A flexible and extensible approach to face extraction and swapping

Yiwen Li, Jiaying Huang*, Dazhi Ning
BNU-HKBU United International College (BNU-HKBU UIC), Zhuhai, Guangdong, China

ABSTRACT

The completion and detail retention of face replacement after face changing are difficult technical problems in face swapping. This paper proposes an efficient and flexible model for face swapping which can realize high-quality face swap and improve the quality of the outputs of the existing networks. The proposed face swapping network is completed by the AEI-Net. After training the AEI-Net (the AEI-Net consists of three subnets: identity encoder, multi-level attribute encoder and ADD generator), the generated image with face change effect can be obtained. In our experiment, we use the training data sets which are CelebA and FFHQ as other face swapping networks to train their model and our model, and use the same test data set to evaluate it. After training, it can recover the abnormal area in a self-supervised way. The results show that compared with the state-of-the-arts, our model achieves good performance in terms of the realism of the generated face and the degree of detail reduction.

Keywords: Face detection, face extraction, face swapping

1. INTRODUCTION

Face swapping is a technique method widely used in film and TV entertainment, and it promotes the emerging development of entertainment and cultural industries. It's easy to explain its ability: replace the face of a person with a specific target image of another person's face[1]. For example: creating virtual characters in movie production, video rendering, computer games and so on. At the same time, this technology also has many negative effects, such as producing fake news, influence tampering, etc. Therefore, various countries and regions have also introduced policies and regulations to limit the malicious abuse of this technology. At present, the more mature face exchange frameworks in the technical field including DeepFaceLab, FaceShifter, FaceSwap and so on. In our work, we concentrated on optimizing the efficiency of face swapping while ensuring high-precision face swapping. We aim to get more attractive face swapping results. Therefore, it is important to ensure that the swapped face is a mixture with both the posture and expression of the target face so that it can be seamlessly fitted to the targeting image.

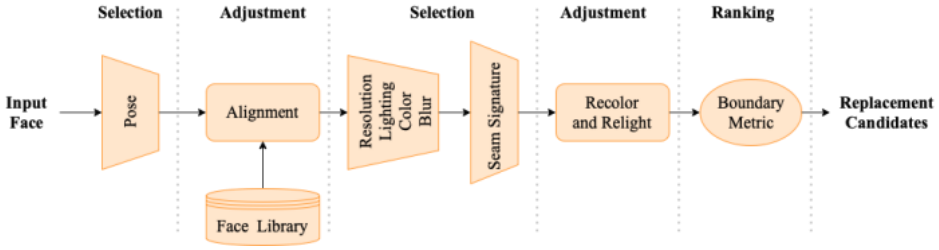

Figure 1. Main steps of automatic face replacement system.

Figure 1 shows the key steps of the auto face replacement system. The first stage is face recognition. This traditional method, which was replaced by DNN (deep neural network) trained with the huge data sets, based on artificially designed features.

Traditional methods based on the combination of artificial design feature analysis and linear discriminant analysis. It is hard to design the robust characteristics manually in a natural environment. It makes former researchers concentrates on particular

* jyhuangjiaying@163.com

Third International Conference on Computer Science and Communication Technology (ICCSCT 2022)
edited by Yingfa Lu, Changbo Cheng, Proc. of SPIE Vol. 12506, 125063H
 · 0277-786X · doi: 10.1117/12.2662657

methods for every type of change, for example, the methods that can deal with the details likes various ages, diverse lighting conditions, different postures, etc. Nowadays, the deep learning method replaces the traditional face recognition method based on CNN. The advantage of deep learning methods is obvious. For example, they can be trained with the huge sets to learn the best characteristics of the datasets. Large numbers of images of faces on the Web have allowed researchers to gather large datasets of faces that encompass a wide range of real-world situations. The face recognition methods trained with existing datasets have achieved a relatively high accuracy, the reason is that they could learn some of the robust features in the image. However, there are still room for improvements.

The following contributions are made to this paper:

● An efficient and flexible model for face swapping was completed. The Arcface was used to extract the identity vector of faces and the U-Net structure was used to extract facial attributions. ResNet-101 is used as the backbone of our model. The result faces can be got from features in the last ResBlk by transposed convolution.

● A better training strategy and a new normalization method are used in our experiments, which accelerate the process while training and develop the performance of our model. These modified parameters maximize the speed of model training. The switchable normalization in our model can let the model selected normalization method for each layer based on our data, or the weighted sum of the three normalization methods.

● More detailed facial features of the target faces can be capture and preserve. The latent space was expanded in our model, which can find the correlations between facial features and decouple them. More high-definition pictures can be got after face exchanging.

2. RELATED WORK

There are lots of methods to achieve face swapping and it has a long history. Some existing frameworks such as DeepFakes and FaceSwap could already get satisfying results. Our research is based on multiple computer fields: such as computer vision, pattern recognition etc. There are three main approaches related to our research: GAN-Based Approaches, 3D-Based Approaches and CNN-Based Approaches. Therefore, the following paragraphs will summarize and review the previous researches.

2.1 GAN-based approaches

GAN is usually equipped in face swapping frameworks for improving the authenticity of the data set. In Reference[2], Wang et al. use GAN to initialize the Deep Face Swap process. The principle of their face exchange algorithm is to train two auto encoder models separately. The two training models have the same encryptor and different decoders. They choose the training set data as real images, and the images output by the auto encoder are fake images, and then they were used as the training set for the CNN-based discriminator.

2.2 3D-based approaches

As the traditional face exchange technology requires the face of the target image and the face of the source image have similar poses and appearances, in order to improve the universality of face exchange technology, Reference[3] proposes a method based on a personalized 3D head model. The framework first constructs a 3D head model from the main view face image uploaded by the user by applying face alignment and feature point matching, which can eliminate the obstacles to face exchange caused by the different postures of the original image and the target image. Firstly, the process goes through the face alignment. Secondly, feature point matching was applied. In addition, in order to seamlessly integrate the synthesized human face into the image, the framework also applied color-transfer and multi-resolution spline technology. Also, in Reference[4], Blanz et al. use this method to generate the 3D conversion among faces with low similarity actions, but require user interaction. Thies et al. use 3DMM (3D Morphable Model) to capture head movements from RGB-D images, making static faces controllable. Without any manually annotations, our method[1] finds these occlusions in a self-supervised way.

2.3 CNN-based approaches

Convolutional neural network can divide the complex problems into several simple problems, then reduce the dimensions of a large number of parameters into a small number of parameters, and do processing. At the same time, it can effectively retain the features of the image. Therefore, in the process of face exchange, convolutional neural network can optimize training Time, and make the image present a high level of reality. In Reference[5], on the basis of Ulyanov et al. and Johnson et al., Korshunova et al. used all the images in the training set as the standard for identifying human face images and used them only in the training process. Therefore, their training results showed a more advanced image realism than previous studies, but in some details, such as the artifacts in the face swap image caused by the inaccurate points along the jaw line, the detection algorithm is based on the key points of the face[6]. Therefore, further improvements can be achieved.

3. METHODOLOGY

Our face changing method consists of AEI-Net. After appropriate training, it can be used for face changing models of any two face images. In fact, AEI-Net itself can get quite good result. The running speed of the whole model is very fast, and the generated face swapping results are of good quality.

3.1 FaceShifter (AET-Net)

As shown in Figure 2, AEI net consists of three sub-networks: Identity Encoder, Multi-level Attributes Encoder and Adaptive Attentional Denormalization Generator.

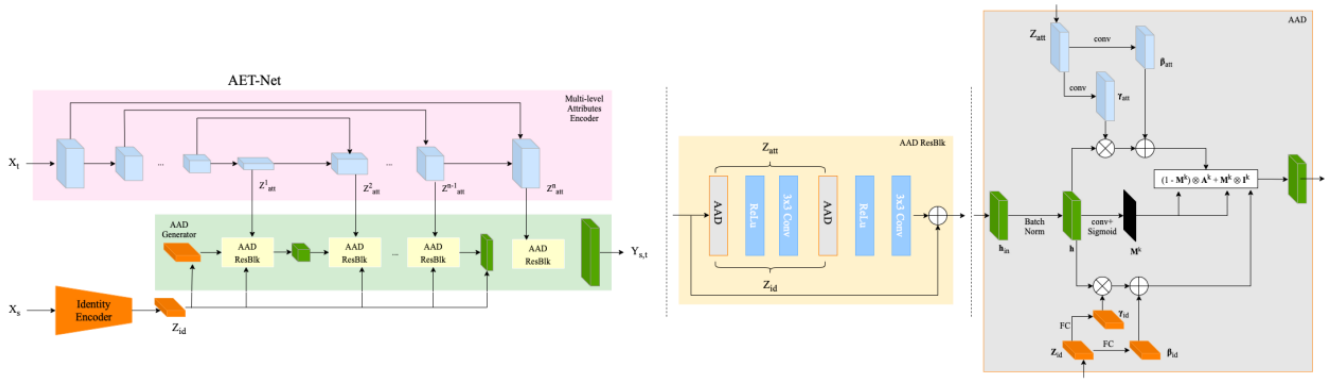

Figure 2. Three parts of the AET-Net[1].

Identity Encoder: An encoder that embeds z_{id} (X_s) was generated into the space describing the identity of the face in the image. The sub-network source image x_s is projected to low-dimensional features. Space is just a three-dimensional, we have z_i, as shown in Figure 2[7]. The characteristics of different people, such as the shape of their eyes, the distance between the eyes and the mouth, the curvature of the lips.

We use an encoder which shown in Figure 3 that has been collected and trained. We used a trained face recognition network. This satisfies our requirements because the network of a specific face must extract and related features.

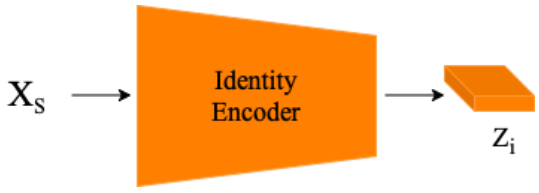

Figure 3. The identity encoder.

Multi-layer Attribute Encoder: An encoder that embeds X_t into a space that describes the attributes to be preserved when exchanging faces[8]. This sub-network encodes the target image X. It generates multiple vectors, and each vector describes the attributes of X_t with a different spatial resolution. Generally, there are 8 feature vectors, called z_a. The attributes refer to the facial structure in the target image, for example, the facial posture, contour, facial expression, hairstyle, skin color, background, scene lighting, etc. As shown in Figure 4, it is a ConvNet with a U-Net structure, where the output vector is only the feature map of each level in the decoding part[9].

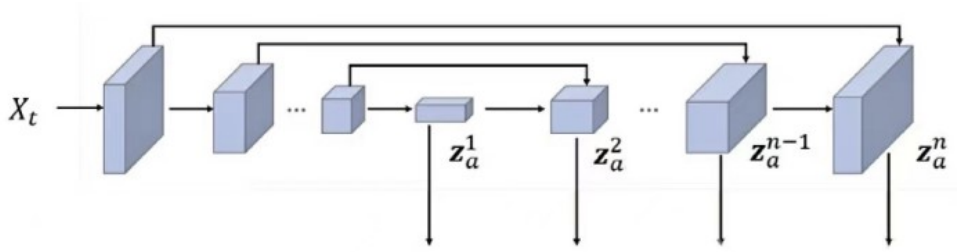

Figure 4. Multi-layer attribute encoder.

We add the target image X_t into a structure likes the U-Net decoder. zatt (Xt) is defined as:

$$z_{att}(Xt) = \{z_{att}^1(X_t), z_{att}^2(X_t), \cdots z_{att}^n(X_t)\} \quad (1)$$

Adaptive Attentional Denormalization Generator: This sub-network generates the final face change result. It combines the output of the first two subnets to improve the spatial resolution, thereby producing the final output of the AEI network[10]. It is achieved by superimposing a new block AAD Resblock. We divide it into 3 parts. As shown in Figure 5, from a high level, Part 1 displays how to edit the input feature map h_i to make it more like Xt in terms of attributes. Specifically, it outputs two tensors with the same size as h_in, one tensor contains the scaling value multiplied by each cell in h_in, and the other tensor contains the shift value. The input of the first part of the layer is one of the attribute vectors. Similarly, Part 2 shows how to edit the feature map h_i to make it more like Xs.

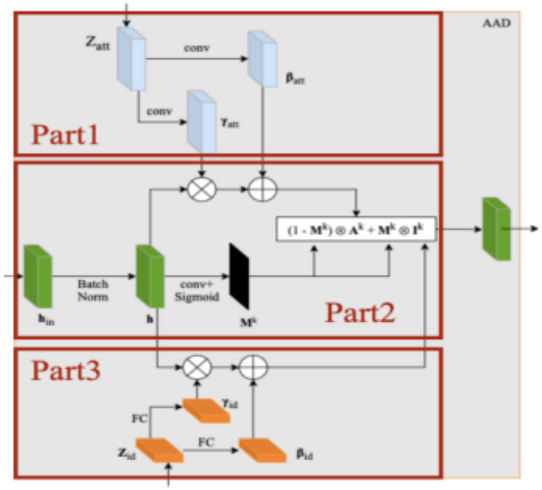

Figure 5. The architecture of the AAD layer.

The task of part 3 is to select the part (2 or 3) that we should focus on at each cell/pixel. For example, at the unit/pixel related to the mouth, the network will tell us to pay more attention to Part 2, because the mouth is more related to identity[11]. This is

empirically proved through an experiment shown in Figure 6. The figure shows the content learned in part 3 of the AAD layer. The image on the right shows the output of part 3 of the total AAD generator with out-of-sync count/spatial resolution. The bright area indicates that we should pay attention to the same cell (Part 2), and the black area indicates that we pay attention to the first part. Note that at high spatial resolution, we are mainly concerned with Part 1.

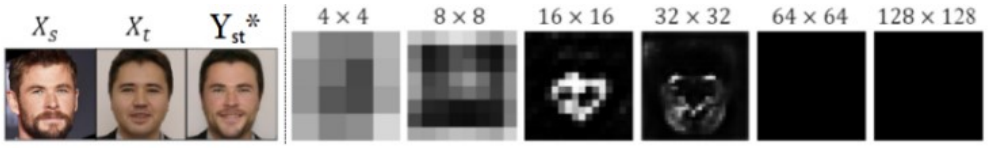

Figure 6. The content learned in Part 3 of the AAD layer.

In this way, the AAD generator will be able to construct the final image step by step, and in each step, it will determine the best way to enlarge the current feature map for a given identity and attribute encoding.

Before generation, we perform batch normalization:

$$\overline{h}^k = \frac{h_{in}^k - \mu^k}{\sigma^k} \tag{2}$$

In this equation, all the variables are either the means or the standard deviations of the channel-wise activation inside the mini-batch. The AAD generator is responsible for an integrated processing of the outputs of the first two modules to obtain $Y_{s,t}$, which is different from the feature concat that may cause image blur. For attributes embedding integration, we constructed an equation formulated as:

$$A^k = \gamma_{att}^k \otimes \overline{h}^k + \beta_{att}^k \tag{3}$$

There are two modulation parameters convolved from z_{katt} which have the same tensor dimensions. For embedding integration, we define a formula:

$$I^k = \gamma_{id}^k \otimes \overline{h}^k + \beta_{id}^k \tag{4}$$

The training process at this stage is essentially the process of image reconstruction, so the loss function includes the following four items: Adversarial Loss, ID Loss, Attribute Loss and Reconstruction Loss. The multi-scale discriminator is used to combat the loss. The ID loss extracts the ID feature and minimizes the difference. The attribute loss uses the multi-level attribute extraction network mentioned above to extract the feature, and the L2 loss is used to reconstruct the loss. This paper proposes an AAD layer, the output of the ADD layer can be obtained as a combination of A^k and I^k it is formulated as:

$$h_{out}^k = (1 - M^k) \otimes A^k + M^k \otimes I^k \tag{5}$$

Now, we form the AEI network, which can embed X_s & X_t and integrate them in a way that achieves our goals. We call the output of AEI Net as Y_{st*}. It is the core method in this paper.

Training Losses:

There are four training losses when training the AEI-Net.

(1) We want it to output a real human face, so we will have an adversarial loss, just like any adversarial network.

(2) We hope that the result generated face has the features of X_s. The only mathematical object we can represent identity is z_i. Therefore, this goal can be expressed by the following loss:

$$1 - \cos(z_i(Y_{st}^*), z_i(X_s)) \tag{6}$$

(3) We want the output to have attributes of X_t. The loss is:

$$0.5 * \sum_{k=1}^{n} \|z_{\alpha}^{k}(Y_{st}^{*}) - z_{\alpha}^{k}(X_t)\|_2^2 \tag{7}$$

(4) According to the view that the network should output X_t (if X_t and X_s are actually the same image), one more loss is added:

$$0.5 * \sum_{k=1}^{n} \|Y_{st}^{*} - X_t\|_2^2 \rightarrow if X_t = X_s \tag{8}$$

3.2 Improvement

Our face changing method consists of AEI-Net. After appropriate training, it can be used for face changing models of any two face images. In fact, AEI-Net itself can get quite good results. The running speed of the whole model is very fast, and the generated face swapping results are of good quality.

3.2.1 Optimizer. We also want to find a better Optimizer to continuously improve the neural network to minimize loss function. Therefore, we compare different Optimizers based on network structure and data set to find one that was more suitable for us.

The equation of Adam Optimizer is as follows:

$$w_{t+1} = w_t - \alpha m_t \tag{9}$$

where,

$$m_t = \beta m_{t-1} + (1 - \beta)\left[\frac{\delta L}{\delta w_t}\right] \tag{10}$$

Adammax Optimizer is:

$$x(t) = x(t-1) - \frac{\propto}{(1-\beta(t))} * \frac{m(t)}{\mu(t)} \tag{11}$$

AdamW Optimizer is:

$$x_t \leftarrow x_{t-1} - \alpha \frac{\beta_1 m_{t-1} + (\nabla f_t + w x_{t-1})}{\sqrt{v_t} + \epsilon} \tag{12}$$

After comparing various optimizers, we believe that Adam and the improved version of Adam's Optimizer will have a better effect on our model training. Among these optimizers, we exclude optimizers like SparseAdam that were targeted at sparse tensors rather than for our network. Also, optimizers such as AdamW, which converge faster but do not work with the Pytorch framework, are not considered.

Finally, we choose Adamax as our Optimizer. Like Adam, it is an algorithm combining Momentum algorithm and RMSProp algorithm, which not only uses Momentum to accumulate gradient, but also makes the convergence speed faster and the amplitude of fluctuation smaller and carries out deviation correction. But Adamax is a step up from Adam by adding the concept of a maximum learning rate. It provides a simpler range for the learning rate upper limit.

3.2.2 Normalization. In order to accelerate the convergence of the training process and the generalization ability of the model, we want to select a normalization method that is more suitable for our model.

In this part, we consider three methods of normalization and compare them through many tests. Here are four different types of normalization that we've done.

Batch Normalization: This type of normalization was used in the original code of FaceShifter we used on GitHub. However, as we mention before, the batch size of our model training is affected by GPU memory. The maximum value of batch size can only reach 16^{12}. However, for Batch Normalization, the best batch size is 32. Therefore, the use of Batch Normalization is not very appropriate for us. So we decide to use the normalization method that is unaffected by the batch size.

Group Normalization: Different from group normalization that applied to a batch, group normalization has nothing to do with batch size. In this method, channels are grouped, that is, the g-layer feature maps of single samples in batch are extracted to

calculate mean and variance together, which has nothing to do with batch size[13]. So, we also consider this type of normalization and take this method in our experiment.

Switchable Normalization: The above three test methods are single, we need to choose a suitable one. But in experiments, we find that this choice is often somewhat difficult. Therefore, we also try the Switchable Normalization approach, which unified Batch Normalization, Layer Normalization, and Instance Normalization. It allows the model to learn from the data the selected normalization methods for each layer, or the weighted sum of the three normalization methods.

It is assumed that the input of an implicit convolutional layer of a convolutional neural network can be represented as a four-dimension feature graph. Each dimension individually represents the mini-batch size, number, height and width of channels respectively[14]. The graph interpretation of SN is shown in Figure 7 and the function to calculate Switchable Normalization is as bellow:

$$\hat{h}_{ncij} = \gamma \frac{h_{ncij} - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta \tag{13}$$

SN has some advantages that other optimization methods haven't, which is suitable for our model.

● Robustness: Although the batch size Settings are different from side to side, the SN can hold its accuracy.

● Generality: Previous normalization methods rely on statistical information from different dimensions. We need to select different network structures for tasks. Delicate manual design and tedious experimental verification make it very difficult to select models in practical applications. SN is effective for a variety of network including CNNs and RNNs.

● Diversity: Switchable Normalization selects different operations for different layers of the neural network, which expands the boundary of the normalization results. The existing methods keep the same normalized operation for each layer in the whole network[15]. However, it is difficult to select different normalized operations for the network manually, on the other hand, the optimal performance may not be achieved by selecting a single normalized operation through experiments.

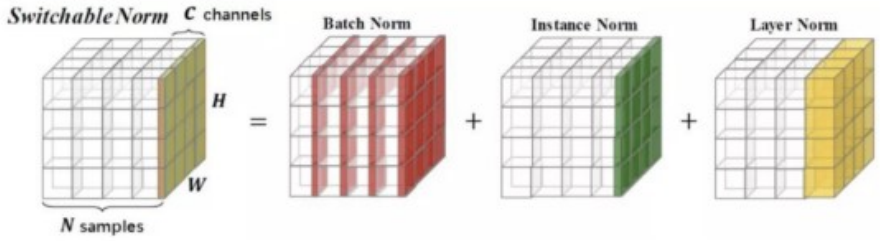

Figure 7. The graph interpretation of switchable normalization.

3.2.3 The Use of Latent Space. We believe that human facial features are diverse, and there is some correlation between these facial features, which means the attributes of human face are coupled. We believe that decoupling these facial features can preserve more facial feature details. Thus, we refer to the latent code and latent space mentioned in StyleGan and use them in our network.

To sort and generate data better, we need to represent the characteristics of the data. But there are various characteristics in the data, which may have some correlations in them[16]. Sometimes, the coupling is higher, and it is difficult to understand the correlation between them, which makes the learning efficiency is very low. So, it is necessary to find these hidden under the surface characteristics of deep relationship. After decoupling, we can find the hidden characteristics, which is just latent code. And the space made up of latent code is called latent space.

Thus, to show the features of the face and keep the details of the face better in the process of exchanging faces, we expand a latent space for our model. We project images in our dataset to latent space. In this process, we have found the matching latent vector for our image file. Then we use id loss function and attribute loss function on this latent space. Similar to the function

of mapping network in StyleGan, we decouple id and attribute of faces in our datasets to get a better performance on face exchanging.

4. EXPERIMENTS

We use FaceShifter to realize face exchange and get a good experiment result. This process includes preparing datasets, do the preprocessing, and training the models[17]. We complete Acknowledging Refinement Network, which is the main part of the FaceShifter model. After completing the AEI-Net, we have already got a good result for face swapping. Since there is no official open source code for Heuristic Error Acknowledging Refinement Network, and the present code for this part which developed by others doesn't have a good performance, we don't consider this part in our work.

4.1 Datasets

Here we choose two widely used face datasets Flickr-Faces-High-Quality (FFHQ) and CelebFaces Attributes (CelebA)[18]. And we randomly choose 80% of both datasets for training and the remaining 20% for validation.

FFHQ: It contains 1024 x 1024 resolution 70000 high-resolution face images, rich variety in age, race, and the image background and clear difference. It also has so many changes on face properties, different age, gender, race, color, face, face, hair, facial gestures, etc., capsule cover ordinary glasses, sunglasses, hats, hair accessories, scarves, and other face accessories.

CelebA: It consists of 202,599 face pictures of 10,177 celebrities, each of which is featured with face bounding box labeling box, 5 individual face feature point coordinates and 40 attribute markers.

4.2 Implementation

4.2.1. Data Preprocessing. Before the training process, we first do the preprocessing for both our training dataset and validation dataset. We build 64-point landmarks for each image and extracted the face. Then, using landmarks to align and then crop the face. The output is of size 256 × 256, which covering both the whole face and some background area.

Considering the hardware support, the size of our datasets, our model structure, and calculations in the training process, we lower the batch size to 16 to train the model successfully with the limit GPU memory. After adjusting the batch size, while we make full use of GPU resource, we maximize the speed of model training.

We also alter the number of works to 2, which depends on batch size and machine performance. After doing several experiments, we find that the data loader create 2 works at once is suitable for our training process. If it is set to over 2, memory overhead is too high for the school GPU to bear.

To find a suitable value for learning rate, we do several experiments. We find that when the learning rate of generator and discriminator were both set to 0.0001, the shock of loss was smaller, and loss decreased faster in our training process without sacrificing training speed. Thus, we set the value of learning rate to $1*10^{-4}$. Adjust batch size, number of work and learning rate

Considering the hardware support, the size of our datasets, our model structure, and calculations in the training process, we lower the batch size to 16 to train the model successfully with the limit GPU memory. After adjusting the batch size, while we make full use of GPU resource, we maximize the speed of model training.

Besides, we also alter the number of works to 2, which depends on batch size and machine performance. After doing several experiments, we find that the data loader create 2 works at once is suitable for our training process. If it is set to over 2, memory overhead is too high for the school GPU to bear.

To find a suitable value for learning rate, we do several experiments. We find that when the learning rate of generator and discriminator were both set to 0.0001, the shock of loss was smaller, and loss decreased faster in our training process without sacrificing training speed. Thus, we set the value of learning rate to $1*10^{-4}$.

Besides, to better detect the training process and evaluate the training effect of our model, we expand the visual content. We visualize all losses in the training, including ID Loss, Attribute Loss, reconstruction Loss, Validation Loss, Gan loss, the loss

of generator, and the loss of discriminator. For each image, we need to recognition the face/faces in it. In this part, we use the trained Arcface model in our network to extract the feature of faces and do the face recognition. Recognized faces will later be used for training the FaceShifter model.

4.2.2. Training Strategies for Face Swapping. We train and test our model under Pytorch framework. CUDA 10.2 and NVIDIA UNIX x86_64 are used in our experiments. According to the paper, in the process of the AEI-Net training, we use the same multi-scale discriminator[19]. The number of attribute embeddings in the AEI-Net is set to $n = 8$. The number of downsamples/upsamples is set to 8. In all the training process, we use ADAMAX with $\beta 1 = 0$, $\beta 2 = 0.999$ and set lr to 0.0004. The AEI-Net is trained 60 epochs with 400K steps, using 4 GPUs. Then, the batch size is set to 16, which is the best choice for our model training.

5. RESULT ANALYSIS AND COMPARISON

5.1 The results and comparisons between different normalization

In the experiments, we find that different type of normalization resulted in a large difference in training process and the result of face exchanging. We use Batch Normalization, Group Normalization, and Switchable Normalization to do the tests and comparison. The result images are shown as Figure 8.

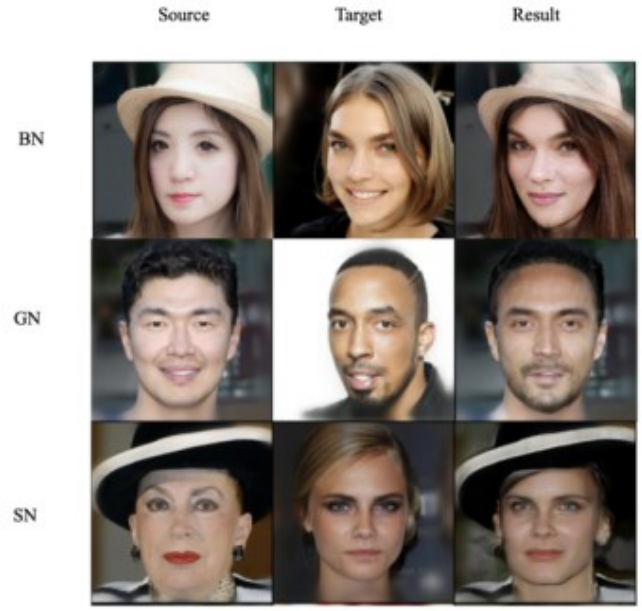

Figure 8. The result images for different methods.

After batch normalization of our model, more characteristic parts of the target face, such as eyebrow shape, facial shape, and eye appearance, are displayed on the source face. However, the faces in the result still retain a skin color close to the source face. Conversely, after group normalization, the swapped faces are more similar to the target face. Furthermore, the results using switchable normalization show that the skin color is similar to the target face color. But it also retains some of the more recognizable facial features, such as beards and eyes. For example, some detailed features, such as the eye shape of the target face, the color of the glasses, and the wrinkles around the eyes, have been reflected on the source face. Simultaneously, more bangs on the target face than the source face are also kept in the face swap.

We also compare the training process and results of the different normalization method. We hope to select an exploratory method that has the best training results. Therefore, we visualize and compare the training process. We believe that face

attributes are a very important point in face exchange. Because whether the attributes of target face can be captured and fully displayed on the source face will directly determine whether better face exchange results can be obtained. Figure 9 shows attribute losses for our model with different normalization methods. Compared with the other two methods, switchable normalization can make the attributes loss decline curve more stable without large fluctuations. At the same time, the value of attribute loss can be reduced to a lower level.

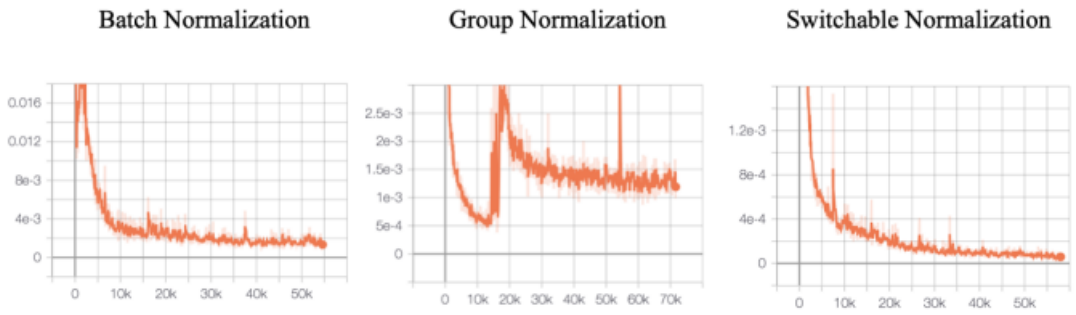

Figure 9. Attribute losses for three different methods.

In addition, we also compare other losses in the training process. As we can see from Table 1, using Switchable Normalization reduces attribute loss and ID loss for faces to a much lower level in training. Compared with other methods, reconstruction Loss can be improved greatly by using SN and validation loss can also be reduced to a minimum.

It is obvious that with using Switchable Normalization in our model, we can get a better performance in training process and have a better result for face exchanging. Thus, we apply the Switchable normalization in all the experiments.

Table 1. The comparison of different loss between different normalization methods.

Loss_Type	Batch_Norm	Group_Norma	Switchable_Norm
Attribute_Loss	1.3008e-3	1.918e-3	5.2872e-5
ID_Loss	0.06027	0.1314	0.06672
Loss_D	1.24	1.422	1.226
Loss_G	0.891	0.07898	1.147
Restruction_Loss	3.4817e-3	1.602e-3	0.01123
Validation_Loss	0.6491	0.9106	0.2748

5.2 The result of using latent space

As we mentioned before, in order to capture more characteristic of faces, we add a latent space for our model, which encodes more details about faces. After projecting the images in the latent space and decoupling, we can get the following result images in Figure 10.

It is shown that after adding the latent space, it seems that the characters of faces can be captured better. We can get more high-definition pictures after face exchanging. The details of faces are preserved more completely in the face exchange. Besides, we can get smoother result images. However, there are still some flaws in the processing of non-face details. For example, in

the first set of face-swapping images, the glasses of the target's face are left with only the lens outside the face. In the second set of images, the resulting face has some inappropriate shadows on the cheek.

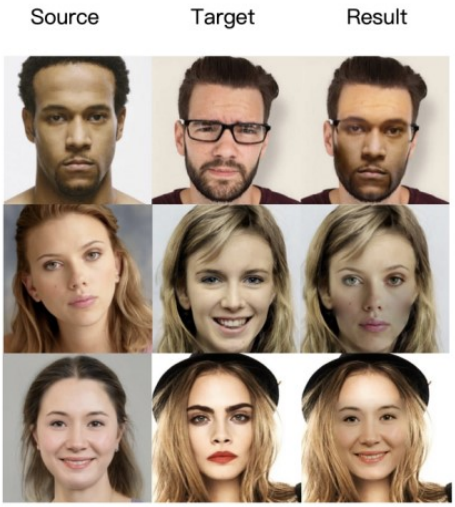

Figure 10. Three results for using latent space.

5.3 Comparison between present methods and ours

The experiments are done to compare our model with other models, includes DeepFakes[20], FaceSwap[21], Nirkin et al.[17], and IPGAN[19]. For each method, 100 images are generated with the same source and the corresponding target image. Then, we make a comparison from the following three metrics: pose error, expression error and ID retrieval.

We use different face recognition models to extract feature vectors and use cosine similarity to measure feature distance. Then, the pose estimator was used to estimate the pose of the head and the real 3D face model to retrieve the expression vector from the faces. We use L-2 distances of pose and expression vectors between the swapped and target face to represent pose and expression errors. The larger the ID retrieval, the better it performs. Also, the smaller the pose error and expression error, the better the model is. As the table shown, our model has an outstanding performance in ID retrieval. Pose error together with the expression error are also in a low level. Thus, our model has a huge advantage in retrieve id and facial features in face exchanging. The result is shown in Table 2.

Table 2. Comparison with other face exchange method.

Method	Id Retrieval	Pose	Expression
DeepFakes[20]	71.96	4.14	2.57
FaceSwap[21]	54.19	2.51	2.14
Nirkin et al.[17]	76.57	3.29	2.33
IPGAN[19]	82.41	4.04	2.50
Ours	**89.27**	**3.15**	2.11

5.4 The cause of the highlights & shortcomings of our results

After many experiments and optimization, we have been able to obtain excellent face exchange results. We can not only generate relatively high-definition face pictures, but also capture many face features and details, such as eyebrow shape, eye shape and skin texture, and transfer these features to source face.

However, our model still has some flaws, and some face exchange results are not very good. For example, since we only implemented AEI-Net, our model doesn't work very well with face occlusions. Some of the occlusions, like glasses, hair and other details might have been erased during the face exchange. For example, in Figure 11, Target Face has a few bangs on its forehead. In the result picture, we can see that the face has no bangs on its forehead or has the same bangs as Target Face. Instead, a partial shadow on the forehead, which is not what we want.

Figure 11. The Experiment result for target face has few bangs.

In addition, our treatment of hair color is flawed. When the hair color of target and source is very different, the hair color in the result picture may be confused. As we can see in Figure 12, source's hair is silver, and target's is black. The result is silvery white and black hair.

Figure 12. The experiment result when persons in target and source image.

We speculate that the reason for this problem is that our model can only better capture facial features (such as eyebrows, eyes, nose, mouth, etc.), but cannot better extract features beyond the facial range. At the same time, we believe that the lack of color sensitivity in our model is also a major reason for this result.

6. CONCLUSION

Overall, the task of our project is to implement a feasible and efficient face swapping framework named FaceShifter. We used the AEI-Net in order to achieve the final result. The framework shows high performance of generating the face images. In our project, we also proposed some methods to optimize the network to improve the accuracy and the performance of the model. The improvements include modify the parameters on our model, change a better optimizer, compare different normalization methods, and find a better method that fits our model. In order to get a better result that pay more attention to attributes of faces, we expanded a latent space for our model to decouple id and attribute of faces. Besides, we also discovered some problems and find out the practical solutions to fix it.

REFERENCES

[1] Wang, C. and Jang, J., "Deep face swap with GAN", Stanford CS 230 Projects, (2019).
[2] Lin, Y., Wang, S., Lin Q. and Tang, F., "Face swapping under large pose variations: A 3D model based approach," 2012 IEEE International Conference on Multimedia and Expo, (2012).
[3] Blanz, V., Scherbaum, K., Vetter, T. and Seidel, H. P., "Exchanging faces in images," Computer Graphics Forum 23, 669-676 (2004).
[4] Korshunova, I., Shi, W., Dambre, J. and Theis, L., "Fast face-swap using convolutional neural networks," 2017 IEEE International Conference on Computer Vision (ICCV), 3697-3705 (2017).
[5] Deng, J., Guo, J., Xue, N. and Zafeiriou, S., "Arcface: Additive angular margin loss for deep face recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 4690-4699 (2019).
[6] Kingma, D. P. and Ba, J., "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, (2014).
[7] Bambach, S., Lee, S., Crandall, D. J. and Yu, C., "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," Proceedings of the IEEE International Conference on Computer Vision, 1949-1957 (2015).
[8] Karras, T., Aila, T., Laine, S and Lehtinen, J., "Progressive growing of GANs for improved quality, stability, and variation," arXiv preprint arXiv:1710.10196, (2017).
[9] Kim, H., Carrido, P., Tewari, A., Xu, W., Thies, J., Niessner, M., Pe´rez, P., Richardt, C., Zollho¨fer, M. and Theobalt, C., "Deep video portraits," ACM Transactions on Graphics (TOG) 37(4), 163 (2018).
[10] Natsume, R., Yatagawa, T and Morishima, S., "Rsgan: Face swapping and editing using face and hair representation in latent spaces," arXiv preprint arXiv:1804.03447, (2018).
[11] Olszewski, K., Li, Z., Yang, C., Zhou, Y., Yu, R., Huang, Z., Xiang, S., Saito, S., Kohli, P. and Li, H., "Realistic dynamic facial textures from a single image using GANs," Proceedings of the IEEE International Conference on Computer Vision, 5429-5438 (2017).
[12] Nirkin, Y., Keller, Y. and Hassner, T., "FSGAN: Subject agnostic face swapping and reenactment," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), (2019).
[13] Ulyanov, D., Lebedev, V., Vedaldi, A. and Lempitsky, V., "Texture networks: Feed-forward synthesis of textures and stylized images," In International Conference on Machine Learning (ICML), (2016).
[14] Ruiz, N., Chong, E and Rehg, J. M., "Fine-grained head pose estimation without keypoints," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2074-2083 (2018).
[15] Chaudhuri, B., Vesdapunt, N. and Wang, B., "Joint face detection and facial motion retargeting for multiple faces," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 9719-9728 (2019).
[16] Nirkin, Y., Masi, I., Tran, A. T., Hassner, T. and Medioni, G., "On face segmentation, face swapping, and face perception," 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), 98-105 (2018).
[17] Ghosh, P., Uziel, R., Bolkart T. and Ranjan, A., "GIF: Generative interpretable faces," arXiv:2009.00149, (2020).
[18] Bao, J., Chen, D., Wen, F., Li, H and Hua, G., "Towards open-set identity preserving face synthesis," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), (2018).
[19] DeepFakes. https://github.com/ondyari/ FaceForensics/tree/master/dataset/ DeepFakes. Accessed: 2019-09-30.
[20] FaceSwap. https://github.com/ondyari/ FaceForensics/tree/master/dataset/ FaceSwapKowalski. Accessed: 2019-09-30.
[21] Li, L., Bao, J., Yang, H., Chen, D. and Wen, F., "FaceShifter: Towards high fidelity and occlusion aware face swapping", arXiv:1912.13457, (2020).