Neural Style Transfer of videos from an image

Rahul Handa
University of Massachusetts
Amherst
rhanda@umass.edu

Rohini Kapoor
University of Massachusetts
Amherst
rohinikapoor@umass.edu

Abstract

Recent groundbreaking work with convolutional neural networks by Johnson[9] and Gatys et al.[2] has shown neural networks can be used to perform style transfer and completely changed the landscape of image and video modification. Style transfer is a method of transferring the style of one style image to the entirety of a content image, resulting in a newly rendered image. After going through these papers,we wondered how the same style transfer will look on a video and what challenges will we face while doing this on the video, if this is possible and can we do the style transfer on only a part of the video frame. We go about it by modifying the loss function as an optimizations of style and content loss, and then implementing this optimization for each frame of a video while keeping track of the previous frame. In the end we have used GrabCut algorithm to produce segmentation masks for one of the frames of a test video and we show how the desired region can be altered while leaving the rest of the frame intact.

1. Introduction

1.1. Motivation

Till now we have seen how back propagation is used to learn valuable information from an image. We do this by learning individual features of the image and then putting them together in higher layers. After becoming familiar with this we were really intrigued by the possibility of what else can we learn if we do a gradient ascent instead and what can be its application. In this project, we attempt to venture into the problem of style transfer. We start with attempting this on images and then use our learning in doing this over videos. So far, the topic of style transfer has been widely studied in computer vision, and we look to apply it on videos using neural networks.

There have recently been a lot of interesting contributions to the issue of style transfer using deep neural networks. Gatys et al.[2] proposed a novel approach using neural networks to capture the style of artistic images and transfer it to real world photographs. We will build upon their approach and try to reproduce state of the art results[1]. We will then conduct our own experiments on images and extend our procedure over videos. The different experiments include changing the pre-trained model, changing the layers that the content is learned from and also experimenting with the structure of the neural network.

1.2. Problem Statement

We aim to extend the notion of style transfer to videos, and if time and resources allow us, we plan to apply the style transform to a particular object in the video(for example just a human figure). We will start by applying style transfer to images. For this, we have used the pretrained vgg-19 network. We will use various images downloaded from the internet as our content and style images(refer to appendix). We plan to spend time to tune the network and parameters to obtain the best balance for human faces. We will experiment with the layers chosen for style and content loss to better tune our output images. Once we have a satisfactory model for our images, we will move onto videos. We will be using the videos from the sintel dataset. We will have to further modify our network to preserve smooth transition between individual frames of the video.

For expected results, we will at least match the state of the art results as can be seen in Artistic style transfer for videos[1], and even improve upon some aspects if resources permit.

Evaluation is based on the human eye as the goal can be producing results closer to either style or content image or even something more aesthetically pleasing.

2. Related Work

The first style transfer using neural networks was proposed by Gatys et al.[2] which performs gradient ascent on a white noise image, minimizing two losses: content loss against an input photo and style loss against a style image. The style loss function is taken at multiple layers in the network, and characterized by the mean squared loss between

the Gram matrices of the input image and the artwork.

Since then, style transfer has been a widely explored topic, and many have worked towards improvements on the algorithm. Johnson et al.,[9] make modifications to the approach in the original paper[2] and use perceptual loss functions based on high-level features instead.

Research in the field of semantic segmentation has made rapid gains in the past decade. The application of a fully convolutional network trained end-to-end on semantic segmentation achieved state-of-theart results on the PASCAL-VOC dataset in 2015 [7]. The Conditional random fields as recurrent neural networks(CRF-RNN network) [8] takes advantage of CRFs to formulate the semantic label assignment as a probabilistic inference problem to achieve finer segmentations. The MNC framework introduced by Dai et al. [5] achieved state of the art results on the 2015 Microsoft COCO dataset, performing instance-aware semantic segmentation using multitask network cascades. Just recently, He et al. [6] presented the Mask R-CNN architecture, which improves upon Fast R-CNN and Faster R-CNN to perform pixel-level segmentation.

Additionally, a recent paper by Gatys, et al. [3] analyzed different ways in controlling color, spatial location, and scale. They used guided Gram matrices and guided sums in controlling spatial location. Results showed notable improvements in maintaining semantic meaning in regards to foreground and background in the content image. In the same vein, as the research done by Gatys, et al., Li, et al. combined Markov Random Fields with Convolutional Neural Networks [18] to more realistically transfer semantically similar parts of a style image over to a content image. Their initial results are very promising. "Neural Style Transfer: A Review" by Jing et al. does an excellent work of summarizing the state of the art work in the style transfer research. They also make some suggestions about what is the future direction of the field and make some suggestions of their own.

Ruder et al.[1] in their paper "Artistic style transfer for videos" transfer a particular style of painting to the entire video, frame by frame. They introduce a temporal constraint that penalizes deviations between two frames to preserve smooth transition between individual frames of the video.

3. Technical Approach

3.1. Architecture

We use a pre-trained VGG19 network to compute our losses. We use a loss network pretrained for image classification to define perceptual loss functions that measure perceptual differences in content and style between images. The loss network remains fixed during the training process. An architecture of the trained network is shown in Figure 1.

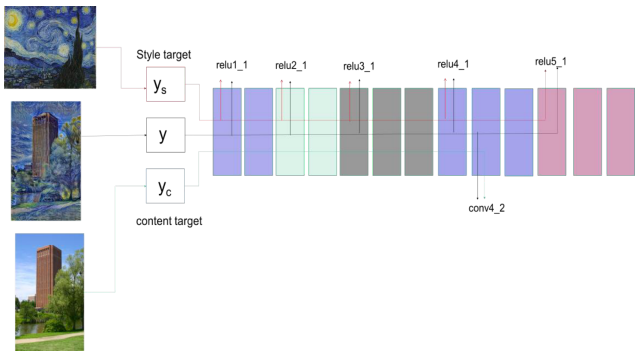


Figure 1: VGG architecture

3.2. Methods

For style transfer, we use the approach suggested by Gatys et al[2]. We treat the problem in hand as an optimization problem of content and style loss. The main idea was that the features extracted by a convolutional network carry information about the image content, while the correlations of these features have details about the style.

It took us a lot of time to setup the whole environment to run our experiments. Setting up Tensorflow with CUDA support on a machine with GPU TODO. We specifically needed opencv with ffmpeg support for video processing, getting the environment up and running made the the whole experimentation extremely smooth

3.2.1 Images

Let s, c, and x be our style, content and output images respectively. We define x randomly as Gaussian noise. We define a feature representation matrix in layer l as $F^l \in \mathbb{R}^{N_l * M_l}$ where N_l is the number of filters, and M_l is the size of the activation map. Let P^l and F^l be the feature representations in layer l of c and x respectively. We thus compute content loss as the mean squared error between P^l and F^l for all layers L_c that we choose for content representation.
The content loss, thus, is given by:

$$L_{content}(c, x) = \sum_{l \in L_c} \frac{1}{N_l M_l} \sum_{i,j} (F^l_{ij} - P^l_{ij})^2 \qquad (1)$$

This helps us to generate an image, x, which replicates the content of image c.
To extract style from an image, we compute the correlations between each activation map. This information is encoded in a matrix $G^l \in \mathbb{R}^{N_l * N_l}$ called the Gram matrix. G^l is calculated as:

$$G^l_{i,j} = \sum_{k=1}^{M_l} F^l_{ik} F^l_{jk} \quad (2)$$

which is the dot product of the vectorized version of each activation map.

The style loss is calculated as a mean-squared error between correlations of the filter responses, expressed as Gram matrices. Let A^l be the Gram matrix for the style image s, and B^l be the Gram matrix of the stylized image x. We then compute the style loss L_{style} for all layers L_s that we choose to represent style as:

$$L_{style}(s, x) = \sum_{l \in L_s} \frac{1}{N_l^2 M_l^2} \sum_{i,j} (B^l_{ij} - A^l_{ij})^2 \quad (3)$$

Thus, we define our overall loss function as:

$$L_{total}(c, s, x) = \alpha L_{content}(c, x) + \beta L_{style}(s, x) \quad (4)$$

with factors α and β that decide the importance of style vs content. We thus attempt to solve this equation for x using gradient-based optimization.

3.2.2 Semantic Segmentation

To transfer style to just a portion of the image, we use the concept of semantic segmentation. Semantic segmentation involves clustering parts of an input image together that belong in the same object class. It classifies each pixel in the input image to recognize any object present in the image. For this, we generate a mask for the input image, which contains ones in the pixels that we want to apply the style to and zeros in all other places. We compute the same content loss as above, but for the style loss, we apply the style gradients to only the part of the image we want to stylize, using the input mask as a filter. Fig x shows an examples of our generated input masks used for segmented style transfer.

3.3. Videos

We cannot tackle videos with the same approach as images as we face a major issue of consistency. When the style transfer for consecutive frames is initialized by independent Gaussian noise, two frames of a video converge to very different local minima, resulting in a strong flickering. To handle this and also to save some computing resources, we will initially use the brute force approach of initializing the input image for $i + 1$ frame with the stylized i^{th} frame. This ensures that pixels that have not been changed in the video are initialized to their desired appearance, while the others will need to be optimised.
However, if the video has an object moving, such an approach results in incorrect initialization of the frames for the objects in motion. To tackle this, we take into account the optical flow. We thus, initialize frame $i + 1$ with the previous stylized frame warped.
Let $p^{(i)}$ and $x^{(i)}$ be the i^{th} frame of the input video and stylized video respectively, and $x'^{(i)}$ be the initialization of the i^{th} output frame. We initialize the first frame randomly. After that we initialize the $i + 1$ frame as-
$x'^{(i+1)} = \omega_i^{i+1}(x^{(i)})$, where ω_i^{i+1} denotes the function that warps a given image using the optical flow field that was estimated between the frame $p^{(i)}$ and $p^{(i+1)}$

Figure 2: Starry night used as style image

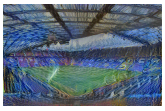
4. Experiments

We started by simple style transfer on images using popular paintings as styles. We have used the pre-trained VGG network and our implementation is in tensorflow. We also had to keep in mind the size constraints of an image and the input type accepted by the network. Training on CPU was taking approx. 2 hours for image after some re-sizing and compression.
At first, we used a similar architecture to the Gatys et al[2]. We used the 'starry night' painting as our style image (refer to Figure 2). The original paper uses the layers conv4_2 as the content layer. We tuned for various parameters. Table 2 shows a stadium that we used as our content image. We generated images for max and well as average pool for this. The max pool image is in 1c and the avg pool is in 1d. As we can see, avg pool helps to keep more details of the content image intact as compared to max pool. This is a trend we saw in other images as well. We saw that these parameters work substantially well for structures like buildings.

From there, we moved to trying to stylize human faces with various cartoon-ish styles. To apply existing styles on human faces, we needed to tune the parameters differently. The tuned parameters for our previous images do not work very well with faces (refer to appendix image). As shown by the Johnson et al.(2016), as we move to the deeper layers, we tend to lose more structural details. Thus, for faces, we have used the the conv2_2 as our style layer. Table 2 was one of our attempts for stylizing a face. It shows the style image, the content image. We show the improvement from moving from the conv4_2 layer as content layer to conv2_2 layer.

(a) Content Image

(b) Style Image



(c) Max pooling

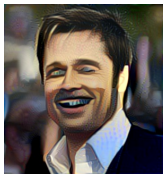
(d) Average Pooling

Table 1: Style transfer with different pooling techniques

(a) Style Image

(b) Content Image

(c) Content layer con4_2



(d) Content layer con2_2

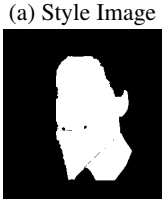
Table 2: Style transfer with different content layers

We then performed semantic segmentation on a single image. We used grab-cut for Foreground Extraction from the image.The algorithm segments a user selected region iteratively to get the best segmentation. Initially user inputs the rectangle which marks the "non-background". Everything outside this rectangle will be taken as sure background. Everything inside rectangle is unknown. The algorithm does an initial labelling depending on the data we gave. It labels the foreground and background pixels (or it hard-labels) Now a Gaussian Mixture Model(GMM) is used to model the foreground and background. Depending on the data we gave, GMM learns and create new pixel distribution. Then a mincut algorithm is used to segment the probability of background and foreground pixels. The process is continued until the classification converges. The content image, style image, mask and stylised image can be seen in Table 3. We pass this mask along with the content and style image to our network.

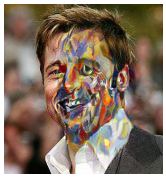
For videos, due to time and computational issues, we used just 5-6 frames of the video to transfer the style.When the style transfer for consecutive frames of a video is initialized by independent Gaussian noise, two frames of a video converge to very different local minima resulting in a strong

(a) Style Image

(b) Content Image



(c) Mask image created from grab cut



(d) Stylized image

Table 3: Style transfer with semantic segmentation

Figure 3: Video frames original and styled

flickering. As stated in Artistic Style Transfer for Videos [1], the most basic way to yield temporal consistency is to initialize the optimization for the frame $(i+1)$ with the stylized frame i. Areas that have not changed between the two frames are then initialized with the desired appearance, while the rest of the image has to be rebuilt through the optimization process. To take moving objects into account,we have to move away from this simple approach as it does not perform well. We take the optical flow into account and initialize the optimization for the frame i+ 1 with the previous stylized frame warped:

$$x^{'(i+1)} = (w_i)^{i+1}(x^{(i)})$$

Here $(w_i)^{i+1}$ denotes the function that warps a given image using the optical flow field that was estimated between image (i) and (i+1). The first frame of the stylized video still has to be initialized randomly.

In figure 2 the first row shows the default frame for each column. The second row shows the style transfer done on

each frame with default parameters. The third row shows results with more weightage given to the content images. The last row shows style transfer applied only on the human body in each frame.

(a) Style Image

(b) Content frame

(c) Mask image created from grab cut

(d) Stylized frame

Table 4: Video frame with semantic segmentation

We then used the ffmpeg converter to stitch the 6 processed frames into a 2 second video with very low fps for testing our results. Table 4 shows the b)content frame, c)style frame and d)output frame. For this video we have used the same mask for all 6 frames as there was no major motion in those frames.

5. Conclusion and Future Work

Our experiments with the neural style algorithm show the flexibility and further potential use cases for style transfer. The fast variant on the neural style algorithm presented by Johnson [9] make these sorts of effects possible to achieve in real-time video processing applications. We have seen style transfer being implemented in popular products such as Prisma, Facebooks messenger service and Snapchat. The extensions that we have presented namely content weighted style transfer and semantically segmented style transfer are simple extensions that will continue to improve the perceptual quality and novel applications achievable through style transfer. We've seen that these extensions enable interesting new perceptual effects.

However, right now we use a handcrafted mask for segmented style transfer. For future,we would like to use CRF-RNN in order to generate a mask for a given input image. Broadly, the CRF for pixel-wise labeling models each pixel in an image as random variables that form a Markov Random Field when conditioned on a global observation. In this context, the global observation is the image upon which we wish to perform semantic segmentation. A CNN predicts labels for each pixel without accounting for the smoothness or consistency of the label assignments. In this paper we have tried adjusting different parameters for the application of localized style transfer on human faces. But since the VGG-19 model is pre-trained on the Imagenet dataset which does not have a lot of human faces, we would like to use an initial model which is trained specifically on human faces and extract content features from this network to use in the style transfer pipeline. Further , we can fine tune the loss functions to account for the spatial considerations of facial features.

From here we plan to take this work to the next stage by achieving region-based style transfer by replicating specific styles in certain regions of the content video.

References

[1] Manuel Ruder, Alexey Dosovitskiy, Thomas Brox *Artistic style transfer for videos*. Department of Computer Science,University of Freiburg

[2] Leon A. Gatys, Alexander S. Ecker, Matthias Bethge *A Neural Algorithm of Artistic Style*. University of Tubingen, Germany

[3] Yongcheng Jing,Yezhou Yang, Zunlei Feng, Jingwen Ye, Mingli Song *Neural Style Transfer: A Review* Zhejiang University, Arizona State University

[4] Harish Narayanan's Blog *https://harishnarayanan.org/writing/artistic-style-transfer/.*

[5] J. Dai, K. He, and J. Sun. *Instance-aware semantic segmentation via multi-task network cascades. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 31503158, 2016.*

[6] K. He, G. Gkioxari, P. Dollar, and R. Girshick. *Mask r-cnn. arXiv preprint arXiv:1703.06870, 2017.*

[7] J. Long, E. Shelhamer, and T. Darrell. *Fully convolutional networks for semantic segmentation, 2015.*

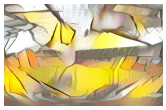
[8] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. *Conditional random fields as recurrent neural networks. In Proceedings of the IEEE International Conference on Computer Vision, pages 15291537, 2015.*

[9] J. Johnson, A. Alahi, and L. Fei-Fei. *P*erceptual losses for real-time style transfer and super-resolution. In European Conference on Computer Vision, pages 694711. Springer, 2016.

6. Appendix

Disclaimer : All input images have been taken from the web. The generated output videos have been attached in the zip file.
Please find below some examples of our experiments

(a) Style Image



(b) Stadium output Image

Table 5: Style tranfer

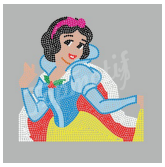
(a) Style Image

(b) Transfer with 1000 ierations

(c) Transfer with style layer weightage increased

(d) Transfer with 2000 iterations

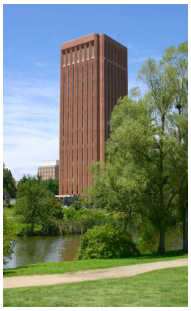
Table 6: Style tranfer with different hyper parameters



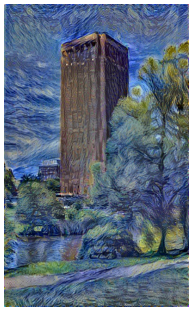
(a) Style Image

(b) Transformed image Image

Table 7: Style Tranfer



(a) Original Image



(b) Starry Library

Table 8: Style Transfer on Du Bois