STYLE TRANSFER

Rithesh R N, Yuvaraj Kakaraparthi, Sai Hanisha Kilari
ECE 285- Machine Learning for Image Processing

June 10, 2019

ABSTRACT

Humans have developed artificial systems like CNN's to create unique visual experiences through composing a complex interplay between the content and style of an image. Neural style transfer is one such optimization technique that renders the content of one image with the style of the other. In this paper we implement the classic technique of style transfer and one of its variants, Image-to-Image Translation using Cycle-GANs.

***K*eywords** Style Transfer · Cycle-GAN · CNN

1 Introduction

While machine learning algorithms have shown superior performance in predictive tasks like classification, object detection etc., they historically lacked the ability to generate artistic images through composing a complex interplay between the content and style of an image. While algorithms existed in other key areas of visual perception such as object and face recognition which demonstrate near-human performance there lacked algorithms for artistic art generation. Neural Style Transfer(NST) is an example of image stylization, a problem studied for over two decades within the field of non-photorealistic rendering. Prior to NST, the transfer of image style was performed using machine learning techniques based on image analogy. Given a training pair of images–a photo and an artwork depicting that photo–a transformation could be learned and then applied to create a new artwork from a new photo, by analogy. The drawback of this method is that such a training pair rarely exists in practice. For example original source material (photos) are rarely available for famous artworks.

2 A Neural Algorithm of Artistic Style

2.1 Introduction

In the paper "A Neural Algorithm of Artistic Style" by Gatys et al. introduce an artificial intelligence system based on a Deep Neural Network that creates artistic images of high perceptual quality. The system uses neural representations to separate and recombine content and style of arbitrary images, providing a neural algorithm for the creation of artistic images. Convolution Neural Networks (CNNs), a class of Deep Neural Networks, are the most powerful in image processing tasks. CNNs consist of layers of small computational units that process visual information hierarchically in a feed-forward manner. Each layer, a collection of image filters, extracts certain features from the input image. The output of a given layer consists of so-called feature maps.

2.2 Model and details

2.2.1 Content Representation

When Convolutional Neural Networks are trained on object recognition, they develop a representation of the image that makes object information increasingly explicit along the processing hierarchy. Therefore, along the processing

hierarchy of the network, the input image is transformed into representations that increasingly care about the actual content of the image compared to its detailed pixel values. The paper by Gatys et al. shows that responses obtained from higher levels in the network capture the high-level content in terms of objects and their arrangement as opposed to responses from lower levels. Therefore the feature responses from higher levels of a network are referred to as the *content representation* of the image.

Let $\hat{p}$ be the original input image that is encoded in each layer of the CNN by the filter responses of that image. Let, there be N_l distinct filters at layer l. It will have N_l feature maps each of size M_l, where M_l is the height $\times$ width of the feature map. Let the responses of layer l be stored in a matrix $F^l \in R^{(N_l \times M_l)}$ where $F^l_{i,j}$ is the activation of the i^{th} filter at position j in layer l. Let $\hat{p}$ and $\hat{x}$ be the original image and the generated image and P^l and F^l be their respective feature representation at layer l. The squared-error loss between the two feature representations is defined as:

$$\mathcal{L}_{content} = \frac{1}{2} \sum_{i,j} (F^l_{i,j} - P^l_{i,j})^2 \tag{1}$$

While F^l and P^l encode the *content representation* of the images at layer l. $\mathcal{L}_{content}$ represents the content loss between the original and the generated image. The derivative of this loss with respect to the activation in layer l equals:

$$\frac{\partial \mathcal{L}_{content}}{\partial f^l_{ij}} = \begin{cases} (F^l - P^l)_{i,j}, & \text{if } F^l_{ij} > 0 \\ 0, & \text{if } F^l_{ij} < 0 \end{cases} \tag{2}$$

from which the gradient with respect to the image $\hat{x}$ can be calculated using standard error back-propagation.

2.2.2 Style Representation

To obtain the representation of style, a feature space originally designed to capture texture information is used. It consists of the correlations between the filter responses over the spatial extent of the feature maps. By including the feature correlations of multiple layers, a multi-scale representation of the input image which captures texture information but not the global arrangement is obtained. This multi-scale representation is referred to as the *style representation*. The feature correlations are given by the Gram Matrix $G^l \in R^{(N_l \times N_l)}$, where G^l_{ij} is the inner product between the vectorized feature maps i and j in layer l. The elements of the Gram Matrix G^l_{ij} are defined as:

$$G^l_{ij} = \sum_k F^l_{ik} F^l_{jk} \tag{3}$$

where F^l_{ij} is the activation of the i^{th} filter at position j in layer l of the generated image.

To generate a texture that matches the style of a given image, we use gradient descent from a white noise image to find another image that matches the style representation of the original image. This is done by minimizing the mean-squared distance between the entries of the Gram matrix from the original image and the Gram matrix of the image to be generated. Let $\vec{a}$ and $\vec{x}$ be the original image and the generate image and A^l and G^l be their respective style representations in layer l. The contribution of that layer to the total loss is then defined as:

$$E_l = \frac{1}{(4N_l^2 M_l^2)} \sum_{i,j} (G^l_{ij} - A^l_{ij})^2 \tag{4}$$

and the total loss is given by:

$$\mathcal{L}_{style}(\vec{a}, \vec{x}) = \sum_{(l=0)}^{L} (w_l E_l) \tag{5}$$

where w_l are weighting factors of the contribution of each layer to the total loss. The derivate of E_l with respect to the activations in layer l can be calculated analytically as:

$$\frac{\partial E_l}{\partial f^l_{ij}} = \begin{cases} \frac{1}{N_l^2 M_l^2} ((F^l)^T (G^l - A^l))_{ij} & \text{if } F^l_{ij} > 0 \\ 0, & \text{if } F^l_{ij} < 0 \end{cases} \tag{6}$$

The gradients of E_l with respect to the activations in lower layers of the network can be readily computed using standard error back-propagation.

2.3 Objective

The objective here is to perform "style transfer". "Style transfer" in the paper is defined as being able to generate an image that captures the content of one image and style of another image. Here we will be using photographs and posters as the content images and paintings as the style images. To generate the images that mix the content of a photograph with the style of a painting we jointly minimise the distance of a white noise image from the content representation of the photograph in one layer of the network and the style representation of the painting in a number of layers of the CNN. So, let $\vec{p}$ be the photograph and $\vec{a}$ be the artwork(painting) and $\vec{x}$ be the image generated, then the overall loss function becomes:

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{a}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x})) \tag{7}$$

where α and β are the weighting factors of the content and style reconstructions respectively.

2.4 Experimental Setup

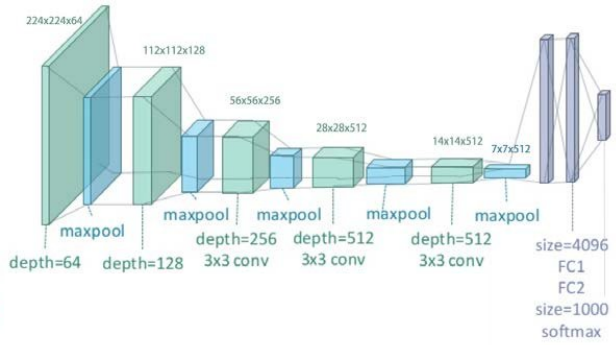

Figure 1: VGG-19 Architecture

For this experiment the pre-trained VGG-Network, a Convolution Neural Network that rivals human performance on a common visual object recognition benchmark task is used. The feature space provided by the 16 convolutions and 5 pooling layers of the 19 layer VGG-Network (Figure 1) are used. A pre-trained VGG network, available in pytorch, is used to synthesize new images that capture the content and style of 2 different images. For this experiment we need 3 images: A *content image*, a *style image* and an *input image*. The *input image* is updated by iterating over the network to minimize the $\mathcal{L}_{total}$ where the $\mathcal{L}_{content}$ is computed against the *content image* and the $\mathcal{L}_{style}$ is computed against the *style image*. For the images shown in Figure 2 the content image is reused as the input image. For the images shown we matched the content representation of layer 'conv_5' and the style representation of layers 'conv_1', 'conv_2', 'conv_3', 'conv_4', 'conv_5' ($w_l = \frac{1}{5}$ in those layers, $w_l = 0$ in all other layers). The $\frac{\alpha}{\beta}$ ration is set to 10^{-7}. Also, we replaced maxpooling layers with avgpooling layres as it was observed that using avgpooling resulted in more appealing images. Number of iterations performed was 300.

To study the effects of variation of $\frac{\alpha}{\beta}$ ratio on the generated image the sets of images shown in Figure 3 were generated. In the Figure 3 grid each column represents a fixed $\frac{\alpha}{\beta}$ ratio and each row represents a fixed style representation. We can see that as we decrease the $\frac{\alpha}{\beta}$ ratio the generated images matched the abstract style of the style image and we can hardly make out any content from the content image. Similarly, as we go down the rows we see that the more layers we use to extract the style representation the richer style is represented in the generated image. For example, we can see that using more layers for style representation captures the red stripes pattern in the sky of the painting and is reflected in the generated image.

In the Figure 4, we can see the variation of the total loss $\mathcal{L}_{total}$ over the iterations while generating a style transferred image. It is evident that the loss decreases steady as we iterate over the network.

Figure 2: Images that combine the content of a original images with the style of several well-known artworks. The images were created by finding an image that simultaneously matches the content representation of the original images and the style representation of the artwork. The images that provided the content representation are shown in the first column and the paintings that provided the style representation are shown in the top row. The paintings from left to right are *The Starry Night* by Vincent van Gough,1889 , *Der Schrei* by Edvard Munch, 1893, Femme nue assise by Pablo Picasso, 1910, *The Shipwreck of the Minotaur* by J.M.W.Turner, 1805.

3 Image-to-Image translation using cyclic-GAN

3.1 Introduction

The objective here is to achieve Image-to-Image translation using Cycle-GAN. i.e. converting an image from one representation to another. The proposed technique tries to solve the problem without the need of paired images in the training data i.e. an algorithm that can learn to translate between domains without paired input-output examples. We achieve Style transfer using this domain to domain mapping algorithm. The assumption here is that there is some underlying relationship between the domains.

Although the supervision in the form of paired examples is lost, supervision at the level of sets can still be exploited: given one set of images in domain X and a different set in domain Y . The objective here is to train a mapping $G : X \rightarrow Y$ such that the output $\hat{y} = G(x), x \in X$, is indistinguishable from images $y \in Y$ by an adversary trained to classify $\hat{y}$ apart from y. The optimal G thereby translates the domain X to a domain $\hat{Y}$ distributed identically to Y . However, such a translation does not guarantee that an individual input x and output y are paired up in a meaningful way. The key idea is that translation should be "cyclic consistent", i.e. for example if we translate a sentence from English to Hindi, and then translate it back from Hindi to English, we should get back the same original English sentence. Mathematically, if we have a translator $G : X \rightarrow Y$ and another translator $F : Y \rightarrow X$, then G and F should be inverses of each other, and both mappings should be bijections. This structural assumptions is incorporated by training both the mapping G and F simultaneously, and adding a cycle consistency loss that encourages $F(G(x)) \approx x$ and $G(F(y)) \approx y$.

3.2 Formulation

The main aim is to convert a Monet's painting to a real image and also, to convert a real image to various styles such as Monet's, Cezanne, Ukiyoe, Vangogh. The objective is to learn mapping functions between two domains X and Y given training samples $\{x_i\}_{i=1}^{N}$ where $x_i \in X$ and $\{y_j\}_{j=1}^{M}$ where $y_j \in Y$. We denote the data distribution as $x \sim p_{data}(x)$ and $y \sim p_{data}(y)$. As explained in the previous section the model has 2 mappings $G : X \rightarrow Y$and $F : Y \rightarrow X$. In addition to this, two adversarial discriminators D_X and D_Y are introduced, where D_X aims to distinguish between images $\{x\}$ and translated images $\{F(y)\}$; in the same way, D_Y aims to discriminate between $\{y\}$ and $\{G(x)\}$. To conclude, The final objective contains two types of terms: adversarial losses for matching the

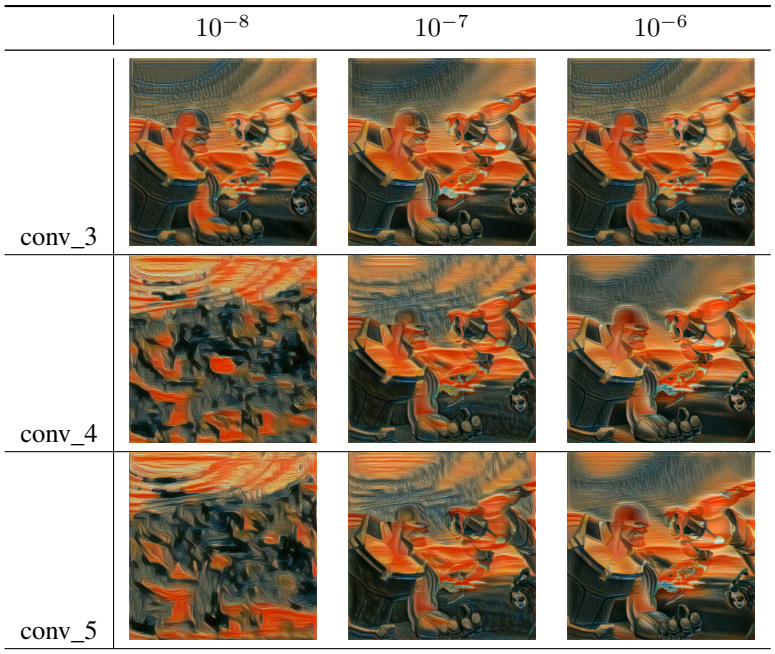

Figure 3: Images obtained by combining the representaion from the original 'Avengers' image and the artwork *Der Schrei* by Edvard Munch, 1893. Along each row we can see the images generated by the variation of the relative weightings between the content and the style reconstruction. The number above each column indicates the ratio $\frac{\alpha}{\beta}$ between the emphasis on matching the content of the image and the style of the artwork. Along each column we see the images generated by matching the style representation of increasing subsets of the CNN layers. The images in the first, second and third rows match the style representation in 'conv_1' to 'conv_3', 'conv_1' to 'conv_4' and 'conv_1' to 'conv_5' respectively.

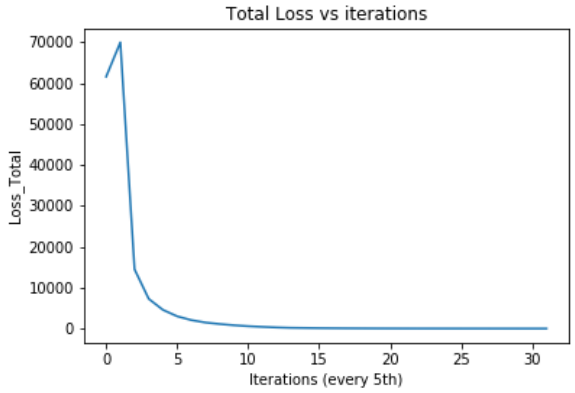

Figure 4: Shows the variation in total loss $\mathcal{L}_{total}$ while generating an image using the 'Avenger' image and the artwork *Der Schrei* by Edvard Munch, 1893. $\frac{\alpha}{\beta}$ used is 10^{-7} and the style representation from layers 'conv_1' to 'conv_5' are matched.

distribution of generated images to the data distribution in the target domain; and cycle consistency losses to prevent the learned mappings G and F from contradicting each other.

3.2.1 Adversarial Loss

The adversarial loss for mapping the function $G : X \rightarrow Y$ and it discriminator D_Y can be expressed as follows :

$$L_{GAN}(G, D_Y, X, Y) = \mathbb{E}\left[y \approx p_{data}(y)\right]\left[\log D_Y(y)\right] + \mathbb{E}\left[x \approx p_{data}(x)\right]\left[\log D_Y(G(x))\right] \tag{8}$$

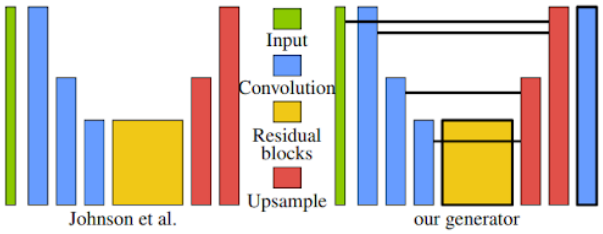

Figure 5: Comparison of architecture(Generator) of Johnson et al. and proposed model.

The generator G tries to generate images $G(x)$ that look similar to images from domain Y , while D_Y aims to distinguish between translated samples $G(x)$ and real samples y. G aims to minimize this objective against an adversary D that tries to maximize it, i.e., $\min_G \max_{D_Y} L_{\text{GAN}}(G, D_Y, X, Y)$. Similarly, for a function $F : Y \rightarrow X$ and its discriminator D_X, the objective becomes $\min_F \max_{D_X} L_{\text{GAN}}(F, D_X, Y, X)$.

3.2.2 Cycle consistency Loss

To induce the cycle consistency in the in the objective function, i.e. for each image x from domain X, the image translation cycle should be able to bring x back to the original image, i.e., $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$. This is referred to as forward cycle consistency. Similarly, for each image y from domain Y , G and F should also satisfy backward cycle consistency: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$. Therefore the final cycle consistency loss can be depicted as follows :

$$L_{cyc}(G, F) = \mathbb{E}\left[y \approx p_{data}(y)\right]\left[\|\log D_Y(y)\|_1\right] + \mathbb{E}\left[x \approx p_{data}(x)\right]\left[\|\log D_Y(G(x))\|_1\right] \tag{9}$$

In practice, 2 additional parameters are used namely, λ_A and λ_B which are multiplied with forward cycle consistency and backward cycle consistency respectively. These values $(\lambda_A and \lambda_B)$ reflect the amount of importance given to forward cycle consistency and backward cycle consistency respectively.

3.2.3 Final Objective function

The final objective function is a combination of Adversarial Loss and Cycle consistency loss, which can be defined as follows :

$$L(G, F, D_X, D_Y) = L_{\text{GAN}}(G, D_Y, X, Y) + L_{\text{GAN}}(F, D_X, Y, X) + \lambda L_{\text{cyc}}(G, F) \tag{10}$$

Where λ controls the relative importance of the two objectives.

3.3 Implementation details

3.3.1 Architecture

Inspired by the Generative network proposed by Johnson et al., the proposed cyclic GAN model uses a 9 layered residual network. The first block of the layers has reflection padding followed by convolution and ReLU activation. Following this, the second block of layers is composed of 2 back-to-back down sampling layers(blue coloured bars in Figure 5). In the third block of layers, 9 ResNet blocks are used, with each residual block comprising of 2 convolution layers(yellow coloured bars in Figure 5). And finally, in the fourth block of the network 2 back-to-back upsampling layers are(red coloured bars in Figure 5) used which is followed by one last convolution layer and $\tanh$ activation function(last blue coloured bar in Figure 5). The discriminator network is inspired by the works of [2], where a 5-layered Convolution network with the input size of 70 x 70 is used to classify whether 70 x 70 overlapping image patches are real or fake. The exact same structure is used for the design of discriminator in our case as well.

3.3.2 Training details

The model is trained for 200 epochs with a batch size of 2 with Adam optimizer. The training dataset has 1337 Monet's painting(domain A) and 6287 Real life pictures(domain B). The objective of the GAN model - negative log likelihood is replaced by least square loss(giving rise to LSGAN). As proved by [3] LSGAN are able to generate higher quality images than regular GANs. And Also, LSGAN tend to be more stable than regular GANs during training. The λ value for all the experiments is set to 10. A lesser value of λ would make the model more stochastic and a value of 0 completely eliminates the cyclic loss from the objective function. The learning rate is initially set to 0.0002 and is linearly reduced after the first 100 epochs. Also, during training we divide the objective by 2 while optimizing D, which

slows down the rate at which D learns, relative to the rate of G. Weights are initialized from a Gaussian distribution $\mathcal{N}(0, 0.02)$.

$Real$: 14.3% $Fake$: 85.7%

$Real$: 60% $Fake$: 40%

$Real$: 50% $Fake$: 50%

$Real$: 57.1% $Fake$: 42.9%

$Real$: 7.1% $Fake$: 92.1%

$Real$: 42.9% $Fake$: 57.1%

Figure 6: Results of the survey

3.4 Results

The main idea of this project is to convert a Monet's painting to a Real life picture and to convert real picture to various styles such as Monet's, Cezanne, Ukiyoe, Vangogh using cyclic GAN model. Once the model was trained with the above mentioned training details, the model was presented with wide variety of testing samples(Monet's paintings) and the resulting output images were expected to resemble real world pictures.

To judge the quality of the images obtained by our model we created a google survey form `https://forms.gle/52d41BmoVCZZUkbV7`, which had around 10 pictures(output of cyclic GAN model) and the reviewer had to make a prediction on whether the corresponding image is REAL(i.e. A actual real picture taken from a camera) or FAKE (i.e. the Image generated by GAN model). We got 14 responses for this and the results of this survey is presented in Figure 6.

The second aim of this project is to convert a real image to various styles such as Monet's, Cezanne, Ukiyoe, Vangogh using cyclic GAN. The results of this can be seen in Figure 7. Every image has 2 parameters 'Real' 'Fake'. 'Real' represents the percentage of people who felt this was a real image and 'Fake' represents the percentage of people who felt the image was machine synthesized(Note : The above responses were recorded from 14 people only). It can be verified from Figure 7 that the style of the artist is captured by the cyclic-GAN than merely capturing the style of a individual painting. I.e. all real images do not take single Monet's painting to learn the style, instead the real images learn from 100s/1000s of Monet's painting trying to learn the style of the artist itself. This very nature of learning the style of the artist itself is what makes cycle-GAN outperform other primitive models.

The training loss plot of Generator and discriminator for the first 1500 iterations can be seen in the figure 8.

4 Discussion

4.1 Learning

The Gatys et al. paper shows that CNNs capture can independently capture the content and style representations of an image. It presents a technique extract the *content representation* and *style representation* of any given image using responses of different layers in a technique. Further it proposes a technique for generating new artistic images by balancing the style loss(L_{style}) and content loss($L_{content}$) using the α/β ratio. In this paper we saw how the variation of

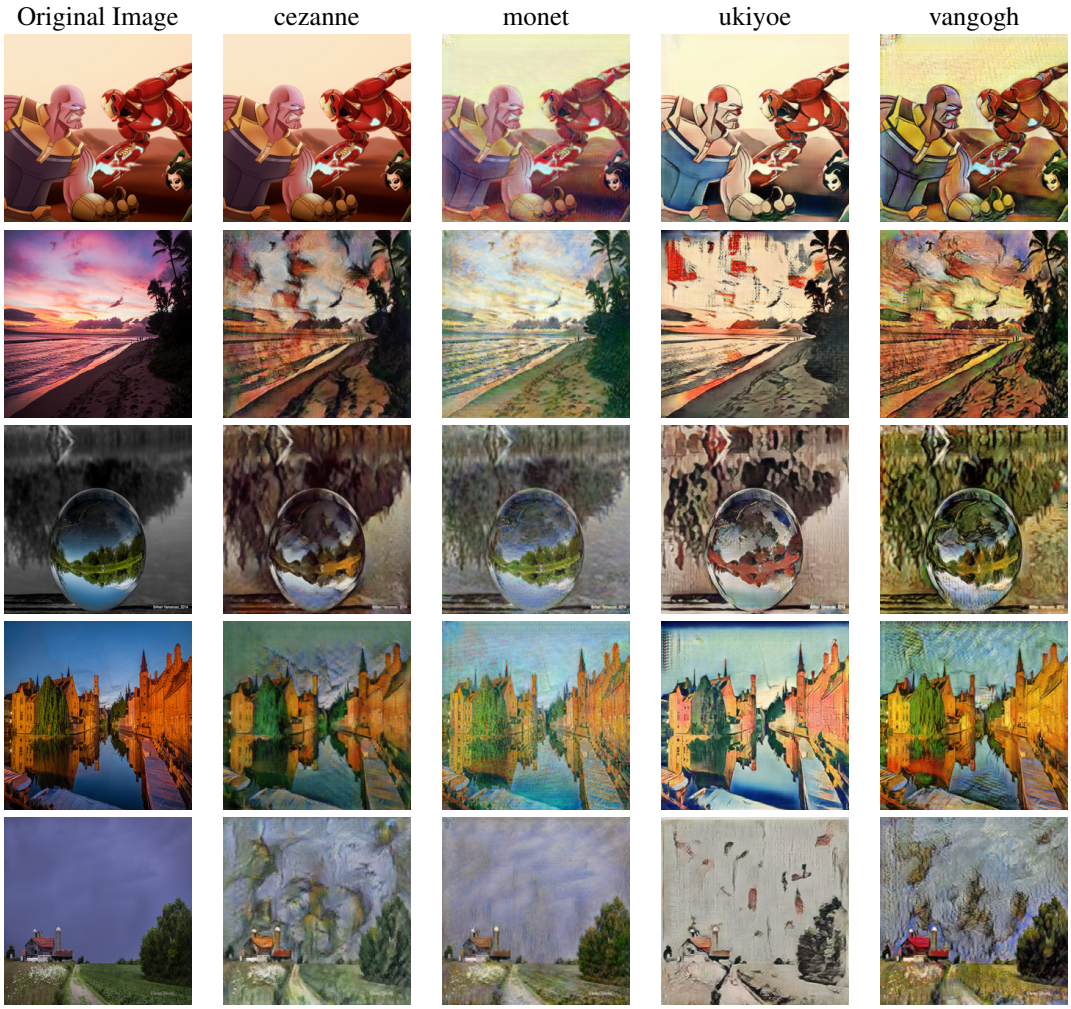

Figure 7: Style transfer of real image to various styles such as Monet's, Cezanne, Ukiyoe, Vangogh using cyclic GAN.

the α/β ration affects the resultant image. We also explored the effects of varying the style representation by extracting the *style representation* from different layers in the network.

In cyclic GAN, all real images do not take single Monet's painting to learn the style, instead the real images learn from 100s/1000s of Monet's painting trying to understand the style of the artist itself. This very nature of learning the style of the artist itself is what makes cycle-GAN outperform other primitive models. The adversarial losses alone cannot guarantee that the learned function can map an individual input to desired output. To reduce the space of possible mapping functions, the learned mapping function should be cycle-consistent. Hence adversarial loss and cycle-consistency loss together are used as the objective of cycle-GAN model.

4.2 Difficulties

In the task of style transfer we could generate visually appealing images only when using paintings as style images. The style image needed to have an abstract definition of style where the style is independent of the content. Using images such as 'anime posters' where the style of the image is dependent on the content of the image (the style of the eyes, the style of the hair, etc) the results were unappealing. This is most likely a result of the primitive mechanism used to capture the *style representation* of an image.

In the task of translating Monet's painting to real image, the quality of the output image was not very close to the real image. We tried using other architectures for generator network but still the quality did not show any significant

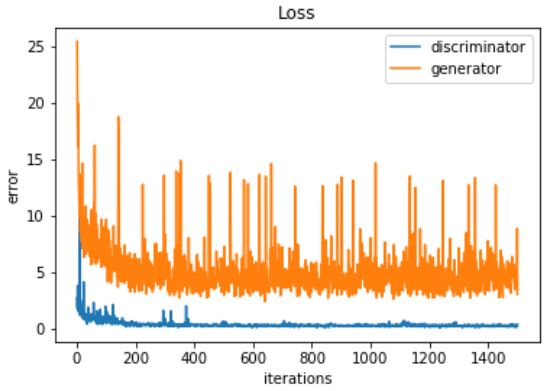

Figure 8: training loss plot of Generator and discriminator for the first 1500 iterations.

improvement. We could only collect 7500 images for training. Probably, with the increase in the size of training data and the computational resources the model can perform relatively better.

4.3 Improvements

For the Gatys et al. style transfer model, we used a VGG-19 network to capture the content and style representation.If sufficient GPU is available we can explore extracting the content and style information from a deeper VGG network and use a richer style representation to capture more information resulting in appealing resultant imagees.

For cyclic GAN model we have currently used 9 residual block architecture. If sufficient GPU resource is available then more complex architectures can be trained to obtain better results. Also, we have used a training data size of 7500 samples. Collecting more data samples would help for the better performance of GAN model. We can explore Ensamble architectures for style transfer model. We can consider the possibility of combining perceptual loss based CNN model and the cyclic GAN model to achieve style transfer of a given input image.

References

[1] Leon A. Gatys, Alexander S. Ecker and Matthias Bethge A Neural Algorithm of Artistic Style In *arXivpreprint arXiv:1508.06576 (2015)*

[2] Justin Johnson, Alexandre Alahi, Li Fei-Fei Perceptual Losses for Real-Time Style Transfer and Super-Resolution In *arXiv:1603.08155*

5 Link to Git repository

The URL of the Git repository is as follows : `https://github.com/ykakarap/style_transfer`

Gatys model and Cycle-GAN model are available in the above Git link. The repository contains two demo folder namely, 'Demonstration' - which is the demo of Cyclic-GAN model and 'gatys-demo' - which is the demo of Gatys et al. implementation. The repository contains all the code used for the training, testing and visualization of Gatys and Cycle-GAN models.