CSE 598: Intro to Deep learning Project Report.

Application of CycleGAN to Convert Real-life Images to Realism Style Paintings and Vice-versa

Pavan Kumar Raja

MS computer science, School of Computing and Augmented Intelligence, ASU

praja3@asu.edu

***Abstract*— Translating real images to Realism art style images. This art style dates back to the 17th century so image-to-image pair datasets are not available to train traditional image translation networks. Ideas expressed in *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*[1] try to define an architecture to build such models without such data. I'll be experimenting with this architecture and building a cycleGAN Network to transfer real-life images to realism art style images, a bi-product of this implementation will also be capable of converting Realism art style images to somewhat real photos.**

I. Introduction

Art styling of real-life images is usually achieved with i. Neural style transfer on pre-trained image models. A drawback of this approach has a drawback where the pipeline requires human intervention to balance style transfers and feature retention else we end with examples as in Fig 1.

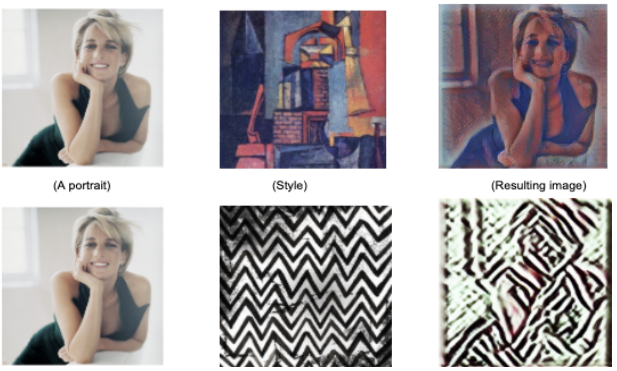


Fig. 1: *[real, style image, result]* - Arbitrary Style Transfer[1]

ii. Another approach is to train a traditional Image-to-Image network and use Real-Image to Art-Image paired dataset, the drawback of this approach is that it is limited to applications where these datasets are readily available, Take an example of my use case where this dataset is not available because my art style is from 17th century. So I am using an idea proposed in the paper *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*[1] - Where they propose having two translation functions such as G : X → Y and another translator F : Y → X, then G and F should be inverses of each other, and introduces cycle consistent loss and minimizing this loss lead to F(G(x)) ≈ x and G(F(y)) ≈ y. In our case, F & G are GAN networks.

II. Datasets

There were no readily available datasets for my use case, so I had to pick data from different sources and combine them to make an Art-image & Real-image dataset.

A. Realism dataset selection.

I have picked the Realism image folder from a dataset from WiKiArt Dataset[2]. There are *10532* images of art painted by various artists. I have split this into a train, test, and evaluation dataset with a split of 70:20:10. All these images are copyrighted and only available for academic and research purposes[2]. Handpicked examples are shown in Fig 2.

Fig 2: Sample of realism art dataset.

B. Real image dataset selection.

I have picked 3 domains for creating this dataset - a) Landscape images dataset picked from Flickr for *cartoonGAN* paper[3]. This dataset consisted of 3900 images and all these images are used. b) Indoor scene images picked from the dataset used in *Recognizing*

[1] Arbitrary Style Transfer https://reiinakano.com/arbitrary-image-stylization-tfjs/

[2] WikiArt term of use - https://www.wikiart.org/en/terms-of-use

Indoor Scenes paper[4]. This dataset consisted of 15620 images, and upon seeing the distribution I only picked interesting scenes from the dataset such as images of bookstores, libraries, casinos, kitchens, etc. which amounted to ~1500 images. c) Images of face portraits from FFHQ defined in *A Style-Based Generator Architecture for Generative Adversarial Networks*[5] were used for face images. I handpicked around 800 images to form my dataset. The resulting dataset was 6318 shuffled images belonging to a, b, and c. This dataset was further divided into train, test, and evaluation datasets with a 70:20:10 ratio. Few images from the dataset are shown in Fig 3.

Fig 3: Indoor, Face portrait and landscape

III. Model

The architecture of the cycle GAN model follows *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*[1].

A. Loss function - The loss function for training is a combination of Cycle consistency loss, Adversarial Loss as per [1], and Identity loss as defined in *Unsupervised Cross-Domain Image Generation*[6]. We can explain these losses by taking 2 GAN networks (F, G) into consideration, where $F : Y \rightarrow X$ and $G : X \rightarrow Y$, in this setup a) Adversarial loss is average discriminator loss i.e, loss of a GAN to discriminate generated vs real image. (There are 2 discriminators one to discriminate X samples and another Y and use generated vs real image to discriminate). b) Cycle consistency loss is a loss function between {(G(F(Y)), X}, i.e the loss calculated between generated X(after a cycle) vs actual X. c) Identity loss is a loss function of {F(Y), Y}. This function helps the network to identify if an image belongs to the same domain and in this case, it doesn't alter the image too much. Ex: when a real-life image is passed to a real-life generator, it has to know that the input belongs to its domain and make a minimal change to it.

The final loss function is a weighted combination of a, b, and c (1, 10, 5).

B. Network architecture - There is a little deviation from the original paper to reduce the complexity (removed fractionally strided convolution layers, and change in convolution layer set up) i. The generator block consists of an initial convolution block to upscale the channels, followed by 2 Conv blocks with a stride of 2 to downscale the input twice, followed by 9 residual networks (as per the paper) and 2 upscaling convolution blocks to output image with the same dimension as the input image. ii. For the Discriminator, I am using a full discriminator instead of a patch discriminator (which is used in the paper[1]). Discriminator consists of 4 (Conv, norm, and Leaky ReLU) layers.

C. Parameters - All the weights of the Conv layer were set to a normal distribution of (0, 0.02), LR was set to 0.0002 it had to decay at every 100 steps by a factor of how many epochs left. Initial epochs were set to 200, but I could only train to 140 epochs due to computational non-availability.

D. Computational matrix - Trainable parameters: Each of the generators had 11378179 and each discriminator had 2764737 trainable parameters. I trained on 140 epochs, on Google colab pro GPU (model of the GPU was not exposed), each epoch took around 40 minutes to train.

IV. Baseline and evaluation

The paper Cycle gan paper *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*[1] uses traditional methods of evaluation. Their models were evaluated on human turkers labeling, automated comparison of Per-pixel accuracy, Per-class accuracy with other models like (BiGAN, CoGAN, GAN, and SimGan). There are a few issues with such metrics, i. Per-pixel accuracy is an evaluation on F(G(X)) which is more of an evaluation of Generator & Discriminator accuracy, I am not interested in such evaluation as our primary output only F(Y)/ G(X) independently, and we have to evaluate them separately. ii. Per-class acc depends on the prediction of the class of the object on a trained object; however, we can't be sure that all the objects in the art/ real image are being trained with this model. Because of the above-mentioned issues, I have deviated from baseline

and evaluation metrics from *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*[1] paper and instead will be using a likeness score as defined in *A Novel Measure to Evaluate Generative Adversarial Networks Based on Direct Analysis of Generated Images*[7]. The likeness score (LS) is a combination of Creativity, Inheritance, and Diversity. The advantage of using LS is that we do not worry about evaluating our new Generative models on old pre-trained models and hence the performance of the network is based solely on the dataset and the generated images. LS is calculated by using Intraclass distance (distance between real and generated image dataset), and Between class distance (Distance between overall real images and generated images). Then calculated LS using the Kolmogorov-Smirnov test for goodness of fit to find if real and generated images belong to both the distribution (it's own and generated). We can say LS approaches 1, as the model improves the quality of the generated images.

To define LS we require a real-art image dataset, Which we didn't have before building the model. So a good measure at this stage was to take the real-life image evaluation dataset and use a couple of images from the realism art dataset and calculate the distance between these sets. This will give a good bound on creativity: Generated images should not be the same as real, Diversity: Generated images are not similar to each other, and Inheritance: Overlap of creativity and diversity. I used 630 samples to define the baseline and it fluctuated from ***0.2*** **to** ***0.3*** ***(****0.26, 0.29 as per the notebook shared****)*** depending on the random images picked.

The LS score of the results was ***0.94***. So the model that I built based on the *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*[1] was successful in generating images with good creativity, diversity, and Inheritance.

V. Experiments

A. Training dataset (Art) - Art style datasets were readily available, and I chose the Realism part of it because it was not used in any of the art style papers that I have come across. Although I was trying to build an architecture based on the paper [1], I didn't want to replicate it and have deviated from it as much as possible.

B. Training dataset (Real-life) - I specifically choose landscape, indoor and face portrait image sets of these only landscape images belonging to the domain of realism art (it had most of the paintings depicted outdoor scenes), the other two were picked to see if the learning could be cross domain.

C. Model architecture - I have deviated from the architecture in the paper [1]. i. Instead of having a fractionally strided convolution layer, I have upsampled the input before doing a normal convolution with stride 1. Advantage of this is the input is extrapolated and new data have average value of neighbors, and this is not learnable unlike transpose convolution (fractionally strided conv), making this change made the model train slightly faster. ii. Instead of using a 70X70 patch discriminator, I have used a full discriminator (conv, norm and activation) layers. Although this is slower, I felt this better suits my use case by considering the entire art style of the image.

D. Evaluation - Although the paper *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks* [1] defines clear evaluation metrics (inception scoring and human annotation), I have decided to go with concepts from *A Novel Measure to Evaluate Generative Adversarial Networks Based on Direct Analysis of Generated Images*[7] as I believe it better in terms of scoring art related GANs.

VI. Results

The models were trained for 144 epochs (140 saved), at the end of this my losses were [discriminator loss: 0.2, Generator loss: 2.4, Adversarial Loss: 0.85, Cycle consistency loss: 0.11 and identity loss of 0.09. The LS score of the model was 0.94. It is to be noted that at the end of the training not all the results were good looking, there are few bad results.

From the results we can infer that i. The model probably needed more epochs to train more. ii. Real-life images (landscape) belonging to the same distribution as the Art style had better artstyle transfers. This could also be a result of having more landscape images in the dataset than other domains. iii. Model also learnt to convert art style images into real-like images, and had better results on human subjects (face/ entire scene with humans), this can be attributed to model learning coloring of skin from portrait images to the real-life dataset.

All the datasets saved models are available on the drive[3]. Code used to build and evaluate the model is available on github[4]. Progression of the generator, positive samples and negative samples are shown at the end of the document.

VII. References

[1] Jun Yan Zhu, Taesung Park, Phillip Isola and Alexei A. Efro. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. 2017. http://arxiv.org/abs/1703.10593

[2] Github - https://github.com/cs-chan/ArtGAN/tree/master/WikiArt/Dataset

[3] Chen, Y., Lai, Y.K., Liu, Y.J. CartoonGAN: Generative adversarial networks for photo cartoonization. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 9465–9474 (2018)

[4] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2009, pp. 413–420.

[5] Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. 2018. http://arxiv.org/abs/1812.04948

[6] Yaniv Taigman, Adam Polyak and Lior Wolf. Unsupervised Cross-Domain Image Generation. 2016. http://arxiv.org/abs/1611.02200

[7] Shuyue Guan and Murray H. Loew. Measures to Evaluate Generative Adversarial Networks Based on Direct Analysis of Generated Images. 2020. https://arxiv.org/abs/2002.12345

[3] Dataset link - https://drive.google.com/drive/folders/1zjOugq_MPH4idMQWd800FCo2DZD285xR?usp=sharing

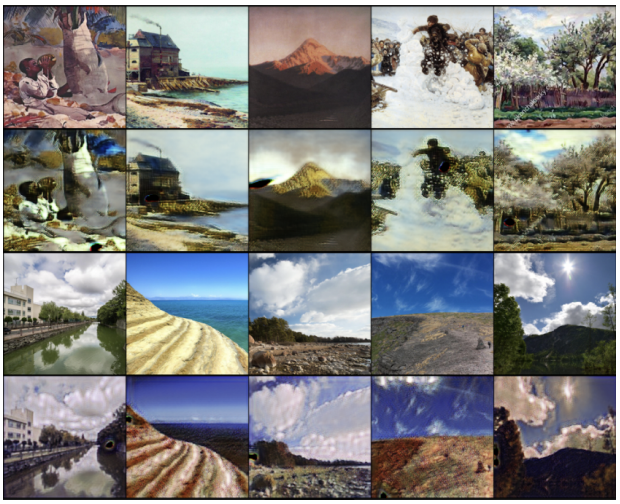
[4]Github - https://github.com/Pavan-pk/ArtGan_Realism

Result images.

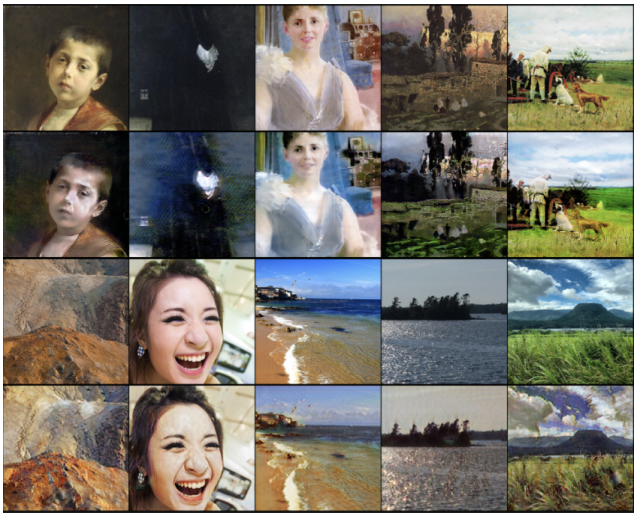
1. Training samples.

a. First row shows Realism art style data.
b. Second row shows images converted from art to real-life kind images.
c. Third row shows images from the Art style dataset.
d. Second row shows images converted from real-life images to Art style.

Step 10000:



Step 100000:



Step 500000:

Step 1000000:

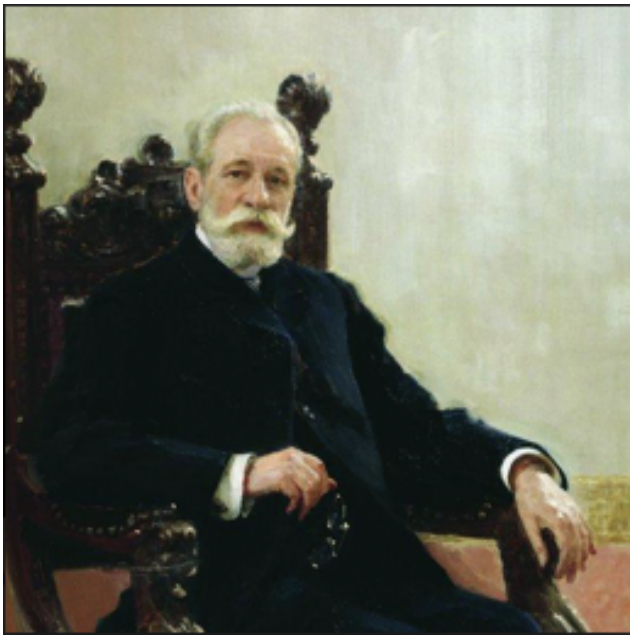
Step 107000:

2. Evaluated Images.

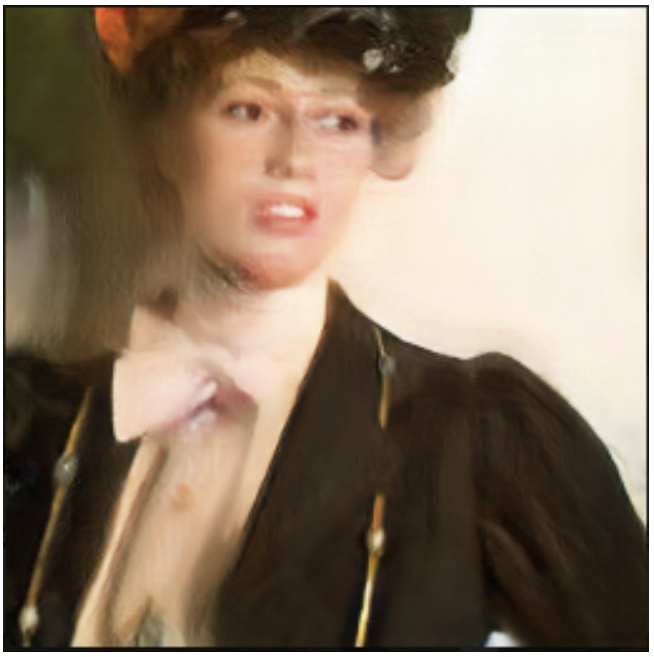
Positive image pairs, Which I thought the model performed well on.

Real **Generated**

Art



Generated

Negative image pairs, which I thought the model performed worse on.

Real **Generated**


ART

Generated