. This thesis provides an in-depth study of GAN-based face image synthesis and its application to FR.

The current facial image synthesizing methods have featured two main research lines, i.e., 2D-based and 3D reconstruction-based. Our works cover both of them. For 2D-based face pose editing, current methods primarily focus on modeling the identity preserving ability but are less able to preserve the image style properly, which refers to the color, brightness, saturation, etc. This thesis proposes a novel two-stage approach to solve the style in-consistency problem, where face pose manipulation is cast into pixel sampling and face inpainting. With pixels sampled directly from the input image, the face editing result faithfully keeps the identity and the image style.

For traditional 3D face reconstruction, due to the linear and low-dimensional nature of the 3D Morphable Model (3DMM), the reconstructed textures hardly capture high-frequency details, resulting in blurred textures that are far from satisfactory. Some recent 3D face reconstruction methods have also leveraged adversarial training to improve the texture quality. However, these methods either rely on scarce, non-public 3D face data or complex and costly optimization approach. This thesis proposes a high-fidelity texture generation method, which predicts the global texture of the 3D face from a single input face image. The training is based on the pseudo ground truth blended by the 3DMM and input face textures. Multiple partial UV map discriminators are leveraged to handle the imperfect artifacts in the pseudo ground truth.

In terms of face de-occlusion, we propose a Segmentation-Reconstruction-Guided face de-occlusion GAN, consisting of three parts, a 3DMM parameter regression module N_θ, a face segmentation module N_S, and an image generation module N_G. With the texture prior provided by N_θ and the occluded parts indicated by N_S, N_G can faithfully recover the missing textures. The proposed method outperforms the state-of-the-art methods quantitatively and qualitatively.

Keywords: face pose editing, 3D face reconstruction, facial image de-occlusion.

Résumé

Récemment, avec le développement des "Convolutional Neural Networks"(CNNs) et des ensembles de données à grande échelle, la reconnaissance des visages (RF) a fait des progrès remarquables. Cependant, la reconnaissance de visages dans de grandes poses et sous forte occlusion reste un défi vital en raison du déséquilibre des données d'entraînement. Grâce aux "Generative Adversarial Networks" (GANs), il est possible de synthétiser des visages multi-vues photoréalistes et de dévoiler les images de visages fortement occlus, ce qui facilite considérablement la RF et offre un large éventail d'applications dans les domaines du divertissement et de l'art. Cette thèse fournit une étude approfondie de la synthèse d'images de visages basée sur les GAN et de son application à la RF.

Les méthodes actuelles de synthèse d'images faciales présentent deux axes de recherche principaux, à savoir les méthodes basées sur la 2D et celles basées sur la reconstruction 3D. Nos travaux couvrent les deux. Pour l'édition de la pose du visage en 2D, les méthodes actuelles se concentrent principalement sur la modélisation de la capacité de préservation de l'identité mais sont moins capables de préserver correctement le style de l'image, qui fait référence à la couleur, la luminosité, la saturation, etc. Cette thèse propose une nouvelle approche en deux étapes pour résoudre le problème d'incohérence de style, où la manipulation de la pose du visage est divisée en échantillonnage de pixels et en peinture du visage. Avec des pixels échantillonnés directement à partir de l'image d'entrée, le résultat de l'édition du visage conserve fidèlement l'identité et le style de l'image.

Pour la reconstruction traditionnelle de visages en 3D, en raison de la nature linéaire et basse dimensionnelle du modèle morphable 3D (3DMM), les textures reconstruites capturent à peine les détails à haute fréquence, ce qui donne des textures floues qui sont loin d'être satisfaisantes. Certaines méthodes récentes de reconstruction de visages en 3D ont également exploité l'entraînement contradictoire pour améliorer la qualité de la texture. Cependant, ces méthodes s'appuient soit sur des données de visage 3D rares et non publiques, soit sur une approche d'optimisation

complexe et coûteuse. Cette thèse propose une méthode de génération de texture haute-fidélité, qui prédit la texture globale du visage 3D à partir d'une seule image de visage en entrée. L'apprentissage est basé sur la pseudo vérité de terrain mélangée par le 3DMM et les textures du visage d'entrée. De multiples discriminateurs de cartes UV partielles sont utilisés pour gérer les artefacts imparfaits de la pseudo-vérité terrain.

En termes de désocclusion de visage, nous proposons un GAN de désocclusion de visage basé sur sengmentation et reconstruction, composé de trois parties, un module de régression des paramètres 3DMM N_θ, un module de segmentation des occlusions N_S, et un module de génération d'images N_G. Avec la texture préalable fournie par N_θ et les parties occluses indiquées par N_S, N_G peut récupérer fidèlement les textures manquantes. La méthode proposée surpasse quantitativement et qualitativement les méthodes de l'état de l'art.

Mots clés: synthèse de pose de visage, reconstruction de visage tridimensionnelle, dé-occlusion d'images faciales.

Acknowledgements

After more than four years of study, I have finally completed my Ph.D. thesis. I am very grateful to everyone for their help during these years.

First of all, I would like to thank my supervisors, Prof. Liming Chen and Prof. Di Huang. Prof. Chen rarely pushed me to do anything. He gave me the greatest freedom to carry out my research at my own pace and interest within the framework of my Ph.D. project. When I encountered difficulties in my study, Prof. Chen would always encourage me and give me enlightening advice. Prof. Huang is the most devoted professor I have ever met. He helped me polish my submission sentence by sentence, and sometimes we discussed until late in the night. I am grateful to be the student of Prof. Chen and Huang, who will be my role models for the rest of my life.

I would also like to thank my friends, Liqun Liu, Shiwei Li, Zehua Fu, Haoyu Li, Qinjie Ju, Ying Lv. We used to have dinner, go on trips together, and take care of each other in this place far from home. I spent an unforgettable time with them. I believe our friendship will last forever.

In addition, I would like to thank my family, who are my strongest spiritual support. Whatever I need, they will always be there for me.

Finally, I give thanks to my girlfriend. We have been separated for two years due to the pandemic, but our hearts are always together.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the advent of deep Convolutional Neural Networks (CNNs) [6] and the construction of large-scale databases, computer vision has made tremendous progress in recent years. Various computer vision-related applications have profoundly affected our lives, among which the most widely used is the ubiquitous face recognition (FR) system. As we all know, for CNN-based FR, the performance of a model is mainly influenced by two factors: one is the algorithm, and the other is the data: On the algorithm side, a series of models and metrics have been proposed, making the current FR models smaller, faster and more accurate; on the data side, researchers and companies have built face datasets in the millions or even tens of millions of volumes. However, it is not enough to increase face data in terms of quantity, as the uneven distribution and lack of diversity in the dataset lead to a decrease in accuracy under some extreme conditions, e.g., recognizing faces across large poses and under heavy occlusions. To address these issues, researchers typically use methods such as constructing more balanced and diverse datasets, augmenting training data with geometric transformations/randomly synthesized occlusions, and designing more efficient loss functions and network structures that decouple poses/occlusions from facial features.

Recently, the development of face image synthesizing provides new solutions to such problems. For example, the technique can be used in both the training and testing phases in large-pose face recognition. In the training phase, one can synthesize face images with extreme poses to make the model robust to the view angle; In the testing phase, the profile images can be transformed into frontal and then recognized. In addition, since the human face is the main organ for recognizing people, expressing/understanding emotions, and communicating in our daily

lives, synthetic photo-realistic face images have broad application scenarios, including media and entertainment industries, art, etc. However, psychological [7, 8] and anatomical [9, 10, 11] studies have demonstrated that humans have a special perceptual mechanism for faces and that minor imperfections in face images can attract human attention and even bring about huge differences in perception, making face image synthesizing a challenging task.

This thesis focuses on high-fidelity face image synthesizing and its application to FR. Specifically, we cover three sub-domains, 1. face pose editing, 2. texture generation for 3D face reconstruction, 3. face image de-occlusion.

1.1 Face Pose Editing

Face pose editing aims to change the view angle of an input face image while keeping its original identity unchanged. Research in this area can be divided into three categories: traditional 3D reconstruction-based methods, GAN-based 2D image generation methods, and the combination of both.

1.1.1 3D Reconstruction-based Methods

Before the widespread use of deep learning, the dominant approaches have focused on 3D reconstruction-based methods. [1] uses 68 facial landmarks to aligns an average 3D face model with the input face image. The face image is then projected onto the aligned model and rotated to present the frontal view. Figure 1.1 shows its process. The method has two obvious weaknesses: firstly, it is unreliable to fit all face images with an average face model, regardless of the expression and the identity, and secondly, it cannot handle the self-occluded textures, resulting in significant artifacts in the frontalized face images. [12] also employs an average 3D face to perform face frontalization. They propose a soft-symmetry strategy to fill the invisible part of the profile image with pixels from its visible symmetric region. Despite the visual improvements, this method still fails to produce photo-realistic face images. A more commonly used 3D face model, in addition to the average 3D face above, is the 3D Morphable Model (3DMM) [13]. It has been widely used in

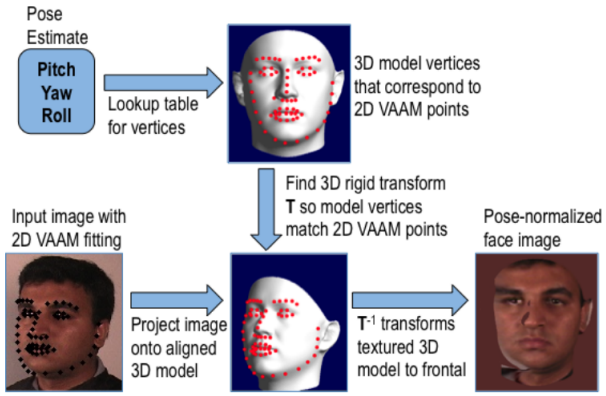


Figure 1.1: Overview of 3D pose normalization of [1]

3D face-related researches since its appearance. The model is a vector basis of the shape and texture learned from a set of 3D face scans. One can get the weights of those vectors, as well as the lighting conditions, by optimizing reconstruction losses of the 2D face images [14]. With the reconstructed 3D face, face images at any target pose can be obtained by 3D geometrical transformation and 2D projection. Nevertheless, the 3DMM is constructed by a small number of face scans under well-controlled conditions, limiting its diversity to identity, race, age, gender, etc. Besides, due to the linear and low-dimensional nature of the model, it can hardly capture high-frequency details. In summary, the traditional 3D reconstruction-based methods mainly face the following difficulties:

- The shape is not accurate enough.
- The synthesized textures are not photo-realistic, either sampled from the input image or reconstructed from the texture model.
- The optimization process is complex and costly.

1.1.2 GAN-based Methods

Thanks to the breakthroughs of Generative Adversarial Networks (GANs) [15], numerous GAN-based face pose editing approaches have been proposed [16, 17, 18, 19], significantly improving the visual realism of the synthesized images. Typically, these methods contain an encoder and a decoder and follow an adversarial training strat-

egy. DR-GAN [16] encodes face images to the feature space, then feeds it to the decoder together with a one-hot pose label. The generator will generate the face image with the indicated pose. A multi-task discriminator is employed to supervise both the identity preservation ability and the pose editing ability of the generator. However, without the supervison of pixel-level loss, DR-GAN's reults are of low quality. TP-GAN [17] introduces a dual-path generator that focuses on local and global transformations, respectively, to frontalize the input face images. CAPG-GAN [18] proposes a couple-agent discriminator to distinguish generated pairs from ground-truth pairs, one agent for pose manipulation and the other for identity consistency; the introduction of 2D landmark heat maps as pose representation makes pose editing more flexible than DR-GAN or TP-GAN. FNM [19] makes the following improvements comparing to TP-GAN: 1. replacing the local path generator by multiple local and global discriminators, 2. applying pixel-wise loss only on frontal inputs (since their corresponding outputs should remain as they are), avoiding the need for paired training data. Most of the above methods are trained on controlled datasets, such as Multi-PIE [20], and focus on the identity preserving ability during pose synthesis, leading to stylistic inconsistencies between the output and the input.

1.1.3 3D Reconstruction & GAN-based Methods

It is natural to think of incorporating the 3D prior in GAN-based methods. FF-GAN [21] made the first attempt. Besides the traditional encoder-decoder structure, it trains a 3DMM regression model which regresses the 3D coefficients of the input face, providing additional information for the generation. However, it is unclear how much this implicit provision of 3D information helps the generation, and the quality of the generated images is low. In comparison, 3D-PIM [22] explicitly leverages the 3D shape to guide the face frontalization. It is actually a combination of the traditional 3D face reconstruction-based method and TP-GAN. The 3D module generates texture-deficient frontalized faces, while TP-GAN serves to refine them to be photo-realistic. The explicit 3D rotation allows the 3D-PIM to converge faster and requires less training data than pure 2D GAN-based approaches. HF-PIM [23] predicts a UV texture map and a sampling map from the input image. The former

is a flattened global facial texture, while the latter stores the coordinates of pixels in the UV map, establishing the correlation between the pixels in the target image and the pixels in the UV map. Although it claims that it does not employ 3DMM to present shape or texture information, its method is equivalent to 3DMM because the ground truth of its sampling map derives from the 3DMM reconstruction results. Most data-driven face pose editing methods rely on paired training data, which is relatively scarce compared to ordinary face datasets. [5] solves the problem by introducing a rotate and render strategy: Given a face Rd_a at pose P_a, they rotate and render it to an arbitrary pose P_b to get Rd_b (Notice that this step is similar to [1]), the artifacts in Rd_b don't matter; then, they rotate and render Rd_b back to P_a to get $Rd_{a'}$. Comparing to Rd_a, the missing textures in $Rd_{a'}$ are due to the self-occlusion from P_b to P_a, which is equivalent to convert a real face image from P_b to P_a. So they only need to train the generator to reconstruct Rd_a from $Rd_{a'}$. In general, all GAN-related methods in this section and the previous section perform better than traditional 3D reconstruction-based methods, but they suffer from more or less the following problems:

- The lack of sufficient paired training data[17, 18, 23, 21].
- Style inconsistency between the input and output [17, 18, 16, 19].
- Texture inconsistency between outputs of different view angles [18, 5].

1.2 Texture Generation for 3D Face Reconstruction

3D face reconstruction has two main goals, shape and texture. Recently, significant advances have been made in single image-based 3D face reconstruction. However, most of the previous works focus on predicting accurate and fine-grained 3D shapes, relatively little work has focused on generating high-fidelity face textures. The existing 3D face texture generation techniques can be divided into three categories:

Texture model-based 3DMM has become an essential foundation for 3D reconstruction since its advent. Earlier approaches regress the 3DMM parameters by solving a non-linear optimization problem [24, 25], which is often slow and costly.

With the development of Convolutional Neural Networks, recent studies tend to predict the parameters using learning-based methods [26, 27, 3]. Nevertheless, as discussed in section 1.1.1, due to the linear, low-dimensional nature of the model, textures generated by the 3DMM model cannot capture fine details of the face image.

Image generation-based GANs provide a powerful tool for generating photorealistic images. Some recent 3D face reconstruction methods have leveraged GAN-based methods to improve the quality of the generated textures [28, 4, 29]. However, those approaches are highly dependent on large 3D face databases. [28] is trained on a synthetic 3D face database [30], leading to artifacts and blurring generation. The training of [4, 29] is based on a large UV map dataset, which is not publicly available.

GAN optimization-based The traditional yet most powerful GANs are trained to synthesize images from noise vectors [31, 32, 33]. To leverage the power of a pre-trained GAN, a series of works are established on inverting the image back to a GAN's latent space using optimization-based approaches [34, 35, 36, 37]. Similar methods are used to generate the UV map of a face image [38, 39]. First, they train a generator that converts noise vectors into UV maps. Then they directly optimize the latent code to minimize the reconstruction error between the input face image and the image rendered by the generated UV map. Instead of training a UV map generator, [40] first rotates the input image in 3D and optimizes the latent code of the pre-trained StyleGAN to fill in the missing textures, then stitches textures of different view angles by alpha blending to form the final UV map. By far, the optimization-based methods can yield the most realistic face UV maps. Nevertheless, they are usually complex and time-consuming, *e.g.*, GANFIT [38] takes 30 seconds to generate the UV map of an input face, while OSTeC [40] takes up to 5 minutes.

In summary, among the current texture generation methods for 3D face reconstruction, those based on texture models cannot yield high-fidelity results due to the model's simplicity; those based on image generation rely heavily on large training dataset; those based on optimization are time-consuming and require a high computational cost.

1.3 Face Image De-occlusion

The study of face images is a popular area in computer vision with vast application scenarios. In the past decade, deep learning has boosted many impressive face-related techniques, such as face recognition, expression recognition, 3D face reconstruction, and facial landmark detection. However, faces obtained in unconstrained environments are often occluded by various objects, e.g., masks, glasses, hands, scarfs, etc., leading to the degraded performance of the techniques above. Therefore, faithfully recovering face images from occlusions is an important research direction.

Before widely used deep learning-based methods, the dominant approaches were based on matching and copying patches from known regions to the invisible regions. A representative one is PatchMatch [41], which recursively searches the nearest neighbor textures to fill in the holes. Such copy-and-paste-based methods work well in recovering simple low-frequency textures and obtaining smooth images; however, they cannot recover high-level textures with complex structures, for example, missing nose/mouth in occluded face images.

SSDA [42] proposes a deep auto-encoder to remove specific noises and occlusions in the image, pioneered the use of deep neural networks for denoising and inpainting images. Its impressive performance led to the prosperity of deep learning-based image inpainting methods. Inspired by the GAN-based image generation method, Context Encoder [43] introduces adversarial training to image inpainting for the first time. Since then, the combination of auto-encoder and discriminator has been adopted as the basic model structure for image inpainting tasks. To make the model concentrate more on the missing parts [44, 45] leverage two discriminators: the global discriminator takes the whole image as input, while the local discriminator takes only the small region around the missing part. However, such a design is only suitable for a single rectangular(or even square) hole and cannot be applied to images with irregular and arbitrarily distributed holes. To properly handle free-form masks, [46] propose a Partial Convolutional Layer, comprising a masked and re-normalized convolution operation followed by a mask-update step. Precisely, they fuse mask

information into the forward path, and each convolution operation considers only the features corresponding to the visible positions. [47] pointed out that each neuron's receptive field expands with the forward propagation, making the corresponding mask inaccurate. Furthermore, it is not reasonable to "hard-gate" features in all channels equally. Therefore, they propose a gated convolution module, which learns a soft mask for the features in different layers. Due to the ineffectiveness of CNNs in modeling long-term correlations between distant textures and the hole regions, the inpainting results often have boundary artifacts and look unreal. [45] propose a Contextual Attention module, which utilizes the image features as a convolution kernel, forcing the distant textures interact with each other.

The common problem of the above methods is that they all require manually masking out the occlusions when applied to the face de-occlusion task. ByeGlass-GAN [48] jointly trains an eyeglass-face segmentation decoder, which shares the same encoder with the face decoder. The implicit segmentation hints extracted by the encoder, rather than the manual mask, guide the face decoder to remove the eyeglasses. [49] propose a semi-supervised face de-occlusion method. Benefiting from an elaborated forward process, the model automatically learns to predict occlusions without ground truth. [50] use the 3D face reconstruction results as priors and feed them into the generator along with the occluded faces. Since the 3D reconstruction results are occlusion-free, implicitly provide the occlusion information, their model directly de-occludes the face image without relying on the segmentation task. [51] proposed a two-stage GAN, where the first GAN aims to reconstruct the occlusion part solely, and the second GAN takes the result of the first GAN as a hint making the input image occlusion-free. Although the above methods do not require manual labeling, they all need synthetic occluded face images for training. Thus the type and amount of synthetic occlusions are crucial.

In summary, there are two factors that determine the effect of face de-occlusion, one is how to detect the occlusions' position and the other is how to generate realistic faces. Existing algorithms either require manually labeled occlusion regions or a large amount of simulated data for training, and the variety of occlusions are often too limited, resulting in poor generalization ability.

1.4 Contributions

The contributions of this thesis are listed as follows:

- In terms of face pose editing, we propose a novel "Pixel Attention Sampling" module, casting the problem into pixel sampling and image in-painting, explicitly preserving more style information of the input image and reducing the reliance on paired training data.
- For 3D face texture generation, a novel weakly supervised learning strategy is proposed, bypassing the need for scarce 3D data and the costly optimization approach.
- A segmentation-reconstruction-guided facial image de-occlusion framework is proposed, which aggregates face segmentation, 3D face reconstruction, and face generation. The method outperforms the SOTAs qualitatively and quantitatively.

1.5 Outline

The rest of this thesis is organized as follows:

- Chapter 2 — Literature Review

  We review all related work in this chapter, including the fundamentals of GANs and the techniques we use in our research.

- Chapter 3 — Pixel Sampling for Style Preserving Face Pose Editing

  This chapter focuses on preserving the style information while performing pose editing of 2D facial images. A novel two-stage face pose editing method is proposed, a Pixel Attention Sampling module is designed.

- Chapter 4 — Texture Generation for 3D Face Reconstruction

  This chapter introduces a weakly-supervised texture generation method for 3D face reconstruction. A large pseudo texture dataset is generated based on

Poisson Blending. A multi-head sampler module is designed to 1) extract the in-complete UV map from face images, 2) mask out the occlusions.

- Chapter 5 — Facial Image De-occlusion GAN

  This chapter proposes a segmentation-reconstruction-guided facial image de-occlusion method. A novel occluded face image dataset is proposed, with both the ground truth occlusion and the synthesized occlusion. An occlusion robust face reconstruction module and a face segmentation module are effectively trained based on the dataset. A novel facial de-occlusion module which can handle arbitrary occlusions is proposed.

- Chapter 6 — Conclusions and Future Work

  The work of this paper is summarized, and the next step of work is proposed.

Chapter 2

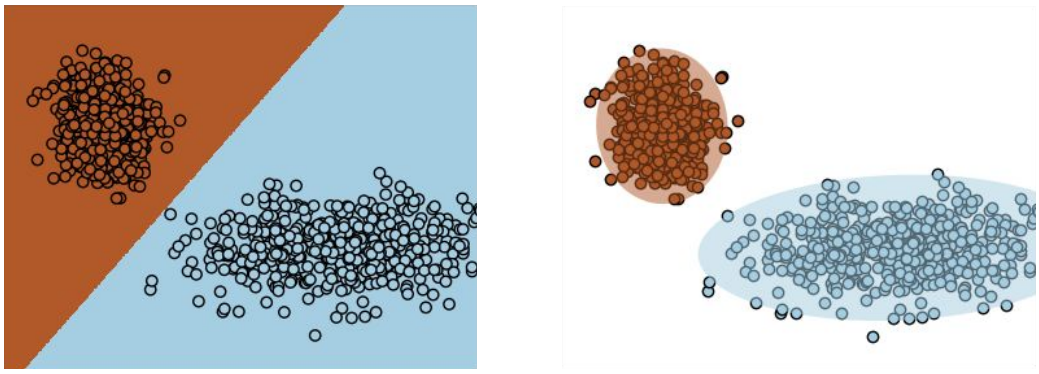
Literature Review

This thesis focuses on synthesizing photo-realistic face images, where the most crucial factor determining the generation quality is the model's design. Therefore, this chapter first introduces several representative generative models, including Variational AutoEncoder(VAE) [52], Generative Adversarial Networks(GANs) [15]. And then focus on several essential modifications to the traditional GANs, including Wasserstein GAN (WGAN), Wasserstein GAN with gradient penalty (WGAN-GP), spectrally normalized GAN (SN-GAN), Self-Attention GAN (SAGAN), followed by the evaluation metrics of the generation quality, i.e., Fréchet Inception Distance (FID), Inception Score(IS). Finally, we summarize the chapter.

2.1 Generative models

Machine learning models can be roughly classified into two categories: the discriminative model and the generative model. Given a set of data X and their corresponding labels Y, the discriminative model captures the probability of Y conditioned by X, $P(Y|X)$; the generative model captures the joint distribution of X and Y, $P(X, Y)$.

As shown in Figure 2, the discriminative model predicts data categories by learning the boundary between categories, whereas the generative model generates data indistinguishable from the real data by modeling the data distribution. Generative models tackle a more difficult task than discriminative models since they have to consider all aspects of the data in the real world. In contrast, the discriminative models only need to concentrate on the differences between classes. For example, it is much more challenging to train a model to generate photo-realistic cat/dog



(a) Discriminative (b) Generative

Figure 2.1: Difference between the discriminative models and the generative model

images than training a model to distinguish between cat and dog images.

Typical discriminative models include logistic regression, support vector machine (SVM), decision trees, and most deep neural networks. Typical generative models include Naive Bayes classifiers, Gaussian Mixture Model(GMM), latent Dirichlet allocation (LDA), Variational Autoencoders (VAE), and Generative Adversarial Network (GAN). Next, we will introduce two representative deep learning-based models, VAE and GAN, in detail.

2.1.1 Variational Autoencoder

Variational Autoencoder (VAE) [52] combines the deep neural networks methods with statistical learning to approximate the real data distribution by simple distributions. Like the traditional autoencoder [53, 54], it is composed of an encoder, which encodes the data into the latent space, and a decoder, aiming to reconstruct the input from the latent code. The difference is that VAE maps the inputs to distributions (regularized by a predefined simple distribution, e.g., standard multivariate normal distribution) rather than isolated points, and the decoder's inputs are sampled from the encoded distributions, Figure 2.2 illustrates the difference between the two models. Therefore, the VAE's hidden space has the following properties: 1. Continuity, two similar codes will decode to similar data; 2. Completeness, any code in the hidden space can be mapped to the data space. Mathematically, the

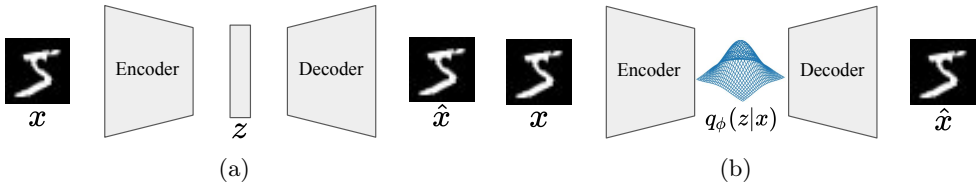


Figure 2.2: Difference between the AE (a) and the VAE (b)

VAE aims to maximize the likelihood shown in Equation 2.1,

$$p(x_i) = \int p(x_i|z;\theta)p(z)dz, \tag{2.1}$$

where x_i is sampled from the real data, z is the latent code sampled from a known distribution $p(z)$, and θ represents the parameter of the decoder. However, the maximization of Equation 2.1 is intractable. Therefore, VAE seeks to maximize the Evidence Lower Bound (ELBO) of $\log p(x_i)$, which is its core idea. The encoder aims to approximate the posterior distribution of $p(z|x_i)$, denoted as $q_\phi(z|x_i)$, where ϕ is its parameter. Thus, the Kullback-Leibler (KL) divergence between two distributions is given by,

$$D_{KL}(q_\phi(z|x_i)\|p(z|x_i)) = -\int q_\phi(z|x_i)\log\left(\frac{p(z|x_i)}{q_\phi(z|x_i)}\right)dz > 0, \tag{2.2}$$

By Bayes' rule and rules of logarithms, Equation 2.2 is further derived as,

$$\begin{gathered}
-\int q_\phi(z|x_i)\log\left(\frac{p(z|x_i)}{q_\phi(z|x_i)}\right)dz > 0, \\
-\int q_\phi(z|x_i)\log\left(\frac{p_\theta(x_i|z)p(z)}{q_\phi(z|x_i)p(x_i)}\right)dz > 0 \\
-\int q_\phi(z|x_i)\log\frac{p(z)}{q_\phi(z|x_i)}dz - \int q_\phi(z|x_i)\log p_\theta(x_i|z)dz + \int q_\phi(z|x_i)\log p(x_i)dz > 0 \\
D_{KL}(q_\phi(z|x_i)\|p(z)) - \mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] + \log p(x_i) > 0 \\
\log p(x_i) > \mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - D_{KL}(q_\phi(z|x_i)\|p(z)),
\end{gathered} \tag{2.3}$$

The right hand side of Equation 2.3 is the optimization objective of VAE, i.e., the ELBO of the log-likelihood. It has two terms, the first term measures the likelihood

of the reconstructed data, and the second term regularizes the posterior distribution by the predefined distribution. The implication of the first term is as follows: we first encode x_i to a specific latent distribution $q_\phi(z|x_i)$, then we sample z from this distribution $\sim q_\phi(z|x_i)$, next we decode z back to a new distribution in data space, the likelihood of x_i under this distribution is denoted as $P_\theta(x_i|z)$, the higher the $P_\theta(x_i|z)$, the better the reconstruction ability of the model. Thus, the first term of the ELBO serves as the reconstruction error like the traditional autoencoder. More specifically, suppose $p_\theta(x_i|z) = \mathcal{N}(f_\theta(z), \sigma^2 I)$, where $x_i \in \mathbb{R}^k$,

$$p_\theta(x_i|z) = \frac{1}{\sqrt{(2\pi)^k \sigma^k}} \exp\left(-\frac{\|x - f_\theta(z)\|^2}{2\sigma^2}\right) \tag{2.4}$$

the first term drives as:

$$\mathbb{E}_{x\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] = -\frac{1}{2\sigma^2}\mathbb{E}_{x\sim q_\phi(z|x_i)}[\|x - f_\theta(z)\|^2] + C, \tag{2.5}$$

where C is a constant. Maximizing the first term of Equation 2.3 is equivalent to minimizing $\|x - f_\theta(z)\|^2$. Although $p_\theta(x_i|z)$ is not necessarily normally distributed, this does not change the fact that the first term represents the reconstruction ability of the model. In practice, we can use L_1 loss, L_2 loss, or cross-entropy loss to guide the training.

All that remains is to derive the closed-form of the second term of Equation 2.3, $D_{KL}(q_\phi(z|x_i)\|p(z))$. $p(z)$ is predefined as $\mathcal{N}(0, I)$ to facilitate computation and to allow good properties of the hidden space (complete, continuous, easy to sample). $q_\phi(z|x_i)$ is also restricted to be a normal distribution $\mathcal{N}(\mu, \Sigma)$, where μ, Σ are outputs of the encoder, and Σ is diagonal. We deduce the KL divergence of two multivariate normal distributions below.

By definition, the distribution function of a multivariate normal distribution is as,

$$N(x; \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^k |\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right), \tag{2.6}$$

where μ is its mean and Σ is its covariance matrix.

The KL divergence between two distributions $p(x)$ and $q(x)$ is given by,

$$\begin{aligned} D_{KL}(p(x)||q(x)) &= \int p(x) \cdot \log\left(\frac{p(x)}{q(x)}\right) dx \\ &= \int p(x) \left(\log p(x) - \log q(x)\right) dx \\ &= \mathbb{E}_{x\sim p}[\log p(x) - \log q(x)]. \end{aligned} \tag{2.7}$$

Let $p(x) = \mathcal{N}(x; \mu_1, \Sigma_1)$, $q(x) = \mathcal{N}(x; \mu_2, \Sigma_2)$, we get,

$$\begin{aligned} &\log p(x) - \log q(x) \\ =&\log\left(\frac{1}{\sqrt{(2\pi)^k|\Sigma_1|}}\exp\left(-\frac{1}{2}(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right)\right) - \\ &\log\left(\frac{1}{\sqrt{(2\pi)^k|\Sigma_2|}}\exp\left(-\frac{1}{2}(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right)\right) \\ =&\frac{1}{2}\log\frac{|\Sigma_2|}{|\Sigma_1|} + \frac{1}{2}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2) - (x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right]. \end{aligned} \tag{2.8}$$

Thus,

$$\begin{aligned} &D_{KL}(p(x)||q(x)) = \mathbb{E}_{x\sim p}[\log p(x) - \log q(x)] \\ =&\mathbb{E}_{x\sim p}\left[\frac{1}{2}\log\frac{|\Sigma_2|}{|\Sigma_1|} + \frac{1}{2}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2) - (x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right]\right] \\ =&\frac{1}{2}\left(\log\frac{|\Sigma_2|}{|\Sigma_1|} + \mathbb{E}_{x\sim p}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right] - \right. \\ &\left.\mathbb{E}_{x\sim p}\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right]\right) \end{aligned} \tag{2.9}$$

For $\mathbb{E}_{x\sim p}\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right]$, we apply the trace trick[1],

$$\begin{aligned} &\mathbb{E}_{x\sim p}\left[(x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right] \\ =&\mathbb{E}_{x\sim p}\left[\mathrm{tr}\left((x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1)\right)\right] \\ =&\mathbb{E}_{x\sim p}\left[\mathrm{tr}\left((x-\mu_1)(x-\mu_1)^T\Sigma_1^{-1}\right)\right], \end{aligned} \tag{2.10}$$

[1] $s = v^TWv = \mathrm{tr}(v^TWv) = \mathrm{tr}(vv^TW)$

the linearity of the trace operator further gives,

$$\begin{aligned}
&\mathbb{E}_{x\sim p}\left[\mathrm{tr}\left((x-\mu_1)(x-\mu_1)^T\Sigma_1^{-1}\right)\right]\\
=&\mathrm{tr}\left(\mathbb{E}_{x\sim p}\left[(x-\mu_1)(x-\mu_1)^T\right]\Sigma_1^{-1}\right)\\
=&\mathrm{tr}\left(\Sigma_1\Sigma_1^{-1}\right)\\
=&\mathrm{tr}(I_d).
\end{aligned} \tag{2.11}$$

For $\mathbb{E}_{x\sim p}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right]$, we need to introduce μ_1 to facilitate the derivation, since $x\sim p=\mathcal{N}(x;\mu_1,\Sigma_1)$,

$$\begin{aligned}
&\mathbb{E}_{x\sim p}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right]\\
=&\mathbb{E}_{x\sim p}\left[((x-\mu_1)+(\mu_1-\mu_2))^T\Sigma_2^{-1}((x-\mu_1)+(\mu_1-\mu_2))\right]\\
=&\mathbb{E}_{x\sim p}\left[(x-\mu_1)^T\Sigma_2^{-1}(x-\mu_1)+(x-\mu_1)^T\Sigma_2^{-1}(\mu_1-\mu_2)+\right.\\
&\left.(\mu_1-\mu_2)^T\Sigma_2^{-1}(x-\mu_1)+(\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)\right]
\end{aligned} \tag{2.12}$$

where $\mu_1-\mu_2$ is a fixed vector, thus the expectation of the second and the third term above is zero. We apply the trace trick again for the first term and get,

$$\begin{aligned}
&\mathbb{E}_{x\sim p}\left[(x-\mu_2)^T\Sigma_2^{-1}(x-\mu_2)\right]\\
=&\mathrm{tr}(\Sigma_1\Sigma_2^{-1})+(\mu_1-\mu_2)\Sigma_2^{-1}(\mu_1-\mu_2).
\end{aligned} \tag{2.13}$$

Taking Equation 2.11, Equation 2.13 into Equation 2.9 gives,

$$\begin{aligned}
&D_{KL}(p(x)\|q(x))\\
=&\frac{1}{2}\left(\log\frac{|\Sigma_2|}{|\Sigma_1|}+\mathrm{tr}(\Sigma_1\Sigma_2^-1)+(\mu_1-\mu_2)^T\Sigma_2^{-1}(\mu_1-\mu_2)-\mathrm{tr}(I_d)\right).
\end{aligned} \tag{2.14}$$

Recall that $q_\phi(z|x_i)\sim\mathcal{N}(\mu,\Sigma)$, where Σ is diagonal,

$$\Sigma=\begin{bmatrix}
\sigma_1^2 & 0 & 0 & \cdots & 0\\
0 & \sigma_2^2 & 0 & \cdots & 0\\
\vdots & \vdots & \vdots & \ddots & \vdots\\
0 & 0 & 0 & \cdots & \sigma_k^2
\end{bmatrix}$$

and $q(z) \sim \mathcal{N}(0, I)$, thus,

$$D_{KL}(q_\phi(z|x_i)\|p(z)) = \frac{1}{2}\sum_{j=1}^{k}(-\log\sigma_i^2 + \sigma_i^2 - 1 + \mu_i^2). \tag{2.15}$$

The ELBO in Equation 2.3 is therefore given by,

$$\log p(x_i) > \mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] - \frac{1}{2}\sum_{j=1}^{k}(-\log\sigma_i^2 + \sigma_i^2 - 1 + \mu_i^2), \tag{2.16}$$

where σ_i^2 is the i-th element on the diagonal of matrix Σ, and μ_i is the i-th element of the mean μ.

Maximizing the ELBO is equivalent to minimize the following loss function,

$$L(x_i) = -\mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)] + \frac{1}{2}\sum_{j=1}^{k}(\sigma_i^2 + \mu_i^2 - \log\sigma_i^2 - 1). \tag{2.17}$$

As discussed before, $\mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\theta(x_i|z)]$ measures the reconstruction ability, thus could be replaced by typical reconstruction losses, e.g., L_1 loss,

$$L(x_i) = \|\hat{x}_i - x_i\|_1 + \frac{C}{2}\sum_{j=1}^{k}(\sigma_i^2 + \mu_i^2 - \log\sigma_i^2 - 1), \tag{2.18}$$

where $\hat{x}_i$ denotes the reconstruction result corresponding to x_i and C is a hyper-parameter that balances the regularization and the reconstruction errors.

As mentioned before, the input z to the decoder follows the distribution $\mathcal{N}(\mu, \Sigma)$, where μ and Σ are the outputs of the encoder. However, when we train the model, sampling directly from the above distribution causes the back-propagation gradient to be blocked, making it impossible for the encoder parameters to be updated, so the reparameterization trick is used here. We sample a random variable ε from $\mathcal{N}(0, I)$, and transform it to have the mean μ and covariance matrix Σ,

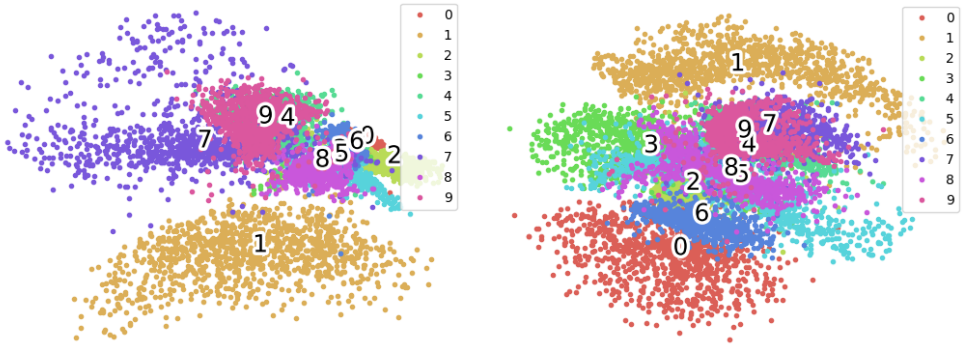
$$z = \Sigma \cdot \varepsilon + \mu. \tag{2.19}$$

In summary, the training algorithm of the VAE is as follows:

Algorithm 1 Training algorithm of VAE

Input: training dataset $\{x_i\}$; Encoder q_ϕ and decoder p_θ; Regularization weight C
Output: trained parameters ϕ and θ

1: **for** number of training iterations **do**
2: Sample minibatch of m samples $x_\mathcal{B} = \{x_1, ..., x_m\}$ from the training data.
3: Encode the minibatch:
$\mu_\mathcal{B}, \Sigma_\mathcal{B} = q_\phi(x_\mathcal{B})$
4: Sample noise for the minibatch:
$\varepsilon_\mathcal{B} \sim \mathcal{N}(0, I)$
5: Reparameterize the noise:
$z_\mathcal{B} = \Sigma_\mathcal{B} \cdot \varepsilon_\mathcal{B} + \mu_\mathcal{B}$
6: Reconstruct the minibatch from the reparameterized parameters:
$\hat{x}_\mathcal{B} = p_\theta(z_\mathcal{B})$
7: Compute the loss:
$L(x_\mathcal{B}) = \|x_\mathcal{B} - \hat{x}_\mathcal{B}\|_1 + C \sum_{j=1}^{k}(\Sigma^2_{\mathcal{B}j} + \mu^2_{\mathcal{B}j} - \log(\Sigma^2_{\mathcal{B}j}) - 1)$
8: Update the parameters ϕ and θ by the gradients of the loss:
$\theta_{k+1} := \theta_k + \nabla_\theta L(x_\mathcal{B})$
$\phi_{k+1} := \phi_k + \nabla_\phi L(x_\mathcal{B})$
9: **end for**



(a) (b)

Figure 2.3: 2D encoding of MNIST by VE (a) and VAE (b)

Figure 2.3 visualizes the 2D encoding of MNIST [55] by AE and VAE. We can observe that the coding of VAE is more compact than that of AE, and the distribution is close to the normal distribution.

2.1.2 Generative Adaversarial Networks

The idea of Generative Adversarial Networks (GAN) [15] originates from the zero-sum game in game theory. As shown in Figure 2.4, it consists of two parts, a

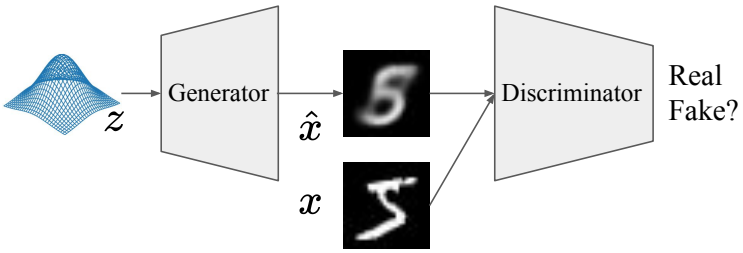


Figure 2.4: Structure of GAN

generative network (generator, G) and a discriminative network (discriminator, D). Similar to the decoder of the VAE, the generator maps random hidden variable $z \sim \mathcal{N}(0, I)$ to the data space to generate fake data $\hat{x}$. The discriminator is a binary classifier predicting whether its input is real.

Mathematically, D and G are trained in an adversarial paradigm with the following value function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))], \quad (2.20)$$

where $p_{data}(x)$ and $p_z(z)$ represent the real data distribution and the random noise distribution, respectively.

The discriminator has two kinds of inputs, the real data $x \sim p_{data}(x)$ and the fake data $G(z)$ generated by the generator from the noise $z \sim p_z(z)$. It is trained by the Cross-Entropy loss:

$$L_D = -\frac{1}{m}\sum_{i=1}^{m}\left[\log D(x_i) + \log(1 - D(G(z_i)))\right], \quad (2.21)$$

where m denotes the batch size.

The generator takes the noise $z \sim p_z(z)$ as input and generates the fake data $G(z)$, which is expected to trick the discriminator. Thus, its loss function is as,

$$L_G = \frac{1}{m}\sum_{i=1}^{m}\log(1 - D(G(z_i))). \quad (2.22)$$

However, the generated data is very unrealistic at the beginning of the training, such that the discriminator can easily identify them correctly. Therefore, $D(G(z_i))$

tends to be zero. The derivative of Equation 2.22 with respect to the generator parameters θ_G is as:

$$\frac{\mathrm{d}L_G(z_i)}{\mathrm{d}\theta_G} = -\frac{1}{1 - D(G(z_i))} \cdot \frac{\mathrm{d}D(G(z_i))}{\mathrm{d}G(z_i)} \cdot \frac{\mathrm{d}G(z_i)}{\mathrm{d}\theta_G}. \tag{2.23}$$

On the right side of the above equation, the first term tends to 1, while the second term tends to 0 (because $D(G(z_i))$ is roughly a constant function whose derivative is 0), causing the overall derivative to tend to 0. Therefore, in practice we usually use the following modified version of the generator loss:

$$L_G = -\frac{1}{m}\sum_{i=1}^{m} \log(D(G(z_i))), \tag{2.24}$$

whose derivative with respect to θ_G is as:

$$\frac{\mathrm{d}L_G(z_i)}{\mathrm{d}\theta_G} = -\frac{1}{D(G(z_i))} \cdot \frac{\mathrm{d}D(G(z_i))}{\mathrm{d}G(z_i)} \cdot \frac{\mathrm{d}G(z_i)}{\mathrm{d}\theta_G}. \tag{2.25}$$

The above equation is similar to Equation 2.24, except for the first term, which tends to infinity rather than 1. Thus the derivative calculated by Equation 2.25 is much bigger than that of Equation 2.24. The training of GAN follows Algorithm 2:

Algorithm 2 Training algorithm of GAN

Input: training dataset $\{x_i\}$; G and D; D update step k per G update
Output: Parameters of G and D: θ_G and θ_D

1: **for** number of training iterations **do**
2: **for** k steps **do**
3: Sample minibatch of m samples $\{x_1, ..., x_m\}$ from real data $p_{data}(x)$.
4: Sample minibatch of m noise samples $\{z_1, ..., z_m\}$ from noise prior $p_z(z)$.
5: Update the discriminator parameters:
$$\theta_D^{k+1} := \theta_D^k + \nabla_{\theta_D} \frac{1}{m} \sum_{i=1}^{m} [\log D(x_i) + \log\left(1 - D(G(z_i))\right)]$$
6: **end for**
7: Sample minibatch of m noise samples $\{z_1, ..., z_m\}$ from noise prior $p_z(z)$.
8: Update the generator parameters:
$$\theta_G^{k+1} := \theta_G^k + \nabla_{\theta_G} \frac{1}{m} \sum_{i=1}^{m} \log(D(G(z_i)))$$
9: **end for**

2.2 Modification of GANs

2.2.1 Problems of the traditional GAN

GANs have faced two main problems since their appearance: 1. training instability, which requires careful balancing of the generator and the discriminator during training; and 2. mode collapse, which leads to a lack of diversity in the generated samples. In this section, we describe it in detail.

As discussed previously, the discriminator (D) minimizes

$$L_D = -\mathbb{E}_{x\sim P_r}[\log D(x)] - \mathbb{E}_{x\sim P_g}[\log(1 - D(x))], \tag{2.26}$$

where P_r is the real data distribution, and P_g is the generated data distribution.

The generator (G) minimizes either

$$L_G = \mathbb{E}_{x\sim P_g}[\log(1 - D(x))] \tag{2.27}$$

or

$$L_G = \mathbb{E}_{x\sim P_g}[-\log D(x)]. \tag{2.28}$$

However, they both have flaws.

- *Equation 2.27 suffers from the vanishing gradients problem: the better the discriminator is trained, the more severe the gradient vanishing.*
- *Equation 2.28 is equivalent to an unreasonable distance metric, resulting in unstable gradient and mode collapse.*

We will prove them separately in the following.

By letting the derivative of Equation 2.26 with respect to $D(x)$ be zero, the optimal discriminator is derived as:

$$D^*(x) = \frac{P_r(x)}{P_r(x) + P_g(x)}. \tag{2.29}$$

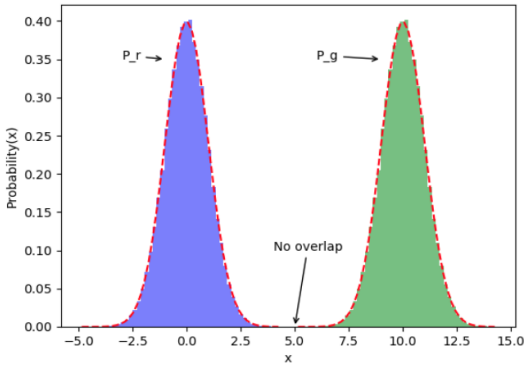


Figure 2.5: Example of two probability distributions that do not overlap.

To facilitate the derivation, we add a generator-independent term to Equation 2.27 (thus not changing the optimization objective): $\mathbb{E}_{x\sim P_r}[\log(D(x))]$, and substitute the discriminator with the optimal one in Equation 2.29.

$$\begin{aligned} L_G &= \mathbb{E}_{x\sim P_g}[\log(1-D(x))] + \mathbb{E}_{x\sim P_r}[\log D(x)] \\ &= \mathbb{E}_{x\sim P_g}\left[\log\frac{P_g(x)}{\frac{1}{2}[P_r(x)+P_g(x)]}\right] + \mathbb{E}_{x\sim P_r}\left[\log\frac{P_r(x)}{\frac{1}{2}[P_r(x)+P_g(x)]}\right] - 2\log 2 \\ &= KL(P_g\|\frac{1}{2}(P_r+P_g)) + KL(P_r\|\frac{1}{2}(P_r+P_g)) - 2\log 2 \end{aligned} \tag{2.30}$$

According to the definition of the JS divergence, we have

$$L_G = 2JSD(P_r\|P_g) - 2\log 2. \tag{2.31}$$

Therefore, minimizing the generator loss of Equation 2.27 is equivalent to minimizing the JS divergence of P_r and P_g, which could be discretized as,

$$\begin{aligned} JSD(P_r(x)\|P_g(x)) &= \frac{1}{2}\sum P_r(x)\log\frac{2P_r(x)}{P_r(x)+P_g(x)} + \frac{1}{2}\sum P_g(x)\log\frac{2P_g(x)}{P_r(x)+P_g(x)} \\ &= \frac{1}{2}\left(\sum P_r(x)\log\left(\frac{P_r(x)}{P_r(x)+P_g(x)}\right) + \sum P_g(x)\log\left(\frac{P_g(x)}{P_r(x)+P_g(x)}\right)\right) + \log 2. \end{aligned} \tag{2.32}$$

P_r and P_g barely overlap at the beginning of training(as shown in Figure 2.5[2], $P_r/(P_r+P_g) \to 1$, $P_g/(P_r+P_g) \to 1$), which makes Equation 2.32 converge to a constant log2 under the optimal discriminator, further resulting in the vanishing

[2]Figure take from https://blog.csdn.net/Invokar/article/details/88917214

gradients of the generator. Thus, when we use Equation 2.27 as the generator loss, it is crucial to balance the training of the generator and the discriminator to avoid the vanishing gradients problem and update the generator efficiently.

Next, we concentrate on the second type of generator loss shown in Equation 2.28. We start by representing $KL(P_g\|P_r)$ using the optimal discriminator of Equation 2.29,

$$\begin{aligned} KL(P_g\|P_r) &= \mathbb{E}_{x\sim P_g}\left[\log\frac{P_g(x)}{P_r(x)}\right] \\ &= \mathbb{E}_{x\sim P_g}\left[\log\frac{P_g(x)/(P_r(x)+P_g(x))}{P_r(x)/(P_r(x)+P_g(x))}\right] \\ &= \mathbb{E}_{x\sim P_g}\left[\log\frac{1-D^*(x)}{D^*(x)}\right] \\ &= \mathbb{E}_{x\sim P_g}\log[1-D^*(x)] - \mathbb{E}_{x\sim P_g}\log D^*(x). \end{aligned} \tag{2.33}$$

Based on Equation 2.30 and Equation 2.31, we have

$$\mathbb{E}_{x\sim P_g}[\log(1-D^*(x))] + \mathbb{E}_{x\sim P_r}[\log D^*(x)] = 2JSD(P_r\|P_g) - 2\log 2. \tag{2.34}$$

Combining Equation 2.34 with Equation 2.33 yields

$$\begin{aligned} \mathbb{E}_{x\sim P_g}[-\log D^*(x)] &= KL(P_g\|P_r) - \mathbb{E}_{x\sim P_g}\log[1-D^*(x)] \\ &= KL(P_g\|P_r) - 2JSD(P_r\|P_g) + \mathbb{E}_{x\sim P_r}[\log D^*(x)] + 2\log 2, \end{aligned} \tag{2.35}$$

where the last two terms are independent to the generator. Thus the second type of generator loss is equivalent to $KL(P_g\|P_r) - 2JSD(P_r\|P_g)$.

The above transformation reveals that the optimization objective defined by Equation 2.28 contains two mutually exclusive terms: the $KL(P_g\|P_r)$ and the $-2JSD(P_r\|P_g)$. The former aims to reduce the distance between P_g and P_r, but the latter increases this distance, resulting in gradient instability during training.

Whats more, $KL(P_g\|P_r) = \int P_g \log P_g/P_r \mathrm{dx}$ is not symmetric:

- If $P_r(x) \to 1$ and $P_g(x) \to 0$, x tends to be real instance that cannot be generated by the generator. This case contributes very little to the loss function.

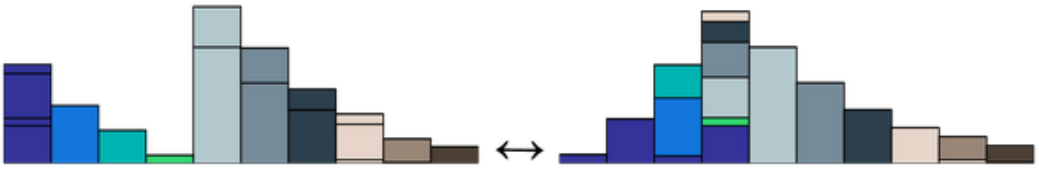


Figure 2.6: Illustration of the EMD. The different color blocks on the left are transported to the position on the right of the same color.

- If $P_r(x) \to 0$ and $P_g(x) \to 1$, x tends to be fake. The corresponding KL divergence will be ∞.

This asymmetry criterion causes the generator to prefer to produce limited kinds of "safe" samples, resulting in insufficient generating diversity, i.e., mode collapse.

2.2.2 Wasserstein GAN

Wasserstein GAN [56] successfully addresses the problems described in the previous section. Instead of the JS/KL divergence used above, it leverages the 1-Wasserstein distance (Earth-Mover distance, EMD)

$$\begin{aligned} \mathcal{W}[P_r, P_g] &= \inf_{\gamma \in \Pi(P_r, P_g)} \iint \gamma(x, y) \|x - y\| \mathrm{d}x \mathrm{d}y \\ &= \inf_{\gamma \in \Pi(P_r, P_g)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] \end{aligned} \tag{2.36}$$

where $\Pi(P_r, P_g)$ denotes the set of all joint distributions $\gamma(x, y)$ whose marginals are respectively P_r and P_g, i.e.,

$$\int \gamma(x, y) \mathrm{d}y = P_r(x), \quad \int \gamma(x, y) \mathrm{d}x = P_g(y). \tag{2.37}$$

Intuitively, if we consider P_r and P_g as two piles of earth, $\Pi(P_r, P_g)$ is a set of transport plans that transform P_r into P_g, the EMD is the minimum "cost" corresponding to the optimal transportion. Figure 2.6[3] illustrates how a pile of earth can be transported into another shape and position, with the transport plan represented by different colors.

[3]Figure taken from https://vincentherrmann.github.io/blog/wasserstein/

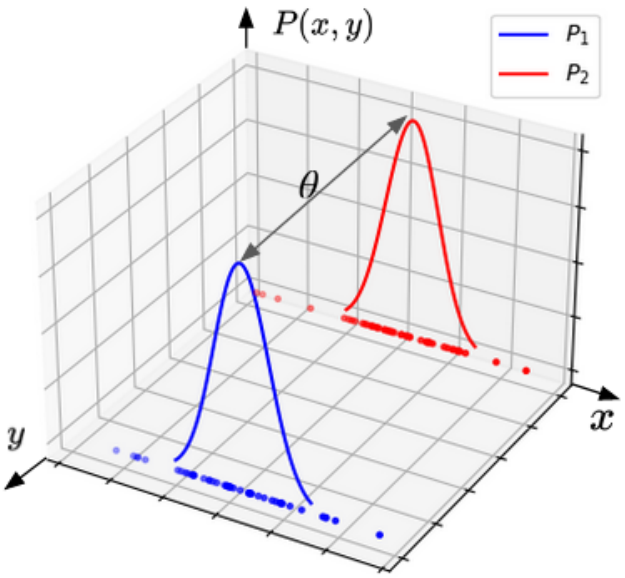


Figure 2.7: Two 2D distributions that do not overlap.

Wasserstein distance is superior to the KL/JS divergences in that it quantifies the distance of two distributions that do not overlap at all. Take two 2D distributions in Figure 2.7 as an example: P_1 and P_2 are one-dimensional distributions with the same shape, placed at different locations along the y-axis of the 3D space at a distance of θ . Obviously,

$$KL(P_1\|P_2) = \begin{cases} +\infty, & \text{if } \theta \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.38}$$

$$JSD(P_1\|P_2) = \begin{cases} \log 2, & \text{if } \theta \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{2.39}$$

$$\mathcal{W}(P_1, P_2) = \inf_{\gamma \in \Pi(P_1, P_2)} \mathbb{E}_{(x,y)\sim\gamma}[\|x - y\|] = \theta. \tag{2.40}$$

The JS/KL divergences are not continuous, taking either the maximum or minimum value. In comparison, the Wasserstein distance is smooth and correctly measures the distance between the two distributions. If we were to optimize θ using gradient descent, the first two could not provide an effective gradient at all, but the Wasserstein distance could. While training GAN to map low-dimensional noise vectors to

the high-dimensional image space, the generated distribution has a high probability of not overlapping with the real distribution, thus making the KL/JS divergences ineffective optimization targets.

The infimum in the definition of the Wasserstein distance (Equation 2.36) is highly intractable, making it difficult to serve as a loss function. Thanks to the Kantorovich-Rubinstein duality [57], it has the following equivalent form:

$$\mathcal{W}(P_r, P_g) = \frac{1}{K} \sup_{\|f\|_L \leq K} \mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)], \tag{2.41}$$

where $\|f\|_L \leq K$ denotes the space of K-Lipschitz continuous functions $f : \mathcal{X} \to \mathbb{R}$.

Given two metric spaces (X, d_x) and (Y, d_Y), where d_x denotes the metric on the set X and d_Y is the metric on the set Y, a function $f : X \to Y$ is called Lipschitz continuous if there exists $K \in \mathbb{R}, K \leq 0$ such that $\forall x_1, x_2 \in X$,

$$d_Y(f(x_1), f(x_2)) \leq K d_x(x_1, x_2), \tag{2.42}$$

K is called the Lipschitz constant of the function f. Simply, if $f : \mathbb{R} \to \mathbb{R}$, the above requirement is equivalent to restricting the absolute value of the derivative of f not exceeding K.

Equation 2.41 searches function $\|f\|_L \leq K$ that maximizes $\mathbb{E}_{x \sim P_r}[f(x)] - \mathbb{E}_{x \sim P_g}[f(x)]$. If we use w to parametrize a family of functions $\{f_w\}_{w \in \mathcal{W}}$ that are all K-Lipschitz for some K, the optimization problem is converted to

$$K\mathcal{W}(P_r, P_g) = \max_{w \in \mathcal{W}} \mathbb{E}_{x \sim P_r}[f_w(x)] - \mathbb{E}_{x \sim P_g}[f_w(x)]. \tag{2.43}$$

In practice, $f_w(x)$ could be approximated by a neural network due to its powerful fitting capabilities. In order for the neural network to be Lipschitz continuous, [56] simply clamps the network's parameters to a fixed range $\mathcal{W} = [-c, c]$, i.e. $[-0.01, 0.01]$ in the paper, after each gradient update. In this case, $\frac{\partial f_w}{\partial x}$ will not exceed a certain value, the Lipshcitz continuity condition is satisfied.Thus we can

approximate the Wasserstein distance by maximizing

$$L = \mathbb{E}_{x\sim P_r}[f_w(x)] - \mathbb{E}_{x\sim P_g}[f_w(x)], \tag{2.44}$$

where f_w is the neural network paramatrized by $w \in [-c, c]$. The Lipschitz constant K is omitted as it is positive and only affects the magnitude of the gradient without changing its direction.

Equation 2.44 is exactly the objective function of the discriminator. It is easy to observe that its goal is to train the discriminator to maximize the distinction between the real and the generated distribution. In traditional GANs, the discriminator outputs a real number between $[0, 1]$, indicating the probability that a sample is real. In comparison, the discriminator of the Wasserstein GAN (WGAN) approximates the Wasserstein distance of two distributions, which is not limited to $[0, 1]$ anymore. Thus the last layer of the W-GAN is no longer the Sigmoid layer.

The generator maps a noise vector $z \sim p_z$ to the data space $G(z) \sim P_g$. And its goal is to minimize approximated Wasserstein distance between the generated distribution P_g and the real distribution P_r. The overall objective function of W-GAN is as

$$\min_{\theta} \max_{w} \mathbb{E}_{x\sim P_r}[D(x; w)] - \mathbb{E}_{z\sim p_z}[D(G(z; \theta); w)], \tag{2.45}$$

where w and θ are the discriminator and generator parameters, respectively.

By replacing the KL/JS divergences with Wasserstein distance, WGAN has the following achievements:

- It solves the problem of instability of adversarial training, and one no longer needs to carefully balance the training process of the generator and the discriminator.

- It solves the problem of mode collapse and generates images with a high degree of diversity.

- The Wasserstein distance could be used as a criterion indicating the training process; the smaller the value, the better the GAN is trained.

The training process of WGAN is described in Algorithm 3.

Algorithm 3 Training algorithm of WGAN. $\alpha = 5e^{-5}$, $c = 0.01$, $n_{critic} = 5$.

Require: α, the learning rate; c, the clipping parameter; m, the batch size;
n_{critic}, the number of iterations of the discriminator per generator iteration.
Require: Initial parameters of D and G: w_0, θ_0.

while θ has not converged **do**
  for $t = 1, ..., n_{critic}$ **do**
    Sample a batch of real data $\{x^{(i)}\}_{i=1}^{m} \sim P_r$.
    Sample a batch of latent variables $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$.
    $g_w \leftarrow \nabla_w[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))]$
    $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
    $w \leftarrow \text{clip}(w, -c, c)$
  end for
  Sample a batch of latent variables $\{z^{(i)}\} \sim p(z)$.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

Despite the success of WGAN, weight clipping in not a good way to ensure the Lipschitz continuity of the discriminator. It suffers from two main problems:

- The hyper-parameter of the weight clipping should be carefully tuned. Due to the cumulative effect across multiple model layers, a too large or too small clipping parameter will lead to exploding or vanishing gradients problem, respectively. As shown in Figure 2.8[4] on the left.
- As shown in the top right of Figure 2.8, weight clipping pushes the weights towards the extremes of the clipping interval, largely restricting the neural network's power.

According to the properties of the Lipschitz continuous functions, if $D : \mathcal{X} \to \mathbb{R}$ is K-Lipschitz continuous, we have,

$$\|\nabla_x D(x)\|_p \leq K. \tag{2.46}$$

Given $p = 2$ and $K = 1$, we can say that the $L2$ norm of the gradient of a 1-Lipschitz continuous function does not exceed 1. Therefore WGAN-GP [58] imposes

[4]Figure taken from [58]

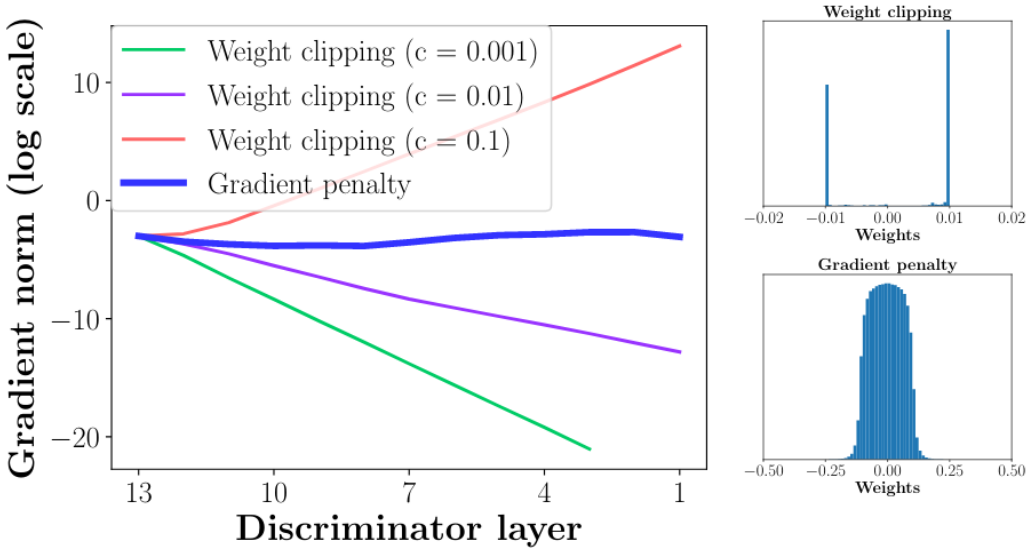


Figure 2.8: Comparison of weight clipping and gradient penalty.

a novel gradient penalty term in the discriminator loss to penalize the gradients with oversized norms. The new discriminator loss is as

$$L = -\mathbb{E}_{x\sim P_r}[D(x)] + \mathbb{E}_{x\sim P_g}[D(x)] + \lambda \mathbb{E}_{\hat{x}\sim P_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2], \tag{2.47}$$

where $P_{\hat{x}}$ is the distribution of the entire sample space. Due to the curse of dimensionality, it is challenging to estimate the expectation over the high-dimensional sample space. Therefore, the proposed method samples data by randomly interpolating the real and generated pairs,

$$\begin{aligned} x_r \sim P_r, x_g \sim P_g, \varepsilon \sim \text{Uniform}[0,1], \\ \hat{x} = \varepsilon x_r + (1-\varepsilon) x_g. \end{aligned} \tag{2.48}$$

The left side of Figure 2.8 shows that the gradients of the WGAN-GP remain stable as the training progresses and do not vanish or explode, as is the case with WGAN. The right side of Figure 2.8 shows that the weights learned by WGAN-GP are more diverse, unlike WGAN, where the weights are stacked at the extremes of the clipping interval. The training of WGAN-GP is described in Algorithm 4.

Algorithm 4 WGAN with gradient penalty. $\lambda = 10$, $n_{critic} = 5$, $\alpha = 1e^{-4}$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: λ, gradient penalty coefficient; m, the batch size; α, β_1, β_2, Adam hyperparameters; n_{critic}, the number of iterations of the discriminator per generator iteration;

Require: Initial parameters of D and G: w_0, θ_0.

while θ has not converged **do**
  for $t = 1, ..., n_{critic}$ **do**
    for $i = 1, ..., m$ **do**
      Sample $x \sim P_r$, latent variable $z \sim p(z)$, a random number $\varepsilon \sim U[0, 1]$.
      $\tilde{x} \leftarrow G_\theta(z)$
      $\hat{x} \leftarrow \varepsilon x + (1 - \varepsilon)\tilde{x}$
      $L^{(i)} \leftarrow D_w(\tilde{x}) - D_w(x) + \lambda(\|\nabla_{\hat{x}} D_w(\hat{x})\|_2 - 1)^2$
    end for
    $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$
  end for
  Sample a batch of latent variables $\{z^{(i)}\} \sim p(z)$.
  $g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
  $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} -D_w(G_\theta(z)), \theta, \alpha, \beta_1, \beta_2)$
end while

2.2.3 Spectral Normalization

As discussed in the previous section, the Lipschitz continuity is a favorable property that allows the discriminator to bypass the unreasonable probability-based loss functions, i.e., KL divergence or JS divergence. Besides, the Lipschitz continuity restricts the upper bound of the gradient norm, thus making the training more stable. To ensure Lipschitz continuity of the discriminator, WGAN uses weight clipping, while WGAN-GP imposes a gradient penalty term in the discriminator loss. For the same purpose, spectrally normalized GAN (SN-GAN) [59] applies the spectral normalization to the weights of the layers in the discriminator, making the discriminator 1-Lipschitz. We present it in detail in this section.

Unlike WGAN-GP, which restricts the overall discriminator gradient, SN-GAN focuses on the weights of each layer and makes the whole network Lipschitz continuous by normalizing the weights corresponding to each layer.

In the case of discriminators $D : I \to \mathbb{R}$, where I denotes the data space. If D

is K-Lipschitz continuous, then $\forall x, y, \in I$,

$$\|D(x) - D(y)\| \leq K\|x - y\|, \tag{2.49}$$

where $\|\cdot\|$ denotes the L_2 norm. Since the linear and convolutional layers are essentially matrix multiplication, we first study the Lipschitz continuity of the matrix.

If a linear function is K-Lipschitz at the origin, then it is K-Lipschitz everywhere. As matrix transformations are linear, a matrix $A : \mathbb{R}^n \to \mathbb{R}^m$ is K-Lipschitz if and only if

$$\|Ax\| \leq K\|x\|, \quad \forall x \in \mathbb{R}^n. \tag{2.50}$$

This is equivalent to

$$\langle Ax, Ax\rangle \leq K^2\langle x, x\rangle, \forall x \in I, \tag{2.51}$$

further equivalent to

$$\langle (A^TA - K^2)x, x\rangle \leq 0, \forall x \in I. \tag{2.52}$$

As A^TA is positive semidefinite, all of its eigenvalues are nonnegative. The orthonormal basis constructed by its eigenvectors is denoted as $\{\nu_1, \nu_2, ..., \nu_n\}$, correspondingly, the eigenvalues are denoted as $\lambda_1, \lambda_2, ..., \lambda_n$. Thus, $\forall x \in I$ could be decomposed as

$$x = x_1 \cdot \nu_1 + x_2 \cdot \nu_2 + ... + x_n\nu_n. \tag{2.53}$$

Substituting the decomposed x back to Equation 2.52 yields

$$\begin{aligned}
\langle (A^TA - K^2)x, x\rangle &= \langle (A^TA - K^2)\sum_{i=1}^{n} x_iv_i, \sum_{j=1}^{n} x_jv_j\rangle \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} x_ix_j\langle (A^TA - K^2)v_i, v_j\rangle \\
&= \sum_{i=1}^{n}(\lambda_i - K^2)x_i^2 \leq 0.
\end{aligned} \tag{2.54}$$

Since λ_i is nonnegative, to make Equation 2.54 hold for $\forall x \in \mathbb{R}^n$, we must have $K^2 \geq \max(\lambda_i)$. Suppose λ_1 is the largest eigenvalue of A^TA, we get $K \geq \sqrt{\lambda_1}$, where $\sqrt{\lambda_1}$ is also called the spectral norm of A. Thus, the matrix $A : \mathbb{R}^n \to \mathbb{R}^m$

is K-Lipschitz and the minimum of K is $\sqrt{\lambda_1}$. To make A further 1-Lipschitz, it suffices to divide $\sqrt{\lambda_1}$ over all elements of A.

For composite function $f \circ g$, we have $\|f \circ g\|_L \leq \|f\|_L \|g\|_L$. A multi-layer neural network is exactly the composition of matrix multiplication and nonlinear activation functions. The commonly used activation functions, e.g., ReLU, LeakyReLU, Sigmoid, are all 1-Lipschitz and do not affect the overall Lipschitz norm of the model. Thus, to make the discriminator 1-Lipschitz, we only need to ensure the convolution/linear layers to be 1-Lipschitz, which can be done by dividing the weights by their spectral norms. The remaining task is to efficiently calculate the spectral norm of the weight in the convolution/linear layer.

SN-GAN uses power iteration to estimate the spectral norm of a matrix (layer). Suppose A is a full rank matrix of $n \times n$. Its eigenvalues are $\lambda_1, \lambda_2, ..., \lambda_n$ (in descending order), and the corresponding normalized eigenvectors are $\nu_1, \nu_2, ..., \nu_n$. $\forall x \in \mathbb{R}^n, x \neq \vec{0}$, we have $x = x_1 \cdot \nu_1 + x_2 \cdot \nu_2 + ... + x_n \cdot \nu_n$, thus

$$\begin{aligned} Ax &= A(x_1 \cdot \nu_1 + x_2 \cdot \nu_2 + ... + x_n \cdot \nu_n) \\ &= x_1(A\nu_1) + x_2(A\nu_2) + ... + x_n(A\nu_n) \\ &= x_1(\lambda_1\nu_1) + x_2(\lambda_2\nu_2) + ... + x_n(\lambda_n\nu_n). \end{aligned} \tag{2.55}$$

We iterate k times of Equation 2.55 and get

$$\begin{aligned} A^k x &= x_1(\lambda_1^k\nu_1) + x_2(\lambda_2^k\nu_2) + ... + x_n(\lambda_n^k\nu_n) \\ &= \lambda_1^k \left[x_1\nu_1 + x_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k \nu_2 + ... + x_n \left(\frac{\lambda_n}{\lambda_1}\right)^k \nu_n \right] \end{aligned} \tag{2.56}$$

Given $\lambda_1 > \lambda_2 > ... > \lambda_n$ (the case where two eigenvalues are equal is not considered as it is too rare for randomly initialized weights), we have

$$\lim_{k \to +\infty} \left(\frac{\lambda_i}{\lambda_1}\right)^k = 0, \quad \text{for} \quad i > 1. \tag{2.57}$$

Thus

$$\lim_{k \to +\infty} A^k x = \lambda_1^k x_1 \nu_1. \tag{2.58}$$

where x_1 and λ_1 are scalars and ν_1 is a vector. That is, $\forall x \in \mathbb{R}^n, x \neq \vec{0}$, after multiplying sufficient times of A, we get the scaled dominant eigenvector of A. Due to the high dimensionality of W, computing W^TW explicitly is computationally intensive. The power iteration in the paper is formulated as

$$\tilde{v} := \frac{W^T\tilde{u}}{\|W^T\tilde{u}\|_2} \tag{2.59a}$$

$$\tilde{u} := \frac{W\tilde{v}}{\|W\tilde{v}\|_2} \tag{2.59b}$$

where $\tilde{u}$ is initialized by a random vector. Substituting Equation 2.59b into Equation 2.59a yields a fractional expression with a numerator of $W^TW\tilde{v}$. The normalization operation in each step ensures $\tilde{v}$ to be a unit vector while making the algorithm numerically stable.

Lastly, we left multiply both sides of Equation 2.59b by $\tilde{u}^T$,

$$\begin{aligned} \tilde{u}^T\tilde{u} &= \frac{\tilde{u}W\tilde{v}}{\|W\tilde{v}\|_2} \\ 1 &= \frac{\tilde{u}^TW\tilde{v}}{\sqrt{\tilde{v}^T(W^TW\tilde{v})}} \\ 1 &= \frac{\tilde{u}^TW\tilde{v}}{\lambda_1\tilde{v}^T\tilde{v}} \\ 1 &= \frac{\tilde{u}^TW\tilde{v}}{\sqrt{\lambda_1}} \end{aligned} \tag{2.60}$$

Thus, the spectral norm is

$$\sqrt{\lambda_1} = \tilde{u}^TW\tilde{v}. \tag{2.61}$$

Combining Equation 2.59a, Equation 2.59b and Equation 2.61 can efficiently approximate the spectral norm corresponding to each convolution/linear layer.

The learning rate is relatively low compared to the weight W. Thus W does not change much in each update and can be considered constant from the power iteration part. Consequently, we can combine the power iteration and the weight update during training, that is, after each weight update, update $\tilde{u}$ and $\tilde{v}$ as well as applying the spectral normalization (divice W by $\sqrt{\lambda_1}$).

Compared to other methods, SN-GAN exhibits better generation performance

as well as better robustness to different training settings. In addition, the spectral normalization is computationally efficient and easy to be integrated into other methods, thus widely used in GAN-based applications, including SA-GAN [60], BigGANs [33], StyleGAN [32], *etc.* Due to its excellent properties, we integrate this module into all GAN-related models in the subsequent chapters.

2.2.4 Self-Attention GAN

Due to the limited kernel size of the convolution layer, each convolution operation can only cover a small area around a pixel and cannot capture the long-term dependencies of different image regions. Traditional convolutional neural networks leverage stacked convolution and pooling layers to mitigate this problem. However, such an approach is neither straightforward nor efficient, and there is no guarantee that feature dependencies are well captured after passing through stacked layers. However, such an approach is neither straightforward nor efficient, and there is no guarantee that feature dependencies are well captured after passing through stacked layers. Specifically for the image generation task, traditional network structures have difficulty learning structural patterns that occur consistently in certain classes (e.g., dogs have well-defined independent four legs that can neither be more nor less).

Contextual Attention [43] reshapes the image features to a convolutional kernel to force the distant textures to interact with each other. However, the method is designed only for image completion and has little follow-up work. Self-Attention GAN (SAGAN) successfully applies the self-attention mechanism [61] to image generation, pioneered the self-attention-based generation methods [62, 63]. The self-attention mechanism is also widely used in natural language processing [64], but this is beyond the scope of this thesis. We focus on SAGAN with the overall structure shown in Figure 2.9[5].

Suppose the input feature to the Self-Attention (SA) layer is as $x \in \mathbb{R}^{B\times C\times H\times W}$, where B is the batch size, C is the number of channels, and $H \times W$ are the height and width of the feature map. To simplify the notation, we assume $B = 1$ and omit this term in the equations, i.e. $x \in \mathbb{R}^{C\times H\times W}$.

[5]Figure taken from[60]

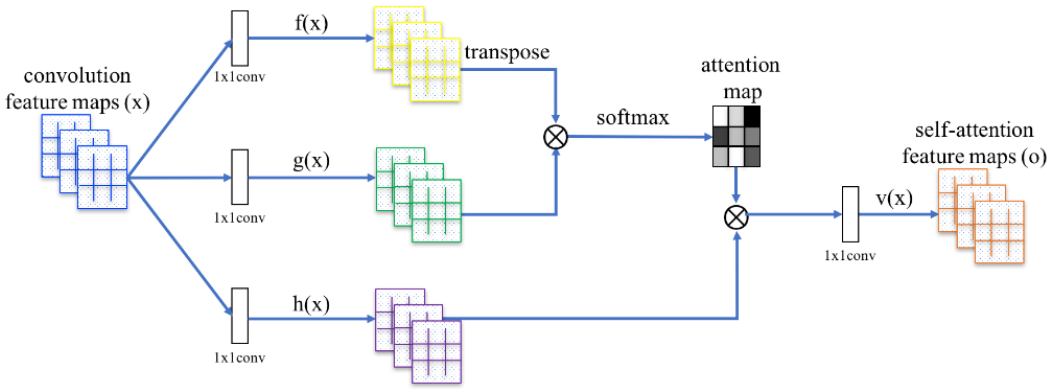


Figure 2.9: Illustration of the Self-Attention layer. $\otimes$ denotes matrix multiplication.

The convolution is essentially the matrix multiplication. Thus the feature x could be reshaped as $C \times N$, and the convolution operation is formulated as Wx where W represents the kernel's weight.

The attention map β is a normalized weight sized $N \times N$ for each feature location,

$$\beta_{j,i} = \frac{\exp(s_{ij})}{\sum_{i=1}^{N} \exp(s_i j)}, \quad \text{where} \quad s = \mathbf{f}(x)^T \mathbf{g}(x), \tag{2.62}$$

$\beta_{j,i}$ indicates the influence of the features on the i^{th} location to the j^{th} location. $\mathbf{f}(x), \mathbf{g}(x) \in \mathbb{R}^{N \times C}$ are two convolution operations calculated by $\mathbf{f}(x) = W_f x$, $\mathbf{g}(x) = W_g x$, where W_f and W_g are the matrixed 1×1 sized kernels' weights, respectively. The input feature itself is also mapped to a new feature by a 1×1 convolution with channels unchanged,

$$\mathrm{h}(x) = W_h x. \tag{2.63}$$

Based on the output of the Equation 2.62 and Equation 2.63, the output of the SA layer is as

$$o = \gamma \mathrm{h}(x)\beta + x, \tag{2.64}$$

where γ is also a learned parameter, and is initialized by 0. The intuition behind this design is to allow the attention map to gradually exerts its influence as the training progresses.

2.3 Evaluation Metrics of GAN

2.3.1 Inception Score

To evaluate the performance of a generative model, we need to focus on two aspects: 1) whether the generated images are photorealistic and 2) whether the generated images are diverse (Some generative models can only simulate the distribution of a subset of the training data, which are caught in the so-called mode collapse).

Inception Score (IS) [65] considers these two aspects in the following way:

- **Reality**: Feed generated image x to the Inception V3 Network pre-trained on the ImageNet dataset. The model will output a 1000-dimensional vector with each dimension representing the probability of x being to one class. Intuitively, a realistic image should have a high probability on one class and low probability on others. Formally, the entropy of $p(y|x)$ should be low.

- **Diversity**: If the generation is sufficiently diverse, the generated distribution should be even across all categories. Assuming that we generate 10 000 images, then ideally, ten images should be generated in each of the 1000 categories. Formally, the marginal distribution of the generated images across all categories $p(y)$ should be high.

A good generator should have a sharp $p(y|x)$ and a smooth $p(y)$, thus the KL divergence of the two distributions should be high. Therefore, the IS is formulated as

$$\mathrm{IS}(G) = \exp\left(\mathbb{E}_{x\sim p_g} D_{KL}(p(y|x)\|p(y)\right), \tag{2.65}$$

where $p(y)$ is approximated by the empirical distribution $\hat{p}(y)$,

$$\hat{p}(y) = \frac{1}{N}\sum_{i=1}^{N} p(y|x^{(i)}), \tag{2.66}$$

where N is the number of synthesized image. And the KL divergence is approximated by

$$D_{KL}(p(y|x)\|p(y)) = \frac{1}{N}\sum_{i=N}^{N} p(y|x^{(i)})\log\frac{p(y|x^{(i)})}{\hat{p}(y)}. \tag{2.67}$$

We take the log of Equation 2.65 and make a simple derivation:

$$\begin{aligned}
\log(\mathrm{IS}(G)) &= \mathbb{E}_{x\sim p_g} D_{KL}(p(y|x)\|p(y)) \\
&= \sum_x p_g(x) \sum_i p(y=i|x) \log \frac{p(y=i|x)}{p(y=i)} \\
&= \sum_{x\sim p_g, y} p(x,y) \log \frac{p(x,y)}{p(y)p(x)} \\
&= \sum_{x\sim p_g, y} p(x,y) \log \frac{p(x,y)}{p(x)} - \sum_y \log p(y) \sum_x p(x,y) \\
&= \sum_{x\sim p_g, y} p(x,y) \log p(y|x) - \sum_y p(y) \log p(y) \\
&= \mathbb{E}_{x\sim p_g} p(y|x) \log p(y|x) - \sum_y p(y) \log p(y)
\end{aligned} \tag{2.68}$$

The effectiveness of IS lies in its ability to recover good estimates of 1) $p(y)$, the marginal class distribution across the generated images, and of 2) $p(y|x)$, the conditional class distribution for generated images x.

However, when the GAN is trained on datasets different from the ImageNet, estimating the above two distributions is problematic. [66] calculated the marginal distribution of CIFAR-10 [67] using Equation 2.66, and find that the top 10 predicted classes are obscured and not relevant to their true categories, which means that $\hat{p}(y)$ is not a good estimation to $p(y)$. As Equation 2.68 shows, the IS uses $\mathbb{E}_{x\sim p_g} p(y|x) \log p(y|x)$ to measure the generation quality, [66] thus calculated this term for CIFAR-10 images, images of random pixels, and images in the test set of ImageNet, and the results are 4.664 bits, 6.512 bits, and 1.97 bits, respectively, meaning that CIFAR-10 images are closer to random pixels than to images of ImageNet under this metric, which is not reasonable.

The root cause of the above problem is that IS only considers the generated images and ignores the real data in its formula, i.e., IS cannot reflect the distance between the real and the synthesized data. The basis for IS to judge the reality of the data is from the training set of Inception V3: ImageNet. Thus, in the "worldview" of Inception V3, any image that does not look like ImageNet is not real.

In conclusion, the Inception Sore is not a good measure of the GANs' perfor-

mance. Most GAN-related literature has now abandoned this metric in favor of the more reasonable FID score [68].

2.3.2 FID score

The Fréchet Inception Distance (FID) was proposed by [69] which is originally used to measure the distance between two multivariate normal distributions. [68] use it to measure the distance between the generated and the real image distributions.

As is known, pre-trained neural networks can extract the high-level information of an image, which reflects the essence of the image to some extent. Therefore, [68] calculates the FID based on the features extracted by the pre-trained Inception V3 network. The FID is formulated as

$$\mathrm{FID}(p_g, p_r) = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}), \tag{2.69}$$

where $\mu\_$, $\Sigma\_$ denote the mean and covariance matrix of the features extracted from the real and synthetic data.

FID only uses Inception V3 as a feature extractor, thus does not rely on an image's predicted class probability distribution. It has the following advantages over IS:

- The training set of the generated model can be different from the training set of Inception V3.
- The calculation involves both the generated data and the real data, which is more reasonable than IS. It can be understood that IS measures the generation quality by comparing the generated data with the ImageNet data. In contrast, FID compares the generated data with the training data and is, therefore, more reasonable.

2.4 Summary

This chapter mainly focuses on the theoretical underpinnings of the various techniques applied later in the thesis.

Section 2.1 describes two representative research directions in generative models, namely, variational autoencoders and the GANs. The former is based on variational inference with a beautiful mathematical form, and the latter leverages adversarial training, which lacks mathematical rigor but can generate surprisingly good results. Traditional GANs are difficult to train and are very sensitive to training parameters and model structure.

Section 2.2 presents several improvements to the traditional GANs. First, we analyze the problems of the JS/KL divergence-based losses. Then we introduce the Wasserstein Distance-guided GAN: WGAN and WGAN-GP. Next, we present the spectral normalization technique, which ensures the Lipschitz continuity of the model. And finally, we introduce the self-attention mechanism, which facilitates the model to capture the long-term dependencies of distant image regions.

Section 2.3 first introduces the Inception Score (IS), a widely used evaluation metric in the early days of GAN. We analyze the limitations of IS and then lead to the FID score, the current standard evaluation criterion.

Chapter 3

Pixel Sampling for Style Preserving Face Pose Editing

3.1 Introduction

Face pose editing aims to change the pose of an input face image while keeping its original identity unchanged. It has many potential applications, *e.g.*, face recognition, movie industry and entertainment. The current state-of-the-art has featured two main research lines in this field, *i.e.*, 3D reconstruction-based, and simple 2D based.

For 3D reconstruction-based approaches, face pose editing is achieved by either mapping the 2D face images to 3D face models with fixed or regressed parameters [70, 71, 21] or directly regressing the UV map [4, 72] of the input face. The advantage of such models is that pose control is not demanding. With the reconstructed 3D face, face images at any target pose can be obtained by 3D geometrical transformation and 2D projection. However, regressing either the parameters of predefined 3D models or the UV map requires large amounts of high-quality training data. Moreover, due to the restriction of the predefined model and the missing texture of extreme poses, fine details of the images are ignored. As a result, the faces generated by these approaches are generally not photo-realistic enough and require further refinements[73].

Thanks to the development of Generative Adversarial Networks (GAN) [15], a number of GAN based 2D approaches to face pose editing have been proposed in recent years. GAN has achieved great success in face image inpainting and facial attribute editing [74, 75, 76, 77, 78]. However, the existing methods are generally

only capable of editing the subtle attributes or local regions of the image, whereas the global structure remains almost unchanged. Regarding face pose manipulation, when changing the view angle from side to front, not only the local texture but also the global shape of the face image dramatically changes. Despite these difficulties, there still exist significant efforts tackling this problem [16, 17, 18, 79, 19]. Most of the methods are implemented by an encoder-decoder structured network, with a bottleneck layer in the middle, where the faces are first encoded into a low dimensional feature vector, and then decoded into the image space conditioned by the pose information, *e.g.*, CR-GAN [79], DR-GAN [16]. However, there exists an intrinsic trade-off between the image style conserving capability and the identity preserving ability in the compact deep feature space, *i.e.*, it is hard to model the expertise of both the face identity and other image properties, such as lightning condition, saturation, background color, *etc.*

To highlight the aforementioned dilemma that commonly incurs in current 2D based methods, we remove the face classification branch of DR-GAN [16] (with the latent feature dimensionality of 320) and train the model only with the adversarial loss and the reconstruction loss. In this case, an adversarial auto-encoder (AE) is achieved, where the reconstruction loss aims to efficiently preserve the style of the input image, and the adversarial loss enforces the generated images photo-realistic. Figure 3.1 illustrates the input images (the first row) and the results obtained by the adversarial auto-encoder (second row) and DR-GAN (third row), respectively. As can be seen, the auto-encoder properly preserves the style of the input image, but it fails maintaining the identities. The reconstructed faces by DR-GAN successfully catch the identity characteristics of the input images, whereas the output ones are distorted and present obvious artifacts. If it is even painful for the model to faithfully rebuild the given input face in terms of both style and the identity without any pose manipulation, how can we further expect it to preserve them after changing the pose?

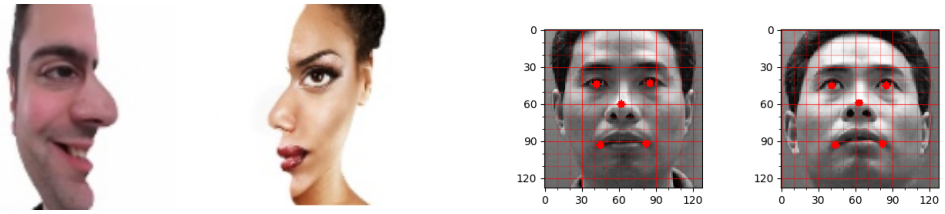
To fight the trade-off incurred by the low-dimensional restriction in the feature space, we seek solutions from the high-dimensional embeddings. But to make the condition label not ignored by the decoder, the encoding dimension should not be

Figure 3.1: Illustration of the trade-off between identity preserving and style preserving.

simply increased. The classical structure of U-Net [80], which adds skip-connections between symmetric layers of the encoder and the decoder, is able to prevent the problem of over-compression by concatenating features from the shallow layers in the reconstruction path. Therefore, we add skip-connections to the DR structure and retrain the model. The last row of Figure 3.1 shows the corresponding reconstruction results, where both the identity and the style of the input face is well preserved. High-dimensional embedding is indeed promising in image synthesis, however, structures like U-Net convey too much low-level details, making it much more challenging to edit the face pose than on the low-dimension features, especially for the extreme shape changes. Therefore, how to enable face pose editing in the high dimensional feature space is the main problem to be solved.

To tackle the challenge above, we present a novel two-stage method and a module named "**P**ixel **A**ttention **S**ampling" (PAS) in this chapter. Inspired by the fact that face images of different view angles also share a large number of similar pixels as highlighted by the optical illusion of face images [81, 82] in Figure 3.2(a), we believe that these pixels are significant to construct the texture of a face image in the target view through sampling. Specifically, given a target pose, this PAS module selects pixels from the input image and slightly change their relative locations in a learning manner to match the target pose (similarly to a non-linear image warping). Thus

the recovered face editing result possesses the target pose and shares the original texture simultaneously, faithfully keeping the identity information and image style unchanged. Due to the lack of texture in invisible regions, the results of PAS would possibly contain noises and holes, then the main task can be cast as image inpainting, which has been extensively studied. We feed the intermediate pose-edited face image into the aforementioned U-Net, so that the noises can be filtered out and holes filled. By incorporating the module of PAS, the low-level details preserved by the U-Net are no longer burdensome for the task of pose editing, instead, they become useful information for generating the visually compelling face images.



(a) (b)

Figure 3.2: (a) Example of front/profile optical illusion. Indicating that face images in different view angles still share pixel-level similarities. (b) The ambiguity of representing 3D face pose by 2D landmarks. The two faces above have almost the same landmark distribution, but are in different poses..

Further, by introducing the 3D landmarks rather than 2D ones to represent the head pose more precisely, we achieve a better flexibility of pose manipulation. On the contrary, the traditional methods like DR-GAN and CR-GAN merely manipulate face images in several discrete yaw angles, and TP-GAN [17] can only frontalize face images. Although CAPG-GAN [18] uses 2D landmarks to guide the generation, it cannot generate faces in arbitrary poses as it claims, since using 2D landmarks to represent 3D angles can bring ambiguity as Figure 3.2(b) shows. Besides, the 3D landmarks tends to provide richer shape-related information, further facilitating the synthesis of face images.

In summary, our main contributions are as follows:

- A novel two-stage face pose editing method is proposed, which casts the task of face pose manipulation as face inpainting, thereby enabling it fully utilize the

fine details of the given input image by exploiting high-dimensional embedding.

- A new "**P**ixel **A**ttention **S**ampling" module is designed, which effectively resolves the conflict between the identity and style preserving.
- The 3D facial landmarks is introduced to represent face poses for the first time, resulting in more flexible pose editing than using the discrete one-hot pose label or ambiguous 2D facial landmarks.
- The proposed method demonstrates competitive performance in comparison with the current state-of-the-art, both qualitatively and quantitatively.

3.2 Related Work

3.2.1 Generative Adversarial Network (GAN)

Traditional GAN is composed of a generator and a discriminator. The training follows an adversarial paradigm. To overcome the problems of unstable gradient and mode collapse, Wasserstein GAN (WGAN) [56] proposes the earth move distance as metric in the discriminator's loss function. To enforce the Lipschitz constraint of the discriminator, SN-GAN [59] applies spectral normalization to the weight parameters. Due to its simplicity and promising effect, most of the recent GAN based algorithms make use of this technique, including SA-GAN [60], BigGANs [33], StyleGAN [32], *etc.* In our method, SN-GAN is also adopted in the structure.

3.2.2 Image-to-Image Translation

The combination of auto-encoder with discriminator has achieved impressive results in image-to-image translation [78, 83, 84, 76]. In multi-domain image translation tasks, the domain information is provided either to the bottleneck layer of the auto-encoder [84, 85, 16], or to the entry of the encoder/generator [78, 76, 18], by simply concatenating the domain label with the features or input images. Conditional batch normalization [86] and conditional instance normalization (CIN) [87] provide another way of introducing the conditional label in addition to concatenation, via

predicting the affine parameters of the normalized feature map (either by batch normalization or by instance normalization) from the input label. Here, the CIN technique is exploited in our decoder to avoid the operation of duplicating the label.

In multi-domain image translation, the discriminator is used to not only estimate the image quality, but also control the target domain of the generated image. Our approach borrows the idea of projection discriminator [88], which introduces the reality score and the inner product of the embedded label with the features of the input data.

3.2.3 Face Pose Manipulation

The existing methods can be roughly divided into two categories: 3D reconstruction based, and simple 2D based.

For the 3D based models, DA-GAN [71] uses a predefined 3D face model to produce the synthesized faces with arbitrary poses, and the dual agents serve to keep the identity information stable and improve the realism, Feng *et al.* [72] train a model to regress the UV map from a single 2D image directly, which records the 3D shape information. Tran *et al.* [89] proposes a framework to learn a nonlinear 3DMM model from a large set of unconstrained face images. FF-GAN [21] incorporates 3DMM [13] into the GAN based structure, where the 3DMM coefficients provide the low-frequency information, while the input image injects high-frequency local information.

For 2D based models, DR-GAN [16] learns a disentangled representation of face identity with the supervision of an auxiliary face classifier of the discriminator. TP-GAN [17] employs a two-pathway architecture to preserve both global and local texture information separately, and generates the frontalized face images. With the guidance of 2D facial landmarks, CAPG-GAN [18] is able to generate faces of arbitrary poses, where the couple-agent discriminator distinguishes the generated face/landmark pairs and profile/front pairs from ground-truth pairs, such design enables the algorithm generate face images of target poses while keeping the identity unchanged. CR-GAN [79] trains the generator to produce face images directly from the noises, together with the training of pose manipulation, maintaining

the completeness of the learned embedding space. FNM [19] employs unsupervised training and synthesizes normalized face images of Multi-PIE [20] style. Most of the above methods only focus on modeling the identity preserving ability, whereas they generally ignore the image style preserving ability, such as color, facial expression, lightning, *etc.* Although it is claimed that the synthesized frontal face images improve the face verification accuracy, the generated face images are visually far from the input images, thus greatly limits their further usage scenarios other than face recognition.[90] frontalizes the face image by predicting the pixel displacement. However, it's hard to extend to the arbitrary face pose editing problem due to the time consuming SIFT feature extraction.

3.3 Method

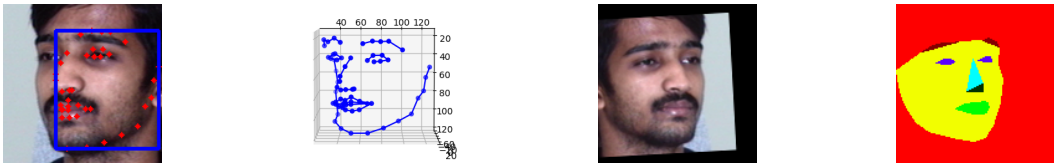
The goal of our method is to keep not only the identity but also the image style during face pose manipulation. We first define several notations: (I, J) denotes paired face images in the training set, where I is the source image, and J is the target one. The 3D facial landmarks are denoted as $ldmk_I$ and $ldmk_J$, which could be detected by an off-the-shelf 2D\3D facial landmark detector [2]. I_{tf} represents the input image after similarity transformation. To guide the training, the landmark based segmentation maps of I_{tf} and J are also required, which we denote as I_{seg_tf} and J_{seg}. These notations are visualized in Figure 3.3.

Our approach is composed of three major steps: preprocessing, pixel attention sampling, and image inpainting. They are described in detail subsequently.

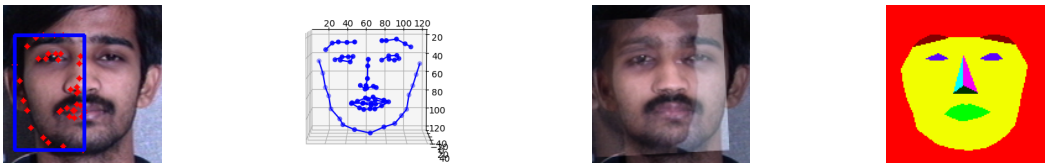
3.3.1 Preprocessing

Given the fact that human faces are roughly left-right symmetrical, thus a face at an arbitrary pose always has at least one side fully exposed to the camera. This preprocessing step aims to align the fully exposed side of face I to that of a target face J.

The inputs of this step are the input face image I, its 3D facial landmarks $ldmk_I$, and the landmarks $ldmk_J$ of the image J at a target pose. We first find the fully ex-



(a) I (b) $ldmk_I$ (c) I_{tf} (d) I_{seg_tf}



(e) J (f) $ldmk_J$ (g) blend (h) J_{seg}

Figure 3.3: (a) and (e) are the source image and the target one, respectively, where the corresponding landmarks and the bounding boxes of their bigger side are shown to illustrate our aligning strategy. (b) and (f) are the 3D facial landmarks detected by [2]. (c) shows the aligned image I_{tf}. (g) is the alpha blend of I_{tf} and J, illustrating that the target image shares pixel-level similarities with the source image. (d) and (h) are the segmentation maps of I_{tf} and J transformed from their 2D facial landmarks.

posed side by calculating the bounding box region of the projected facial landmarks, as illustrated in Figure 3.3(a) and Figure 3.3(e). Then, the least square regression on the corresponding landmarks is applied to calculate the transformation matrix, based on which the aligned image I_{tf} could be obtained, as shown in Figure 3.3(c). From Figure 3.3(g), we observe that I_{tf} and J indeed share pixel-level similarities. Finally, with the 2D facial landmarks of I and the transformation matrix obtained above, we obtain the 2D facial landmarks of I_{tf}, as well as the landmark based segmentation map I_{seg_tf}. Besides, to guide the training process of the PAS module, the segmentation map of the target image J_{seg} is also prepared at this stage.

3.3.2 Pixel Attention Sampling

The previous preprocessing step delivers the input face image with the larger side aligned to the target pose. Despite the fact that the transformed input face image I_{tf} and the face at the target pose J share many similarities in terms of texture, there still exist great gaps between them, from the global shape to the finer details of textures. Therefore, our goal at this stage is to preserve and fine-tune their similar

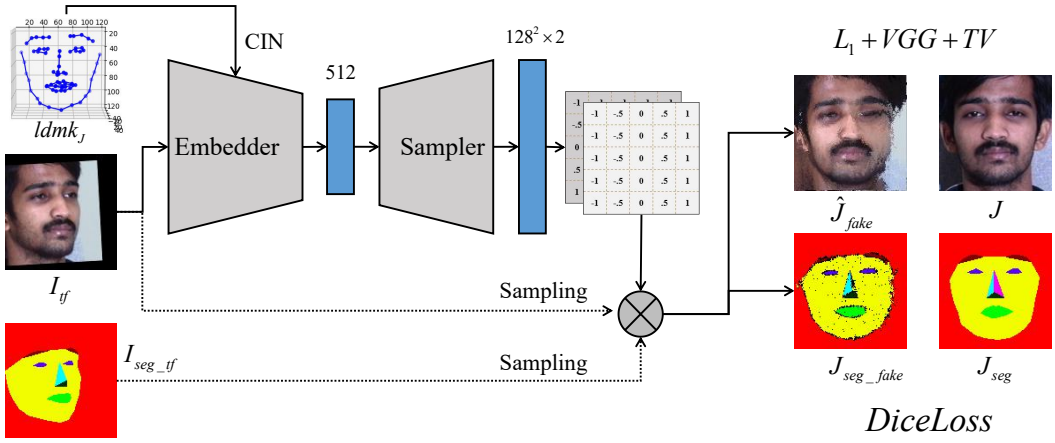


Figure 3.4: Structure of the proposed **P**ixel **A**ttention **S**ampling (PAS) module.

face regions while eliminating the major differences. This is achieved by a novel pixel sampling based module, which we call **P**ixel **A**ttention **S**ampling module (PAS), since the process of sampling mainly “focuses" on bridging the gaps. Figure 3.4 depicts the corresponding diagram.

Specifically, given the transformed image I_{tf} and the target pose $ldmk_J$, PAS generates a two-channel coordinate sampling map of the same size as I_{tf}. The first channel holds the abscissa while the second one for the ordinate. Each pixel location of the map is registered a coordinate, indicating which input pixel of I_{tf} that location will sample from. Note, the original pixel indices are converted into decimal coordinates ranging from -1 to 1, for the purpose of gradient backpropagation, and the final sampling is achieved by interpolating the adjacent pixels. Our sampling map is similar to the one used in the spatial transformer network[91]. The difference lies in that the one in [91] is determined by a 2D affine transform matrix, with only six parameters, whereas our sampling map is directly predicted by the neural network, resulting in $height \times width \times 2$ parameters in total. The PAS module is composed of two parts *i.e.*, the image embedder and the sampler. The embedder consists of stacked convolution layers, conditional instance normalization [87] layers (CIN), and self-attention [60] layers (SA). The CIN layers incorporates the 3D facial landmarks of the target pose to guide the embedding, and the SA layers enable the embedder focus more on the global structure of the face.The embedder finally

outputs a 512-dimensional feature vector, which is further fed into the sampler to generate the sampling maps. The sampler is composed of fully connected layers and ReLU layers. After applying the obtained sampling map to the transformed input face image I_{tf} and its corresponding segmentation map I_{seg_tf}, we could obtain the intermediate face image at the target pose, denoted as $\hat{J}_{fake}$, and its corresponding fake segmentation map, denoted as J_{seg_fake}. In order to maintain the reconstruction ability of the module, the original image I and its corresponding 3D landmarks $ldmk_I$ are also fed into the PAS module and the reconstructed output $\hat{I}_{recon}$ is achieved.

The training process of the PAS is guided by the following losses:

Pixel-wise loss between $\hat{J}_{fake}$ and J, $\hat{I}_{recon}$ and I, which is commonly used in the image-to-image translation algorithms. It can be formulated as:

$$L_{pix} = L_1(\hat{J}_{fake}, J) + 0.1 \cdot L_1(\hat{I}_{recon}, I) \tag{3.1}$$

where

$$L_1(I, J) = \frac{1}{WHC} \sum_{x,y,c=1}^{W,H,C} |I(x, y, c) - J(x, y, c)| \tag{3.2}$$

Since it does not take much effort to learn an identity mapping, we set the weight of the reconstruction loss to 0.1, which makes the PAS module concentrate much more on the pose manipulation task.

Segmentation loss between J_{seg_fake} and J_{seg}. Based on the assumption that if the sampled image $\hat{J}_{fake}$ is close to the target image J, the segmentation map J_{seg_fake} should be close to the target segmentation map J_{seg} as well. We therefore introduce a segmentation-related loss so as to push $\hat{J}_{fake}$ close to J. To facilitate the training converge, the segmentation loss is used as a complement to the aforementioned pixel-wise loss L_{pix}. Here, we make use of the Dice loss[92], which has been widely exploited in image segmentation tasks, and it can be formulated as:

$$L_{seg} = \sum_{c=1}^{N} 1 - \frac{2 \sum_{x,y} J^c_{seg}(x, y) \cdot J^c_{seg\_fake}(x, y)}{\sum_{x,y} J^c_{seg}(x, y) + \sum_{x,y} J^c_{seg\_fake}(x, y)} \tag{3.3}$$

where c represents the different classes of facial attributes. Since each pixel location (x, y) of the segmentation map is represented by a c-dimensional one-hot vector, the fraction in Equation 3.3 is thus a simple intersection over union. The benefit of the adopted loss function is that it is independent to the amount of pixels of different classes.

Perceptual loss[93] between $\hat{J}_{fake}$ and J. Perceptual loss is significant to preserve the identity information and high-level semantic features of the face images. We follow the work of [94] and employ the pre-trained VGG-Face [95] network to extract the features:

$$L_{per} = VGG_{loss}(\hat{J}_{fake}, J) \tag{3.4}$$

with

$$VGG_{loss}(I, J) = \sum_{i} |VGG_{Face}(I)_i - VGG_{Face}(J)_i| \tag{3.5}$$

where i is the layer index of the pre-trained model and $i \in \{3,8,15,22,29\}$, which are the last convolutional layer of each feature map scale.

Total variation loss. Total variation [96] loss has been widely used in GAN based algorithms for its powerful ability of reducing the noises and smoothing the generated results. In our PAS module, there inevitably exist obvious noises, since the resultant face image is pixel-wise sampled from the transformed input face image. Therefore, the TV loss is incorporated:

$$L_{tv} = TV(\hat{J}_{fake}) + TV(\hat{I}_{recon}) \tag{3.6}$$

where

$$\begin{aligned} TV(I) = &\sum_{x,y,c=1}^{W-1,H,C} |I(x+1, y, c) - I(x, y, c)|^2 \\ + &\sum_{x,y,c=1}^{W,H-1,C} |I(x, y+1, c) - I(x, y, c)|^2 \end{aligned} \tag{3.7}$$

The overall training loss of the PAS module is a sum of the above losses:

$$L_{sampler} = L_{pix} + L_{seg} + L_{per} + L_{tv} \tag{3.8}$$

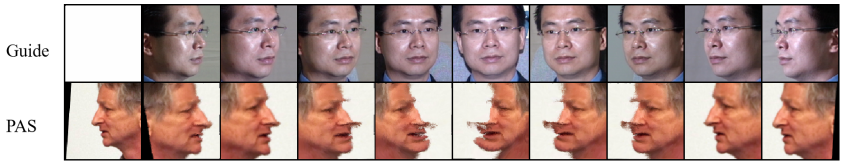


Figure 3.5: The result of PAS. The first row shows the guiding face images at target poses, the first image in the second row is the input face image, and the remaining images are synthesized faces based on the landmarks of the guiding face images. We can see that the pixels are sampled and adjusted to the target pose. The noises and holes will be removed or filled at the next image inpainting stage.

Thanks to the PAS module, we achieve a face image whose facial attributes have been aligned to the target pose location, with the original identity and style characteristics well preserved. It should be noted that, as the sampling is accomplished by interpolating adjacent pixels, it only modifies the location of the pixels within a small area around them, the PAS module is thus not able to sample for instance the left eye from the right one or the opposite. As a result, the sampled face images possibly contain artifacts, holes and noises, as illustrated in Figure 3.5. In order to further improve the generated image quality, image inpainting is introduced subsequently.

3.3.3 Image Inpainting

The image inpainting stage is to restore the holes and remove the noises and artifacts on the intermediate faces generated by PAS, and finally generate photo-realistic face images. To accomplish this goal, we introduce a Conditional Adversarial Auto-Encoder, where the discriminator is implemented by a projection discriminator [88], and the auto-encoder is based on the U-Net structure [80]. We also make use of CIN layer to merge the information provided by 3D facial landmarks, the identity features, and the image features, thereby making the generated face image in desired pose and shape. More precisely, the inputs of the encoder are the images generated by PAS together with their target poses, *i.e.*, $\hat{J}_{fake}$ with $ldmk_J$ for the task of pose manipulation, and $\hat{I}_{recon}$ with $ldmk_I$ for the task of reconstruction. To well preserve the face identity, the decoder is conditioned by the high level feature extracted by

the pre-trained LightCNN [97] model, where the parameters of the fully connected layer is fine-tuned during training. The outputs of the auto-encoder are denoted as J_{fake} and I_{recon}, whose ground truths are J and I, respectively. To further improve the model's generalization ability, unpaired face images could also be exploited to supplement the training set. This is achieved by feeding the network with the partially occluded face images S_{occ} and their 3D landmarks, and expect the network to output S_{recon} restoring the original S. For the discriminator, we feed all the generated images, including J_{fake}, I_{recon} and S_{recon}, as fake samples, while their corresponding ground truth as the genuine ones, with dis_{real} and dis_{fake} as output, respectively.

The loss function of the inpainting network is composed of four parts:

Pixel-wise loss, formulated as:

$$L_{pix} = L_1(J_{fake}, J) + \lambda \cdot L_1(I_{recon}, I) + L_1(S_{recon}, S) \tag{3.9}$$

where L_1 is defined in equation 3.2.

Perceptual loss to capture the semantic similarity:

$$L_{per} = VGG_{loss}(J_{fake}, J) + VGG_{loss}(S_{recon}, S) \tag{3.10}$$

where VGG_{loss} is defined in equation 3.5. We do not include (I_{recon}, I) here, because compared to image reconstruction task, image inpainting and pose manipulation are more likely to lose the identity consistency.

Identity loss to maintain the identity-related characteristics stable. We use the pre-trained LightCNN [97] to extract the identity feature of the synthesized image and the target image, and minimize the L_1 loss of them:

$$L_{id} = \frac{1}{N} \sum_{i=1}^{N} |F(J_{fake})_i - F(J)_i| \tag{3.11}$$

Adversarial loss to guarantee the generated image quality:

$$L_{adv} = -dis_{fake} \tag{3.12}$$

Besides, we also incorporate the total variation loss to reduce the spike artifacts. The overall training loss of the generator of the image inpainting network is a sum of the aforementioned losses:

$$L_{gen} = L_{pix} + L_{per} + L_{id} + L_{tv} + L_{adv} \tag{3.13}$$

Following the work of [98], the **discriminator loss** is defined as:

$$L_{dis} = \mathbf{max}(1 - dis_{real}, 0) + \mathbf{max}(1 + dis_{fake}, 0) \tag{3.14}$$

3.4 Experiments

Given an input face image, the proposed method aims to manipulate its pose while keeping the identity unchanged along with its style. Correspondingly, we evaluate it in two aspects: the style-conserving skill and the identity-preserving ability during face pose editing. In this section, we present the training details first, then the qualitative analysis for face style conserving, followed by the quantitative results for identity preserving. Ablation studies are also carried out to highlight the effectiveness of the proposed PAS module.

3.4.1 Training details

The training is based on four databases: Multi-PIE [20], 300W-LP [30], CAS-PEAL-R1 [99], and CelebA [100]. **Multi-PIE** has four sessions with face images under 13 poses and 20 illuminations. We follow Setting 1 of TP-GAN [17] and train the proposed algorithm on the first 150 subjects of session 1, then test on the remained 99 subjects. **300W-LP** contains large-pose face images synthesized from 300W [101]. After manually filtering out the low-quality images, we have 40,159 images from 2,815 subjects in total. **CAS-PEAL-R1** contains 1,040 subjects. For each subject, gray-scale images across 21 different poses are included. **CelebA** is a large-scale face attributes dataset with more than 200K celebrity images in it.

During the training process, we use the occluded face images as input, and

Figure 3.6: The final result of our approach. The first row shows the guidance images. The input images are in the first column, and the simple reconstructed images are in the second column. The rest images are the pose editing results based on the landmarks of the exemplars.

train the U-Net based generator to restore the original face images. This operation improves the generalization ability of the network, and make the generated images photo-realistic. All of the training images are cropped to 128×128 pixels. The learning rate is set to $1\mathbf{e}^{-4}$, and the Adam [102] optimizer is utilized with betas of [0.9, 0.999]. We first pre-train the generator and the discriminator on CelebA for 20000 iterations, making it a fundamental image inpainting model, which facilitates the subsequent training procedure. Then, we train the proposed PAS model and the image inpainting model jointly for 110000 iterations in total. Observing that the CAS-PEAL-R1 dataset consists of gray-scale images, which degenerates the color saturation of the generated images, we thus exclud the data of CAS-PEAL-R1 for the last 10000 iterations.

3.4.2 Style-conserving validation

Multi-PIE images under different poses are used as the guiding images, the pose of faces from CelebA are edited accordingly. As shown in Figure 3.6, the synthesized face images comply with the guiding faces in term of pose. They are visually photo-realistic and both the identities and the styles are well preserved, clearly validating

(a) (b) (c)

Figure 3.7: (a) From top to bottom shows the input images, results of CR-GAN, DR-GAN, FNM and our method. (b) From left to right are the input images and the generated images with both yaw angles and pitch angles changed. (c) From left to right are the input images, half-face aligned images, frontalized images w/o PAS module, results of the proposed algorithm, and the ground-truth images.

the effectiveness of the proposed method. A further qualitative comparison of our method and CR-GAN [79], DR-GAN [16] and FNM [19] are demonstrated in Figure 3.7(a). As can be seen, our results are more visually convincing and the styles are closer to the input images compared to DR-GAN, and the identities are better preserved than CR-GAN. As for FNM, the generated image style is more similar to the training set, where the lighting and expressions are normalized, and the color has been changed, by contrast, our results better preserve those characteristics of the input face images. Moreover, the proposed approach changes the face poses in three degrees of freedom, resulting in more flexible pose editing results than merely controlling the yaw angles as previous methods. Figure 3.7(b) shows the results of editing both pitch and yaw angles of input face images (leftmost).

Quantitative evaluations are further performed. We calculate the FID score of the above models, on the frontalized large pose face images from CelebA, the results are shown in Table 3.1, indicating that the proposed method generates face images with styles closer to the input face images.

Table 3.1: FID score of frontalized face images (lower is better)

CR-GAN	DR-GAN	FNM	ours
204	122	150	105

Table 3.2: Rank-1 recognition rates (%) across views under Setting 1.

Method	±90°	±75°	±60°	±45°	±30°	±15°
HPN [103]	29.82	47.57	61.24	72.77	78.26	84.23
c-CNN [104]	47.26	60.7	74.4	89	94.1	97.0
TP-GAN [17]	64.0	84.1	92.9	98.6	99.9	99.8
PIM [105]	75.0	91.2	97.7	98.3	99.4	99.9
CAPG-GAN [18]	77.1	87.4	93.7	98.3	99.4	99.9
FNM [19]	55.8	81.3	93.7	98.2	99.5	99.9
Light CNN [97]	2.6	10.5	32.7	71.2	95.1	99.8
Ours	45.5	78.7	90.0	99.6	99.9	100

3.4.3 Identity-preserving ability evaluation

There are 249 subjects in Session 1 of Multi-PIE. Following the Setting 1 of TP-GAN, we use the first 150 subjects for training, and the remaining 99 subjects for testing. The identity preserving ability is evaluated by Rank-1 recognition rate. The face with frontal view and normal illumination in the testing set compose the gallery, and the rest non-frontal images are used as probe.

The evaluation is conducted based on the features extracted by the pre-trained Light-CNN model. We directly extract the features of the probe images as baseline. For the proposed method, we first frontalize the probe face images, based on which their face representations are extracted. As can be seen from Table 3.2, the proposed method achieves similar or even better Rank-1 recognition rate in comparison with the baseline and state-of-the-art algorithms when the rotation angle is smaller than 60°. For larger rotation angles ($\geq 60°$), the proposed algorithm drastically outperforms the baseline, whereas it does not perform as well as the SOTA algorithms. There exist two possible reasons: 1) The face images of extreme poses share relatively less pixels with the face images of front view, thus the pixels sampled by the PAS module are not sufficient enough for the following inpainting stage, and 2) most of the SOTA algorithms normalize the face images into a consistent style, where the information irrelevant to identity is filtered out, in contrast, our method preserves relatively more style information.

3.4.4 Ablation Study

To highlight the effectiveness of the PAS module, the ablation study is conducted by removing it and training the U-Net based conditional adversarial auto-encoder directly. For the sake of fair comparison, we apply the same preprocessing pipeline (*i.e.*, align the larger side of the input image to match the target pose) and train the model with the same number of iterations. Figure 3.7(c) shows the results. As can be seen, the synthesized images without PAS are blurred. More specifically, in the second row and the fourth row, the mouths are not well aligned, and the unexpected edges of the aligned input images are not well removed. The results indicate that it is indeed difficult for the single U-Net based model to change the original patterns of the input image thus results in undesired artifacts.

3.5 Conclusion

In this work, we first carefully analyze the trade-off between the style-preserving ability and the identity-preserving ability of the existing 2D based pose manipulation methods. Based on the observation that face images in different poses share a large number of pixels, we propose a novel pose editing method and a sophisticatedly designed PAS module. The method selectively samples pixels from the input face and adjust their relative locations with the PAS module, so that the recovered face editing result match the target pose and faithfully keeps the original identity and style information unchanged. In this way, we convert the pose manipulation problem to a image inpainting problem, and further make the best of the finer details in the original face images to obtain convincing pose editing results. We also utilize 3D facial landmarks to represent the face pose, which is more precise and flexible comparing to the one-hot labels and the 2D facial landmarks adopted in previous studies. Extensive experiments validate that the proposed pose editing approach preserves the style information of the input images better than the existing methods.

Chapter 4

Texture Generation for 3D Face Reconstruction

4.1 Introduction

3D face reconstruction is an important yet challenging domain in computer vision, aiming to faithfully restore the shape and texture of a face from one or more face images. It has a wide range of applications, such as face recognition, face editing, face animation, and other artistic and entertainment fields. Recently, there has been a surge of interest in single-image based 3D face reconstruction [3, 106, 107, 26, 27, 3, 72, 108, 28]. While most previous work has been devoted to predicting more accurate and detailed 3D shapes, not much work has focused on generating photo-realistic face textures. However, studies [109, 12] have shown that the texture plays a more significant role than that of the shape in face recognition tasks. Thus we can never ignore the importance of the texture in 3D face reconstruction.

Existing 3D face texture generation methods can be broadly classified into three categories: texture model-based, image generation-based, and GAN optimization-based.

Texture model-based Since the 3D Morphable Model (3DMM) [13] was proposed, it has been widely used in 3D face reconstruction. The model is a vector basis of the shape and texture learned from a set of 3D face scans. Earlier approaches regress the 3DMM parameters by solving a non-linear optimization problem [24, 26, 25], which is often slow and costly. With the development of Convolutional Neural Networks, recent studies tend to predict the parameters using learning-based methods [26, 27, 3]. However, the 3DMM is constructed by a small

Figure 4.1: Results of the proposed method. The left column shows the input images. Images on the right are synthesized using the predicted UV-map.

number of face scans under well-controlled conditions, limiting its diversity to identity, race, age, gender, etc. Besides, due to the linear and low-dimensional nature of the model, it can hardly capture high-frequency details, resulting in blurred textures that are far from satisfactory.

Image generation-based Generative Adversarial Network (GAN) [15] provides a powerful tool for generating photorealistic images. Since its appearance, numerous image generation methods with stunning results have been proposed. Thanks to various large databases and the highly structured geometry of the human face, 2D face image generation is one of the most prosperous areas [74, 75, 76, 77, 78, 32, 31]. Influenced by this trend, some recent 3D face reconstruction methods have also leveraged adversarial training to improve the texture quality [28, 4, 29]. However, such an image generation approach is highly dependent on a large 3D face database. [28] is trained on a synthesized 3D face database [30], where originally self-occluded textures are obtained by simple interpolation of

visible parts, resulting in imperfect generation. [4, 29] are trained on a large UV map dataset, which is not publicly available.

GAN optimization-based The traditional yet most powerful GANs are trained to synthesize images from noise vectors [31, 32, 33]. To leverage the power of a pre-trained GAN, a series of works are established on inverting the image back to a GAN's latent space using optimization-based approaches [34, 35, 36, 37]. Similar methods are used to generate the UV map of a face image [38, 39]. First, they train a generator that converts noise vectors into UV maps. Then they directly optimize the latent code to minimize the reconstruction error between the input face image and the image rendered by the generated UV map. Instead of training a UV map generator, [40] first rotates the input image in 3D and optimizes the latent code of the pre-trained StyleGAN to fill in the missing textures, then stitches textures of different view angles by alpha blending to form the final UV map. By far, the optimization-based methods can yield the most realistic face UV maps. Nevertheless, they are usually complex and time-consuming, *e.g.*, GANFIT [38] takes 30 seconds to generate the UV map of an input face, while OSTeC [40] takes up to 5 minutes.

Besides generating a global face texture, we note that a series of pure 2D image generation methods can also synthesize face images of different view angles [16, 5, 18]. However, the generation consistency is poor due to the absence of global consistency constraints and a priori knowledge of the 3D shape.

In summary, among the current texture generation methods for 3D face reconstruction, those based on texture models cannot yield high-fidelity results due to the model's simplicity; those based on image generation rely heavily on large training dataset; those based on optimization are time-consuming and require a high computational cost.

To this end, we propose a novel image-to-image translation model that converts the input face image into its corresponding UV map. The proposed method is image generation-based, therefore much faster than optimization-based methods. We use the pseudo UV map for training, bypassing the dependency on the real UV map dataset. Thanks to multiple partial UV discriminators, we can use cropped parts

of incomplete UV maps (acquired using the data pre-processing method provided in [4]) for training to improve the generation quality. Our contributions are as follows:

- A novel image generation-based UV map prediction framework is proposed. The generated results are comparable to the optimization-based method but much faster.
- With the proposed UV sampler module, the visible face textures can be directly mapped to the UV space, forming an incomplete UV map. No 3D information (shape, occlusion) is required during the inference stage. Therefore our model can be stitched seamlessly with any 3D shape reconstruction models.
- The training is doesn't rely on the real UV map dataset, and the design of multiple discriminators can compensate well for the imperfect ground truth.
- The proposed method outperforms the state-of-the-art methods, both qualitatively and quantitatively.

4.2 Related Work

3D shape reconstruction From earlier optimization-based methods to CNN prediction-based methods, acquiring accurate 3D face shape becomes easier and faster, bringing powerful tools and significant opportunities for face-related tasks. Our training process relies on 3D shape reconstruction of a given face, where numerous 3D shape fitting methods are applicable. In this paper, we adopt an off-the-shelf model [3] as our shape re-constructor, which is the current SOTA 3DMM-based method. The model will predict its corresponding pose and 3DMM shape/texture parameters with a single face image as input.

UV map generation There exist mainly two texture representation methods for 3D models, vertex-based and UV map-based. The vertex-based representation is very intuitive, where each vertex has a color, and the interpolation of those colors generates the texture of the 3D surface. However, such representation flattens

the texture into a linear vector, destroys the spatial relationship of texture patches, thus prevents it from leveraging powerful CNN-based methods. The UV map-based representation unwraps the 3D texture into a 2D space. Briefly, each 3D vertex's color is mapped to its corresponding location of a 2D image, and adjacent vertices are mapped to adjacent regions so that the positional relationships between vertices are well preserved. [4] first sample the color of visible 3D vertices from the input face image, then map them to UV space to get the incomplete UV map, in which the generative model will further complete the missing parts. However, their method is highly dependent on the precise 3D shape and ground truth UV maps. In contrast, our method does not need the UV map data for training or 3D shape for inference. [28] propose a non-linear 3DMM, where the predicted texture takes the UV map-based representation. Nevertheless, their UV map generator's input is a low-dimensional encoding of the input image, resulting in an loss of detail of the predicted UV map. In addition, their model is trained on linear 3DMM synthesized images [30], where artifacts caused by self-occlusion appear frequently. Unlike [28], our model is trained on real face images, and the coding keeps a large dimension across the forward path, making the generated UV map photorealistic.

Differentiable renderer To obtain the gradient of the loss function and thus train the network, a differentiable renderer is widely used in 3D face-related algorithms [26, 27, 28, 3]. Briefly, a renderer is composed of a rasterizer and a shader. The rasterizer applies depth-buffering to select the mesh triangles corresponding to each pixel, and the shader computes the pixel colors as follows:

$$\bar{c} = w_0 c_0 + w_1 c_1 + w_2 c_2 \tag{4.1}$$

where c_i is the color of the i^{th} vertex of the mesh triangle the pixel resides in, w_i is the barycentric coordinate of the pixel in the triangle. During backward propagation, the gradients are passed from each pixel to the vertices:

$$\frac{\mathrm{d}L}{\mathrm{d}c_i} = \frac{\mathrm{d}L}{\mathrm{d}\bar{c}} \frac{\mathrm{d}\bar{c}}{\mathrm{d}c_i} = \frac{\mathrm{d}L}{\mathrm{d}\bar{c}} w_i \tag{4.2}$$

where L is the loss function. Since c_i is sampled from the output of the texture generator, *i.e.*, the UV map, the gradients could be further backpropagated. In our project, we adopt the off-the-shelf differentiable renderer of PyTorch3D [110].

Pixel attention sampling To get the UV map of visible parts, UV-GAN [4] first fits a 3DMM to the input image, then use the vertices' projected 2D coordinates to sample their corresponding colors, and the incomplete UV map is further generated. However, their method relies on accurate 3D shape fitting and facial landmark detection. Furthermore, such a method does not have a mechanism to deal with face occlusions(hands, hair, eyeglasses, etc.). Inspired by [111], we apply a pixel attention sampling (PAS) module to sample the incomplete UV map from the input image directly. Thanks to this module, the inference process is free from 3D shape or facial landmarks. Besides, different from [111], where input images require landmark-based pre-alignment due to the arbitrary target poses. The target output, *i.e.*, the UV map, is highly structured, so neither spatial transformation to the input image nor the target pose condition is demanded.

4.3 Proposed Method

The goal of our method is to predict the face UV map from a single face image. As illustrated in Figure 4.3, the proposed model consists of two parts: a UV attention sampling module (UV sampler) and a UV map inpainting module (UV generator). During the inference process, the UV sampler will sample the pixels from the input image to generate an incomplete UV map, and then the UV generator will further complete the semi-finished UV map. We describe the details of each component as follows.

4.3.1 UV Attention Sampling

The UV map is a two-dimensional representation of the global texture of a 3D object. Due to self-occlusion, it is an ill-posed problem to get the UV map from a single image. This section studies how to generate an incomplete UV map that contains only visible textures of the input face image. As a comparison, we recall

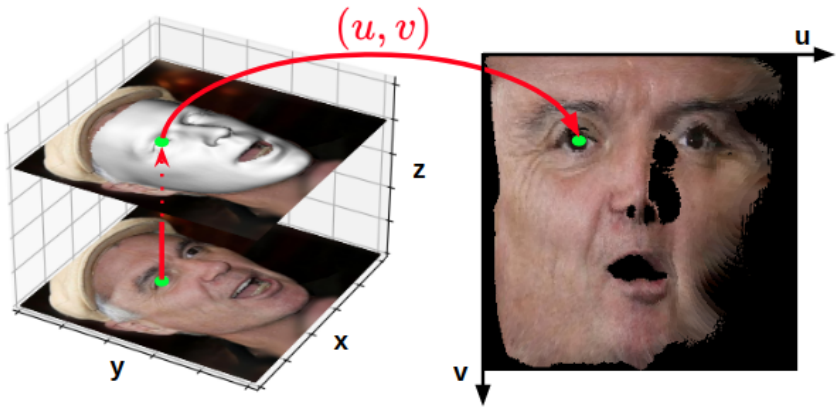


Figure 4.2: The traditional method for incomplete UV map generation. Which is used for generating the target output of the UV sampler.

the traditional method, which consists of four steps: (1) Get the 3D face shape based on the input image. (2) Determine the visible vertices using depth-buffer-based methods. (3) Project these visible vertices onto the image plane and index their colors according to their coordinates. (4) Render the UV map with the colors and the pre-defined UV-coordinates corresponding to each visible vertex. Figure 4.3 illustrates the above steps. Obviously, such a method is tedious and relies on an accurate 3D shape fitting. Since the UV map contains *only* the texture information of a 3D surface, is it really necessary to fit the exact 3D shape before getting the UV map? We do not think so. In fact, the only purpose of the 3D shape is to establish a one-to-one relationship between the pixel in the 2D face image and the pixel in the UV map, so why not learn such a mapping relationship in a data-driven manner? To achieve such a goal, we designed the UV sampler, a CNN-based model that maps the face image's pixels directly to the UV map.

The model has three parts, *i.e.*, the feature extractor, the segmentation head, and the sampler head. Similar to most generative models, the feature extractor is composed of stacked residual blocks [112]. Spectral normalization [59] is applied to each convolution layer to stabilize the training. With this module, 2D feature maps of different scales and a 512-dimensional vector are extracted from the input image. The 2D feature maps are fed into the FPN structured [113] segmentation head and

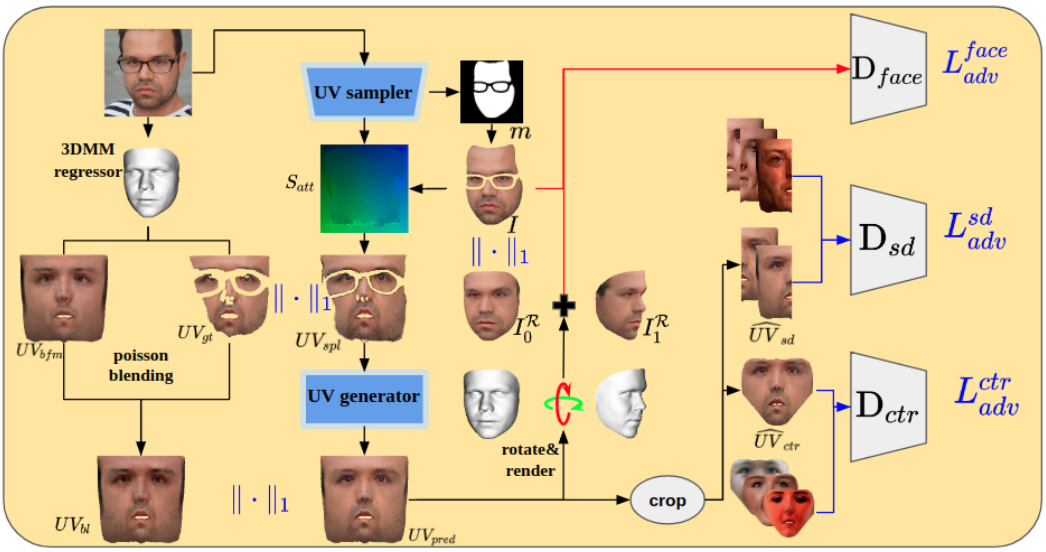


Figure 4.3: Overview of our approach. (1) Given an input face image, the UV sampler predicts its face mask m and sampling map S_{att}, based on which samples an incomplete UV map UV_{spl} (2) The UV generator will further complete the sampled UV map and output the UV_{pred}. (3) With an off-the-shelf 3DMM regressor, we predict the shape and texture of the input face image, which is used for getting the ground truth of the UV_{spl}: UV_{gt} and the pseudo ground truth of the UV_{pred}: UV_{bl}. (4) The predicted UV map is used to render face images of different poses: $I_0^{\mathcal{R}}$ and $I_1^{\mathcal{R}}$, which are further fed into a face discriminator. (5) UV_{pred} is cropped to the side part $\widehat{UV}_{sd}$ and center part $\widehat{UV}_{ctr}$, fed into their corresponding discriminators.

output an attention mask m. Besides, the 1D feature vector is fed into the sampler head, a stack of fully connected layers interspersed with ReLU activations. The sampler head's output is reshaped as $S_{att} \in \mathbb{R}^{B\times 256\times 256\times 2}$, which is the attention sampling map, where B is the batch size, 256 is the height/width of the UV map, and the last two channels hold the normalized abscissa and ordinate of the pixel in the input image to sample. Based on S_{att} and m, differentiable sampling [91] is applied to the masked input image I, and an incomplete UV map $\widehat{UV}_{spl}$ is finally obtained.

To train the model, we use the above-mentioned traditional method to generate the ground truth (incomplete) UV map, UV_{gt}. One issue to note is that the 3D shape fitting is not completely accurate. The projected 3D vertices sometimes could lie on the background part of the face image, resulting in wrong vertices' colors and further

wrong UV_{gt}. Moreover, since the surface normal of the face edge is almost parallel to the image plane, numerous projected vertices are piled up in the narrow edge region, aggravating the UV map's inaccuracy. The occlusions such as eye glasses and hairs also cause the wrong UV texture. To mitigate this problem, we first generate a binary mask for the face region based on a pre-trained face segmentation model, then erode the mask's edge to ensure that the region inside of which must be the face, the generation of UV_{gt} only takes into account the vertices that fall inside the mask. Although this would result in a loss of texture near the edge, it is worth sacrificing the unimportant edges to ensure the accuracy of UV_{gt}.

The training is guided by the following loss function:

$$\begin{aligned}\mathcal{L}_{spl} = \|\widehat{UV}_{spl} - UV_{gt}\|_1 + \|S_{att} - S_{gt}\|_1 + \\ L_{seg}(m, m_{gt}) + \lambda TV(\widehat{UV}_{spl})\end{aligned} \tag{4.3}$$

where S_{att} and m are the outputs of the UV sampler, $\widehat{UV}_{spl}$ is the sampled UV map based on them. S_{gt} is the ground truth sampling map, which is obtained by mapping the normalized x,y coordinates of the visible 3D vertices into the UV space, *i.e.*, UV position map [72]. $L_{seg}(m, m_{gt})$ is the binary cross-entropy loss of the predicted face mask.

$$L_{seg} = -[m_{gt}\log m + (1 - m_{gt})\log(1 - m)] \tag{4.4}$$

$TV(\widehat{UV}_{spl})$ is the total variation loss [96] of the predicted UV map, which is powerful in smoothing the noises of the generated UV map.

$$\begin{aligned}TV(\widehat{UV}_{spl}) = \sum_{x,y,c=1}^{W-1,H,C} \left|\widehat{UV}_{spl}(x+1, y, c) - \widehat{UV}_{spl}(x, y, c)\right|^2 + \\ \sum_{x,y,c=1}^{W,H-1,C} \left|\widehat{UV}_{spl}(x, y+1, c) - \widehat{UV}_{spl}(x, y, c)\right|^2\end{aligned} \tag{4.5}$$

Thanks to the UV sampler, an incomplete UV map could be sampled directly from the input image, bypassing a series of complex and expensive steps of traditional methods, including 3D shape fitting, visible vertices determination, UV map rendering, etc.

4.3.2 UV Map Inpainting

With the UV sampler described above, we can sample an incomplete UV map from a face image. The next task is to fill the missing parts with textures consistent with the sampled parts. This is an image inpainting problem, which has been extensively studied. However, most image inpainting methods are trained on paired images, meaning the ground truth image is uniquely determined. In contrast, in our case, the ground truth is not available. This section studies how to train a UV map inpainting model without the supervision of the ground truth. Briefly, our approach is to generate a pseudo ground truth UV map to assist the training. Then, we work with multiple discriminators to make the generated images as photorealistic as possible.

4.3.2.1 Pseudo UV Map Generation

Generating the pseudo UV map consists of three steps: 1) incomplete ground truth UV map generation, 2) 3DMM texture fitting, 3) seamless image blending. The first step has been described in detail in the previous section. For the second step, we use directly the BFM [114] texture parameter predicted by [3]. The UV map representation of the reconstructed BFM texture is denoted as UV_{bfm}. Obviously, due to the linear, low-dimensional nature of the BFM model, UV_{bfm} is far from reality, as can be seen in Figure 4.4(d). Therefore, we move to the third step: seamless image blending.

The UV_{gt} obtained in the first step can faithfully restore the input image((face edge excluded)), but it is incomplete due to self-occlusion. While the UV_{bfm} obtained in the second step is complete, but it is only a rough approximation in the 3DMM solution space. Thanks to Poisson image editing [115], we can seamlessly blend the two results by solving the following Poisson equation with Dirichlet boundary conditions:

$$\Delta f = \text{div } \mathbf{v} \text{ over } \Omega, \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega} \tag{4.6}$$

where in our case, Ω denotes the domain of real textures in UV_{gt}, f is the texture to be modified in the UV_{bfm}, $\mathbf{v}$ is the gradient of the texture in UV_{gt}, f^* denote the

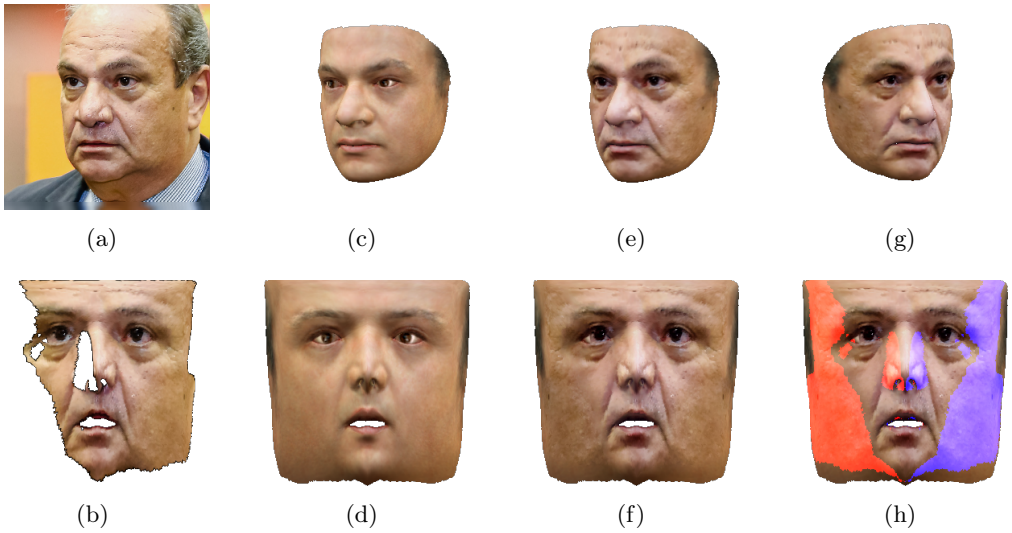


Figure 4.4: **(a)** The input image. **(b)** UV_{gt}. **(c)** The face reconstructed from UV_{bfm}. **(d)** UV_{bfm}. **(e)** The face reconstructed from UV_{bl}. **(f)** UV_{bl}. **(g)** The face in (e) under different view angle. **(h)** The texture marked in blue is used to fill the missing texture in its symmetric area (marked in red).

texture of UV_{bfm} outside the Ω. We also leverage the texture of the visible region to fill its missing symmetric region. That is, we do two times Poisson Blending, the first time blends the UV_{gt} to the UV_{bfm}, the second time blends the flipped UV_{gt} to its symmetric missing parts, as illustrated in Figure 4.4(h). The final blending result is denoted as UV_{bl}, as in Figure 4.4(f), both Figure 4.4(e) and Figure 4.4(g) are generated from it, which is far more photorealistic than the BFM reconstruction result in Figure 4.4(c).

4.3.2.2 Multiple discriminators

The training of the UV generator follows an adversarial paradigm; therefore, a large amount of data from the target domain is essential. However, the pseudo UV map U_{bl} generated above is not very reliable. Its quality depends on the accuracy of UV_{bfm}, the texture area of UV_{gt}, and the accuracy of the 3D shape. We only use the pseudo UV map to calculate the reconstruction loss, which is a rough guide to the generator's output. Although the complete UV map data is not available, we might as well collect a bunch of partial UV maps using the traditional method,

i.e., for UV maps generated from frontal face images, the central region, denoted as UV_{ctr}, is accurate, and for UV maps generated from profile face images, the visible half side, UV_{sd} is precise. Note that the partial UV maps collected in this way are not paired with UV_{pred}, so they are only used for adversarial loss, thus indirectly force the UV_{pred} lying in the real domain.

We design two partial UV map discriminators, one for the half side, the other for the center region. Together with the *masked face* discriminator, the system has three discriminators in total, as shown in Figure 4.3. The training is guided by the following losses.

Adversarial loss Given an output of the UV sampler, UV_{spl}, the generator will predict a global UV map, UV_{pred}. Three UV patches can be cropped from UV_{pred}, namely $\widehat{UV}_{ctr}$, $\widehat{UV}_{left}$, $\widehat{UV}_{right}$. Due to UV map's symmetry, the latter two can be put together and denoted as $\widehat{UV}_{sd}$. With the UV_{pred} and the 3D shape/pose parameters predicted by the model of [3], a reconstructed face image $I_0^{\mathcal{R}}$ could be rendered. By changing the pose parameter, we can get a face image in a different view angle, denoted as $I_1^{\mathcal{R}}$. So far, we have three types of fake data: $\widehat{UV}_{sd}$, $\widehat{UV}_{ctr}$, and $I_{0,1}^{\mathcal{R}}$, each of which corresponds to real data represented as UV_{sd}, UV_{ctr}, and I^m, where I^m is the input face image with occlusions/background masked.

The adversarial loss is thus formulated as:

$$\mathcal{L}_{adv} = \mathbb{E}_x[\log D(x)] + \mathbb{E}_{\hat{x}}[\log(1 - D(\hat{x}))] \tag{4.7}$$

where

$$\begin{aligned}(x, \hat{x}, D) \in \{&(UV_{ctr}, \widehat{UV}_{ctr}, D_{ctr}),\\ &(UV_{sd}, \widehat{UV}_{sd}, D_{sd})\\ &(\{I^m, I_{0,1}^{\mathcal{R}}\} \odot m_{gt}, D_{face})\}\end{aligned}$$

Reconstruction loss The reconstruction loss consists of two terms, the UV reconstruction loss and the face reconstruction loss.

$$L_{rec} = \|UV_{pred} - UV_{bl}\|_1 + \|I_0^{\mathcal{R}} \odot m_{gt} - I^m\|_1 \tag{4.8}$$

Symmetry loss Since the UV map of the face is left-right symmetrical, we design the symmetry loss to help the model learn this property.

$$L_{sym} = \|UV_{pred} - FlipLR(UV_{pred})\|_1 \tag{4.9}$$

Identity loss Since the pose is arbitrary, the ground truth of $I_1^{\mathcal{R}}$ is not available. Thus we use the pre-trained FaceNet [116] to extract the identity feature of $I_1^{\mathcal{R}}$ and I^m, and minimize their L_1 distance.

$$L_{id} = \|\mathcal{F}(I_1^{\mathcal{R}}) - \mathcal{F}(I^m)\|_1 \tag{4.10}$$

TV loss TV loss of Equation 4.5 is also applied to UV_{pred}. The total loss function is as follows:

$$L = L_{rec} + \lambda_1 L_{adv} + \lambda_2 L_{sym} + \lambda_3 L_{id} + \lambda_4 TV \tag{4.11}$$

4.4 Experiments

The proposed method can faithfully convert the input face image to its corresponding UV map. To demonstrate the conversion ability, we qualitatively compare the 3D reconstruction results with the current state-of-the-art methods, both 2D-based and 3D-based. A quantitative evaluation is also presented.

4.4.1 Implementation details

Our training is based on two datasets:, CelebA-HQ [31], and FFHQ [32]. Face images are pre-aligned with landmarks detected by [2]. The input image size is 256×256, and the predicted UV map is the same size as the input. As for the ground-truth face mask, we first train a stand-alone face segmentation model, using the attribute mask of the CelebA-HQ and our manually labeled occlusions (eyeglasses, hands, etc.). Then we use this model to detect the face masks of the training data, and use them as the ground truth. We set the learning rate to $1e^{-4}$ and use Adam[102] optimizer with betas of [0.5, 0.999], the batch size is set to 6. We first pre-train the

Figure 4.5: Frontalization results comparing with 2D-based face pose editing methods. Zoom-in for a better view.

UV sampler until it outputs an incomplete UV map that can perfectly reconstruct the input image, which takes about 100K steps. Then we train the UV generator for 150K steps with the UV sampler's weights fixed. The training of the whole model takes about 100 hours on two Titan X Pascal graphics cards. Since most of the face images in the training sets are frontal, making the model not robust to the large view angles, to solve this problem, one trick we adopt is to rotate and render the input faces with their corresponding shapes and pseudo UV maps, then train the model to reconstruct the original face images.

4.4.2 Qualitative results

We use the predicted UV maps to render 3D shapes. By changing the pose parameters, images of different view angles are generated. For the qualitative evaluation, as a usual convention, we take the same inputs as others and paste the generated results after them. Figure 4.5 compares our frontalization results with 2D-based face pose editing methods, including TP-GAN [117], CAPG-GAN [18], HF-PIM [23], FNM [19] and Zhou*el al.*[5]. As shown in Figure 4.5, TP-GAN doesn't convert the pose well, and the third face image it generates is obviously left-skewed. Furthermore, the images generated by TP-GAN, CAPG-GAN, and FNM have large color

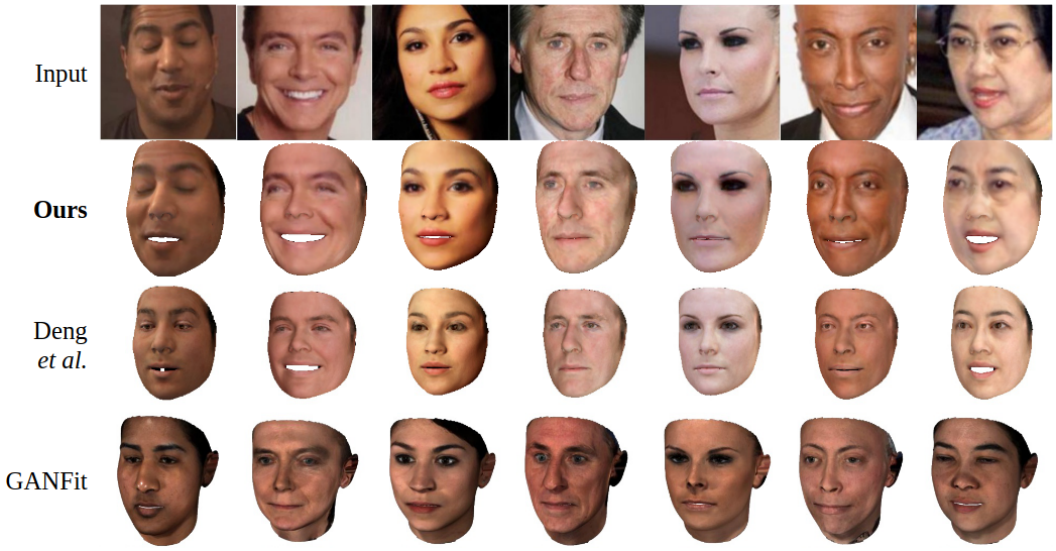


Figure 4.6: Qualitative comparison with other state-of-the art 3D reconstruction methods.

deviations with the input images due to the influence of Multi-PIE [20] data in the training set. Besides our method, only HF-PIM and Zhou *et al.* maintain a consistent texture style with the input image. However, due to the lack of a priori knowledge of the 3D shape, HF-PIM cannot preserve the face shape well while editing the face pose. In addition, the second image generated by it preserves the finger on the mouth corner, showing that it cannot handle the face occlusions well. Our method achieves similar performance to the current state-of-the-art, Zhou *et al.*, we both use an off-the-shelf shape regressor. However, their method is based on face image generation, which means that we need to re-infer the missing texture each time we change the view angle. Another limitation of face image generation-based method is that the training settings greatly limit their pose editing freedom. The method of Zhou *et al.* cannot well generate face images of a large yaw angles; TP-GAN, FNM, and HF-PIM can only generate face images in frontal view.

A further qualitative comparison of our method and two representative 3D reconstruction-based methods are demonstrated in Figure 4.6. The method proposed by Deng *et al.* [3] is based on 3DMM parameter regression, and GANFIT [38] is based on latent-code optimization of a pre-trained GAN model. As can be seen,

our results are more visually pleasant: large amounts of details are well preserved, including freckles, wrinkles, and expressions. Due to the model's low-dimensional nature, it's difficult for 3DMM-based methods to restore the input image's details faithfully. As can be seen in the 3rd-row of Figure 4.6, freckles and wrinkles are not well reconstructed. The results of GANFIT do contain richer details, but the resulting textures' styles are very homogeneous and differ considerably from their corresponding input images. We believe this is due to the lack of diversity in their training data, as the face UV map datasets are not easily accessible.

4.4.3 Quantitative results

Training Data	Method	ACC(%)	AUC(%)
CASIA(baseline)	Zhou et al.	98.77	99.90
CASIA+rot	Zhou et al.	98.95	**99.91**
CASIA(baseline)	UV-GAN	99.02	-
CASIA+augUV	UV-GAN	**99.22**	-
CASIA(baseline)	ours	98.75	99.88
CASIA+augUV	ours	98.98	99.90

Table 4.1: Comparison of the face augmentation ability with UV-GAN [4] and [5]

Data augmentation As in many previous works, we use our proposed method to synthesize face images for face data augmentation and evaluate the performance of the model trained on the augmented dataset to demonstrate the merits of our approach. The experiment is based on the CASIA [118] dataset. Due to the low resolution of the images in CASIA, we retrained a model with an input/output resolution of 128×128. Moreover, we remove the segmentation head in the UV sampler to increase the diversity of the augmented images. For each image with less than 30° yaw angle, we randomly increase its yaw angle from 15° to 60° and get a synthesized images. Our basic training settings are the same as [5], with ResNet18 [112] for the backbone and ArcFace [119] for the loss function. Results are shown in Table 4.1. Since UV-GAN [4] uses ResNet27 as its backbone, which is deeper than ours and [5], it is not surprising that it achieves the highest accuracy. Although we take the same settings as Zhou et al., we achieve a slightly lower baseline due to numerous differences in training details (learning rate, batch size,

optimizer, etc.). However, by training on the augmented dataset, our model exceeds their accuracy, and our AUC closes the gap with them, demonstrating the efficiency of our data augmentation ability.

	Reconstruction		Recognition	
Method	L_1	SSIM	Recon	Front
Deng *et al.*	0.064	0.698	0.554	0.501
Zhou *et al.*	0.069	0.613	0.780	0.675
Ours	**0.021**	**0.913**	**0.862**	**0.684**

Table 4.2: Pixel-wise reconstruction and the identity-preserving ability on AFLW2000-3D, non-facial areas of all images are masked out for fair comparison.

Face reconstruction We evaluate the proposed method in two aspects: the pixel-wise reconstruction ability and the identity-preserving ability. As most previous works are not open-sourced, we only compare with Deng *et al.* [3] and Zhou *et al.* [5], SOTA methods based on 3DMM and 2D face image generation, respectively. We conduct the experiments on the AFLW2000-3D [30], which contains 2000 face images with ground truth shape parameters.

For the reconstruction ability evaluation, we calculate the L1 loss and the structural similarity [120] of the reconstructed face images. As can be seen from Table 4.2, our method outperforms others in both these metrics.

As for the identity-preserving ability, the evaluation is conducted by features extracted by the pre-trained LightCNN-29 v2 [97] model. We calculate the cosine similarity of the features corresponding to the input images and the reconstructed/frontalized images. Results are shown in the two rightmost columns of Table 4.2. An interesting thing to notice is that, although Zhou *et al.* is inferior to the 3DMM-based model in terms of reconstruction loss, they are more capable of preserving the face identity. However, our proposed method achieves the best performance in both aspects.

4.5 Ablation Study

UV sampler Due to the inaccurate 3D shape, textures in the edge areas of the incomplete UV maps are not reliable, especially when the face has a large yaw an-

gle. As shown in Figure 4.7, the incomplete UV map generated by the UV sampler discards the inaccurate parts, forcing the subsequent UV generator to work on optimizing this region. The incomplete UV map generated by the traditional method, on the other hand, has preserved the inaccurate edges, causing the subsequent UV generator to lack the urge to optimize the edge area. In consequence, the generated UV map is blurred and full of noise.

Figure 4.7: Ablation study of the UV sampler. The UV sampler not only makes the first stage independent of the 3D shape, but also helps to generate more accurate textures in the second stage.

Partial UV discriminators The partial UV discriminators are designed to mitigate the problem of lack of ground truth UV map. If we use the face discriminator solely, only the reconstructed face images are guaranteed to be photo-realistic. However, the textures corresponding to the input face image's occluded parts will still suffer from blurring and artifacts. The 4th and 5th row of Figure 4.8 shows the problems caused by the absence of partial discriminators.

Segmentation head During pixel sampling, the segmentation head of the UV sampler could mask out all face occlusions (glasses, hands, scarf, etc.) to avoid the occlusions appear in the predicted UV map. Figure 4.9 illustrates its effect.

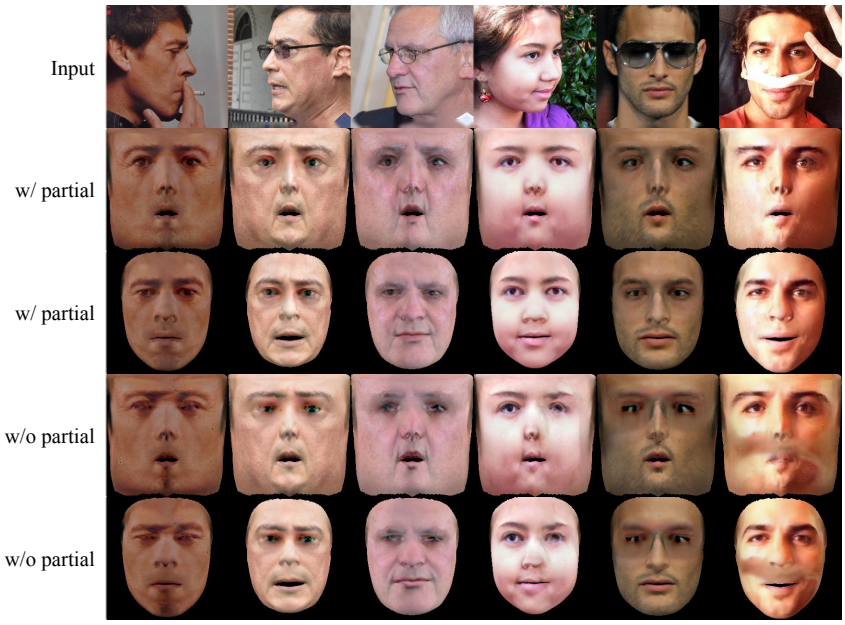


Figure 4.8: Ablation study of partial discriminators.

Figure 4.9: Ablation study of the segmentation head.

4.6 Application

An obvious application of our method is the profiling and frontalization of face images.

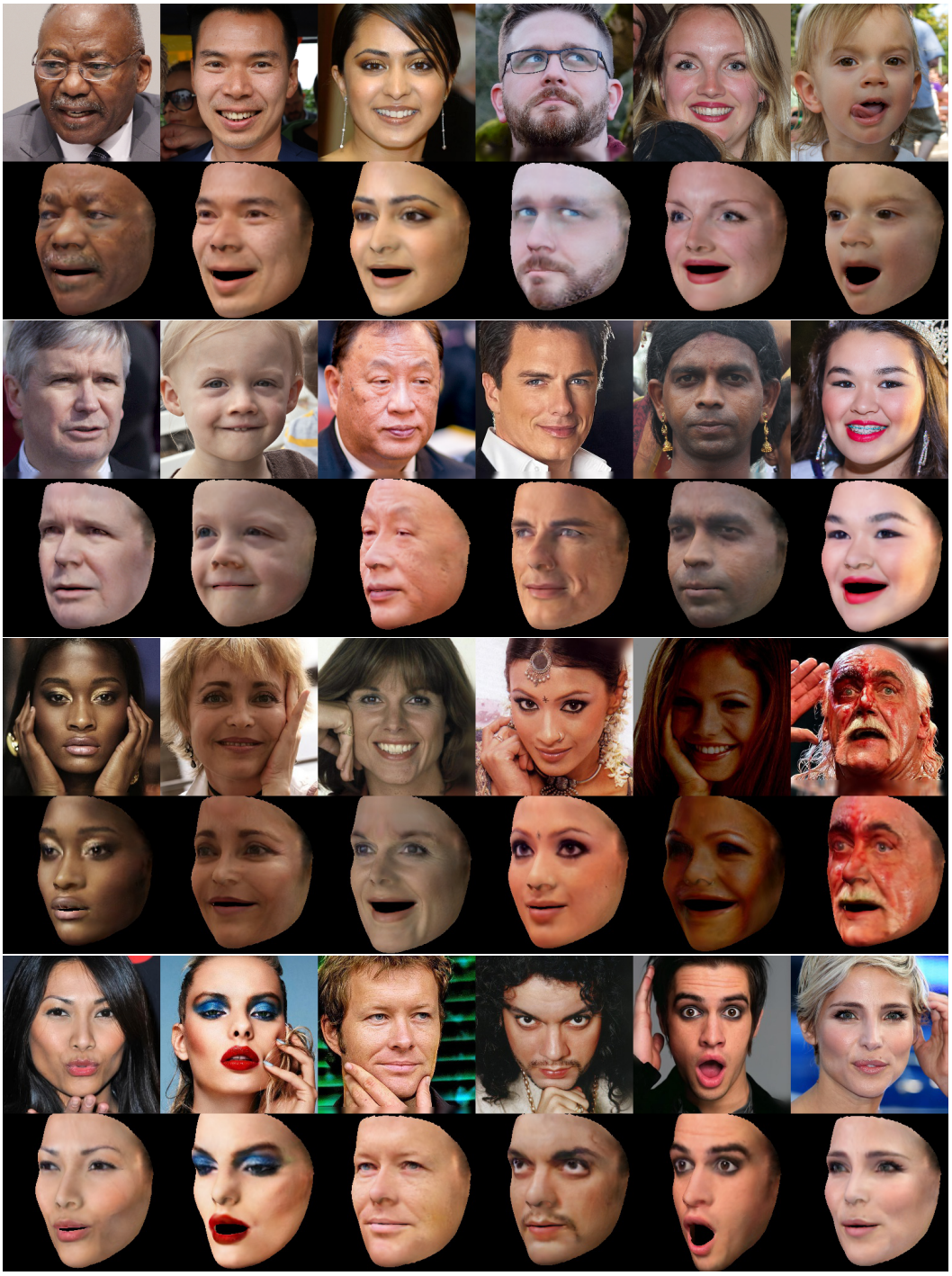


Figure 4.10: Face profiling results

Figure 4.11: Face frontalization results

4.7 Conclusion

This work proposes a novel 2-stage image-to-image translation model that can convert the input face image into its corresponding UV map. In the first stage, with

the proposed UV sampler, pixels in the input face images are selectively sampled and adjusted to form an incomplete UV map, which contains all the visible textures of the face. With the help of this module, the inference stage no longer requires the intervention of 3D shapes. In the second stage, the incomplete UV map is further completed by a UV generator. The training is conducted on purely pseudo UV maps, thus weakly-supervised. With the help of two carefully designed partial UV discriminators, we can generate photo-realistic face textures without the supervision of the complete UV map. Qualitative and quantitative experiments validate the reconstruction ability and the identity-preserving ability of the proposed method.

Chapter 5

Facial Image De-occlusion GAN

5.1 Introduction

Thanks to deep Convolutional Neural Networks (CNNs), massive training data, and the widespread use of cameras, face-related techniques have successfully found many applications in our daily life, covering the fields of entertainment, media, art, security, etc. However, face images in the wild may be occluded by various objects, which leads to loss of information and undesirable noise, further resulting in degraded performance of algorithms for face analysis. To address this issue, researchers typically combine methods such as augmenting training data with synthesized occlusions [121, 122], designing more sophisticated metrics and network structures [123, 124, 125, 126], exploiting elaborated training strategies [127, 128, 129]. Despite their effectiveness, most of them are task-specific and cannot be migrated to other face-related tasks. A relatively more general solution is to de-occlude the face image before passing it to the downstream tasks.

Before the widespread use of deep learning-based methods, the dominant approaches were based on matching and copying visible patches to the missing parts. A representative one is PatchMatch [41], which recursively searches the nearest neighbor textures to fill in the holes. Such copy-and-past-based methods work well in recovering simple low-frequency textures and obtain smooth results; however, they cannot recover high-level textures with complex structures, e.g., eyes, nose, mouth. [130, 131, 132, 133] leveraged Principal Component Analysis (PCA) to recover high-level semantic face textures. Nevertheless, such methods are based on the assumption that face images are in a linear space. Even for images acquired under constrained conditions, the results are blurry and far from satisfactory. As

Figure 5.1: Results of the proposed method. The first row and the third row are input images, the second row and the last row shows the de-occlusion results. Zoom in for a better view.

pointed out by [134], the deep autoencoder is a non-linear generalization of the PCA. Therefore, it is reasonable to replace the PCA part of the above methods with an autoencoder. SSDA [42] proposed a deep sparse autoencoder to remove Gaussian noises and superimposed text from images, pioneered the use of deep neural networks in this area. Influenced by the great success of Generative Adversarial Networks (GAN) [15] in image generation, Context Encoder [43] introduced adversarial training to image inpainting for the first time. Its impressive performance led to the prosperity of GAN-based image inpainting methods. Since then, the combination of autoencoder and discriminator has been adopted as the basic model structure for image inpainting tasks [44, 45, 46, 47]. Despite the remarkable progress so far achieved, current methods mainly focus on filling the holes with visually plausible textures and neglect the detection of imperfectness. When applied to the face de-occlusion task, images must be accompanied by manually labeled masks. One strategy to address this problem is to train the model to detect the occlusions au-

tomatically. Due to the lack of well-labeled paired training data, existing methods mainly train on the synthetic occluded face images [135, 49, 51, 50, 48, 136]. Unfortunately, these methods either generate low-resolution results or only tackle specific occlusion types. We attribute this to the limited variety of synthetic occlusions. Real occlusions have different shapes, textures, and blurred boundaries, making it infeasible for synthesis-based methods to cover all possibilities of occlusion. In addition, collecting completely occlusion-free face images as ground truth is not easy because the forehead part of the face is often occluded by bangs, which is perhaps why none of the existing methods consider hair as a source of occlusion.

Given the limitations of existing methods, this paper explores how to enable neural networks to remove arbitrary kinds of face occlusions. Our main contributions are summarized as follows:

- We propose a novel face de-occlusion framework robust to all types of occlusions, regardless of their shapes and textures.
- We build a large occlusion dataset with extensive manually labeled occlusions from real face images.
- The proposed method outperforms the state-of-the-art baselines quantitatively and qualitatively.

5.2 Related Work

5.2.1 Mask-Dependent Face Inpainting

Image inpainting aims to recover the missing textures of the input image. Recently, the autoencoder-GAN-based models have achieved impressive results. Benefiting from such structure, Context Encoder [43] successfully predicts the 64×64 missing part of 128×128 images. To make the model further concentrate on the missing parts, [45, 44] leverage two discriminators: a global discriminator, which takes the whole image as input, and a local discriminator, which only takes the small region around the missing part. However, such a design is only suitable for a single rectangular(or even square) hole and cannot be applied to images with irregular and

arbitrarily distributed holes. To properly handle free-form masks, [46] proposed a Partial Convolutional Layer, comprising a masked and re-normalized convolution operation followed by a mask-update step. [47] proposed a Gated Convolution module, which learns a soft mask for the features in different layers. However, due to the ineffectiveness of CNNs in modeling long-range correlations between distant textures and the hole regions, the inpainting results often have boundary artifacts and look unreal. [45] proposed a Contextual Attention module, which utilizes the image features as a convolution kernel, forcing the distant textures to interact with each other; this could be simply achieved by attention. [60] proposed a Self Attention module, which allows long-range dependency modeling of features; since then the attention mechanism has been widely applied in image generation tasks [62, 137, 63, 138]. To make the generated image faithfully recover the facial topology, [139] leverages facial landmarks as a shape prior. [138] proposed a two-path probabilistic framework to generate pluralistic inpainting results. The common problem of the above methods is that they cannot detect the occluded parts automatically and rely on manually specified masks, limiting their usage scenarios.

5.2.2 Mask Free Face De-occlusion

Some methods are devoted to getting rid of masks' dependence and making the face de-occlusion process fully automated. They are typically trained on face pairs with synthesized occlusions. [135] proposed an LSTM-Autoencoder to gradually substitute the occlusions with facial textures. Benefiting from an elaborated forward pass, [49] proposed a semi-supervised face de-occlusion method, which is trained to predict the occlusion mask without the supervision of ground-truth. [51] proposed a two-stage GAN, where the first GAN aims to reconstruct the occlusion part solely, and the second GAN takes the result of the first GAN as a hint making the input image occlusion-free. Due to the limited variety of occlusions in the training set, the above methods can only generate images in 128×128 resolution with noticeable artifacts, restricting their application to face recognition. Given the complexity of occlusion types, a relatively simple task would be to deal with only specific occlusion: [48, 136] for eyeglasses and [140] for masks. Besides being limited by the

occlusion variety, as they lack face topology information, the results are often far from satisfactory, especially on large-angled face images.

5.2.3 3D-Guided Face De-occlusion

The work closest to ours is FD3D [50], where we both use 3D Morphable Model [13] (3DMM)-based reconstruction results as guidance. FD3D feeds the synthetic occluded face and the 3D reconstructed one to an adversarial auto-encoder and expects the output to be occlusion-free. Our method surpasses theirs in the following aspects: 1) They inpaint directly on the occluded face despite the challenge of converting various occlusion textures to face textures; in contrast, our method explicitly removes occlusions with predicted masks before inpainting, bypassing the disturbance of the occlusion textures. 2) FD3D entirely relies on supervised learning, which requires occlusion-free images to serve as the ground truth, while our method can be trained on initially occluded images. 3) They synthesize images with fixed occlusions, namely, cup, glasses, hand, mask, and scarf, which have 99 variations in total; by detecting the face mask instead of innumerable types of occlusions, our method can handle all types of occlusion.

5.3 Proposed Method

5.3.1 Overview

Figure 5.2 illustrates the overview of our method. Given that covering all possible occlusions is impossible, we consider the opposite direction. We employ a face segmentation module N_S to predict the face mask M_f, which is much easier than predicting the mask of all kinds of occlusions. Then, we use the 3D reconstruction module N_R to predict the prior 3DMM-based texture and the corresponding mask for the whole face, denoted as I_m and M_m, respectively. The mask of the occlusions M_o can then be calculated as:

$$M_o = M_m - M_m \odot M_f, \tag{5.1}$$

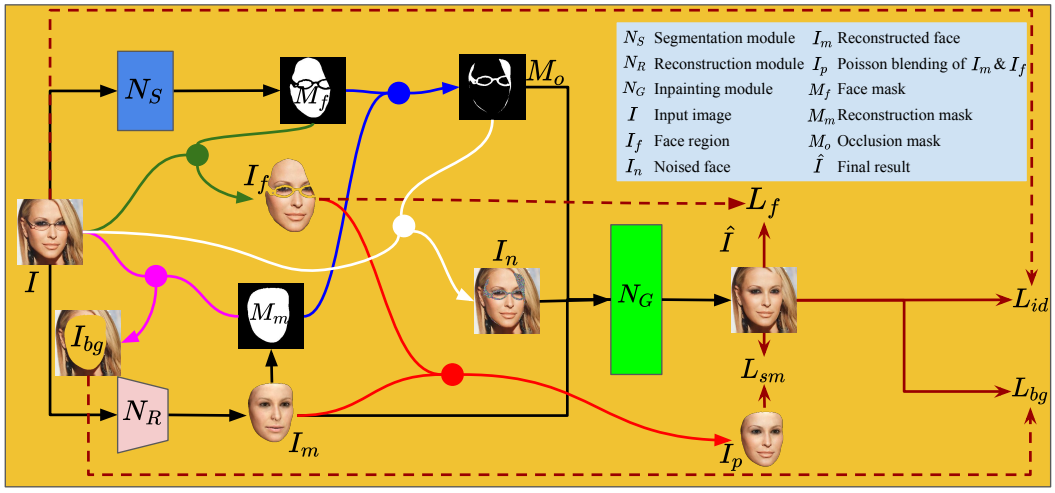


Figure 5.2: Overview of the proposed method, which does not include the discriminator part for space limitation

where $\odot$ denotes element-wise product. For face inpainting, we replace the occluded part of the input image I with Gaussian noise to obtain I_n, which minimizes the disturbance of the occlusion textures. I_n, I_m, and M_o are concatenated and fed into the inpainting module N_G, and the output $\hat{I}$ is expected to be occlusion-free. The following terms supervise the inpainting: 1) the background region I_{bg}, 2) the non-occluded face region I_f, 3) the Poisson blending result of I_f and I_m, denoted as I_p.

The overall framework contains three modules, N_S, N_R, and N_G, for face segmentation, 3D reconstruction, and image inpainting. We present the implementation details of each module separately in the following.

5.3.2 Face Segmentation

Extracting face regions from occluded face images is a simple, valuable, but never seriously addressed problem. As a result, the recent 3D face reconstruction method [3] requires this functionality still needs to resort to a Naive Bayesian classifier. We present our solution to the problem in this section.

Data Preparation. We build our training data based on CelebAMask-HQ [141], which has 30,000 high-resolution face images, each with a segmentation

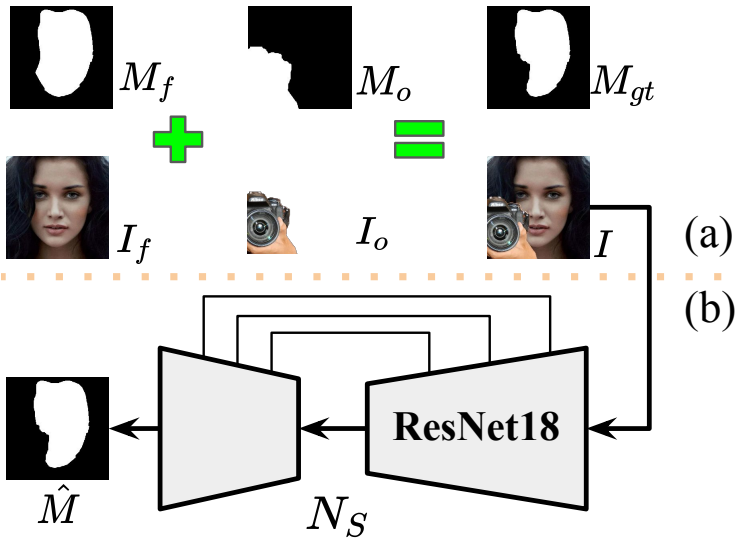


Figure 5.3: (a) Data augmentation method. I_f: face image, M_f: face mask of I_f, I_o: occlusion, M_o: occlusion mask, I: occluded image, M_{gt}: face mask of I. (b) Structure of the segmentation model N_S, $\hat{M}$: predicted face mask.

mask of facial attributes. Basically, given a segmentation mask of a face image, we can easily get the face region by gathering the labels of eyes, nose, lips, mouth, and skin. Unfortunately, the segmentation mask in the dataset does not consider the occlusions, e.g., hands, accessories, microphones, which are all misclassified as skin. In addition, the segmentation masks in the dataset do not distinguish between sunglasses and spectacles; however, in our task, we should not discard the skin and eyes under the transparent lenses of the spectacles. To solve the above problems, we manually picked out the occluded face images with incorrect segmentation masks and labeled more than 1700 occlusions from them. We also collected about 300 occlusions from Google to get a more balanced occlusion type (e.g., masks, scarfs). In addition, we collected more than 800 texture patches covering every type we can think of, which will substitute for the original occlusion textures during training to produce more occlusion variations. Those occlusions are utilized for data augmentation, as illustrated in Figure 5.3 (a) and the following equation.

$$\begin{aligned} M_{gt} &= M_f \odot (1 - M_o), \\ I &= I_f \odot (1 - M_o) + I_o \odot M_o. \end{aligned} \tag{5.2}$$

Model Structure. N_S follows the classical U-Net structure [80], with ResNet18 [112] as encoder, while the decoder is a stack of convolutional blocks corresponding to each stage of the encoder. Each decoder block takes features from its previous block and its symmetric encoder stage through skip-connection. Since the target mask is binary, a single channel output for the decoder suffices.

Losses. The training is guided by the Dice loss [92] and the Binary Cross-Entropy loss (BCE):

$$L_{dice} = 1 - \frac{2\sum \hat{M} \odot M_{gt}}{\sum \hat{M} + \sum M_{gt}}, \tag{5.3}$$

$$\begin{aligned} L_{bce} = -\frac{1}{WH}\sum \big(M_{gt} \odot \log\hat{M} + \\ (1 - M_{gt}) \odot \log(1 - \hat{M})\big), \end{aligned} \tag{5.4}$$

where W and H represent the width and height of the mask, respectively. Further, we apply the “online hard example mining" (OHEM) [142] strategy to the BCE loss to make the training focus on hard examples, thus achieving more effective and efficient training.

The proposed segmentation module achieves a precision of 0.981 and an IoU score of 0.954 on the validation dataset, which is acceptable considering the inaccurate edge region of the ground truth. By random checking the results, we surprisingly find that the predicted face region is often more accurate than the ground truth. We use this module to extract the face regions of the images in CelebA-HQ and FFHQ, which greatly facilitates the training of the other two modules, i.e., the reconstruction module N_R and the inpainting module N_G.

5.3.3 3D Face Reconstruction

The reconstruction module is adapted from [3], the state-of-the-art 3D reconstruction method with a ResNet50 [112] as its backbone. Given a face image, it predicts a vector $c \in \mathbb{R}^{239}$, containing 6 translation and rotation parameters; 144 shape and 80 texture coefficients of the 3DMM; and nine illumination coefficients of the Spherical Harmonics [143, 144] model. With vector c, the face is reconstructed and further rendered to the image through a differentiable renderer.

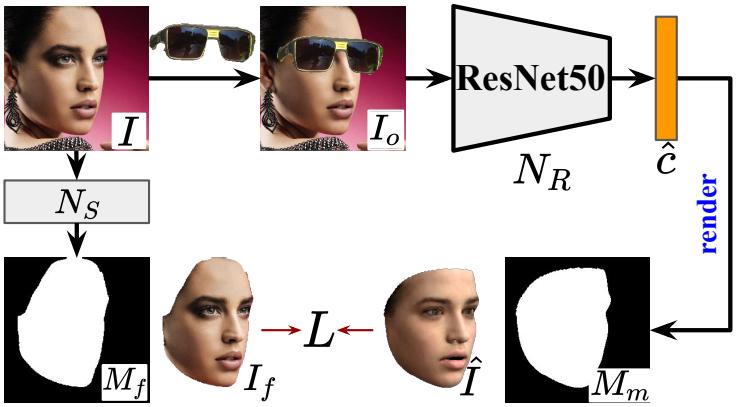


Figure 5.4: Overview of the 3D reconstruction module. I: original image, I_o: occluded image, $\hat{c}$: predicted 3D parameters, $\hat{I}$: reconstructed face, M_m: mask of $\hat{I}$, M_f: face mask of I, I_f: $I \odot M_f$

Although [3] performs well on occlusion-free faces and even demonstrates certain robustness to small occlusions, it still cannot handle severe occlusions, e.g., sunglasses and masks. As shown in the second row of Figure 5.5, sunglasses make the reconstructed face have dark circles. In the third column, they even change the color of the reconstructed skin. Therefore, in our usage scenario, i.e., face de-occlusion, it is necessary to retrain an occlusion-robust 3D face reconstruction module. Now we present, as shown Figure 5.4, our training strategy.

Training data. The training is based on CelebA-HQ and FFHQ. Firstly, we use the pre-trained model of [3] to predict the reconstruction parameters of the training data, denoted as c_{gt}. Then, we filter out all the images with sunglasses since their corresponding c_{gt} risk to be inaccurate. Next, we leverage the face segmentation module N_S to detect the face masks of the training data, denoted as M_f, for calculating the face-related losses. At last, as with the previous section, we randomly superimpose heavy occlusions to I during training to get occluded input images for N_R.

Losses. Since we already have c_{gt}, the most straightforward option is to use it as supervision; the loss is as:

$$L_{coef} = \frac{1}{N}\|\hat{c} - c_{gt}\|_1, \tag{5.5}$$

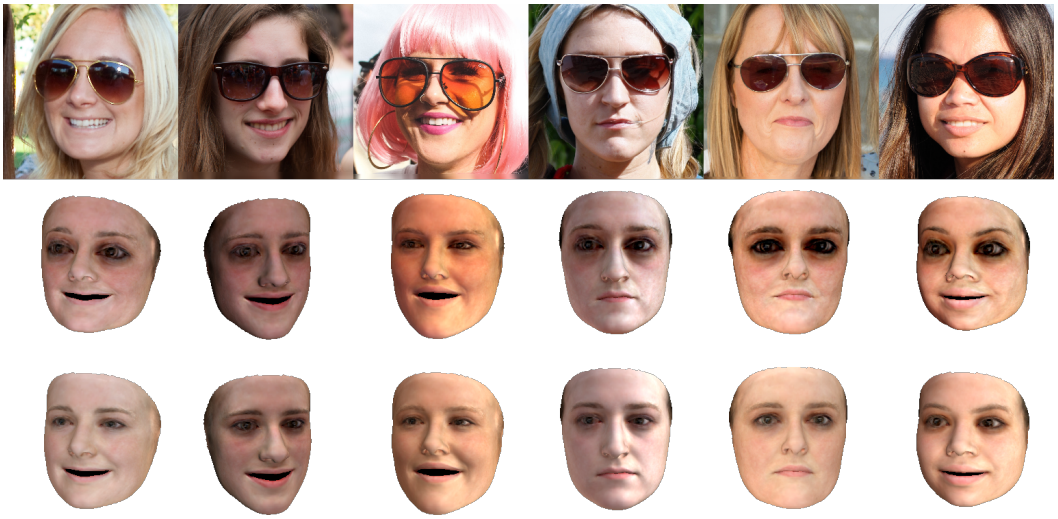


Figure 5.5: The first row shows the input images; the second row and the last row are reconstructed faces by [3] and ours, respectively.

where $\hat{c}$ is the predicted coefficient, and N is its dimension.

Training solely with Equation 5.5 will result in inaccurate and non-discriminative results. Since it equally optimizes all the coefficients. However, different parts of $\hat{c}$ obviously have different impacts on the reconstruction result, e.g., the poses play a more critical role than the illumination coefficients. Therefore, we also leverage the pixel-wise and the perceptual [93] losses to reduce the discrepancy between the reconstructed face $\hat{I}$ and the real face I_f:

$$L_{pix} = \frac{1}{\sum M} \|\hat{I} \odot M - I_f \odot M\|_2, \tag{5.6}$$

$$L_{id} = 1 - \frac{\mathcal{F}(\hat{I})^T \cdot \mathcal{F}(I_f)}{\|\mathcal{F}(\hat{I})\| \cdot \|\mathcal{F}(I)\|}, \tag{5.7}$$

where M is the overlap of the original face mask M_f and the reconstruction mask M_m; $\mathcal{F}(\cdot)$ denotes the feature embedding function of a pre-trained face recognition model, and we use Arcface [119] here. We only extract the feature of I_f, rather than the entire image I, to avoid the noises introduced by the non-facial patterns.

To accelerate the training, we also use the landmark loss. Thanks to c_{gt}, we have ground truth 3D facial landmarks $\mathbf{q}$, thus eliminating the need for a 2D facial

landmark detector as required in other methods. The loss is calculated as :

$$L_{ldmk} = \frac{1}{n_{pt}} \sum_{i=1}^{n_{pt}} \omega_n \|\hat{\mathbf{q}}_i - \mathbf{q}_i\|^2, \tag{5.8}$$

where n_{pt} denotes the number of landmarks, $\hat{\mathbf{q}}_i$ denotes the predicted 3D coordinates of the i-th facial landmark. The weights ω_n are set to 20 for the nose and inner mouth points and 1 for others.

As c_{gt} provides sufficient regularity, we discard the complex regularization terms in [3]. The overall loss is the weighted sum of the above losses:

$$L = L_{coef} + \lambda_{pix} L_{pix} + \lambda_{id} L_{id} + \lambda_{ldmk} L_{ldmk}, \tag{5.9}$$

where $\lambda_{pix} = 1.92$, $\lambda_{id} = 0.2$, $\lambda_{ldmk} = 1.6e^{-3}$.

Figure 5.5 compares our results with those of [3], demonstrating the effectiveness of the proposed training strategy in improving the occlusion-robustness of the model. A quantitative comparison is also performed: we use [3] and our model to reconstruct 1000 face images with synthetic occlusions, the L_{pix} of our model is 0.153, which is much less than 0.177 of [3].

5.3.4 Face Inpainting

With the N_S and N_R described above, we get the following information of an occluded face image: face mask M_f, face I_f, reconstructed face I_m, reconstructed face mask M_m. Based on M_f and M_m, Equation 5.1 further calculates the occlusion mask M_o. The goal of this section is to restore the missing textures indicated by M_o.

The generator N_G is constructed from stacked gated residual blocks [47] and follows the classical encoder-decoder structure; a self-attention [60] module is applied to the bottleneck features. As shown in Figure 5.6, the input is composed of three parts: the occlusion mask M_o, the face image with noised occlusion I_n, and the reconstructed face I_m. The output $\hat{I}$ is the de-occluded image. A VGG [145] shaped discriminator is further employed to increase the photorealism to $\hat{I}$.

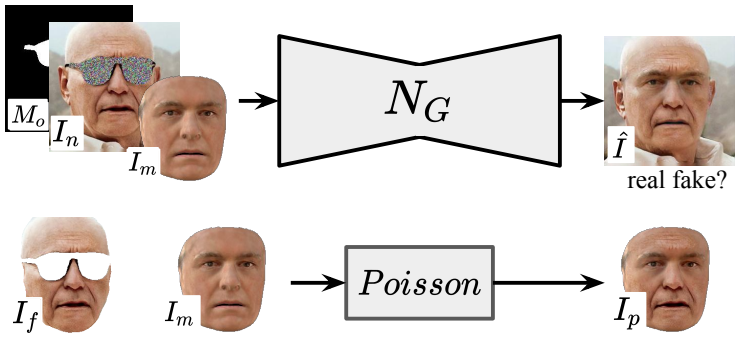


Figure 5.6: Overview of the inpainting module.

We randomly superimpose occlusions to the training data and train N_G to recover the missing textures. One issue to note is that the input image itself always contains occlusions, such as bangs, and we do not have ground truth for these parts. So we use Poisson Blending [115] to seamlessly merge I_f and I_m and get I_p, which provides weak supervision for the inpainting task. The training is guided by the following losses:

Pixel-wise face loss.

$$L_{pix} = \frac{1}{\sum M} \|\hat{I} \odot M - I_f \odot M\|_1, \tag{5.10}$$

where $M = M_m \odot M_f$, limiting the loss to be calculated on the face region solely. This loss serves for recovering textures occluded by the synthetic occlusion and cannot recover the initially occluded textures.

SSIM loss. We use I_p to guide the generation of initially occluded textures. Although the Poisson blending result is visually pleasing, it changes the color of I_f, so we cannot simply apply L_1 or L_2 loss. Instead, we leverage the Structural Similarity loss (SSIM) [146], emphasizing the structural level discrepancy:

$$L_{sm} = \frac{-1}{\sum \bar{M}_m} SSIM(\hat{I} \odot \bar{M}_m, I_p \odot \bar{M}_m), \tag{5.11}$$

where $SSIM$ stands for the similarity mapping function; $\bar{M}_m$ is the eroded M_m, eliminating the edge effects in the similarity map. To further mitigate the impact of inaccurate color of I_p and make the loss focus on the missing textures, we apply

the OHEM again as in Equation 5.4.

Background loss. Areas other than the face should remain unchanged:

$$L_{bg} = \frac{1}{\sum M_{bg}} \|\hat{I} \odot M_{bg} - I \odot M_{bg}\|, \tag{5.12}$$

where M_{bg} denotes the background mask, calculated by $1 - M_m$. We also erode M_bg to alleviate the edge effects.

Identity loss. The generated image should have the same identity as the original image, so we also leverage Equation 5.7 to add feature level identity constraints.

TV loss. To penalize the noises in $\hat{I}$, we adopt the total variation loss [96].

$$L_{tv} = \frac{1}{WHC} \|\nabla_x \hat{I}\|^2 + \|\nabla_y \hat{I}\|^2, \tag{5.13}$$

where W, H, C are the width, height, and the number of channels, respectively, $\nabla\_$ calculates the image gradient along a direction.

Adversarial loss. To further make $\hat{I}$ photorealistic, we employ the adversarial loss:

$$L_{adv} = -\mathbb{E}_{\hat{I}}[\log(D(\hat{I}))], \tag{5.14}$$

where $D(\cdot)$ denotes the output of the discriminator, which is the term to be maximized by the generator.

The global objective function of the generator can be summarized as follows:

$$\begin{aligned} L = &\lambda_{pix} L_{pix} + \lambda_{sm} L_{sm} + \lambda_{bg} L_{bg} + \\ &\lambda_{id} L_{id} + \lambda_{tv} L_{tv} + \lambda_{adv} L_{adv}, \end{aligned} \tag{5.15}$$

where the weights are empirically set to $\lambda_{pix} = 10$, $\lambda_{sm} = 5$, $\lambda_{bg} = 5$, $\lambda_{id} = 0.2$, $\lambda_{tv} = 0.1$, $\lambda_{adv} = 0.01$.

Discriminator loss. We use the BCE loss to train the discriminator, aiming to distinguish $\hat{I}$ from real images:

$$L_{adv} = \mathbb{E}_I[\log(D(I))] + \mathbb{E}_{\hat{I}}[\log(1 - D(\hat{I}))], \tag{5.16}$$

where I is the real image with less occlusion, selected from CelebA-HQ and FFHQ according to the overlap rate of its corresponding M_m and M_f.

5.4 Experiments

The proposed method can effectively remove the occlusions from face images. To demonstrate the power of our model, we qualitatively compare our results with state-of-the-art methods, including those for face de-occlusion and for image inpainting. Quantitative experiments are conducted in two aspects as well, namely reconstruction ability and identity recovery ability.

5.4.1 Implementation Details

We follow an incremental training strategy: 1) We train the segmentation module N_S. 2) we train the reconstruction module N_R based on N_S. 3) We train the inpainting module N_G based on the result of the first two. The training is performed on two public datasets, CelebAMask-HQ [141] and FFHQ [31], and our manually labeled/collected occlusions and textures. All the images are aligned with facial landmarks detected by [2] and cropped to 256×256.

The training of N_S consists of two rounds: In the first round, we train on 300,000 images of CelebAMask-HQ with our manually labeled occlusions to obtain a coarse model. In the second round, we apply the coarse model to both CelebAMask-HQ and FFHQ [32], select 500 hard examples from the results, relabel and add them to the occlusion dataset to retrain a more accurate model. Each round is trained for 30 epochs with a batch size of 16 and a learning rate of $1e^{-4}$. The training takes about two hours on two Nvidia GTX 1080 GPUs. The 3D reconstruction module and the face inpainting modules are trained on 100,000 images of CelebAMask-HQ and FFHQ, with an initial learning rate of $1e^{-4}$. For the reconstruction module N_R, we train 50 epochs with a batch size of 16, and for the inpainting module N_G, we train 80 epochs with a batch size of 4. The learning rates of both modules are dropped halfway through the training with ratios of 0.2 and 0.1, respectively. All three modules are optimized using Adam [102] with a weight decay of 0. Betas are

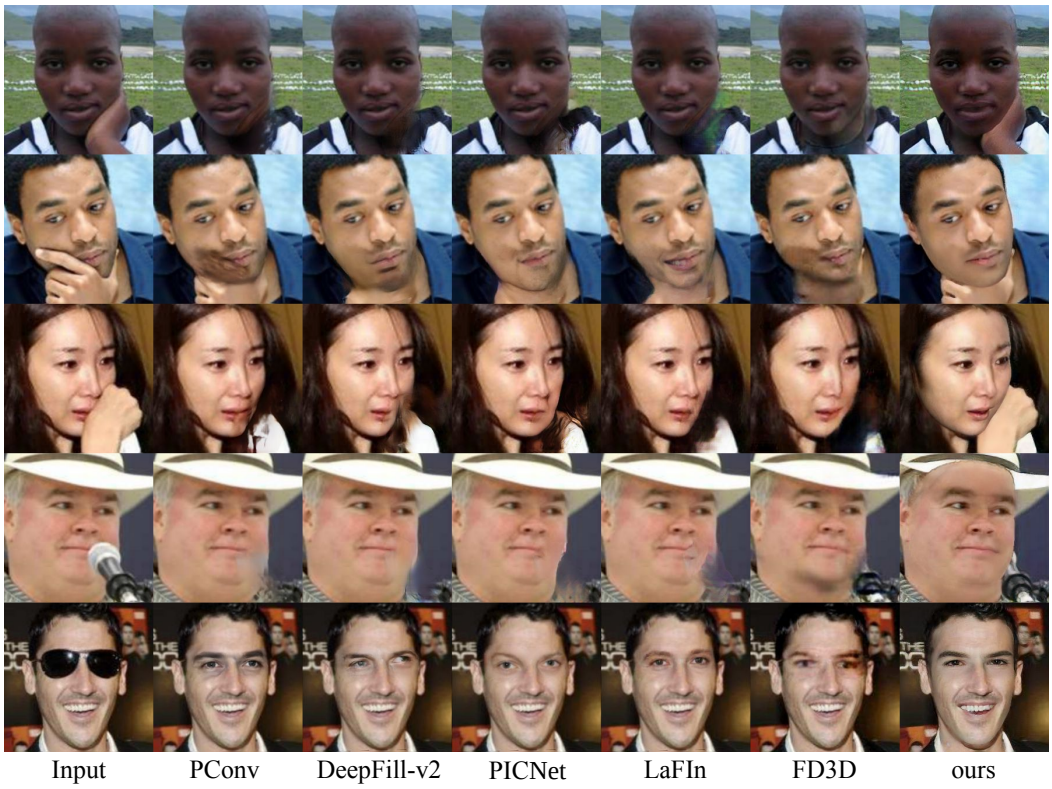


Figure 5.7: De-occlusion results compared with state-of-the-art methods. Input images are aligned and cropped according to the rules of each method, and we re-paste the outputs back to facilitate comparison.

set to $[0.9, 0.999]$ for N_S and N_R, $[0.5, 0.999]$ for N_G. It takes about 80 hours to train N_R and 40 hours to train N_G on two Titan X Pascal GPUs.

5.4.2 Qualitative Results

The method closest to ours is FD3D [50]; we both leverage 3D face reconstruction for face de-occlusion. Unfortunately, they do not release their code, so we use images from their paper to conduct the qualitative evaluation. Figure 5.7 compares our method with FD3D and several other publicly available image inpainting methods, including DeepFill-v2 [47], PConv [46], PICNet [147], and LaFIn [139]. The FD3D results are taken from their paper, while the rest are generated from their official implementations with models pre-trained on face images. We provide those inpainting methods with manually labeled occlusion masks for a fair comparison.

One issue to note is that when FD3D compares with other methods in their paper, the images are not pre-aligned according to the rules of the corresponding method, resulting in questionable inferior results.

As shown in Figure 5.7, our method generally outperforms FD3D and other image inpainting methods, especially on face images with large yaw angles. We argue that the image inpainting methods are purely based on statistical learning, thus highly dependent on the training data distribution. When a few face images with large yaw angles occur, these methods record dramatic performance degradation. Although FD3D does not require a manual mask for the de-occlusion, they apparently can only handle limited types of occlusions: the bangs in the third row and the hat in the fourth row are not identified as occlusions. In addition, FD3D produces low-quality face images with blurred boundaries and unreal textures, as evidenced in the images from the second to the last row. We believe this is due to the following reasons: 1) they do not explicitly detect the occlusions and mask them out, causing the generator to only cope with the occlusions it has ever seen while failing to de-occlude arbitrary occlusions; 2) they use a very coarse 3D reconstruction method, which cannot provide correct and effective face prior to the generator.

5.4.3 Quantitative Results

This section evaluates the proposed method in two aspects: the reconstruction ability and the identity recovery ability. We synthesize 1000 sunglasses-occluded face images and use DeepFill-v2, PICNet, LaFIn, and our method to recover the initial images. Since the image inpainting methods only synthesize the missing region, the reconstruction ability is evaluated only in that region, with the following metrics: L1 loss, SSIM [146] score, and PSNR score. The identity recovery ability is evaluated by the cosine similarity of the features extracted by the pre-trained ArcFace [119]. Results are reported in Table 5.1.

As can be seen, our method outperforms the state-of-the-art image inpainting methods across all listed metrics. It performs particularly well in recovering the identity, much exceeding its comparators. We attribute this mainly to the occlusion-robust face reconstruction module.

Method	$L_1\downarrow$	SSIM ↑	PSNR↑	ID↑
LaFIn	0.067	0.607	27.944	0.639
PICNet	0.0654	0.604	28.137	0.615
DeepFill	0.065	0.604	28.105	0.659
w/o SSIM	0.065	0.619	28.491	0.665
Ours	**0.062**	**0.623**	**28.902**	**0.690**

Table 5.1: Comparison of the proposed method with state-of-the-art image inpainting methods.

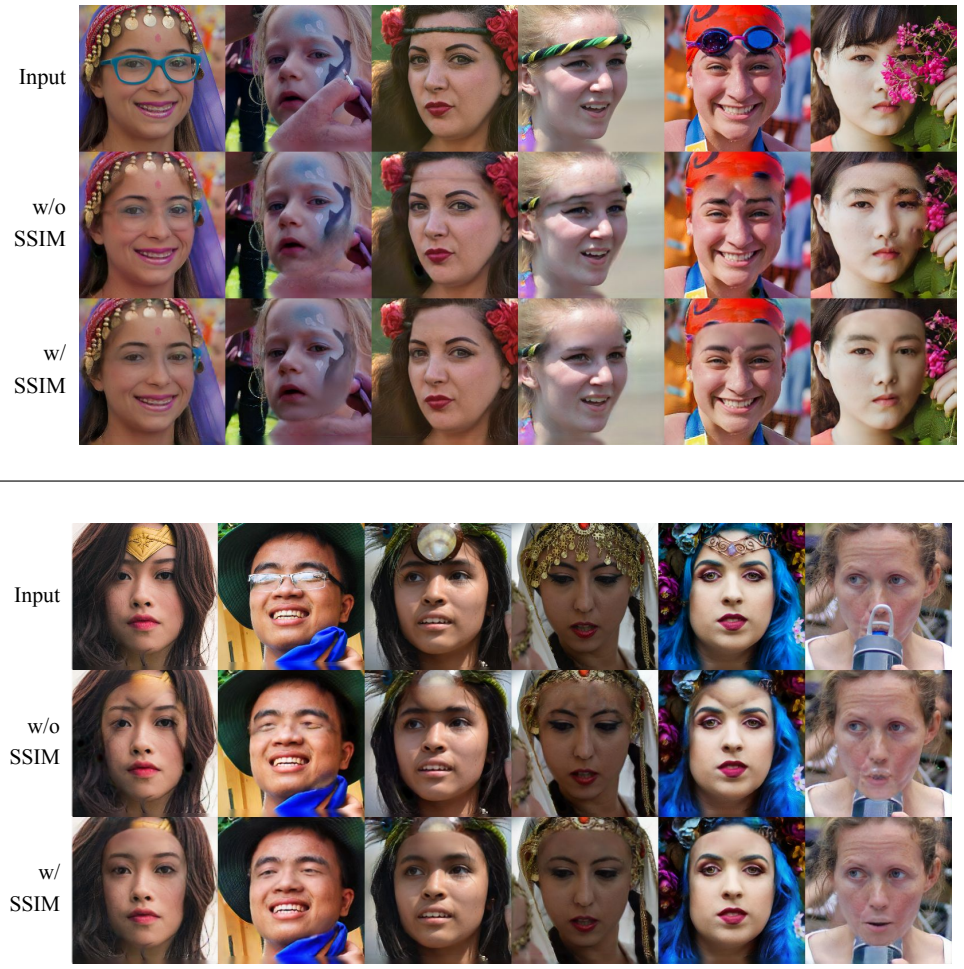


Figure 5.8: Ablation study of the SSIM loss.

5.4.4 Ablation Study

The three modules in our framework are interdependent, making it impossible to remove one for ablation study. Instead, we mainly focus on the SSIM loss of the inpainting module, as the reconstruction and identity losses are conventional prac-

Figure 5.9: Limitation of the proposed method.

tices.

As Table 4.2 shows, without the SSIM loss, the model degrades in all aspects of metrics (nevertheless, benefiting from the 3D prior, it still outperforms others). Moreover, the initially occluded textures are now only supervised by the adversarial loss; thus, the results are not guaranteed to be occlusion-free. As shown in Figure 5.8, the shadows on the occlusion edges are difficult to eliminate with mere adversarial loss.

5.4.5 Discussion

Limitation. The proposed method focuses on removing the occlusions within the 3D reconstructed face mask. For occlusions spanning the face and the background, it can only rigidly remove the parts above the face without producing a smooth transition, creating an unrealistic scene where the face seems to hover above the occlusions, as Figure 5.9 shows. We have tried to create a gap between the face area and the background through mask erosion, and the gap is solely supervised by the adversarial loss during training. This trick alleviates the boundary effect to some extent but still cannot cope with large-sized occlusions.

Social impacts. Our method helps pre-process face images to avoid the negative impact of occlusions such as hair, glasses, hands, etc., on downstream tasks (e.g., fine-grained 3D face reconstruction, face recognition). Despite the benefits it brings, it also risks violating human privacy. To quantify this risk, we further conduct experiments to analyze the identity recovery ability across different types of occlusions.

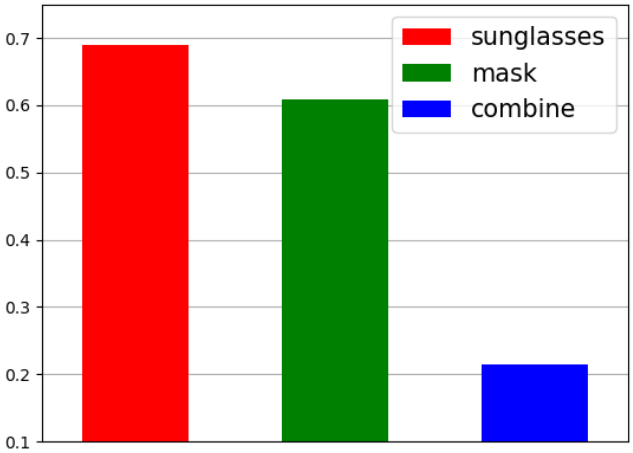


Figure 5.10: Identity recovery ability for different types of occlusion.

Figure 5.10 shows that the proposed method performs well on sunglasses-occluded faces; the performance decreases for the mask-type occlusion; and when combining the mask with sunglasses, the method can no longer recover the face identity. The above observation proves that the method is controllable for privacy violations. People's identities can be safely protected when they simultaneously wear sunglasses and masks.

5.5 Additional results

Figure 5.11 are the de-occlusion results of the initially occluded images in the FFHQ [32] dataset.

5.6 Conclusion

This work proposes a segmentation-3D reconstruction-guided facial image de-occlusion method that automatically removes all types of occlusions. We first analyze the limitations of current face de-occlusion methods: they either require a manually labeled mask or can only handle a limited number of occlusion types, which mainly stems from the vast diversity of possible occlusions. Our key innovation is bypassing the segmentation of infinite occlusions and instead segmenting the face regions, which is much easier. We manually labeled a large face occlusion

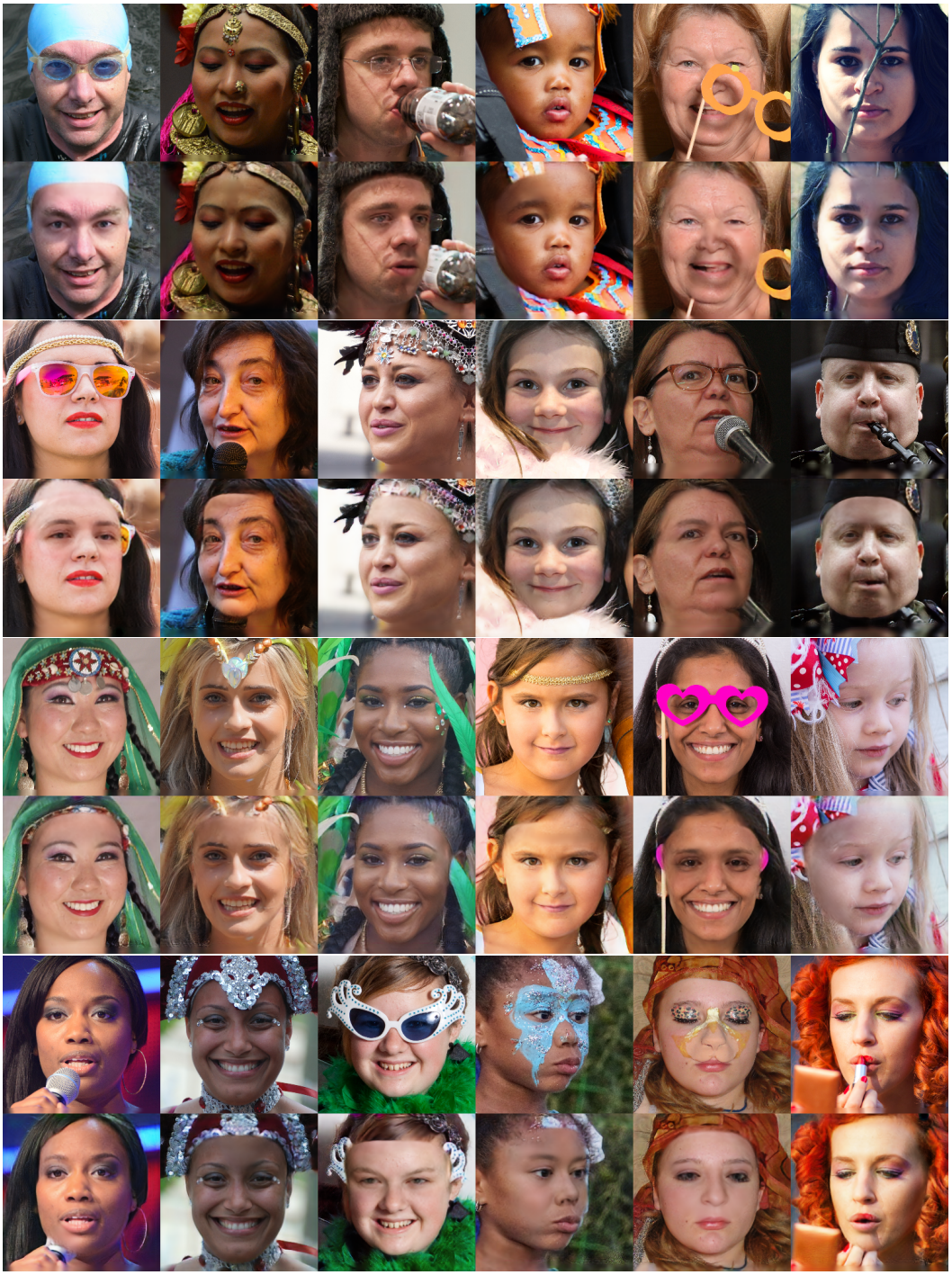


Figure 5.11: De-occlusion results.

dataset, based on which we trained a face segmentation module N_S and an occlusion-robust 3D reconstruction module N_G. Given an occluded face image, N_S and N_G

work collaboratively to mask the occlusions and provide beneficial priors to guide the subsequent face inpainting module. Qualitative and quantitative evaluations demonstrate the superiority of the proposed method.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

We focused on GAN-based face image generation and its application to face recognition. Specifically, we proposed several novel methods for 1) face pose editing of 2D facial images, 2) texture generation for 3D face reconstruction, 3) facial image de-occlusion. The methods employed 2D and 3D-based techniques in computer vision and computer graphics, including image generation, segmentation, and 3D face reconstruction.

Chapter 3 first analyzed the problem of the existing encoder-decoder-based face-pose editing methods: they primarily focus on modeling the identity preservation ability during pose synthesis but are less able to preserve the image style properly, which refers to the color, brightness, saturation, etc., resulting in a style discrepancy between the output image and the input image. Then, it proposed a two-stage framework that converts the problem into pixel-sampling and image inpainting. In the first stage, the pixel attention sampling module directly samples pixels from the input image, thus explicitly preserving more style information. In the second stage, the missing textures are further completed, and the noises are removed. Thanks to the pixel sampling module and the high-dimensional embedding in the inpainting module, the image style is well preserved during face pose editing.

Although pure 2D-based methods could generate face images of different view angles, the generation consistency is poor due to the lack of global consistency constraints and prior knowledge of the 3D shape. Chapter 4 solved the problem by generating high-fidelity global facial textures for 3D face reconstruction. The proposed method also benefited from a pixel-sampling module similar to Chapter3

and exploited a two-stage framework. In the first stage, the sampling module directly predicts an incomplete *global* face texture (UV map) from the input image without relying on 3D shapes as traditional methods. In the second stage, the incomplete UV map is completed by a texture generator supervised by the pseudo UV map and multiple partial UV discriminators, bypassing the need for scarce UV datasets and the costly optimization approach. The generated textures are of high-fidelity and could be used for data augmentation in face recognition tasks to improve the robustness of the model to view angles.

In addition to the view angle, another factor that harms face recognition accuracy is the occlusion. Chapter 5 tackled this problem by proposing a facial image de-occlusion framework. We trained a practical face mask prediction module and an occlusion-robust face reconstruction module, with extensive manually labeled occlusions from real images. Any occlusions could be masked out by collaboratively exploiting the two modules, despite their shapes and textures. The inpainting module takes the occlusion masks and the 3D reconstruction results as prior and generates plausible non-occluded face images. The proposed method far exceeds the state-of-the-arts qualitatively and quantitatively.

Our work advanced incrementally. In the beginning, we only worked with pure 2D images, without incorporating 3D shape information. Chapter 4 focused only on the texture for 3D face reconstruction; the shape part still relied on an off-the-shelf model. While chapter 5 trained all texture and shape modules from scratch. Our work has spanned from 2D to 3D, from low resolution to high resolution, from a single model to a combination of multiple models, from following the state-of-the-art methods to significantly exceeding them.

6.2 Future Work

The 3D reconstruction method used in this work is based on the 3D Morphable Model [13], a low-dimensional linear model that cannot capture high-frequency details of shapes and textures. Chapter 4 solved the problem of unreal textures by training a texture generator; however, generating fine-grained 3D face shapes still

remains unsolved. There are two categories of research addressing this problem: one is to construct larger 3D face datasets as training data, which are costly and involve privacy and legal issues; the other is to follow a analysis-by-synthesis training strategy: the model render the reconstructed 3D face into image space and uses the reconstruction loss to guide the generation of fine-grained shapes. However, such a method could be corrupted by the occlusions (e.g., bangs, eyeglasses) in the input face image. In the future, we plan to use the method of chapter 5 to de-occlude the face image before applying the above method, thus eliminating the negative influences of the occlusions.

Bibliography

[1] A. Asthana, T. K. Marks, M. J. Jones, K. H. Tieu, and M. Rohith, "Fully automatic pose-invariant face recognition via 3d pose normalization," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 937–944. xi, 2, 3, 5

[2] A. Bulat and G. Tzimiropoulos, "How far are we from solving the 2d & 3d face alignment problem? (and a dataset of 230,000 3d facial landmarks)," in *International Conference on Computer Vision*, 2017. xi, 47, 48, 71, 94

[3] Y. Deng, J. Yang, S. Xu, D. Chen, Y. Jia, and X. Tong, "Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0. xiv, 6, 59, 62, 63, 68, 70, 73, 75, 86, 88, 89, 90, 91

[4] J. Deng, S. Cheng, N. Xue, Y. Zhou, and S. Zafeiriou, "Uv-gan: Adversarial facial uv map completion for pose-invariant face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7093–7102. xv, 6, 41, 60, 61, 62, 63, 64, 74

[5] H. Zhou, J. Liu, Z. Liu, Y. Liu, and X. Wang, "Rotate-and-render: Unsupervised photorealistic face rotation from single-view images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5911–5920. xv, 5, 61, 72, 74, 75

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012. 1

[7] M. Mori, K. F. MacDorman, and N. Kageki, "The uncanny valley [from the field]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 98–100, 2012. 2

[8] A. W. Young, E. H. De Haan, and R. M. Bauer, "Face perception: A very special issue," 2008. 2

[9] J. Liu, A. Harris, and N. Kanwisher, "Perception of face parts and face configurations: an fmri study," *Journal of cognitive neuroscience*, vol. 22, no. 1, pp. 203–211, 2010. 2

[10] B. Rossion, R. Caldara, M. Seghier, A.-M. Schuller, F. Lazeyras, and E. Mayer, "A network of occipito-temporal face-sensitive areas besides the right middle fusiform gyrus is necessary for normal face processing," *Brain*, vol. 126, no. 11, pp. 2381–2395, 2003. 2

[11] E. de Vries and D. Baldauf, "Attentional weighting in the face processing network: a magnetic response image-guided magnetoencephalography study using multiple cyclic entrainments," *Journal of cognitive neuroscience*, vol. 31, no. 10, pp. 1573–1588, 2019. 2

[12] T. Hassner, S. Harel, E. Paz, and R. Enbar, "Effective face frontalization in unconstrained images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4295–4304. 2, 59

[13] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 1999, pp. 187–194. 2, 46, 59, 85, 104

[14] ——, "Face recognition based on fitting a 3d morphable model," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 25, no. 9, pp. 1063–1074, 2003. 3

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014. 3, 11, 18, 41, 60, 82

[16] L. Tran, X. Yin, and X. Liu, "Disentangled representation learning gan for pose-invariant face recognition," in *Proceedings of the IEEE conference on*

computer vision and pattern recognition, 2017, pp. 1415–1424. 3, 4, 5, 42, 45, 46, 56, 61

[17] R. Huang, S. Zhang, T. Li, and R. He, "Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2439–2448. 3, 4, 5, 42, 44, 46, 54, 57

[18] Y. Hu, X. Wu, B. Yu, R. He, and Z. Sun, "Pose-guided photorealistic face rotation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8398–8406. 3, 4, 5, 42, 44, 45, 46, 57, 61, 72

[19] Y. Qian, W. Deng, and J. Hu, "Unsupervised face normalization with extreme pose and expression in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9851–9858. 3, 4, 5, 42, 47, 56, 57, 72

[20] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "Multi-pie," *Image and vision computing*, vol. 28, no. 5, pp. 807–813, 2010. 4, 47, 54, 73

[21] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, "Towards large-pose face frontalization in the wild," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3990–3999. 4, 5, 41, 46

[22] J. Zhao, L. Xiong, Y. Cheng, Y. Cheng, J. Li, L. Zhou, Y. Xu, J. Karlekar, S. Pranata, S. Shen *et al.*, "3d-aided deep pose-invariant face recognition." in *IJCAI*, vol. 2, no. 3, 2018, p. 11. 4

[23] J. Cao, Y. Hu, H. Zhang, R. He, and Z. Sun, "Learning a high fidelity pose invariant model for high-resolution face frontalization," *arXiv preprint arXiv:1806.08472*, 2018. 4, 5, 72

[24] E. Richardson, M. Sela, and R. Kimmel, "3d face reconstruction by learning from synthetic data," in *2016 fourth international conference on 3D vision (3DV)*. IEEE, 2016, pp. 460–469. 5, 59

[25] J. Booth, E. Antonakos, S. Ploumpis, G. Trigeorgis, Y. Panagakis, and S. Zafeiriou, "3d face morphable models" in-the-wild"," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 48–57. 5, 59

[26] E. Richardson, M. Sela, R. Or-El, and R. Kimmel, "Learning detailed face reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1259–1268. 6, 59, 63

[27] Y. Guo, J. Cai, B. Jiang, J. Zheng *et al.*, "Cnn-based real-time dense face reconstruction with inverse-rendered photo-realistic face images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 6, pp. 1294–1307, 2018. 6, 59, 63

[28] L. Tran and X. Liu, "Nonlinear 3d face morphable model," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7346–7355. 6, 59, 60, 63

[29] G.-H. Lee and S.-W. Lee, "Uncertainty-aware mesh decoder for high fidelity 3d face reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6100–6109. 6, 60, 61

[30] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, "Face alignment across large poses: A 3d solution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 146–155. 6, 54, 60, 63, 75

[31] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017. 6, 60, 61, 71, 94

[32] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410. 6, 34, 45, 60, 61, 71, 94, 99

[33] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=B1xsqj09Fm 6, 34, 45, 61

[34] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of gans for semantic face editing," in *CVPR*, 2020. 6, 61

[35] F. Ma, U. Ayaz, and S. Karaman, "Invertibility of convolutional generative networks from partial measurements," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper/2018/file/e0ae4561193dbf6e4cf7e8f4006948e3-Paper.pdf 6, 61

[36] ——, "Invertibility of convolutional generative networks from partial measurements," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 9651–9660. 6, 61

[37] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *European conference on computer vision*. Springer, 2016, pp. 597–613. 6, 61

[38] B. Gecer, S. Ploumpis, I. Kotsia, and S. Zafeiriou, "Ganfit: Generative adversarial network fitting for high fidelity 3d face reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1155–1164. 6, 61, 73

[39] M. Lee, W. Cho, M. Kim, D. Inouye, and N. Kwak, "Styleuv: Diverse and high-fidelity uv map generative model," *arXiv preprint arXiv:2011.12893*, 2020. 6, 61

[40] B. Gecer, J. Deng, and S. Zafeiriou, "Ostec: One-shot texture completion," *arXiv preprint arXiv:2012.15370*, 2020. 6, 61

[41] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009. 7, 81

[42] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in neural information processing systems*, 2012, pp. 341–349. 7, 82

[43] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544. 7, 34, 82, 83

[44] Y. Li, S. Liu, J. Yang, and M.-H. Yang, "Generative face completion," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3911–3919. 7, 82, 83

[45] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5505–5514. 7, 8, 82, 83, 84

[46] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 85–100. 7, 82, 84, 95

[47] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4471–4480. 8, 82, 84, 91, 95

[48] Y.-H. Lee and S.-H. Lai, "Byeglassesgan: Identity preserving eyeglasses removal for face images," in *European Conference on Computer Vision*. Springer, 2020, pp. 243–258. 8, 83, 84

[49] J. Cai, H. Han, J. Cui, J. Chen, L. Liu, and S. K. Zhou, "Semi-supervised natural face de-occlusion," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1044–1057, 2020. 8, 83, 84

[50] X. Yuan and I. K. Park, "Face de-occlusion using 3d morphable model and generative adversarial network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 10 062–10 071. 8, 83, 85, 95

[51] J. Dong, L. Zhang, H. Zhang, and W. Liu, "Occlusion-aware gan for face de-occlusion in the wild," in *2020 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2020, pp. 1–6. 8, 83, 84

[52] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013. 11, 12

[53] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011. 12

[54] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, vol. 11, no. 12, 2010. 12

[55] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012. 18

[56] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017. 24, 26, 45

[57] C. Villani, *Optimal transport: old and new*. Springer, 2009, vol. 338. 26

[58] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," *arXiv preprint arXiv:1704.00028*, 2017. 28

[59] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018. 30, 45, 65

[60] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *International conference on machine learning*. PMLR, 2019, pp. 7354–7363. 34, 45, 49, 84, 91

[61] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803. 34

[62] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0. 34, 84

[63] Y. Lei, W. Du, and Q. Hu, "Face sketch-to-photo transformation with multi-scale self-attention gan," *Neurocomputing*, vol. 396, pp. 13–23, 2020. 34, 84

[64] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008. 34

[65] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, pp. 2234–2242, 2016. 36

[66] S. Barratt and R. Sharma, "A note on the inception score," *arXiv preprint arXiv:1801.01973*, 2018. 37

[67] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research)," ., 2009. [Online]. Available: http://www.cs.toronto.edu/~kriz/cifar.html 37

[68] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017. 38

[69] D. Dowson and B. Landau, "The fréchet distance between multivariate normal distributions," *Journal of multivariate analysis*, vol. 12, no. 3, pp. 450–455, 1982. 38

[70] F. Liu, Q. Zhao, D. Zeng *et al.*, "Joint face alignment and 3d face reconstruction with application to face recognition," *IEEE transactions on pattern analysis and machine intelligence*, 2018. 41

[71] J. Zhao, L. Xiong, P. K. Jayashree, J. Li, F. Zhao, Z. Wang, P. S. Pranata, P. S. Shen, S. Yan, and J. Feng, "Dual-agent gans for photorealistic and identity preserving profile face synthesis," in *Advances in Neural Information Processing Systems*, 2017, pp. 66–76. 41, 46

[72] Y. Feng, F. Wu, X. Shao, Y. Wang, and X. Zhou, "Joint 3d face reconstruction and dense alignment with position map regression network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 534–551. 41, 46, 59, 67

[73] B. Gecer, B. Bhattarai, J. Kittler, and T.-K. Kim, "Semi-supervised adversarial learning to generate photorealistic face images of new identities from 3d morphable model," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 217–234. 41

[74] J. Cai, H. Hu, S. Shan, and X. Chen, "Fcsr-gan: End-to-end learning for joint face completion and super-resolution," in *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*. IEEE, 2019, pp. 1–8. 41, 60

[75] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 172–189. 41, 60

[76] A. Pumarola, A. Agudo, A. M. Martinez, A. Sanfeliu, and F. Moreno-Noguer, "Ganimation: Anatomically-aware facial animation from a single image," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 818–833. 41, 45, 60

[77] T. Portenier, Q. Hu, A. Szabo, S. A. Bigdeli, P. Favaro, and M. Zwicker, "Faceshop: Deep sketch-based face image editing," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 99, 2018. 41, 60

[78] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8789–8797. 41, 45, 60

[79] Y. Tian, X. Peng, L. Zhao, S. Zhang, and D. N. Metaxas, "Cr-gan: learning complete representations for multi-view generation," *arXiv preprint arXiv:1806.11191*, 2018. 42, 46, 56

[80] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241. 43, 52, 88

[81] "Combine side and front portrait shots to create optical illusion | photoshop cc," https://www.youtube.com/watch?v=KxdXaAxdKBQ, accessed: 2020-2-25. 43

[82] "Photoshop tutorial: How to make a bi-directional, optical illusion, photo portrait," https://www.youtube.com/watch?v=H8PpBInBEDM&t=29s, accessed: 2020-2-25. 43

[83] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232. 45

[84] Z. Zhang, Y. Song, and H. Qi, "Age progression/regression by conditional adversarial autoencoder," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5810–5818. 45

[85] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen, "Attgan: Facial attribute editing by only changing what you want," *IEEE Transactions on Image Processing*, 2019. 45

[86] H. De Vries, F. Strub, J. Mary, H. Larochelle, O. Pietquin, and A. C. Courville, "Modulating early visual processing by language," in *Advances in Neural Information Processing Systems*, 2017, pp. 6594–6604. 45

[87] V. Dumoulin, J. Shlens, and M. Kudlur, "A learned representation for artistic style," *arXiv preprint arXiv:1610.07629*, 2016. 45, 49

[88] T. Miyato and M. Koyama, "cgans with projection discriminator," *arXiv preprint arXiv:1802.05637*, 2018. 46, 52

[89] L. Tran and X. Liu, "On learning 3d face morphable model from in-the-wild images," *IEEE transactions on pattern analysis and machine intelligence*, 2019. 46

[90] Z. Zhang, X. Chen, B. Wang, G. Hu, W. Zuo, and E. R. Hancock, "Face frontalization using an appearance-flow-based convolutional neural network," *IEEE Transactions on Image Processing*, vol. 28, no. 5, pp. 2187–2199, 2018. 47

[91] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in neural information processing systems*, 2015, pp. 2017–2025. 49, 66

[92] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, 2016, pp. 565–571. 50, 88

[93] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European conference on computer vision*. Springer, 2016, pp. 694–711. 51, 90

[94] E. Zakharov, A. Shysheya, E. Burkov, and V. Lempitsky, "Few-shot adversarial learning of realistic neural talking head models," *arXiv preprint arXiv:1905.08233*, 2019. 51

[95] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *British Machine Vision Conference*, 2015. 51

[96] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5188–5196. 51, 67, 93

[97] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, 2018. 53, 57, 75

[98] J. H. Lim and J. C. Ye, "Geometric gan," *arXiv preprint arXiv:1705.02894*, 2017. 54

[99] W. Gao, B. Cao, S. Shan, X. Chen, D. Zhou, X. Zhang, and D. Zhao, "The cas-peal large-scale chinese face database and baseline evaluations," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 38, no. 1, pp. 149–161, 2007. 54

[100] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3730–3738. 54

[101] C. Sagonas, G. Tzimiropoulos, S. Zafeiriou, and M. Pantic, "300 faces in-the-wild challenge: The first facial landmark localization challenge," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013, pp. 397–403. 54

[102] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 55, 71, 94

[103] C. Ding and D. Tao, "Pose-invariant face recognition with homography-based normalization," *Pattern Recognition*, vol. 66, pp. 144–152, 2017. 57

[104] C. Xiong, X. Zhao, D. Tang, K. Jayashree, S. Yan, and T.-K. Kim, "Conditional convolutional neural network for modality-aware face recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3667–3675. 57

[105] J. Zhao, Y. Cheng, Y. Xu, L. Xiong, J. Li, F. Zhao, K. Jayashree, S. Pranata, S. Shen, J. Xing *et al.*, "Towards pose invariant face recognition in the wild," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2207–2216. 57

[106] J. Guo, X. Zhu, Y. Yang, F. Yang, Z. Lei, and S. Z. Li, "Towards fast, accurate and stable 3d dense face alignment," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 59

[107] X. Zhu, X. Liu, Z. Lei, and S. Z. Li, "Face alignment in full pose range: A 3d total solution," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 1, pp. 78–92, 2017. 59

[108] Z. Bai, Z. Cui, J. A. Rahim, X. Liu, and P. Tan, "Deep facial non-rigid multi-view stereo," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5850–5860. 59

[109] I. Masi, A. T. Tran, T. Hassner, G. Sahin, and G. Medioni, "Face-specific data augmentation for unconstrained face recognition," *International Journal of Computer Vision*, vol. 127, no. 6, pp. 642–667, 2019. 59

[110] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari, "Accelerating 3d deep learning with pytorch3d," *arXiv:2007.08501*, 2020. 64

[111] X. Yin, D. Huang, H. Yang, Z. Fu, Y. Wang, and L. Chen, "Pixel sampling for style preserving face pose editing," in *2020 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, 2020, pp. 1–10. 64

[112] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 65, 74, 88

[113] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125. 65

[114] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, "A 3d face model for pose and illumination invariant face recognition," in *Proceedings of the 6th IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS) for Security, Safety and Monitoring in Smart Environments*. Genova, Italy: IEEE, 2009. 68

[115] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *ACM SIGGRAPH 2003 Papers*, 2003, pp. 313–318. 68, 92

[116] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823. 71

[117] R. Huang, S. Zhang, T. Li, and R. He, "Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 72

[118] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014. 74

[119] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4690–4699. 74, 90, 96

[120] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2. Ieee, 2003, pp. 1398–1402. 75

[121] J.-J. Lv, X.-H. Shao, J.-S. Huang, X.-D. Zhou, and X. Zhou, "Data augmentation for face recognition," *Neurocomputing*, vol. 230, pp. 184–196, 2017. 81

[122] D. S. Trigueros, L. Meng, and M. Hartnett, "Enhancing convolutional neural networks for face recognition with occlusion maps and batch triplet loss," *Image and Vision Computing*, vol. 79, pp. 99–108, 2018. 81

[123] Y. Xia, B. Zhang, and F. Coenen, "Face occlusion detection based on multi-task convolution neural network," in *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*. IEEE, 2015, pp. 375–379. 81

[124] W. Wan and J. Chen, "Occlusion robust face recognition based on mask learning," in *2017 IEEE international conference on image processing (ICIP)*. IEEE, 2017, pp. 3795–3799. 81

[125] L. Song, D. Gong, Z. Li, C. Liu, and W. Liu, "Occlusion robust face recognition based on mask learning with pairwise differential siamese network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 773–782. 81

[126] H. Ben Fredj, S. Bouguezzi, and C. Souani, "Face recognition in unconstrained environment with cnn," *The Visual Computer*, vol. 37, no. 2, pp. 217–226, 2021. 81

[127] L. He, H. Li, Q. Zhang, Z. Sun, and Z. He, "Multiscale representation for partial face recognition under near infrared illumination," in *2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems (BTAS)*. IEEE, 2016, pp. 1–7. 81

[128] Y. Wu and Q. Ji, "Robust facial landmark detection under significant head poses and occlusion," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3658–3666. 81

[129] L. He, H. Li, Q. Zhang, and Z. Sun, "Dynamic feature learning for partial face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7054–7063. 81

[130] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991. 81

[131] C. Wu, C. Liu, H.-Y. Shum, Y.-Q. Xy, and Z. Zhang, "Automatic eyeglasses removal from face images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 3, pp. 322–336, 2004. 81

[132] J.-S. Park, Y. H. Oh, S. C. Ahn, and S.-W. Lee, "Glasses removal from facial image using recursive error compensation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 5, pp. 805–811, 2005. 81

[133] Z.-M. Wang and J.-H. Tao, "Reconstruction of partially occluded face by fast recursive pca," in *2007 International Conference on Computational Intelligence and Security Workshops (CISW 2007)*. IEEE, 2007, pp. 304–307. 81

[134] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006. 82

[135] F. Zhao, J. Feng, J. Zhao, W. Yang, and S. Yan, "Robust lstm-autoencoders for face de-occlusion in the wild," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 778–790, 2017. 83, 84

[136] B. Hu, Z. Zheng, P. Liu, W. Yang, and M. Ren, "Unsupervised eyeglasses removal in the wild," *IEEE Transactions on Cybernetics*, 2020. 83, 84

[137] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised attention-guided image to image translation," *arXiv preprint arXiv:1806.02311*, 2018. 84

[138] C. Zheng, T.-J. Cham, and J. Cai, "Pluralistic image completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1438–1447. 84

[139] Y. Yang and X. Guo, "Generative landmark guided face inpainting," in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*. Springer, 2020, pp. 14–26. 84, 95

[140] N. U. Din, K. Javed, S. Bae, and J. Yi, "A novel gan-based network for unmasking of masked face," *IEEE Access*, vol. 8, pp. 44 276–44 287, 2020. 84

[141] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 86, 94

[142] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 761–769. 88

[143] R. Ramamoorthi and P. Hanrahan, "An efficient representation for irradiance environment maps," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 497–500. 88

[144] ——, "A signal-processing framework for inverse rendering," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001, pp. 117–128. 88

[145] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 91

[146] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004. 92, 96

[147] C. Zheng, T.-J. Cham, and J. Cai, "Pluralistic free-from image completion," *International Journal of Computer Vision*, pp. 1–20, 2021. 95