Face Swap using Autoencoders & image-to-image Translation Techniques

Yash Sanjeev
Electrical Department
IIT Bombay
Mumbai, India
180070068@iitb.ac.in

Shubham Seth
Electrical Department
IIT Bombay
Mumbai, India
18B090009@iitb.ac.in

Rohan Gupta
Electrical Department
IIT Bombay
Mumbai, India
180010048@iitb.ac.in

***Abstract*—DeepFakes is a popular image synthesis technique based on artificial intelligence. In this paper, we only consider the problem of transferring human faces, with different architectures & models and better ways of training them. FaceForensic++ is the main dataset used, albeit with a lot of perturbations and preprocessing. We have used a different training approach for CycleGANs and Autoencoders, which aims at making the training more stable.**

I. Introduction

Image-to-image translation has been researched for a long time by scientists from fields of computer vision, computational photography, image processing and so on. It has a wide range of applications ranging from entertainment to design-assistance. the goal is to learn the mapping between an input image and an output image using a training set of aligned image pairs. However, for many tasks, paired training data will not be available. In these cases, a popular approach for learning is to translate an image from a source domain X to a target domain Y in the absence of paired examples. Our goal is to learn a mapping $\mathcal{G} : X \rightarrow Y$ such that the distribution of images from $\mathcal{G}(X)$ is indistinguishable from the distribution Y using an adversarial loss. Because this mapping is highly under-constrained, we couple it with an inverse mapping $\mathcal{F} : Y \rightarrow X$ and introduce a cycle consistency loss to enforce $\mathcal{F}(\mathcal{G}(X)) \approx X$ (and vice versa).

In this particular report, we consider the problem of transferring the face of a selected individual over another fixed individual without changing the facial expression or context of the image. Thus, X and Y translate to the space of different facial expressions and contortions. Our primary objective is to find the aforementioned mappings $\mathcal{F}$ & $\mathcal{G}$ between these spaces. These mappings are learnt by formulating a loss function and letting the model backpropagate to a (preferably global) minima. This problem can be approached by two relatively modern methods - **CycleGANs** and **Autoencoders**.

CycleGANs involve the simultaneous training of two Generative Adversarial Networks (GANs). One of these generators focus on generating outputs from the distribution of X given an input from Y, while the other one generates Y from an input from X. Similarly the two discriminators are used to distinguish images from the actual distributions X and Y respectively from the geenrated images. Thus these generators and discriminators have opposing objectives, which helps them in highlighting each other's shortcomings and becoming better simultaneously. However during actual training one of them starts dominating the other resulting in diminishing gradients. In CycleGAN, where we train two such models, the problem amplifies considerably and prevents us from reaching stable configurations. Even more subtle is the problem of mode collapse where either of the learned functions $\hat{\mathcal{F}}$ and $\hat{\mathcal{G}}$ end up mapping to a mode of the original distributions X and Y, and hit a useless local minima. To escape these undesirable outcomes, we outline a less well-known approach (described in subsequent sections) of training these GANs that is apparently less prone to these issues.

Autoencoders have a Siamese Network type of architecture for DeepFake realisation, where we train a common encoder but use separate decoders to decode images from spaces X and Y. At test time, an image from X is encoded and then decoded by the decoder for Y or vice versa. The main idea behind this architecture is that encoding the image may help us obtain a rich feature vector which can be used to reconstruct images of either of the people. Thus, an encoder may be able to overlay the face of the person whose space it was trained on, over the original image background. This architecture also has various dangers associated with its training. It makes an inherent assumption that all faces have a common feature vector space, hence a common encoder can be used, which, if not true, may result in poor training. Moreover, the features extracted may have certain recurring portions of the background, as well as other body parts not related to just the face. This may result in unnecessary artifacts which only get more profound with increased training. However, with careful hyperparameter tuning and a lot of training epochs did give us a satisfactory result.

II. Related Work

Generative Adversarial Networks (GANs) [3] have achieved tremendous improvements in various fields like (conditional) image generation, image editing and representation learning. The fundamental concept behind GANs is th eiddea of an *adversarial loss* that forces generated images to be as close to real images as theoretically possible. We adopt this idea to learn our mappings $\mathcal{F}$ & $\mathcal{G}$ so that the translated images

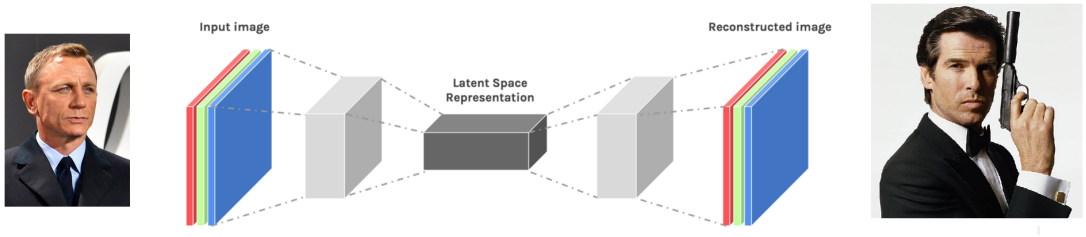

Fig. 1: The Autoencoder Architecture

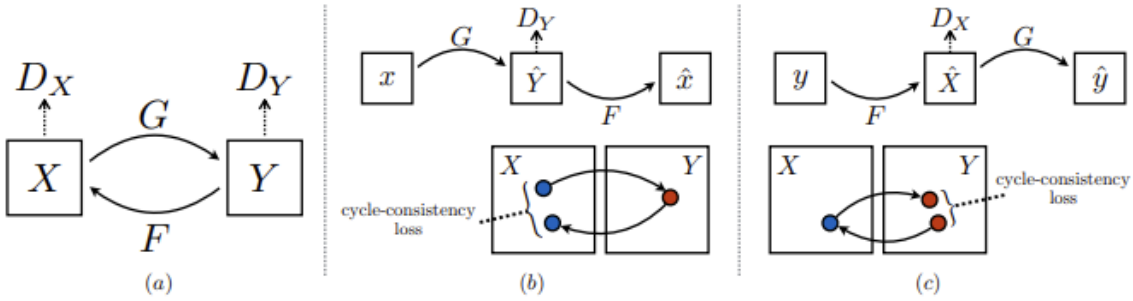

Fig. 2: CycleGAN fundamental principle

are as close to the real images from the respective distributions as possible.

Unpaired Image-to-Image Translation [4] have devised a general cyclic formulation that does not rely on task-specific, predefined similarity function between input and output, nor is it assumed that input and output lie in the same low-dimensional embedding space, thus escaping the weakness of an autoencoder. This general formulation was tailored specific to our needs and used in the present report.

Cycle Consistency [6] have used a cyclic objective to make the two mappings consistent, drawing inspiration from dual learning in machine translation. However, this technique has been a standard trick in various other fields for decades. Enforcing simple forward-backward consistency in visual tracking, verifying and improving translations via *back translation & reconciliation* are all examples of this technique being utilised in other application fields. In our particular CycleGAN example, we have used to types of cyclic losses, explained in the latter sections. We have also used a certain identity loss to stabilise our training.

III. Dataset

The datasets used are subsets of two popular datasets, namely, Face Forensic++ [1] and the custom dataset by OldPan [2].

Face forensic is a video database. So, different frames of a video were extracted with a custom script and were used as training samples. This was a dataset of 1000 images.

The second dataset was a custom dataset we created by picking images of two public figures, PewDiePie and Elon Musk from the internet. This dataset contained only 20 images of each person, and due to the small datset we carefully picked images of them in a similar pose, which was their faces looking straight ahead.

Data Preprocessing and Augmentation The images fed to both the CycleGAN and Auto-Encoder were flipped along the horizontal axis with a probability of 1/2, after preprocessing them to output square images.

IV. Auto Encoders

A. Formulation

Auto-encoders are a kind of neural network that try to learn a latent space mapping of the original input, through a series of two networks, the encoder and the decoder. In this implementation we try to see how we can use this simple network architecture to construct a face swapping algorithm, and what are the drawbacks of using this algorithm for swapping faces. For this implementation, we used [2] as our starter code.

B. Implementation

Network Architecture The architecture we used was composed of a pair of auto-encoders. One was a reference to reference auto-encoder, the other was a target to target auto-encoder. The idea was to take a reference image, and run it through the reference encoder, and then run the resulting vector through the target decoder.

The idea behind this process was, that if both the reference and target auto-encoders learnt how to extract the features into a latent space and then learn to re-construct the person's face from just these particular features, we would be able to map the features extracted from the reference's image, such as the position of eye, the mouth, etc. and create an image of the target with those features.

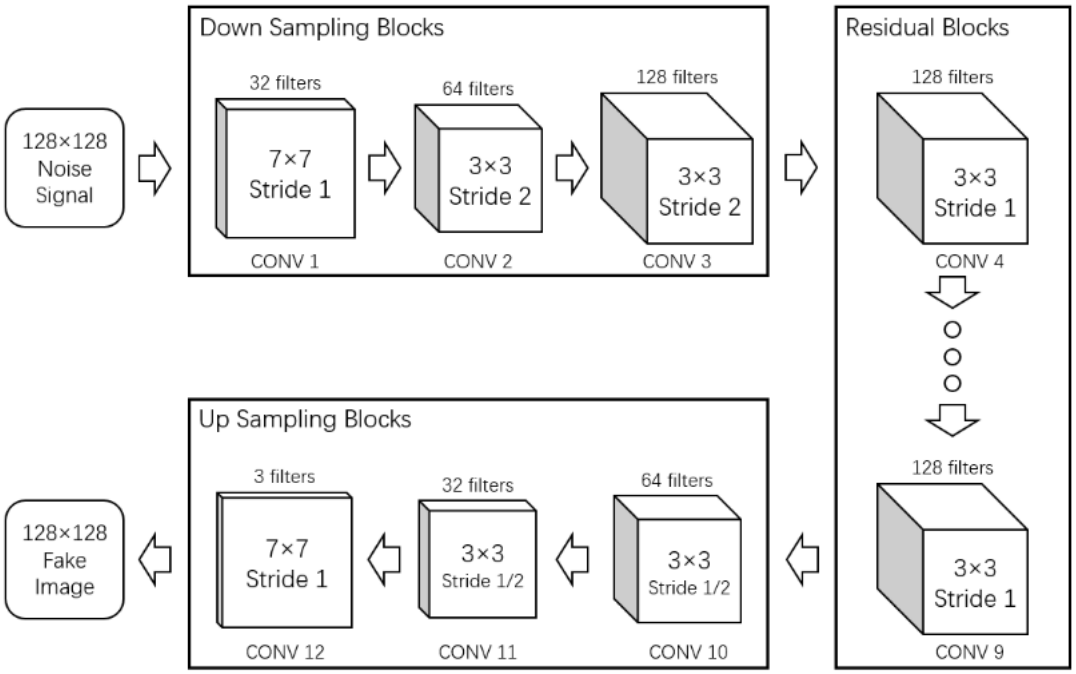

Fig. 3: Generator Architecture for CycleGAN

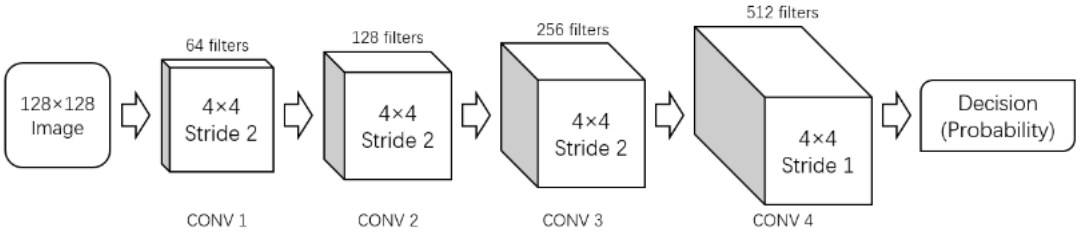

Fig. 4: Discriminator Architecture for CycleGAN

Training Details Due to the nature of the model as a pure auto-encoder based model, we only use mean absolute error as our loss function. First, we randomly apply a transform to the original image, and then pass through the auto-encoder and compare with the original image, so as to promote learning features from the images.

Since the dataset was comparatively small, we specifically picked images with similar looking orientations of the reference and target. We trained for an approximate of 50,000 epochs

Drawbacks As you can see, while the model is producing images from which you can see some likeness to our target being achieved, these images look clearly fake and are not believable. This, we can attribute to two reasons:

The dataset being too small Since we trained this on a relatively small dataset, at some point the auto-encoders stop learning the actual features and start overfitting. This results in some of the images produced after the face-swap to have a target image's background, and not the reference image's background.

The lack of a discriminative component This is the main reason why the training stagnates after a while in our model. The lack of a discriminative component which separates between images looking fake or real means that the model only tries to minimize the loss function from the generated images, and not try to produce a realistic image. A proper discriminative component would also ensure that the images produced by us are significantly less blurry as compared to now.

V. CycleGAN

A. Formulation

The loss function used is as described in [4]. The full objective consists of the following terms:

Adversarial Loss For the mapping $\mathcal{G} : X \rightarrow Y$, and its discriminator D_Y, the following objective is the usual tradition:

$$\mathcal{L}_{GAN}(\mathcal{G}, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(Y)}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(X)}[\log(1 - D_Y(\mathcal{G}(x))]$$

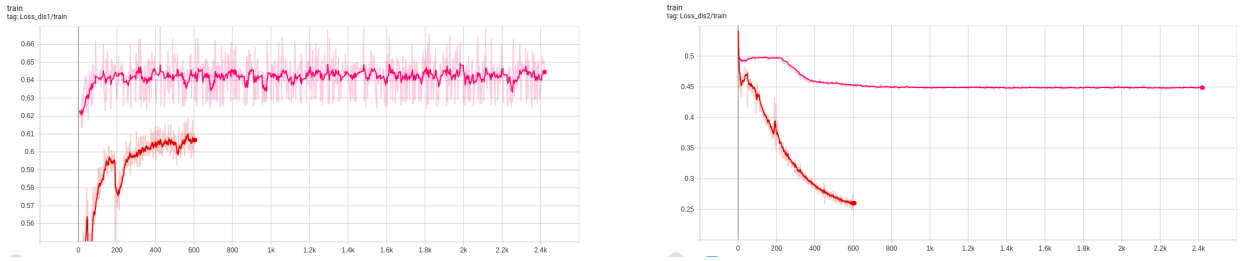

Fig. 5: Discriminator loss

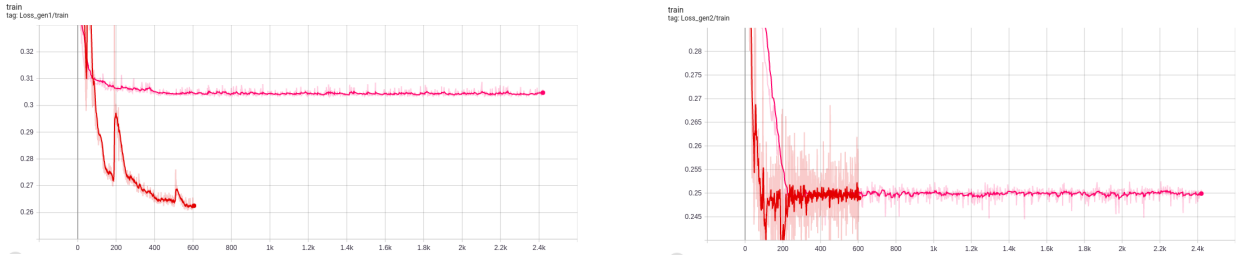

Fig. 6: Generator loss

where the first term is to identify real images, while the second term is to identify generated images as fake. This loss function with a log results in discontinuities and gradient issue, hence we formulated the loss as:

$$\mathcal{L}_{GAN}(\mathcal{G}, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}(Y)}[(1 - D_Y(y))^2] + \mathbb{E}_{x \sim p_{data}(X)}[D_Y(\mathcal{G}(x)^2]$$

and a similar expression for the other adverserial loss:

$$\mathcal{L}_{GAN}(\mathcal{F}, D_X, X, Y) = \mathbb{E}_{x \sim p_{data}(X)}[(1 - D_X(x))^2] + \mathbb{E}_{y \sim p_{data}(Y)}[D_X(\mathcal{F}(x)^2]$$

where $\mathcal{F}, \mathcal{G}$ and D_X, D_Y are the generators and discriminators respectively.

Cyclic Loss As also depicted in Figs. 2(b)(c), the traditional cyclic losses aim to push $\mathcal{F} \circ \mathcal{G} : X \to X$ and $\mathcal{G} \circ \mathcal{F} : Y \to Y$ to identity mappings, so that the two mutually exclusive GANs can be bound together and the mappings $\mathcal{F}$ and $\mathcal{G}$ ideally become inverses of each other. This loss term is given considerable importance since the entire stability of the system is delicate and depends on the relation between the two GANs. Thus this term is an L1 loss term described as:

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

Identity Loss This term was added so that the images drawn from X remain unchanged when passed through $\mathcal{F}$ and similarly, images from Y remain unchanged when passed through $\mathcal{G}$. This makes $\mathcal{F}$ & $\mathcal{G}$ identity on their respective images and provides a further constraint on their distributions. Thus the loss expression is:

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

however the results did not show a considerable improvement with this loss term and we are unsure if it actually produces a positive effect on the model training.

Full Objective Thus our full objective is:

$$\begin{aligned}\mathcal{L}(\mathcal{F}, \mathcal{G}, D_X, D_Y) = \\ &\mathcal{L}_{\text{GAN}}(\mathcal{F}, D_X, X, Y) + \\ &\mathcal{L}_{\text{GAN}}(\mathcal{G}, D_Y, X, Y) + \\ &\alpha \mathcal{L}_{\text{cyc}}(\mathcal{F}, \mathcal{G}) + \\ &\beta \mathcal{L}_{\text{id}}(\mathcal{F}, \mathcal{G})\end{aligned}$$

According to the constraints put forward by the cycle loss and identity loss, we see that the generators $\mathcal{F}$ and $\mathcal{G}$ are autoencoders which together model an identity transformation. But the intermediate encoded information actually is the translation of an image from space X to space Y. Moreover, these generators do not affect any image in there latent image spaces. Training such a complex architecture requires a lot of care and hyperparameter tuning, and most of our training attempts resulted in issues like mode collapse.

It was then that we implemented an ingenious method of GAN training known as Progressive GAN to escape these undesirable situations, with the details presented in the next section.

B. Implementation

Network Architecture The architecture of our generators and discriminators, taken from [7], are shown in Figure 3 and 4 respectively. The activation functions used are RELU and Leaky-RELU for the generator and discriminator architectures respectively, as suggested in [5].

Training Details To deal with dying gradients, a learning rate scheduler has been used. It abates the learning rate to one tenth of its original value if the loss remains invariant over two

Fig. 7: The Auto-Encoder Output, Every 3 Image pair is the reference image, its output through its own decoder, and output through target decoder

runs. Respective generator and discriminator validation losses were fed to their schedulers.

C. Challenges Faced

As you know, there are several challenges one faces during training of a GAN. Here is how we dealt with one of those challenges. During training, we noticed that the discriminator was too weak and wasn't training well as compared to the generator. Thus, in order to strengthen the discriminator, we applied several techniques like:-

Increasing the learning rate: In order to strengthen the discriminator, we increased the learning rate of the discriminator one as compared to the other learning rates. This leads to discriminator one strengthening in comparison to the other.

LeakyReLU: We added the LeakyReLU to the discriminator for better flow of the gradients. This lead to somewhat bettering of the gradient strength.

The results of these can be seen in , where the pink curve is the original model, and the red curve is the model trained with the modified parameters.

VI. A Progressive CycleGAN

NVIDIA's paper on progressive GANs [8] describes a new technique for training GANs for image generation. Inspired by this, this paper proposes a similar method for training GANs for image-to-image translation. We train a CycleGAN using this method and show that better results can be obtained with similar time for training.

A. Implementation

We start with a very simple 2-layer generator and discriminator. The generator has one downsampling layer and one upsampling layer. After training this configuration for a few epochs, we increase the number of layers in the generators by 2 in each step and the discriminators by 1 in each step.

According to [8], these intermediate steps are needed to be trained until near perfect results are achieved, which can take

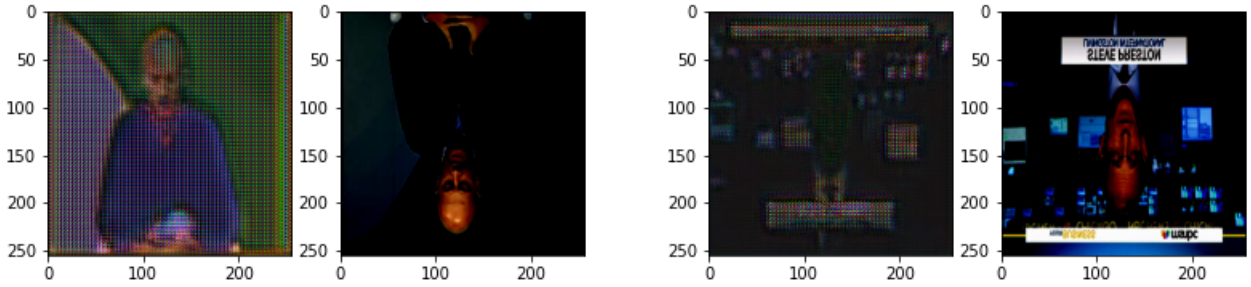

Fig. 8: Outputs from the CycleGAN

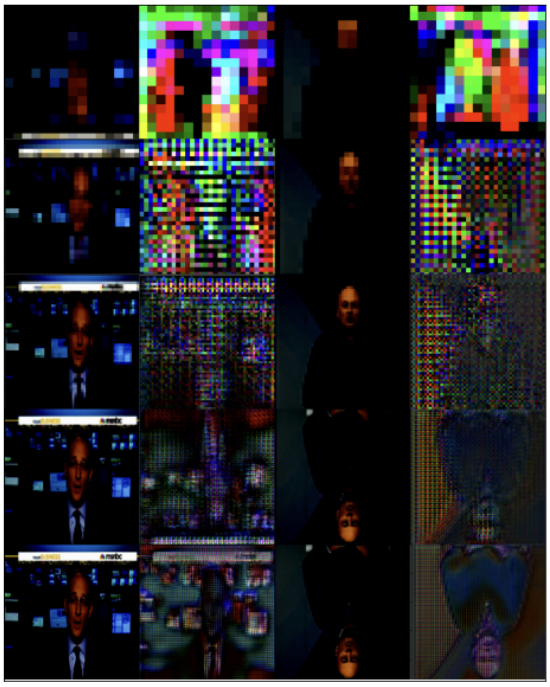

Fig. 9: Progressive GAN results after each progression

days on even a high-end GPU. Due to the lack of resource and time, we have trained for only a few rounds to overview the effect of this unique method of training.

Network Architecture The final architecture is the same as the one used for CycleGAN, but it is built in a total of five steps. Starting with only two layers in the models, we start adding the intermediate layers after training the models at every stage. In the initial small models, we escape the modes and gradient diminishing quite easily. Training the model bit by bit helps us in avoiding most of the undesirable outcomes

VII. Results

A. Autoencoders

Even though the images generated from the Auto-Encoders are really blurry and morphed, you can make out to some extent the face of the target in a similar pose to that of the reference. In some cases, you might also see a change in the background of the target image, which is due to overfitting. Some of the results are shown in Figure 7.

B. CycleGAN

The loss for the generators decreases over several epochs, while the discriminator loss decreases. The cyclic loss also decreases with increasing epochs, however the sheer number of epochs required to train the model for even slightly good results proved to be very high, almost impossible to do on public platforms.

Moreover, for most of the simulations, one of the models ended up in a low loss case while the other models have high losses preventing further training. We even had cases of mode collapse. It was due to these constant problems that we came up with the idea of Progressive CycleGANs.

C. Progressive CycleGAN

The results of CycleGAN and our 'Progressive CycleGAN' are shown in figures 8 and 9 respectively. A key observable difference between the two results is the strange textures visible in 8 but not in its counter part. Even the segmentation of the face is more clearly visible in the latter of the two. The training time for both of them is similar. However, the Progressive CycleGAN model shows better results. This is mainly because the gradients died out in the conventional model.

VIII. Limitations and further Discussion

One drawback of GAN neural network is it needs multiple data samples to train it and the speed of the training process is rather slow. CycleGANs have many interesting applications in all kinds of industry apart from generating new objects and style transfer. One typical application is season transfer. We can train the network using photos of different seasons and turn it into a season transferring system. Likewise, our method can be implemented as a photo enhancement technique. It can generate photos with shallower depth of field given photos captured by DLSR camera. Recently, GANs have modeled patterns of motion in video. They have also been used to reconstruct 3D models of objects from images. There will be an increasing number of improved GANs come into play in the near future.

One major drawback of the CycleGAN is that it is very hard to train and optimize over the function, whilst the progressive GAN is comparatively less harder to train than the former. CycleGAN usually works for smaller image sizes, while the Progressive GAN is also capable to handle relatively higher image sizes.

References

[1] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, Matthias Nießner. FaceForensics++: Learning to Detect Manipulated Facial Images

[2] Oldpan. Faceswap-Deepfake-Pytorch

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS, 2014*.

[4] Phillip Isola, Jun-Yan Zhu, Taesung Park and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks *2017*

[5] Soumith. Ganhacks. https://github.com/soumith/ganhacks

[6] Z. Yi, H. Zhang, T. Gong, Tan, and M. Gong. Dual-gan: Unsupervised dual learning for image-to-image translation. In *ICCV, 2017*.

[7] Tianxiang Shen, Ruixian Liu, Ju Bai and Zheng Li. Deep fakes using generative adversarial networks

[8] Tero Karras, Timo Aila, Samuli Laine and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *2017*