Generating Controllable and Realistic Images by Unsupervised Image-to-Image Translation

Janaranjani Palaniswamy
Matriculation Number: 3441420
University of Stuttgart
st170584@stud.uni-stuttgart.de

Koushik Ragavendran
Matriculation Number: 3439128
University of Stuttgart
st169478@stud.uni-stuttgart.de

Abstract

Automatic transformation of the image from one domain to another domain while retaining original semantics, is defined as image-to-image translation problem. In many real-world applications, it is expensive or difficult to collect sufficient paired training dataset for image translation. Unpaired image-to-image translation has gained a great deal of attention for applications where paired data are unavailable. The aim of this work is to translate a comic face into a real face image. We employ an unsupervised learning technique using Generative Adversarial Networks (GAN) to realize our objective. In this project, the CycleGAN model is used as a baseline to perform unpaired image translation. Besides, we make two proposals to overcome the drawback of CycleGAN in the generation of human faces from comic images. We propose Wasserstein loss to improve the realism of the translation and a modified cycle loss to translate based on general features. Quantitative results of our improved model are presented using FID score.

1. Introduction

Image-to-Image translation is a major phenomenon in computer vision applications. It refers to the task of converting an image to another image using paired dataset. The paired dataset has own aligned images pairs as shown in Figure 1. However, the collection of paired datasets is merely impossible in many of the practical scenarios and it's in turn very expensive. Thus, the need for translating an image from one domain to another domain using unpaired datasets is the major concern in many computer vision and image generation applications. Predominantly, Generative Adversarial Networks (GANs) [4] provides a generic framework when dealing with unsupervised learning algorithms for image translation using an unpaired dataset. In this task, the GAN is trained to transfer images between two domains. An input image is then modified by the trained algorithm to resemble the images of another domain by obtaining its characteristics.

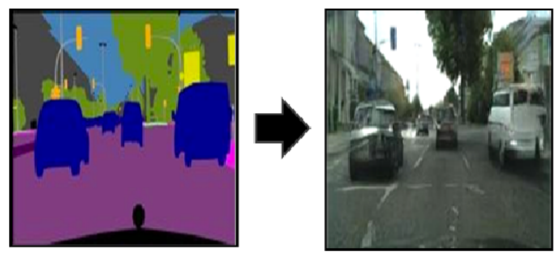

Figure 1. Label to photo translation using paired images.

Figure 2. Unpaired translation of a comic face to a real human face.

Comic to Real translation is a unique yet challenging use case, which aims to generate a celebrity face with a comic facemaker image as an input. Here the comic and real human face images are unaligned images and the translation is illustrated in Figure 2. This task could be used in many real-world image processing applications. This use case is not addressed by various existing models for unpaired image translation. In this report, we consider the work of Zhu et al. [10] to satisfy our objective. CycleGAN is a pioneer to present an approach for learning to translate an image from

source domain X to target domain Y without paired data. Moreover, we investigate the efficiency of this approach to our use case of a Comic to Real face translation. We propose a simple, yet effective way to improve the model's performance in the generation of a realistic human faces from comic face images. We validate these claims through a variety of experiments on facemaker and CelebA datasets.
Our contributions are the following:

1. We propose Wasserstein loss function to calculate the adversarial losses in the CycleGAN.
2. We introduce Feature-wise cycle loss and Weight Decay to overcome the drawbacks of Cycle-consistency loss.
3. We evaluate the model with different parameters using Frechet Inception Distance (FID) score.

1.1. Related Work

Image-to-Image translation aims to transform an image from the source domain to the target. It involves paired and unpaired-data translation. For paired data, pix2pix [5] applies adversarial loss with L1-loss to train the network. For unpaired data, there is no corresponding ground truth in the target domain. Thus, it is more difficult. Moreover, A key problem of unpaired image-to-image translation is determining which properties of the source domain to preserve in the translated domain, and how to preserve them. CycleGAN [10] uses cycle-consistency criteria to achieve a better reconstruction of realistic images. It also helps in reducing the possible learned mapping of the generator. PuppetGAN [7] uses encoders to extract attributes from real and synthetic images and uses these attributes to induce control over the generated images. CartoonGAN [3] proposes semantic content loss using VGG network and edge-promoting adversarial loss for cartoon stylization. In this work, we use the cycle-consistency loss to enforce controllability and we incorporate the idea of semantic content loss from CartoonGAN to calculate feature distance loss.

2. Datasets

In this task, there is a need for two domains of data for comic and real human faces. For real faces, we choose classical face dataset: CelebA and for comic faces, we use the given facemaker dataset. Celebrity Attributes Dataset (CelebA) is a face attribute dataset with more than 200K celebrity face images. CelebA contains 40 binary attributes like eyeglasses, bangs, smile, etc. and it is widely used for various computer vision tasks such as face attribute recognition and face detection. As we are not making use of annotations, we do not need more operations. The facemaker dataset is generated from the 3D facemaker application based on WebGL. This application generates the avatar based on the configured facial parameters. The given facemaker dataset contains 130K male comic face images. We use both CelebA and facemaker datasets for training and testing.

Figure 3. CycleGAN model with two mapping functions and associated discriminators.

3. CycleGAN

The main goal here is to obtain the mapping between two domains A and B such that the output distribution is identical to the target domain (B) distribution. The resulting mappings are G_A: $A \rightarrow B$ and G_B: $B \rightarrow A$. Such a mapping will result in infinitely many mappings between the source and target domains. Domain A contains the comic images from facemaker dataset and Domain B contains real images from CelebA dataset. The model has two adversarial discriminators D_A and D_B which aims to distinguish the images from the target domain and the generated images. The overview of the mapping functions and adversarial discriminators are shown in Figure 3. The overall objective of CycleGAN contains two major loss functions: adversarial losses to encourage the generator to produce outputs that are indistinguishable from the data distribution in the target domain and cycle-consistency loss that capture the intuition that if we translate from one domain to the other and back again we should arrive at where we started [10].

In this work, we use the Forward CycleGAN and Full CycleGAN with both forward and backward cycles for various experiments. We present the details of the individual architectures in Section 3.2. This section describes the details of the generator and discriminator networks, model architectures, Adversarial and cycle loss functions, the proposed approach, and the evaluation criteria for CycleGAN.

3.1. Generator and Discriminator Network

A GAN framework consists of two CNNs. One is the generator G which is trained to produce output that fools the discriminator. The other is the discriminator D which classifies whether the image is from the real target manifold or synthetic. Like other GAN frameworks, a discriminator function D is trained for pushing G to reach its goal by distinguishing images in the CelebA dataset from the generated output of G and providing the adversarial losses for G. We adopt the network architecture of generator and discriminator used by Zhu et al. [10]. The detailed network architecture is as follows. This network contains three convolutions, several residual blocks [18], two fractionally-strided convolutions with stride 1/2, and one convolution that maps features to RGB. We use 9 blocks for 64 × 64 and higher-resolution training images. We use instance normalization [6] as specified in CycleGAN paper.

On the other hand, the discriminator network D is used to decide whether the input image is a real or comic image. Since this task of judging whether real image or not is less demanding and hence a 70 × 70 PatchGANs is used. This PatchGAN network will classify whether 70 × 70 overlapping image patches are real or comic. The major reason for using this architecture is that it requires only fewer parameters in D than a regular full-image discriminator [10].

3.2. Model Architecture

Forward CycleGAN: In this architecture, there are two generators and one discriminator. Let us consider these generators as G_A and G_B and the discriminator as D_A. G_B is used to ensure the back translation and reconciliation. The architecture diagram with input and output images of our use case is given in Figure 4. For each image from domain A, the forward cycle should be able to bring back the original image i.e., $A \rightarrow G_A(A) \rightarrow G_B(G_A(A)) = A$.

Generator G_A is used to generate the real image (B*) from a given comic image. The Discriminator D_A discriminates the generated fake real image (B*) by comparing with the real image (B) in the target dataset. In this way, Generator G_A is learning to generate image similar to the real image from the target domain. Meanwhile, the Generator G_B takes the generated fake image (B*) as input and tries to regenerate the comic image (A*). This regenerated comic image is compared with original comic image either pixel-wise or feature-wise. The generator G_A and G_B is trained to decrease the pixel-wise or feature-wise loss. Thus, the Generator G_B learns to regenerate comic image from generated fake real image and Generator G_A learns to generate a fake real image with features of comic image.

Full CycleGAN: The Full CycleGAN architecture is an extension of forward CycleGAN architecture with backward cycle. Backward cycle involves translation from domain $B \rightarrow A : B \rightarrow G_B(B) \rightarrow G_A(G_B(B)) = B$. Here there is an additional discriminator and it is named as D_B. The architecture details concerning Comic to Real image translation are shown in Figure 5. This discriminator D_B is used to discriminate the generated fake comic image (A*) and comic image (A) from the training dataset. The discriminator D_B learns to differentiate the generated fake comic image and comic image during training.

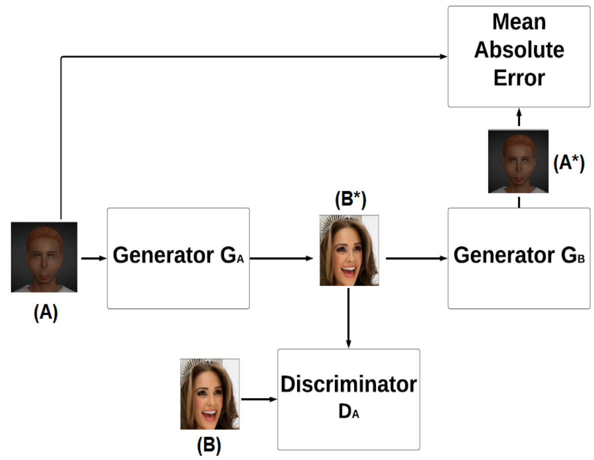

Figure 4. Architecture of Forward CycleGAN for Comic to Real Image Translation.

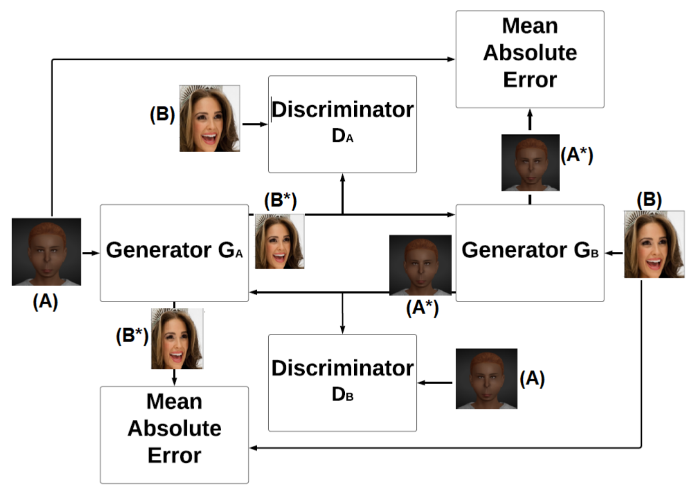

Figure 5. The architecture details for Comic to Real image translation of Full CycleGAN with Forward and Backward cycle.

Generator G_A maps $A \rightarrow B$ and Generator G_B tries to map $B \rightarrow A$. The associated discriminators D_A aims to distinguish between images (A) and translated images (A*) from G_B, whereas D_B distinguishes between images (B) and translated images (B*) from G_A. To satisfy the cycle-consistency criteria, the generated fake real (B*) and comic (A*) images are used to regenerate comic (A) and real (B) image using Generator G_B and G_A respectively. The regenerated comic and real images are compared using pixel-wise or feature-wise with the original images in the train sets. We aim to solve:

3.3. Losses

In general, CycleGAN has two objectives: one is to ensure that the translated image looks like the target domain called as GAN objective. Another objective is to ensure that the translated image still looks like the original and it is called as reconstruction objective. Adversarial losses are used to satisfy the GAN objective and Cycle loss is used to satisfy the reconstruction objective. Various losses are involved to realize these two objectives. The Overall loss function of our implementation is shown in (1).

$$Loss(G_{\text{A}}, G_{\text{B}}, D_{\text{A}}, D_{\text{B}}) = Loss_1(G_{\text{A}}, D_{\text{B}}, A, B)+ \quad (1)$$

$$Loss_2(G_{\text{B}}, D_{\text{A}}, B, A) + \lambda Loss_{\text{cyc}}(G_{\text{A}}, G_{\text{B}}) + \lambda Loss_{\text{identity}}$$

Where Loss_1 and Loss_2 are the adversarial losses for forward and backward cycle and lambda is the weight value used to control the relative importance of the two objectives. Loss_{cyc} and $\text{Loss}_{\text{identity}}$ are the cycle and identity losses respectively. The identity loss says that if an image from the target domain (B) is fed as an input to the generator, it should yield the output image B which is close to the actual input image in B. The detailed description of adversarial and cycle loss are discussed in Section 3.3.1 and Section 3.3.2.

3.3.1 Adversarial Loss

Adversarial losses are used for matching the distribution of generated images to the data distribution in the target domain. In the CycleGAN architecture from Figure 5, the discriminators D_{A} and D_{B} tries to maximize the loss. Whereas the Generator G_{A} and G_{B} tries to minimize the loss against the respective discriminator. The adversarial losses are calculated for both forward and backward cycles.

In our work, we use different loss functions to calculate the adversarial losses. The functionality of Binary Cross Entropy loss, Wasserstein loss and Least squared loss functions are described below.

Binary Cross Entropy (BCE) Loss: This loss function compares the ones or zeros and predicted value. While using BCE loss the model is trained to predict value equal to ones or zeros. In the case of discriminator, the output of discriminator when dataset image is given as input is compared to ones and output of discriminator when the generated image is given as input is compared to zeros. In this manner, discriminator learns to convert train dataset images to ones and generated images to zeros. In the case of generator, the output of the discriminator when the generated image is given as input is compared to ones. In this way, the generator learns to generate images that can be converted to ones by the discriminator. The BCE loss function is shown in (2).

$$Loss_{\text{bce}} = y_{\text{i}} * log(p(y_{\text{i}})) + (1 - y_{\text{i}}) * log(1 - p(y_{\text{i}})) \quad (2)$$

Wasserstein Loss (W loss): This loss function compares the mean of predicted value with the mean of ground truth. When using W loss the model is trained to predict value equal to the ground truth. In the case of discriminator output of discriminator for given dataset image mean is maximized and output of discriminator for given generated image the mean is minimized. The discriminator learns to convert the train dataset to the maximum value and convert generated image to the minimum value. In the case of generator output of discriminator for given generator image is maximized. The generator learns to generate images that can be maximized by the discriminator. W loss formula is given in (3), where x is the true image and y is the predicted image.

$$Loss_{\text{W}} = E_{(\text{x,y})}||x - y|| \quad (3)$$

Least Squared (LS) Loss: This loss function compares the ones or zeros and predicted value. With LS loss the model is trained to predict value equal to ones or zeros. In the case of discriminator, the output of discriminator when dataset image is given as input is compared to ones and squared. The output of the discriminator when the generated image is given as input is compared to zeros and squared. In this manner, the discriminator learns to convert the train dataset images to ones and generated images to zeros. In the case generator, the output of the discriminator with generated images as input are compared to ones. In this way, the generator learns to generate images that can be convert to ones by the discriminator. LS loss function is described in (4).

$$Loss_{\text{ls}} = (x)^2 + (1 - y)^2 \quad (4)$$

3.3.2 Cycle-Consistency Loss

To learn the mapping between individual input image from the source domain to the desired output image in the target domain, cycle-consistency loss is used [10]. In general, cycle-consistency loss compares the images by pixel distance, where the generated images are just a 3D tensor with values. In pixel-wise comparison, the corresponding element in the 3D tensor is compared. In this way, the predicted image is expected to look exactly like the ground truth image. It also helps to reduce the possible solution space mapping functions. In simple terms, cycle loss is the absolute difference between the original input image and the reconstructed input image through the cyclic network as shown in Figure 5.

A sample from the training is shown in Figure 6. The behavior induced by the cycle consistency loss can be ob-

Figure 6. The Input Comic, Generated Real and Reconstructed Comic images from Cycle-consistency Network.

served. The reconstructed comic image looks very similar to the input comic image.

3.4. Proposed Approach

After choosing CycleGAN as our base model to realize comic to real image translation, we made few proposals based on our research. The following are the ideas we proposed to satisfy our objective.

3.4.1 W Loss for Adversary

Among the three different loss functions for adversarial loss as discussed in Section 3.3.1, we propose to utilize the W loss function. The main reason behind the W loss function is that it helps to increase the realism and diversity of the generated output images. Also W loss helps to increase the stability of GAN and also reduces the problems of vanishing gradient and mode collapse [1].

3.4.2 Feature-wise Cycle-consistency

During our research, we found that the cycle loss is performing pixel-wise comparison and it tries to recover the original image pixels. To overcome the drawback of pixel-wise comparison, we decided to do the comparison based on the feature level. Thus, the cycle-consistency loss is calculated based on the feature-wise distance. In this method, the generated image is given as input to the Inception pre-trained model which returns a feature vector of size 2048. In feature-wise comparison, the corresponding element from the feature vector is compared. While doing this type of comparison the predicted image has the feature of ground truth image. In this case, the output shares similar features to the ground truth image [8].

3.4.3 Weight Decay

A weight is assigned to regenerated cycle loss and identity loss. This weight is used to decide the impact of loss values while training the generator. As the cycle loss helps in stabilizing the training a lot in the early stages but it becomes an obstacle towards realistic images in later stages. We propose to linearly decay the weight value so that the generator is trained less based on cycle loss and identity loss as the training progresses [8].

3.5. Evaluation Criteria

A definite quantitative evaluation criterion for GAN is still an open research question. We use the Frechet Inception Distance (FID) [2] score to evaluate the performance of the CycleGAN with our proposed changes. FID score summarizes the distance between the Inception feature vectors for real and generated images in the same domain. Ideally, lower the FID score, more closer the generated image is to the original real image. Normally FID score is calculated using the Inception model that returns the feature vector with 2048 values. This feature vector is obtained for two sets of images. These two sets of vectors are used to calculate the mean and variance and then this is compared to compare the distribution of the image generated by using the formula in (5).

$$d^2 = (\mu_1 - \mu_2)^2 + trace(\sigma_1 + \sigma_2 - 2 * sqrt(\sigma_1 * \sigma_2)) \quad (5)$$

There exists no standardized metric to evaluate the unconditional generation of images, we reply upon visual inspection of generated samples for the quality assessment.

4. Implementation Results

This section provides a thorough analysis of the performance of our various experiments with the proposals to obtain Comic to Real image generation. Also, a detailed description of data pre-processing and training steps are discussed.

4.1. Data Pre-processing

To reduce the training time, we only use 4000 images from both facemaker and CelebA the datasets. We perform normalization, cropping, resizing face images to 64 × 64, and create random mirroring for data augmentation purposes on both datasets. We fixed the pixel size as 64 × 64 to obtain a stable GAN training. Since the given datasets are clean without noise, we do not require much pre-processing.

4.2. Training Details

The CycleGAN architectures as detailed in Section 3.2 is used to train the comic and real face images. The hyperparameters like Batch Size, Learning Rate, pool size, weight penalty and Mean Absolute Error as the identity loss function were kept constant as per the specifications given in [10], our reference model. The Number of Epochs the models were trained for is 100. The learning rate remains the same for the first 50 epochs and linearly decays the rate to zero over the next 50 epochs. Similar to Zhu et al. [10] we use a buffer that stores 50 previously generated images to update the discriminator to reduce the model oscillation.

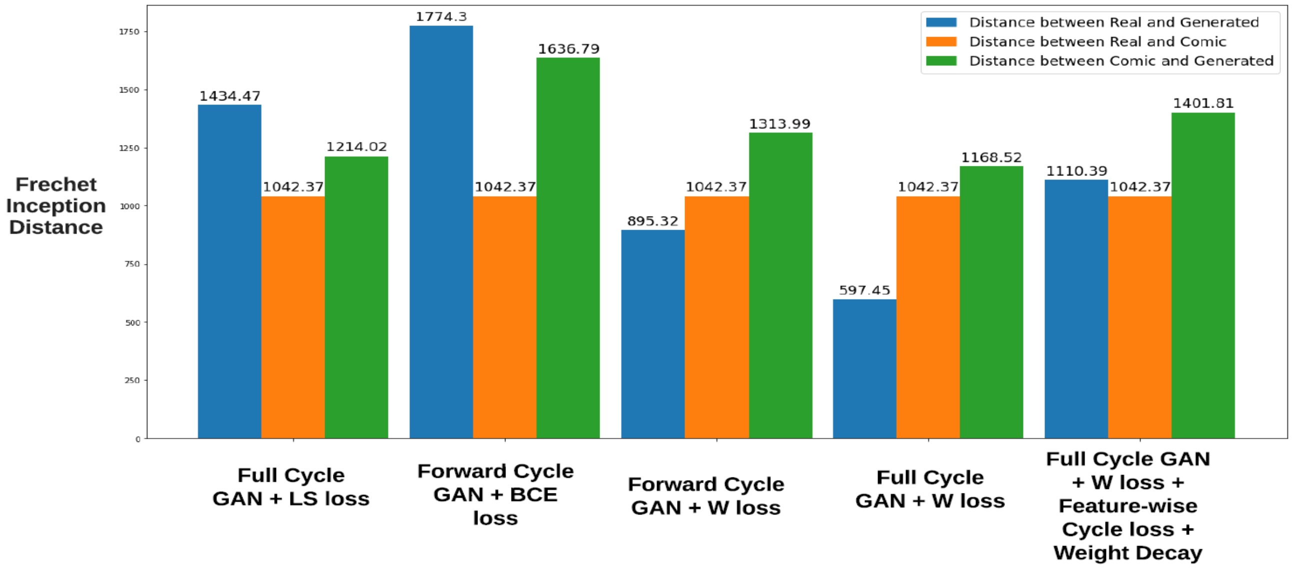

Figure 7. The FID score of Real and Generated images, Real and Comic images, and Comic and Generated images for different experiments. From left to right: Full CycleGAN + LS loss, Forward CycleGAN + BCE loss, Forward CycleGAN + W loss, Full CycleGAN + W loss and Full CycleGAN + W loss + Feature-wise cycle loss + Weight decay.

As a first step, we re-implemented the original CycleGAN with LS loss. Later, to get the better output, we performed different experiments which are discussed in Sections 4.3 to 4.7. Finally, we evaluate the models with different parameters using FID scores. The generated image for the given facemaker comic image using Full CycleGAN + LS loss, Forward CycleGAN + BCE loss, Forward CycleGAN + W loss, Full CycleGAN + W loss, and Full CycleGAN + W loss + Feature-wise Cycle loss + Weight decay is visualized in Figure 8. The detailed visualization of the FID score of Real and Generated image, Real and Comic image, and Comic and Generated image using above models are shown in Figure 7.

4.3. Full CycleGAN with LS loss

In the first step, we re-implemented the CycleGAN model. It is also mentioned in the CycleGAN paper [10] that the model with the Least Squared loss is not suitable for human faces. We can see from the results shown in Figure 8 that regardless of the varying features of the input image like the shape of eyes, mouth, and nose it generates the same output image. From Figure 7, we can see that the FID score between real and generated images is increased to 1434.47 from 1042.37 (expected distance between comic and real image) and FID score between comic and generated images is increased to 1214.02 from 0 (ideal).

4.4. Forward CycleGAN with BCE loss

We started our experiment using the CycleGAN with the forward cycle as explained in Section 3.2 to verify and analyze the behavior with the forward cycle alone. Firstly, BCE loss is employed to calculate the adversarial losses. This experiment did not give better results. We can see from results in Figure 8 that the feature like eyes, mouth, nose and skin tone are not translated from input image. Moreover, the FID score between real and generated images is increased to 1774.3 from 1042.37 (distance between comic and real) and the distance between comic and generated images is increased to 1636.79 from 0, it is visualized in Figure 7. Forward CycleGAN + BCE loss is performing worse than the Full CycleGAN + LS loss. But Both of them have the same issue of generating similar images for all images called mode-collapse.

4.5. Forward CycleGAN with W Loss

In an attempt to solve the mode-collapse problem in the generated image we used W loss as stated in Wasserstein GAN paper [1]. So, we added W loss as the adversarial loss function to the Forward CycleGAN. From the results in Figure 8 we can see that the features like eyes, mouth, and nose are translated to the generated image. In this case, the FID score between real and generated images is decreased to 805.32 and the FID score of 1313.99 is obtained for comic and generated images as shown in Figure 7. Though certain features from the comic image are translated to generated image the other features look the same from the real dataset and in turn, the output images for various inputs look almost similar.

4.6. Full CycleGAN with W Loss

It is suggested in the CycleGAN paper [10] that using both forward and backward cycle helps in regularizing the training for this under-constrained problem and it also avoids mode-collapse. In this case, we utilized the full CYcleGAN model with W loss function for adversarial loss. From the results in Figure 8 we can see features like eyes, mouth, and nose are translated and the other features from the real dataset are also not similar in generated images. Here, the FID score between real and generated images is decreased to 597.45 from 1042.37 as shown in Figure 7 and it appears to be the optimum value among other experiments.

4.7. CycleGAN with Feature-wise Cycle Loss and Weight Decay

Full CycleGAN with W loss is performing better than the earlier experiments. Though the distance between real and generated is decreased, the distance between comic and generated images is not decreased. Also, the features like skin tone are not translated from the input comic image. We employed the W loss, modified feature-wise cycle loss, and Weight Decay to the Full CycleGAN. We observed that in addition to the feature like eyes, mouth, and nose the model also tries to translate the skin tone of the input image to the generated output image. This is depicted in Figure 8. Unfortunately, the FID score between real and generated images, comic and generated images are increased than the full CycleGAN model with W loss method as shown in Figure 7. We could not achieve our intention of further reducing the distance between comic and generated images. Maybe the intended goal could be achieved by using only feature loss.

5. Future Scope

As discussed in Section 4.7 the experiment with feature-wise cycle loss and weight decay ended up in unsatisfactory results. The reason could be the Weight Decay phenomenon as it reduces the weightage of regeneration loss as training progresses. As future work, this can be extended by calculating feature-wise cycle loss alone and observe the model output. In addition, from Figure 7, it is observed that though Full CycleGAN + W loss gives a better FID score, the visual results are not completely satisfying. As a next step, analysis can be done to change the model architecture using global and local discriminator as done by Wu et al. [9] for obtaining the desired results. Furthermore, the given facemaker dataset contains only male comic face images, steps can be taken to include female comic face images and train the network to map a female comic face to a female celebrity (real) face image.

6. Conclusion

Unpaired Image-to-image translation is a challenging and important task in computer vision and machine learning. Also, unsupervised training with GAN is generally unstable and hard. For Comic to Real face translation, we utilized two losses to make the generated distribution close to the target domain distribution. Besides, we also introduced the FID score as a metric to evaluate the performance of the CycleGAN model. We have observed that CycleGAN from the base paper [10] do not perform as expected for the desired translation. From our work, we can see that the CycleGAN + forward cycle alone generates the same output image for any given input image. We have shown that Full CycleGAN with W loss can generate realistic images and captures the facial attributes from the input image except skin color resulting in the best FID score between real and generated images. We demonstrate that when adding feature-wise cycle loss and decay of loss weightage to the full CycleGAN, the model does not result in better Comic to Real translation. This method in turn increased the FID score between real and generated images. Thus, we conclude that among our two proposals only W loss function for adversarial losses in Full CycleGAN results in better model performance compared to the baseline CycleGAN model, whereas the modified feature-wise cycle loss did not end up in better results providing a space to improvise the translation as stated in Section 5.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

[2] Ali Borji. Pros and cons of gan evaluation measures, 2018.

[3] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[4] Ian J. Goodfellow, Jean Pouget-Abadie, Bing Xu Mehdi Mirza, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[5] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[6] Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. A large-scale study on regularization and normalization in gans, 2019.

[7] B. Usman, N. Dufour, K. Saenko, and C. Bregler. Puppetgan: Cross-domain image manipulation by demonstration. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9449–9457, 2019.

[8] Tongzhou Wang and Yihan Lin. Cyclegan with better cycles, 2019.

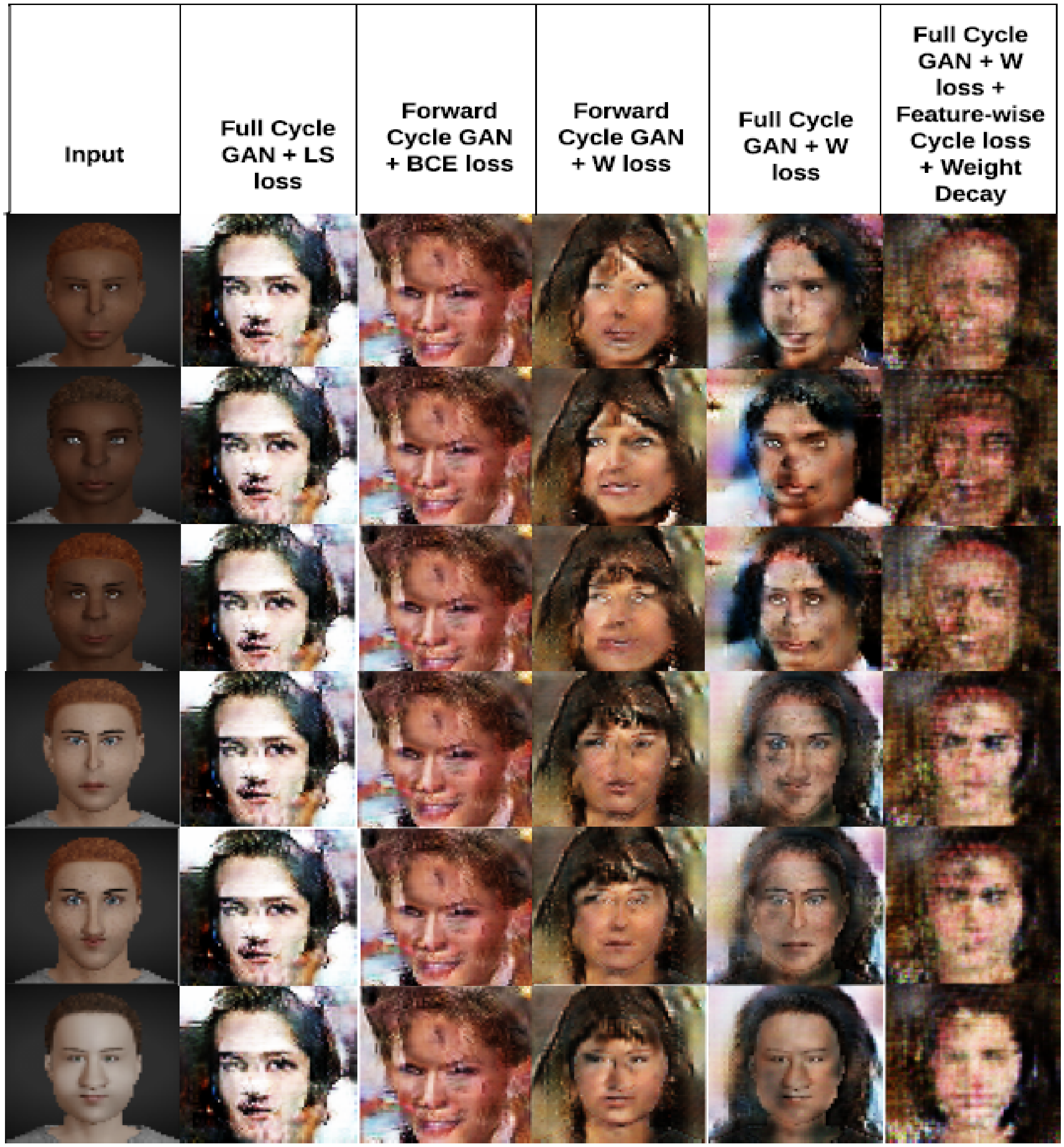

Figure 8. Results of different methods for Comic to Real image translation. From left to right: Full CycleGAN + LS loss, Forward CycleGAN + BCE loss, Forward CycleGAN + W loss, Full CycleGAN + W loss, Full CycleGAN + W loss + Feature-wise cycle loss + Weight decay trained on facemaker and CelebA dataset.

[9] Ruizheng Wu, Xiaodong Gu, Xin Tao, Xiaoyong Shen, Yu-Wing Tai, and J iaya Jia. Landmark assisted cyclegan for cartoon face generation, 2019.

[10] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *CoRR*, abs/1703.10593, 2017.

A. Dataset

• The facemaker and CelebA datasets used in this work was provided by the department. The datasets are available in the server path: '/home/palaniswamyji/Datasets'.