Accurate face swap using cycleGAN

Subtitle as needed (*paper subtitle*)

Qingyuan Liu[*,†]
Huazhong University of Science and Technology
line 2: name of organization, acronyms acceptable
Wuhan, Hubei, China
[*] u201911111@hust.edu.cn

Yuxuan Zhou[†]
Hebei University Of Technology
line 2: name of organization, acronyms acceptable
Tianjin, China
184969@stu.hebut.edu.cn

Shuai Bao[†]
Southwest Jiaotong University
line 2: name of organization, acronyms acceptable
Chengdu, Sichuan, China
sc18sb@leeds.ac.uk
[†]*These authors contributed equally.*

Abstract**—***Face swap is to transfer a face image of one person to another given person. It can generate false but very realistic images and video data. At present, numerous frameworks have been proposed, such as CNN, FCN, GAN. Recently, GAN-based methods are popular with satisfying performance. In this paper, we aim to improve the GAN-based face swap methods under the cycleGAN framework and achieve a more realistic face-swapping result. In this method, cycleGAN is applied to to face swap task, which mainly solves the problems of unavailable paired training images, and our model can be trained without pairing real images. More than 1,000 pictures in total have participated in the model training. The test needs to input a video and output a video of the facial changes in a few minutes. We collect Hillary and Trump videos data on the google website for testing and analysis, obtain a trained model, and accurately change the face of the video. Our method can handle face swap effectively with reasonable speed. The quality and speed of its generation are no less than that of other frameworks like GAN, which is more popular nowadays.*

Keywords-Face swap; CycleGAN; GAN; Unpaired image-to-image translation

I. INTRODUCTION *(HEADING 1)*

Face swapping is the task of transferring a face from a source to a target one, seamlessly replaces a face appearing in the target and produces a realistic result. In recent years, deep forgery technology is becoming a new driving force for the entertainment and cultural industries. Among them, face swap has become one of the hot topics. Although this technology is being condemned by most people, it can also provide us with many conveniences. For example, it can change the faces of characters in movies and provide customers with virtual fittings. At present, in the field of deep learning face swapping, the main technologies are the Generative adversarial network(GAN), convolutional neural network(CNN), and recurrent neural network(RNN). Korshunova et al. [1], for example, proposed a quick way to swap face with CNN. They use an affine transformation which aligns 68 facial key points from a given image to the reference key points, so as to align the image with a frontal-view reference face, and then match the 68 key points with the target face to achieve face-swapping. Nirkin et al. [3] used the detected facial landmarks to establish 3D pose and facial expression for a 3D face shape, and then used FCN to segment faces from background and occlusion. And cover this 3D mask to the target face's position. Kim et al. [4] recently used cGAN to achieve face-reenactment. They render a reconstructed 3D facial model using a classic graphic pipeline. Then the rendered image is processed by a generator network, which trained to map synthetic views of each subject to the photo-realistic images. With the continuous development of the GAN network, using the GAN framework for face swap is a mainstream research direction in recent years.

In our work, we hope to use cycleGAN for more realistic video face-swapping. We have changed the structure of the original GAN and auto-coder. For each person who swaps face, such as swapping face of A to B, we get an image B', and then we encode and decode the result B' again, this time we use decoder A to try to restore the result to A. And we got the B". The discriminator to distinguish the difference between B" and the original image A. Also, the model tries to minimize the loss between B" and A.

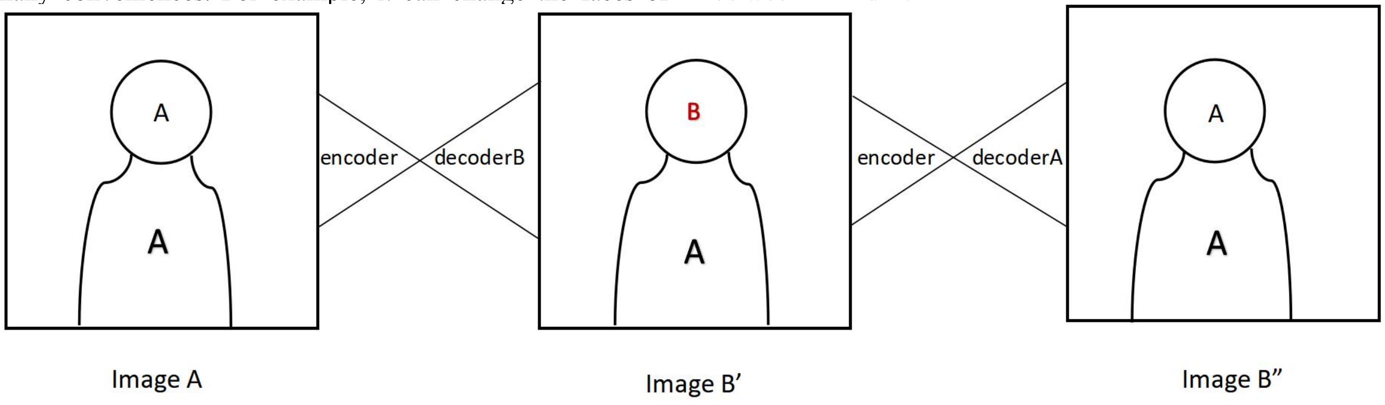


Figure 1. The structure of our algorithm

Through the cycle GAN network structure, we successfully realized face-swapping.

II. METHOD

A. GAN Overview

Using game theory, GAN can generate numerous training samples to handle few-shot learning tasks, including image reconstruction, super-resolution tasks, etc.

1) Principle

GAN includes two models, one is the generative model G (Generator), and the other is the discriminant model D (Discriminator). Their respective functions include:

G is mainly responsible for generating pictures, he receives a random noise z, generates pictures through the noise, and denote the generated pictures as G(z).

D is responsible for judging whether a picture is "real". Its input is x, x represents a picture, and the output D(x) represents the probability that x is a real picture. If it is 1, it means that the probability of a real picture is 100%, and the output is 0, which means it cannot be real. Picture (real examples come from the data set, fake examples come from the generative model).

In the training process, the goal of generative model G is to generate images that look similar to the original data to deceive model D. The goal of discriminating model D is to distinguish between the pictures generated by generative model G and the real pictures as much as possible. In this way, the generator tries to fool the discriminator, and the discriminator tries not to be fooled by the generator. The two models are optimized and trained alternately to improve each other. G and D form a dynamic "game", which is the basic idea of GAN.

2) Training

Purpose: Pass a random Gaussian noise z through a generating network G to obtain a generated data distribution pG(x; θ) similar to the real data distribution pdata(x), where θ is a network parameter, we want to find θ such that pG(x; θ) and pdata(x) are as close as possible.

First of all, when we think about the problem from the perspective of discriminating the network, the discriminator must be able to distinguish between the real data and the generated data. In mathematical expression, D(x)=1 and D(G(z)) = 0. Based on these two formulas, we construct the log loss functions of [positive class] (identifying x belongs to real data) and [negative class] (identifying G(z) belongs to generated data).

The loss function of the generating network G is log(1−D(G(z))) or −logD(G(z)).

The loss function of discriminant network D is −(logD(x)+log(1−D(G(z)))).

It can be seen from the formula that the image of the loss function is a graph similar to the y=log(x) function when x<0, y>0, when x=1, y=0, the generation network, and the discriminant network confrontation (The purpose of training) is to minimize their respective loss functions. The training purpose of generating network G is to hope that D(G(z)) is close to 1, which is the positive class so that the loss function log (1−D(G(z))) will be the smallest. The training purpose of the discriminant network is a 2-classification, the purpose is to make the discriminative probability D of the real data x approach 1, and the discriminative probability D(G(z)) of the generated data G(z) approach 0, which is Negative class.

When the judgment network encounters real data: Ex~pdata(x)[logD(x)], this expectation should be the largest, only when D(x)=1, that is to say, the judgment network recognizes that the data is true.

When the discriminant network encounters generated data: Ez~Pz(z)[log(1−D(G(z)))], because 0<probability<1, and the logarithm of x<1 is negative, in order to get the maximum value of this mathematical expectation, you need to set D(G(z)) = 0, D(G(z)) = 0, which means that the discriminator found that the generated data G(z) is false.

To sum up these two concepts, the objective function of discriminating network maximization is:

$$E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim P_z(z)}[\log(1-D(G(z)))]$$

The above formula is to find the optimal discriminator D*G for a given generator G, that is, to discriminate the maximum value of the network, so a value function can be defined as follows

$$V(G, D)=E_{x\sim p_{data}(x)}[\log D(x)]+E_{z\sim P_z(z)}[\log(1-D(G(z)))]$$

So we get a problem, for the discriminant network D, the expected objective function (discrimination formula V(D, G)) is maximized, while the generating network is expected to be minimized (discrimination formula V(D,G), then the objective function Is it to maximize or minimize? In fact, the entire training process is an iterative process, which can be understood separately. Given G, first maximize V(D, G) and take D*G, then fix D, and Minimize V(D,G) to get G*D. Given G, maximize V(D, G) evaluates the difference or distance between real data and generated data.

When we find the optimal D*G, namely D=D*G, we in turn substitute D=D*G into the above formula to find the optimal (minimum) G, namely G*D. Finally, we can express the optimization problem as: G*D=argmaxGV(G, D*G).

3) Loss

GANs are different from some other models. The GANs model needs to train two optimization algorithms at the same time. Define optimizers for the discriminator and generator respectively. One minimizes the loss for the discriminator and the other minimizes the loss for the generator. Finally get Loss = d_Loss + g_Loss.

For discriminator, the loss function is equal to the sum of the loss of the real picture and the generated picture, that is: d_loss = d_loss_real + d_loss_fake, Loss is calculated by cross entropy, the function can be expressed by Tensorflw: tf.nn.sigmoid_cross_entropy_with_logits(logits=logits , labels=labels).

We hope that when the discriminator calculates the loss d_loss_real generated by the real data, it outputs 1, and when calculating the generated data d_loss_fake, we want the discriminator to output 0.

Therefore, when calculating the loss of real data, you can set all labels to 1 because they are real. You can also set labels to 0.9 to enhance the generalization ability of the discriminator.

When calculating the loss of the generated data d_loss_fake, we set all labels to 0.

G_loss calculation method:

Finally, the generator's loss is calculated by the logits (ie d_logits_fake) of the generated data, but now all labels are set to 1 (that is, we want the generator to output 1). In this way, through training, the generator tries to'cheat' the discriminator.

B. Cycle GAN

1) Application

CycleGAN mainly solves the problems that are the training model to create images without paired real images. Such as image style conversion, we have the original input image and we want to change the image to another style. however, we don't have the target image. To make style conversion based on retaining original features, we can use cycleGAN.

2) Method

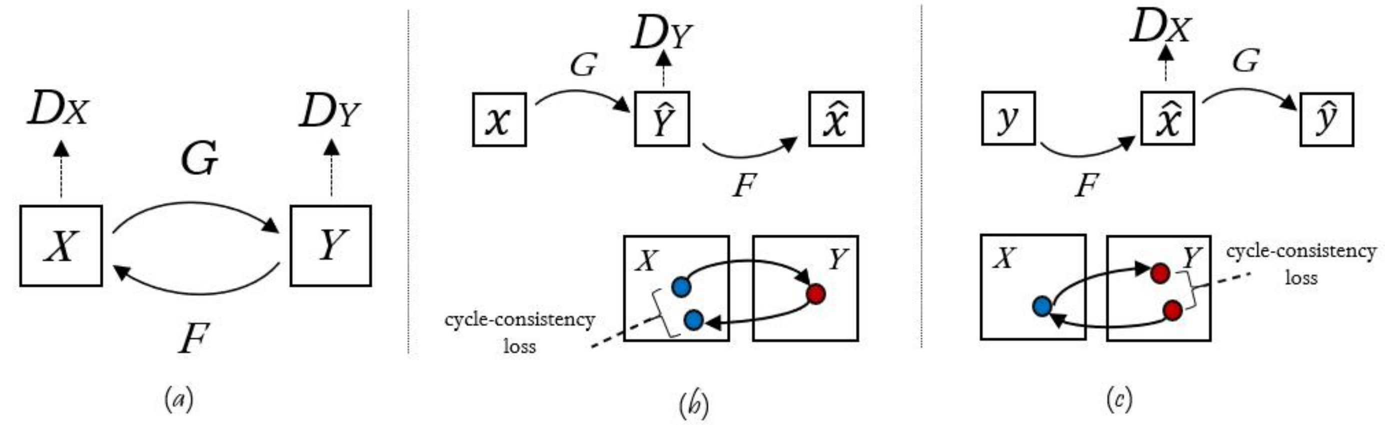


Figure 2. it shows the structure of cycleGAN [5]

Cycle GAN uses two normal generative adversarial networks, the first network A translates images from domain-X to domain-Y, and the second one reverses the process which translates back from domain-Y to domain-X. Here domain-X represents images in which we want to make the style conversion, but it has no paired target images. Domain-Y represents the images that have a certain style, this style we want to migrate to the input image X.

The first process translates images into G(x) with a certain purpose (like image style conversion). However, if we train generator Gx->y with y, we will have a big problem because the generator may turn all x into the same picture in y. For this single GAN, it really makes sense because the images generated by Gx->y are just one of the real samples in domain-Y. But this is not what we want. We want to get an image with original features, we want the model just to make a single change to the picture, such as image style conversion.

Therefore, in order to retain the original features of x, we add another generative adversarial network B. Through the network B, we translate G(x) back to our input x as F(G(x)). Network B solved the problem correctly. because if all input x is translated to a single image in domain-Y, that means the second generator Fy->x only has one input image but it has to translate this input G(x) to different x in domain-X.It is impossible and the discriminator can easy to tell the difference. In order to narrow this difference, the generator Gx->y will retain the original features of the original input.

3) Loss function

Synchronously, the discriminator A will determine whether the image G(x) belongs to domain-Y. For the first function G:X->Y. We have the loss function:

L(a)= $\log D_A(y) + \log(1 - D_A(G(x))$.

Here G tries to generate images G(x) to look similar to the images in domain-Y, DA will try to distinguish G(x) from y.G aims to minimize the loss while tries to maximize it. The second GAN is the same:

minFmaxL(b)=log(x)+log(1-(F(y)).

With a large enough capacity, the network can map the same set of input images to any random arrangement of images in the target domain, in this way any of the learned mappings can induce an output distribution that matches the target distribution. Thus, adversarial losses cannot guarantee that the learned function can map an individual input xi to the desired output Yi alone. To further avoid the wrong possible mapping functions, we argue that the learned mapping functions should be cycle-consistent. That means the image translation cycle should bring x back to the original input image: x->G(x)->F(G(x)) $\approx$ x. For y in

Identify applicable sponsor/s here. *(sponsors)*

domain-Y, the cycle also should satisfy y->F(y)->G(F(y)) ≈ y. We incentivize this behavior using a cycle consistency loss:

$$\mathcal{L}_{\text{cyc}}(G,F) = \mathbb{E}_{x\sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] + \mathbb{E}_{y\sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].$$

The full objective is:

$$L(G,F,D_A,D_B) = L_{GAN}(G,D_A,X,Y) + L_{GAN}(F,D_B,Y,X) + \lambda L_{cyc}(G,F)$$

Where λ controls the relative importance of the two objectives.

C. Train

1) Algorithm structure

For the traditional GAN network face swapping, it is achieved through an auto-encoder. First, we use the same encoder to reduce the dimensionality of the photos of the two people A and B that we want to swap faces. Then we use different decoder A/B to decode the result of the encoder. We need to train the respective decoder to produce results that are as similar as possible to the input face of the encoder. In the end, by using the decoder A/B corresponding to the character A/B, we can swap the face of the input photo. The whole auto-coder formed a generator, which needs to minimize the difference between the image after the face swap and the original face.

This paper uses cycleGAN to change the face in order to achieve a more realistic faceswapresult. we change the structure of the original GAN network. For each person who swaps face, such as swapping face of A to B, we get an image B', and then we encode and decode the result B' again, this time we use decoder A to try to restore the result back to A. And we get the B''. The discriminator will try to distinguish the difference between B'' and the original image A like cycleGAN. Also, this difference is cycle consistency loss.

Cycle loss:

loss_GA += 10 * cyclic_loss(A)

loss_GB += 10 * cyclic_loss(B)

Cyclic_loss(A) = $\sum 0.1 * \| GA(GB(A)) - A \|_2$

Adversarial loss: Adv_loss(A) = $\| D(G(A)) - D(A) \|_2$

2) Train detail

We use our own computer for training, the configuration of the computer is with I9-9900K CPU and Nvidia RTX 2080ti.

We used the photos we collected for training. A total of about 1,000 photos participated in the model training, including for 483 Trump and 507 for Hillary. The training image pixels are 256×256.The model supports 64x64, 128x128, and 256x256 outupt resolutions.

The code using MTCNN [6] and Kalman filter in video conversion in face detection and alignment:

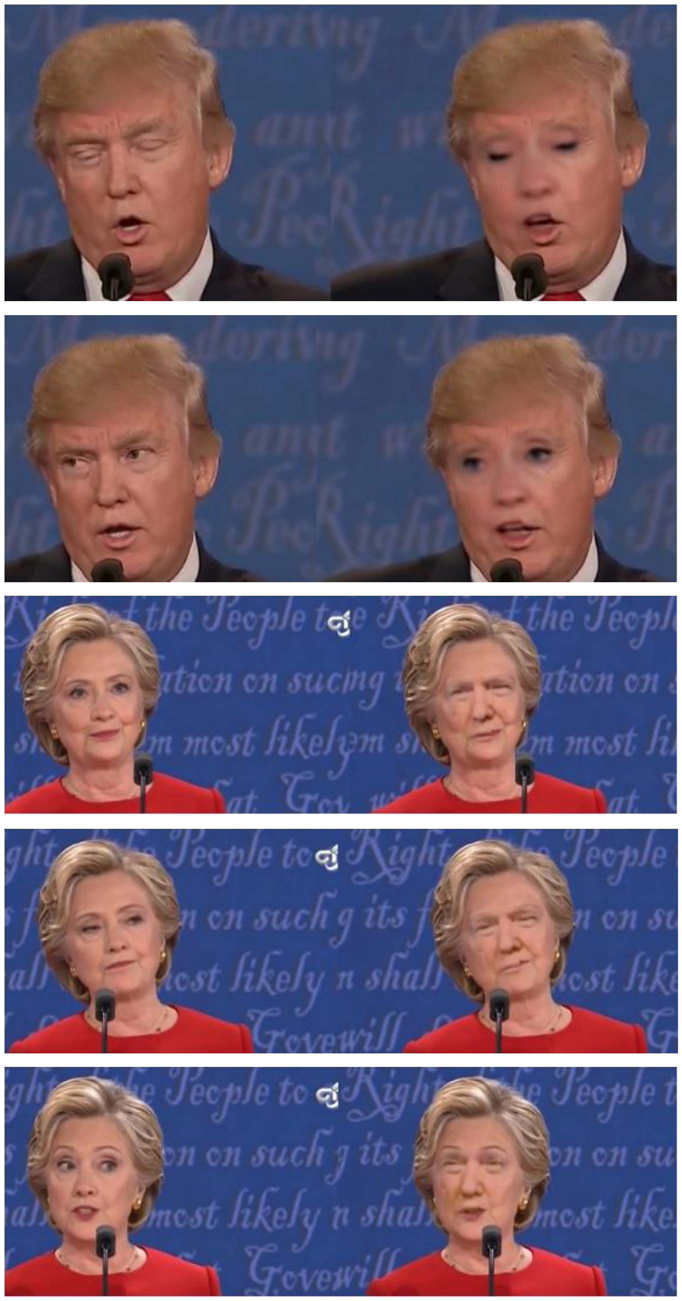
MTCNN is introduced for more stable detections and reliable face detection;

Kalman filter smoothen the bounding box positions over frames and eliminate jitter on the swapped face.

III. RESULTS AND DISCUSSION

A. Test data

We test our model using Trump and Hillary’s videos [7], which are open-sourced1. The result of our method can be shown in Figure 1.



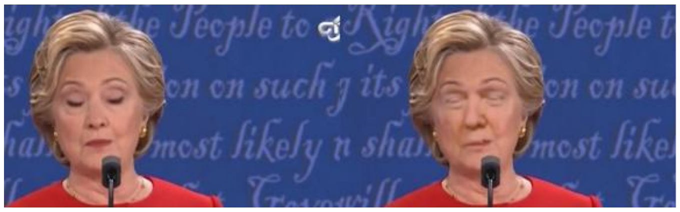


Figure 3. The visualization results of our method

Generally speaking, the effect of face change is ideal. The person who changes face can be clearly seen, and at the same time, the facial expression and poses of the person can be basically recognized. For example, the color, size and orientation of the eyes of the characters change obviously. We can generate very crisp videos. Because of the fluctuation of computer computing performance, it has an impact on the speed of generating video. Therefore, after many experiments, we take the average value, each frame needs 0.11s processing time under normal circumstances.

But through the results, we can see that there is an obvious defect in the face change result, which is the face recognition problem. Some faces have not been replaced. Our research found that the main problem is the inaccurate face detection. The impact of MTCNN's detection results, so MTCNN has a certain error for people's profile. Because of this, we found that the face detection step has a significant impact on the face change effect, and we will also use other face detection algorithms to test our face change results later. Another problem is that the face transition is not smooth after the character changes face. We think we need to increase the number of training sets, including different lighting, different environment, and posture expression.

IV. CONCLUSION

In this paper, we propose a face swap method based on CycleGAN. Our approach applies the cycleGAN face swap task to generate images when there are no paired real images in the training model. In the end, on the training scale of the process, more than 1, 000 images of Hillary Clinton and Donald Trump are used to train the models. As for the test process, the video input of Donald Trump and Hillary Clinton will be required and the output of the videos of face swap will be completed within a certain period of time. The purpose of testing and analyzing the video data of Hillary Clinton and Donald Trump from online resources is to cultivate a qualified model to achieve the face exchange technique with the required accuracy. Although our model can process face swap with appropriate accuracy and reasonable speed, which is nothing less than the more popular cycleGAN framework, the quality of generated face exchange does not reach the expected goal, that is, some edge faces exchange fails. Because MTCNN has the certain errors on people ’ s edge faces, it will cause influences toward the result of MTCNN. When the face is detected, the error of detecting the side face will be more than that of detecting the front face. As a result, the face detection errors leads to a certain part of human faces unchanged. Our method can effectively deal with face swap tasks with reasonable speed. It can be applied to lots of fields, such as entertainment and cultural industries to improve their accuracy.

REFERENCES

[1] Korshunova, Iryna, et al. "Fast face-swap using convolutional neural networks." Proceedings of the IEEE international conference on computer vision. 2017.

[2] Thies, Justus, et al. "Face2face: Real-time face capture and reenactment of rgb videos." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[3] Nirkin, Yuval, et al. "On face segmentation, face swapping, and face perception." 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). IEEE, 2018.

[4] Kim, Hyeongwoo, et al. "Deep video portraits." ACM Transactions on Graphics (TOG) 37.4 (2018): 1-14.

[5] Zhu, Jun-Yan, et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks." Proceedings of the IEEE international conference on computer vision. 2017.

[6] Zhang, Kaipeng, et al. "Joint face detection and alignment using multitask cascaded convolutional networks." IEEE Signal Processing Letters 23.10 (2016): 1499-1503.

[7] Information from https://www.youtube.com/watch?v=3M1QBzdpu4Y&t=434s