Artistic Style Transfer using Deep Neural Networks

S.M. Haider Ali Shuvo
Student ID: 0422062512
Grad Student,Dept of EEE
Bangladesh University of Engineering and Technology

***Abstract*—From artistic drawing to fraud detection deep neural networks have been immensely successful in most of the learning fields. In this project paper, we exploit neural style transfer which is a deep network based algorithm to achieve artistic style in any desired image keeping the contents similar. Classical statistic based methods as well as current Deep network based methods are discussed here with their relative pros and cons. As a mother paper we implement and discuss in details abput "Image Style Transfer Using Convolutional Neural Networks, Gatys et. al. CVPR 2016" as this paper captures the central theme of deep network based style transfer algorithms and new algorithms are mostly tweaks of ideas presented in that paper.**

I. Introduction

Style transfer is the process of mixing two images - one which we call style image and other which we call content image. The output image should preserve the contents of the content image but overlaid in the texture and artistic style of the style image. The process is pictorially depicted in Fig. 1. The use of style transfer is in image editing, manipulation, enhancement etc.

Fig. 1. Example of style transfer.

Deep neural networks are bunch of affine layers with nonlinearities inserted between them. The intermediate layers of deep convolutional networks can be thought of extracting features. When Deep neural Networks features are used to generate style transferred image we call it neural style transfer. Recently, Neural methods have been the state of art in style transfer problems. They produce realistic image while having more appealing visual qualities. In this project, a CNN based algorithm to generate style transfer is implemented using pytorch.

The organization of this paper is as follows - In section II we give a brief literature review, In section III we describe the our implemented method and In section IV the results are shown and comments are made.

II. Literature Review

A. Classical approaches

Artistic style transfer is a long hunted problem even before the popularity of deep neural networks. Some of this classical methods without CNN's are mentioned here. Stroke-Based Rendering(SBR) is one of the oldest methods of style transfer. It is the process of synthesizing artwork by placing virtual strokes on a digital campus. The process starts from a source photo and then incrementally places strokes to match the photo and then rendered to appear in artistic style. Region-Based Rendering(RBR) uses the shape of regions to guide stroke placement. This allows local spatial control over the style. Example-Based Rendering(EBR) is based on framework named image analogy. In the training phase pairs of source image and target stylised images are provided and then the model learns mapping from unstylised to stylised images. Besides these, there are some classical filtering techniques to incorporate style using particular kernels. The Non-CNN classical approaches have limitation in learning power, robustness, flexibility and diversity.

B. Neural Style Transfer

Upon the evolution of deep learning based models and their success in image classification, effort in applying neural approaches to other classes of computer vision problems took interest. In that trend, people started applying convolutional neural networks to solve style transfer problems. The general theme in NST is to have an indicator of style and texture from deep learning features and use optimization to achieve output image. The approaches can be loosely classified in following four criterions.

1)Image Optimization Based Methods - Here the idea is to have a target content representation and a target style representation. Then the CNN features are reversed to achieve an image that matchies the desired content and style. Gradient descent is applied on image pixels rather than on model and the model is usually pretrained on some large datasets that has learnt about images well enough.

2)Model Optimization Based Methods - The image optimization based methods are slow since for every content and style pairs, we need to go through an optimization process consisting of forward pass and backward pass in a deep networks. To achieve fast style transfer, modifications have been made that learns rather a dedicated neural network on train time to give stylised image directly in one pass in test

time. This real time style transfer models can be style specific (trained to produce one specific style) or can be universal (trained to produce output given any arbitrary style and content image).

3)GAN Based Methods - Generative Adversarial Networks can generate state of art realistic images. Their methodology requires a generator and a discriminator. While the generator learns mapping from feature space to image space, the discriminator classifies the image as real or fake. Generator learns to fool the discriminator by creating realistic images. StyleGAN is a style based GAN architecture which learns unsupervised high level separation of different image attributes and enables different control on image synthesis.

4)AutoEncoder Based Methods - Here image is encoded into independent style and content components. For style transfer, style features of original image is replaced by latent style features of desired style image. This architecture is called Swapping Autoencoder.

III. Implemented Method

For this project we implement the paper "Image Style Transfer Using Convolutional Neural Networks", Gatys et. al. CVPR 2016 and the followup paper "Controlling Perceptual Factors in Neural Style Transfer", Gatys el. al. CVPR 2017. Choosing this methods to implement over more recent methods is due to the fact most of the recent methods have inspiration from this paper with some tweaks and implementing this the mother papers provide maximum insight about neural style transfer. It is an image optimization based method.

Since its a feature inversion method with optimization on pixel level, the key question is which features to match for correct content and style? An usual CNN is nothing but a cascade of convolutional layers with nonlinarities inserted between and a Fully Connectd network at the end for the classification. A CNN can be thought as a combination of feature extractor consisting of conv layers followed by a classifier with fully connected network. The output from the cascade of conv layer are high level features having latent description of image.

We want to preserve the content of the content image. A good indicator of content is the high level feature outputs from the convolutional layers. The CNN should be pretrained on a large dataset such that it has knowledge of vision in general to extract meaningful features. Then the content representation is the raw features when the content image is propagated through the network. The features can be taken from any layer in principle features from l^{th} layer be denoted as C^l.

It is desired that the output image should have artistic style from the style image. It is in general observed that correlation in image pixel is a good descriptor of textures. In this paper rather correlation in feature space is proposed as the indicator of style. In principle, second order statistics captures style good enough for most cases. The gram matrix used here is an approximation of covariance matrix. Let the features of style image from l^{th} layer be denoted as S^l, where $S^l \in R^{C\times H\times W}$. The spatial dimentions are flattened to get a matrix of $R^{C\times H.W}$. The dot product between features of each channel is called the images gram matrix which is denoted as $G^l = S^l.S^{l^T}, G^l \in R^{C\times C}$. The gram matrix is an approximation of covariance matrix and is used as a descriptor of style and texture in an image.

With the descriptors of content and style in hand, we are now ready to describe style transfer algorithm. A random image is initated at first and propagated through the pretrained CNN. Let the features and gram matrix of the output image at l^{th} layer be denoted as CO^l and GO^l. The content is matched usually by matching features to only one top level layer whereas style is matched by matching gram matrix in several stages from lower and higher level layers since each layer captures style from a different scale. Let the layer matched for content be denoted as L_c and the set of layers matched for style be denoted as L_s. The content loss is then the mean square loss between features of output image and features of content image in the L_c layer i.e. $\text{Loss}_c = \sum_{l=L_c} (C_l - CO_l)^2$. And the style loss is the mean square loss between gram matrix of output image and gram matrix of style images in the layers from the set L_s i.e. $\text{Loss}_s = \sum_{l\in L_s} (G_l - GO_l)^2$. The total loss is then $Loss_{tot} = Loss_c + \alpha \times Loss_s$.

It turns out just training with those two losses doesn't ensure a smooth image. To ensure realistic touch and smoothness on image, an extra total variation loss is added which penalizes difference between immediate neighbourhood pixels. If x is the output image then the total variation loss is

$$Loss_{tv} = \sum_{c=1}^{3}\sum_{i=1}^{H-1}\sum_{j=1}^{W}(x_{i+1,j,c} - x_{i,j,c})^2 + \sum_{c=1}^{3}\sum_{i=1}^{H}\sum_{j=1}^{W-1}(x_{i,j+1,c} - x_{i,j,c})^2$$

Then given a content image and a style image, the algorithm first generates a random image and computes the losses. Then gradient is propagated to the pixel level and update is made to the image while keeping the network fixed by turning it's gradient update off. The image changes to match the desired content and style with smoothness over time. This usually requires some good number of iterations before we start to see fruitful results. In some cases, instead of initiating from random image initialization is done to content image. The weights of the losses are tuned by random search usually. Style loss from earlier layers are given higher weights since they capture much fine grained textures.

IV. Results

The Fig. 2 and 3 shows the process when tubingen image and starry night is taken as content and style respectively. The features from layer 3 of squeezenet is used for content matching whereas features from layer 1,4,6,7 is used for gram matrix matching. Higher weights is used to lower layers in gram matrix matching since they capture more pinpoint fine grained texture statistics. The process starts with random noise sampled from gaussian distribution as output image. At each

Fig. 2. Here the tubingen image is taken as content and starry night as style. The output image is initiated to random noise initially.

Fig. 3. The evolution of output image as iteration proceeds. Output image slows catches the desired content and style.

iteration, features of output image is matched to features of content image and gram matrix of output image is matched to gram matrix of style image. As the number of updates increases, the output image slowly catches the desired content and style. As a result we see after 100 iteration we start to see desired content in bluish starry night style and texture strokes.

One fact that gets evident is the subjectivity in the process. It is not evidently clear when to stop or what is a good output image. Like many computer vision problems, here goodness is subjective and depends on the user at the end. Also the weights of the losses and learning rate requires good amount of hyper-tuning to get good visual results.

Even though the simple method is quite powerful and insightful for the problem, it has few shortcomings. One issue is how to achieve more control over the style transfer algorithm. For example the style image might have different styles at different style location and we are interested in one of them. Another issue is the large inference time in this algorithm. Whole computational burden is one the inference time and for every pair of content and style image whole process has to be repeated. This factors of having perceptual control and enabling fast style transfer is introduced in later section.

V. Spatial Control Using Guided Gram Matrix

The method described till now sort of transfers the overall style of the style image into the content image where the fundamental assumption being that the texture of the style image is same everywhere. However, the style image can have different textures at different spatial locations in which case the elementary method will sort of transfer the mixture of the textures which is unwanted. The key solution then is to control the mapping i.e. texture from which locations in style image should be transferred to which locations in content image. This can be achieved using guided gram matrix. We illustrate the concept below.

The general problem can be stated below. We have a content image which we want to stylize differently in different spatial locations. There are R style images and the style mapping is described by R spatial guidance channels for both content and style image. Elaborately, the guidance channel acts as a mask and Where the r^{th} content guidance channel is high those spatial region should inherit texture from the r^{th} style image where its corresponding guidance channel is high valued. The guidance channels are given beforehand from user specifications.

Let the features of r^{th} style image from l^{th} layer be denoted as $S_r^l \in R^{C \times H.W}$. The flattened r^{th} guidance channel for that layer be denoted as $T_r^l \in R^{H.W}$. Then the guided feature map

is the broadcasted schur product between the original feature map and guidance feature map i.e. $S^l_{gr} = S^l_r \circ T^l_r$. The guided gram matrix is then usual gram matrix computed on the guided feature map i.e. $G^l_r = S^l_{gr}.S^{l^T}_{gr}, G^l_r \in R^{C\times C}$. The working of the guidance channel be intuitive explained. Assume that the hard guidance channel is either 0 or 1. Then, where the guidance channels are zero guidance features maps value at those locations are zero and hence the gram matrix isn't contributed by those spatial locations. The gram matrix then is only affected by the spatial positions where the guidance channel is one i.e. it captures the local texture of style image where the guidance channel is high which was our original goal.

Among the three loss term, only the style loss will change. Now it will consist of R separate terms. Each of them is a squared error loss between guided gram matrix of content image and guided gram matrix of corresponding style image. As a result, the transfer of style to content image will be spatially controlled and several styles can be combined. Fig. 4. and Fig. 5. shows effect and comparison between uncontrolled and spatially controlled NST.

Fig. 4. Content image and the two style images. The inlet shows the hard guidance channel indicating style mapping.

Fig. 5. In first image uncontrolled NST is done with respect first style image. We can see that the ground artifacts appear in the sky of the final image which is undesired. This happens because unlike previous cases now the style image has different textures at different locations. The second image shows spatialled controlled NST with the guidance channels shown in inlet of Fig. 4. The final image achieves the ground style style I and sky style from style II.

VI. Fast Style Transfer

The methods presented so far have a large inference load in the sense that for every pair of content and style image the algorithm has to kick-start again from random guess. Several methods have been proposed that tries to put the burden on train time instead and allow fast style transfer in test time. Here we present the fundamental concept behind fast style transfer.

The central idea of fast style transfer is to train a separate convolutional neural network to produce the style transferred image. Johnson et al. in 2016 proposed an architecture that can produce fast style transfer for any specific style in inference time. The architecture as depicted in Fig. 4. allows an additional image transform network. The image transform network is a fully convolutional neural network that has downsampling followed by upsampling and can be treated as an image to image transformation. The content image is passed as input to the fully convolutional network and the output is then propagated through a pretrained CNN along which features and gram matrices are matched like before. The key difference is that instead of having gradient descent on pixel level for every pair of style and content combination we allow a trainable neural network which learns to produce style transferred image given any content image. At test time only the image transform network is needed and one forward pass through it will be able produce desired style transferred image.

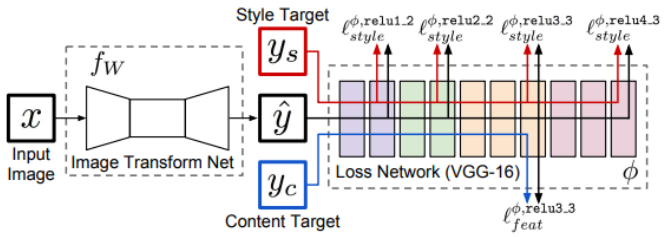

Fig. 6. Fast style transfer network. At train time the image transform network which is a fully convolutional network is trained to produce style transferred image. At inference time only the image transform network is needed and no iterative gradient descent on pixel level is required. Hence style transfer can be achieved with one forward pass.

VII. Conclusion

In this project, the literature of style transfer have been reviewed briefly. A deep learning based Style transfer based on convolutional neural network have been implemented and analyzed. The shortcomings and current state of the art modifications and research trend on neural style transfer is also presented.