Neural Style Transfer

Ran Lu
Department of Statistics
Columbia University
New York, NY 10027
rl3077@columbia.edu
Group 22

Jiaxiang Wang
Department of Statistics
Columbia University
New York, NY 10027
jw3864@columbia.edu
Group 22

Kevin Cho
Department of Statistics
Columbia University
New York, NY 10027
kc3313@columbia.edu
Group 22

Zhenyu Jia
Department of Statistics
Columbia University
New York, NY 10027
zj2268@columbia.edu
Group 22

Abstract

Neural Style Transfer algorithms allow us to combine the content of an image with the style of another image. More specifically, utilizing Convolutional Neural Networks for neural style transfer enabled wider usage of neural style transfer. Here we study the performance of one of the most promising algorithms, ***A Neural Algorithm of Artistic Style***. Our results illustrate how each element of the algorithm is essential for successful transfer learning and show how different hyperparameters affect the performance. We also compare how building on from different pre-trained networks leads to different transformed images.

1 Introduction

Deep learning has been applied to many real-world applications. The Go community and many people around the globe were shocked when AlphaGo beat two professional Go players, Fan Hui and Lee Sedol. Zastrow [12] This was a momentous moment for the deep learning community as people realized the potential of profound impact deep learning or artificial intelligence can have on our lives. bbc [2] More specifically, Convolutional Neural Network (CNN) has been a predominant force in solving image classification problems. ImageNet Large Scale Visual Recognition Challenge (ILSVRC) attracted many deep learning researchers and models like ResNet, VGG-16 and VGG-19 that showed outstanding performance were developed through the annual challenge. Russakovsky et al. [9] In 2015, Researchers in Microsoft Research, Kaiming He et al., created a model called ResNet and won ILSVRC 2015. ResNet decreased the classification error to 3.57%. He et al. [5] This model is remarkable as it even exceeded human abilities. Markoff [8] Once considered a field that is unsolvable, computer vision gained promising improvement as researchers developed state of the art methods.

Building on from computer vision, our goal is to utilize CNN to combine two images. More specifically, we want to transform a content image by blending the style of a style image. Gatys et al. [3] proposed a method called A Neural Algorithm of Artistic Style that synthesizes two images, content image and style image. The general approach of performing the algorithm is the followings:

1. Extract features of both content and style image using a pre-trained CNN model
2. Define the representation differences using a loss function

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

3. Perform gradient descent and apply gradient on the input image to achieve the blend of both style and content.

Because it successfully synthesizes texture to content image, we follow the general procedure that is proposed by Gatys et al. [3], but we will implement different hyperparameters to evaluate the performance. We also compare how algorithms proposed by Gatys et al. [3] and Ghiasi et al. [4] are different. The algorithm in Gatys et al. [3] and its modified methods have been implemented in real-life applications. Photo editors like Prisma, Deep Art Effects use the Neural Algorithm of Artistic Style. More advanced algorithms are implemented in Virtual Reality graphics and video games. Our objective is to combine two images, hence we will focus on the Neural Algorithm of Artistic Style. To assess the performance of the algorithm, we use two content and two style images to have four different combinations of transformed images. Two content images are pictures of Columbia Low library and the Empire States Building in New York City, and the two style images are acrylic painting of fire and oil paint of different colors (Korovilas [7] and Yavnik [11]). Regardless of size of the input images, we use 512×512 pixels for the images.

2 Methods

2.1 Neural Style Transfer

The neural style transfer extracts the content and style representation features with CNN layers. The two images have similar content if their high-level features are similar, and two images are similar in style if their low-level features are similar Ghiasi et al. [4]. Let F_{ij}^l be the activations of l^{th} layer at position j on filter i, then the content loss between generated image and target image can be defined as

$$L_{content}(v, w, l) = \frac{1}{2} \sum_{i,j} (F_{ij}^l - P_{ij}^l)^2$$

where F_{ij}^l and P_{ij}^l are the response matrices respectively. The style loss is the weighted sum of the loss of each style layer. For each layer, the style loss is defined as

$$L_{slayer} = \frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l(v) - G_{ij}^l(w))^2$$

where G_{ij}^l is the gram matrix in layer l:

$$G_{ij}^l = \sum_k F_{ik}^l F_{jk}^l$$

The total loss function we aim to minimize is:

$$L_{total} = \alpha L_{content} + \beta \sum_{l=0}^{L} w_l L_{slayer}$$

where α and β are the weights of content and style loss, and w_l is the weighting factor of the contribution of layer l. In the original paper, have proposed an algorithm with L-BFGS, and in this work, we proceed with Adam optimizer. Since the VGG-19 model can recognize the low and high level features with different layers, it can outperform many other models. In our implementation, we noticed that the style loss value resulted from dividing the squared error by $\frac{1}{4N_l^2 M_l^2} \sum_{i,j} (G_{ij}^l(v) - G_{ij}^l(w))^2$ is similar to that of using the simple average. For simplicity, we decided to just implement the mean function to calculate the style loss.

2.2 Model Implementation through Progression

To begin with, we referred to the tutorial provided by TensorFlow [10]. Because the tutorial had many redundant and unnecessary steps, we successfully implemented the algorithm using our own functions. In order to understand the effectiveness of each transformation component introduced

by the style transfer model from Gatys et al. [3], we decided to first strip the model to its simplest form and then add each component step by step to see how the result progresses. With this in mind, we introduced our simplest version of the neural style transfer model, referred to as NST_{Naive1}. It did not implement a pre-trained model to perform feature extraction on content and style images, instead it simply used the images themselves as the features. In addition, the Gram matrix was not introduced to measure the style feature correlations, and style loss was simply calculated by as a simple MSE between style image and input image. We then created a slightly improved version, referred as NST_{Naive2}, by including a pre-trained VGG19 model proposed by Gatys et al. [3] to extract high-level features from content and style images. The feature extraction process in this model utilized the same convolutional layers specified by Gatys et al. [3], but for style feature extraction, unlike using all specified layers like the paper did, NST_{Naive2} only used the top layer to perform style feature extraction. As for calculating the style loss, the same method from NST_{Naive1} was kept to ensure each Naive model only introduces one improvement at a time. After NST_{Naive2}, we introduced our last naive neural style transfer model, NST_{Naive3}. This model implemented the same design structure from NST_{Naive2} with one improvement. The Gram matrix was used in calculating the style loss for the style feature extracted from the top layer. Finally, we recreated the Neural Style Transfer model proposed by Gatys et al. [3] as our complete version by extending style loss calculation to all style layers from NST_{Naive3}.

2.3 Hyperparameter Tuning

Transformed images can differ depending on how we change hyperparameters. And tuning hyperparameters can often improve performance of deep learning algorithms. The hyperparameters of our interest are number of iterations, learning rate, weights of content and style images and random seed for the white noise.

We evaluate how many iterations is required for the algorithm to produce meaningful outcomes. There is no definite answer to perfectly combine style and content images. Because the L_{total} is a linear combination between loss function of content and style images, we investigate how different relative weights affect the transformed images. We also study the effect of different learning rates to find optimal learning rate for the algorithm. Lastly, we compare how different random white noise images affect the output image.

2.4 Comparison with Other Models

For deeper exploration, we also changed our pre-train model, and compared the output image with our original output image. We have explored the neural style transfer and slow style transfer technique implemented with VGG-19, and in this section, we explored the style transfer with VGG-16 and ResNet50, and how they differed from VGG-19. Additionally, a simple extension, which performs fast style transfer that works on arbitrary painting styles as proposed by Ghiasi et al. [4], were also tested.

3 Results

The evaluation of an algorithm usually includes the qualitative and the quantitative results by Jing et al. [6]. Prior work from Gatys et al. [3] has generated a detailed and successful transferring network with per-pixel loss, and in this paper, we reconstructed this innovative neural network and compared it with the results using different pre-trained model proposed by Ghiasi et al. [4].

3.1 Naive Model Progression Results

Throughout our Naive Neural Style Transfer model series, we used the same white noise image as our input image and set the runtime iterations to 1000. As the naive models gradually improved, they generated some interesting results that deserved their own attention. The output from NST_{Naive1} had mostly white space, with a rough outline of the content in the content image. There were some color variations in the outline, and we believed these variations could come from the style image. Overall, although there was a slight indication of performing style transfer procedures, we could safely say that NST_{Naive1} had too simple of a structure to produce any meaningful results. The output from NST_{Naive2} started to display the effect of style transfer more clearly. The content outline from

the content image was captured much more vividly compared to the output from NST_{Naive1}, and the style pattern in the style image was also exhibited across the output. By comparing the result from NST_{Naive2} to that of NST_{Naive1}, we could state that the implementation of top-level feature extraction procedure using a pre-trained Deep Learning model improved the performance of style transfer significantly. However, as we could see, the capturing effects were still not strong enough as the overall image still displayed the greyish color from the white noise input image. By adding the Gram matrix into style loss, NST_{Naive3} generated an output with much stronger style transfer effect. However, the output quality was still some distance away from being artistic, and this was due to the fact that only the top style layer was used in extracting the style feature.

(a) NST Naive1 Output (b) NST Naive2 Output (c) NST Naive3 Output (d) NST Complete Output

Figure 1: Naive models progression result

3.2 Quantitative Results

To demonstrate the quantitative results of our work, we focused on several metrics, including the total loss during the training process and the running time for generating a single model.

With different pre-trained models, the total losses against the number of iterations with VGG-19, VGG-16, and ResNet50 models can be found in 2, and the total loss of VGG-19 pre-trained model is much larger at the beginning of training process, and it decreased very fast to the similar level with all other pre-trained model, and at the end, the total loss tended to be stable for all three methods. The total loss of the ResNet50 model is notably small among these three models, which has the initial total loss of value 114438 while VGG-19 has 5E8 and VGG-16 has 5E7. The total running time of each training process is described in 1.

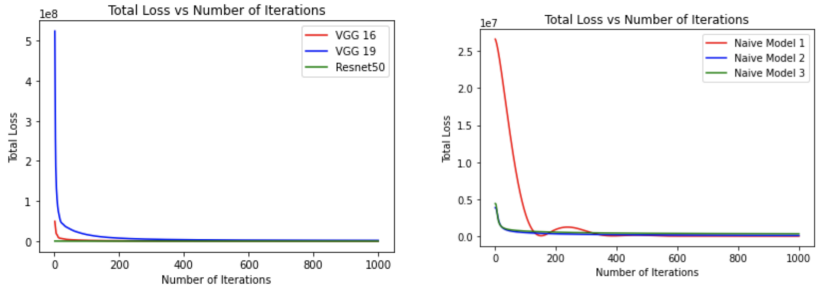
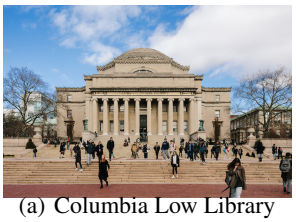


Figure 2: The total loss against the number of iteration with different pre-trained models

Table 1: Runtime of Different Training Networks

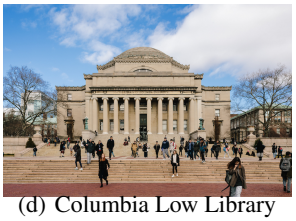
Network	**Average Runtime (seconds)**
naive1	4.8597
naive2	332.1941
naive3	103.0080
VGG-19	111.3773
VGG-16	98.2377
RESNET50	77.3531
Image Style Transfer by Ghiasi et al. [4]	395.1770



(a) Columbia Low Library

(b) Macro Closeup Of Different Colors Oil Paint by Yavnik Pavel.2020

(c) Transformed Image 1



(d) Columbia Low Library

(e) Fire by Korovilas Kostas.2020

(f) Transformed Image 2

(g) Empire State Building in New York City

(h) Macro Closeup Of Different Colors Oil Paint by Yavnik Pavel.2020

(i) Transformed Image 3

(j) Empire State Building in New York City

(k) Fire by Korovilas Kostas.2020

(l) Transformed Image 4

Figure 3: Four transformed images combining 2 contents and 2 styles

3.3 Qualitative Results

For the qualitative evaluation, we will compare the output figures of different networks. An indication of a good transfer of the content image with the chosen style image is that we are able to capture the texture or the pattern of the style image and also maintain our content structure. We have trained the network with different pre-train models and different metrics as described in the Methods section and compare the effects of these transferring neural networks.

5 illustrate how different hyperparameters affect the outcome image. To evaluate how individual hyperparameters affect the outcome, other hyperparameters are fixed based on our empirical results. The Columbia Low library content image and the oil paint style image are used for this purpose. Texture of style image is applied on different parts of the content image depending on different random white noise images. However, different random white noise images do not have a significant effect on the transformed images. Whereas learning rate has a significant effect on the outputs. Once we set learning rate = 0.01, minimal texture of style image is applied to the content image. Once we increase the rate to 1, we observe more texture of the style image, but it is not enough. Setting the learning rate to 5 or higher, two images are well blended. We see similar patterns for different numbers of iterations. Once we set the number of iterations as 100, two images do not synthesize well. Surprisingly, fixing the relative weight of content and style images and same random seed, we observe that learning rate $= 1$ and 100 number of iterations produce similar outputs. From 500 iterations, two images are synthesized well. As we increase the number of iterations, a more diverse range of colors in the style image can be seen. Different ratios of weights of style to content images also produce different transferred images.

Once we place more weight on content relative to style, less texture of style is blended into the output image, whereas putting more weight on style relative to content, the content images become more abstract.

As we discussed above, the VGG-19 pre-trained model outperforms many other models, and we reconstructed a network with VGG-16 and a new image style transfer network introduced by Ghiasi et al. [4]. The transferred output images can be found in 4. Based on our experiment, we found that the VGG-19 model was able to maintain the highest content, and we saw much more details of the content image but lost the information in the style image as it showed more of the edges and the texture of the style image. Meanwhile, Ghiasi et al. [4]'s model captured the highest style effects and lost the content information. We were able to observe some of the objects of the style input, and the content object in Ghiasi et al. [4]'s work is more abstract. The ResNet50 pre-trained model was not able to show enough style information, and it kept almost all the content information, which resulted in an imperceptible style transfer.

4 Discussion

Style transfer is an inspiring research area in computer vision recently. In this paper, we have developed several neural style transferring networks along with different training methodologies. By reconstruction the style feature with different subsets of the style layers from Convolutional neural network (CNN), we are able to capture different properties of the style image input. With more layers included in the training process, we can learn more complex features from the style image, from the edges to the texture, the pattern or some objects of the style input image. With different pre-trained CNN models, hyperparameters, and loss functions, we were able to evaluate and compare the effects of such modifications. Another thing worth discussing is that executing the algorithm using a GPU can effectively save runtime. When we did not use a GPU, it took hours to fit the model, but using a GPU, we were able to get the results in several minutes. In the future, we hope to explore the possibility of adding multiple style images, as well as the runtime reduction.

References

[1] magenta/arbitrary-image-stylization-v1-256. URL https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2.

[2] Google achieves ai 'breakthrough' by beating go champion. *BBC*, Jan 2016. URL https://www.bbc.com/news/technology-35420579.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2415–2423, 2016.

[4] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network, Aug 2017. URL https://arxiv.org/abs/1705.06830.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song. Neural style transfer: A review. *IEEE Transactions on Visualization and Computer Graphics*, pages 3365–3385, 2020.

[7] Kostas Korovilas. Fire, 2020.

[8] John Markoff. A learning advance in artificial intelligence rivals human abilities. *The New York Times*, Dec 2015. URL https://www.nytimes.com/2015/12/11/science/an-advance-in-artificial-intelligence-rivals-human-vision-abilities.html.

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[10] TensorFlow. Neural style transfer: Creating art with deep learning using tf.keras and eager execution, Sep 2018. URL https://medium.com/tensorflow/neural-style-transfer-creating-art-with-deep-learning-using-tf-keras-and-eager-execution-7d541ac31398.

[11] Pavel Yavnik. Macro close up of different color oil paint, 2020.

[12] Mark Zastrow. 'i'm in shock!' how an ai beat the world's best human at go. *NewScientist*, Mar 2016. URL https://www.newscientist.com/article/2079871-im-in-shock-how-an-ai-beat-the-worlds-best-human-at-go/.