Northeastern University

PROJECT REPORT

CS5100

Foundations of Artificial Intelligence

(Prof. Lawson Wong)

Neural Style Transfer

(Analysis and Improvement)

by

Krit Goyal

Bishwarup Neogy

14th December, 2018

1. Introduction

It's the 21st Century and AI is on the rise again. Owing to the recent developments and advances in manufacturing of GPUs and CPUs with hefty processing powers, all the ideas that scientists had envisioned in the mid-late 1900s can now finally be tested and verified experimentally. One of the areas that have particularly benefited from this boom is, *image processing.* Multi-billion dollar companies like Facebook, Google, Snapchat have raised the bar to a whole new level with their state of the art image processing techniques that allow them to identify individual faces in a photograph, enhance images taken from a SmartPhone camera to DSLR level of quality and crispness, and much more. Now, Google has gone one step further with their brand new *Night Sight* mode in their smartphone camera app, which essentially elevates the quality of a photograph taken in low light conditions to that of one taken during the day. All these intriguing ideas and implementations made us wonder about the other applications of this sorcery with colors and pixels.

2. Related Work

After a lot of searching and browsing we came across the Neural Style Transfer algorithm formulated by Gatys et. al. in their 2015 paper bearing the same title. Imagine having Vincent Van Gogh as your personal artist who could paint you a Starry Night rendition of your favorite pictures in a few minutes. Or Pablo Picasso painting a portrait of you using his renowned style with various shapes, colors and defined borders. This is exactly what was achieved by Gatys et. al. using Convolution Neural Networks (CNNs). This paper describes a technique to extract the style of an image (Style Image, preferably a painting) and transfer it to the contents of another image (Content image). Thrilled by our findings, we quickly became curious to see if this technique could be applied to real life photographs. Instead of capturing the style of a painting, we wanted to extract the style from a photograph. For example, taking the style of an image of a city skyline during sunset and applying it to another image of a different city taken at a different time of the day, *a photo-realistic style transfer*. Here, we looked at the Deep Photo Style Transfer technique described by Luan, Fujun et al. in a 2017 paper by the same name, for motivation.

3. How does Neural Style Transfer work?

In simple terms, neural style transfer is a technique used to blend the style of one image with the content of another with the help of CNNs. By style, we normally refer to the patterns, like the brush strokes in a painting. This technique involves 3 images, the content image(C), the style image(S) and the generated image(G). The principle behind this technique lies in the definition of two distance functions, more formally referred to as loss functions. These two functions are defined as follows :

$J_{Content}$: Describes how different the content of the two images (C and G) are.

J_{Style} : Describes how different the style of the two images (S and G) are.

$$J_{total}(G) = \alpha \, . \, J_{content}(C,G) + \beta \, . \, J_{style}(S,G) \qquad (1)$$

In summary, we will transform the base input image by minimizing the content and style distances (losses) using gradient descent. This will hopefully generate an image that has the content of the content image (C) and the style of the style image (S). In the above equation, α and β are hyper parameters that can be tuned to decide the trade-off between the style and the content of the generated image.

$$J_{content}(C, G) = \tfrac{1}{2} \sum (F^l_{ij} - P^l_{ij})^2 \qquad (2)$$

In order to define the content loss $J_{Content}$, we pass our content image through the VGG19 CNN and probe it's output (activation) at some middle layer L_C and then we pass the input image and probe its output at the same layer L_C. The difference between these activation outputs will help us define the content loss. If we probe too shallow in the CNN, the generated image will contain pixel values too similar to the actual image. Whereas, if we probe the output too deep in the CNN, then we will end up identifying entire objects in the image and then try to check if this object is present somewhere in the generated image or not. So, L_C shouldn't be too shallow or too deep in the CNN.

$$J_{style}(S, G) = \sum w_l E_l \qquad (3)$$

$$E_l = 1/(4N^2_l M^2_l) \sum (G^l_{ij} - A^l_{ij}) \qquad (3.a)$$

$$G^l_{ij} = \sum (F^l_{ik} F^l_{jk}) \qquad (3.b)$$

To represent the style of of an image, we once again use our CNN and pass the style image and generated image through it and sample their respective outputs at another hidden layer L_S. The style of the image can be defined as a measure of the correlation between activations across channels in this chosen layer L_S By correlation of the channels, we are trying to find how many times neurons from different channels detect the same type of thing together. For ex: let's say channel-A corresponds to neurons detecting black stripes and another channel-B corresponds to neurons detecting blue tints/patches. The number of times blue tints are present in a part of the image along with black stripes will give you a measure of the correlation between channels A and B.

Mathematically we represent the correlation between channels using *Gram matrices*. It turns out that we get more visually pleasing results when the style cost function is used from multiple layers (5 layers as per Gatys et al.). Thus, as defined above, the style cost function is represented as a function of these correlations over several layers of the CNN.

4. Challenges and Contributions

Before beginning to write any code, we first needed a better understanding of a few concepts in mathematics and machine learning:

- Convolutional Neural Networks (CNN)
- VGG CNN
- Tensorflow and eager execution
- Gradient descent

Then, to go about achieving a photorealistic transfer between two real and similar images, we used [1] as the concept, and an implementation(6.1) of this method as our starting point. Reading through [1], we gathered a better understanding of the concepts mentioned in there by trying to map them to their implementation in 6.1. Then, we formulated our next steps, to analyse and possibly try to improve upon this, as follows:

4.1. Try to reduce noise during style transfer by varying algorithm parameters

We experimented with different combinations of content and style images by varying the following parameters in a bid to achieve a lower value of best total loss in the style transfer. Keeping all other parameters at their default values, we varied the following parameters:

- Set of layers used to extract style features
- Loss function hyperparameters

In some cases, we observed upto 19% reduction in loss (4.1). Although this may seem like a good figure, but this loss does not translate to a significant and visually discernible change in the generated image.

We also realised that this alone was not enough to achieve a photorealistic transfer. As soon as we applied the same technique, by replacing the reference painting image with a real image, we got the following output:

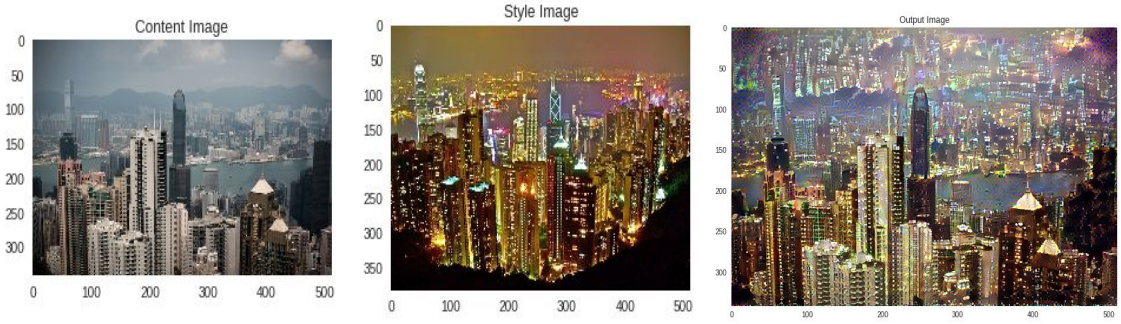


Fig. 1

Here, we observe that the output image seems too noisy and distorted. This is because our current technique cannot differentiate between different types of objects in an image. It is a universal transfer of style from reference to content.

This brings us to the next step: *image segmentation and masking*.

4.2. Deep Photo Style Transfer

The theory behind Deep Photo Style Transfer as explained by Luan, Fujun et al in [2] is based on the concept of Neural Style Transfer. But in order to transfer the style of a photo-realistic image to another photo-realistic image we need to take care of a few problems. Let's consider the example depicted in *Fig.1.* Here the style was generalized for the entire image and transferred to all parts of the content image. In case of paintings, this is exactly what we would want, but as is quite evident, this is not something desirable for real life photographs. In simple words, what we would like is to be able to transfer the style from the buildings of the style image to the buildings of the content image and similarly transfer the style of the sky of the style image to the sky of the content image. As depicted by Fig.2, this was very convincingly achieved by Luan, Fujan et al.

Fig. 2

Another aspect that needs to be redefined is the representation of style. In the artistic model, style was a general representation of the way the artist painted (for ex: brush strokes). But in the case with real photographs, there are no

brush strokes. Instead the style is represented by colors, brightness and contrast of each part of the style image.

4.3. Apply the concepts described in [2]

We understood and realized the root cause of some of the problems we encountered when trying to apply artistic Neural Style Transfer to real photographs. We studied the work of Luan, Fujan et al. and attempted to break down the steps needed to achieve deep photo style transfer. As described in 4.2, one cannot represent the style in the same way as brush strokes of a painting. As explained in [2], representing the style in this fashion would distort edges and regular patterns. When transferring the style to a content image, we must keep, for example the windows to remain aligned in a grid. In other words, we would require a method that strongly and accurately transfers the color representation of the image without having any drastic effects on the geometry of the content image.

This was achieved in [2] by introducing a *Photorealism Regularization* term in the *loss function*. This term helps define an image transformation that is locally affine in color space. This is such that for each output patch of the image there is an affine function that maps the input RGB values to its respective output part. In simpler terms, we now have a way of mapping the RGB values of the pixels representing a window in a house to the pixels that represent the window of the house in the content image.

By considering only the RGB values of the pixels, we make sure that there are no distortions in the edges and features, inherently preserving the overall structure of the content image.

$$L_m = \sum_{c=1}^{3} V_c[O]^T M_I V_c[O] \qquad (4)$$

This regularization term has been defined using the *Matting Laplacian* of Levin et al. We find the gradient descent of this term, $\boldsymbol{L_m}$ along with other terms in our loss function.

$$d\,L_m / \; d\,V_c[O] = 2M_I V_c[O] \qquad (5)$$

The style loss function(L_s) is also modified to overcome the “spillovers” in the original methodology. The gram matrix is computed over the entire image and thus captures the style across the reference image as a whole. As mentioned earlier, this works for a painting which has a uniform style/pattern

throughout. But in the case of real life photographs, the style of a building is different from that of the sky or the lake.

Here, we can use image segmentation techniques to generate masks for input and reference image for a set of common labels. These masks can then be used to do a selective transfer for each label from the reference to the content.

The masks are input as additional parameters while calculating the gram matrix. The augmented style loss term now looks like this:

$$L^{l}_{s+} = \sum_{c=}^{C} (1/2N^{2}_{l,c}) \sum_{ij} (G_{l,c}[O] - G_{l,c}[S])^{2}_{i,j} \qquad (6)$$

$$F_{l,c}[O] = F_{l}[O]M_{l,c}[I],\ F_{l,c}[S] = F_{l}[S]M_{l,c}[S] \qquad (6.b)$$

where C is the number of channels in the semantic segmentation mask, $M_{l,c}[\ldots]$ denotes the channel c of the segmentation mask in layer l, and $G_{l,c}[\ldots]$ is the Gram matrix corresponding to $F_{l,c}[\ldots]$. We need to further downsample the masks to match the features space size at each layer accordingly.

After combining the above two modifications, the total loss function can now be written as:

$$L_{total} = \sum_{l=1}^{L} \alpha_{l} L^{l}_{c} + \Gamma \sum_{l=1}^{L} \beta_{l} L^{l}_{s+} + \lambda L_{m} \qquad (7)$$

where L is the total number of convolutional layers and l indicates the l-th convolutional layer of the deep neural network. Γ is a weight that controls the style loss. α_l and β_l are the weights to configure layer preferences. λ is a weight that controls the photorealism regularization.

5. Results

5.1. Neural style transfer

5.1.1. Choosing a different set of style layers

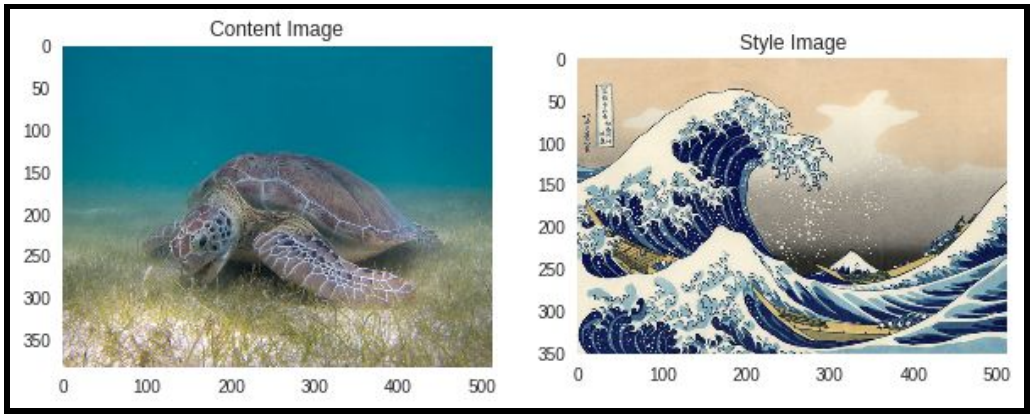
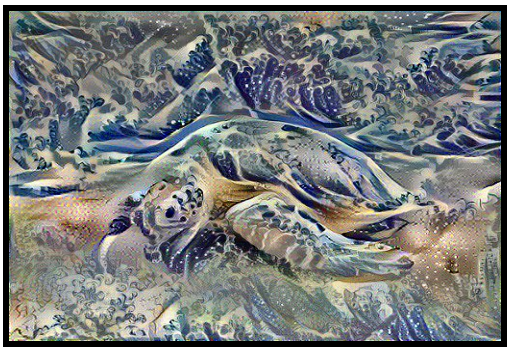
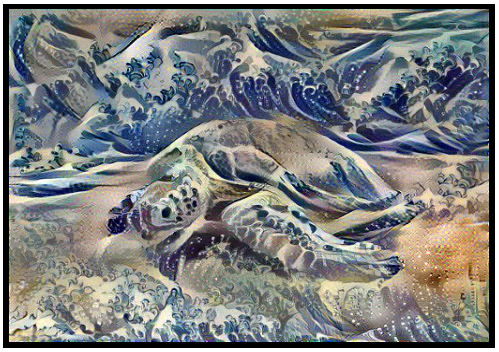


Fig 3



Default



Style layers 4,5 = '_conv2'

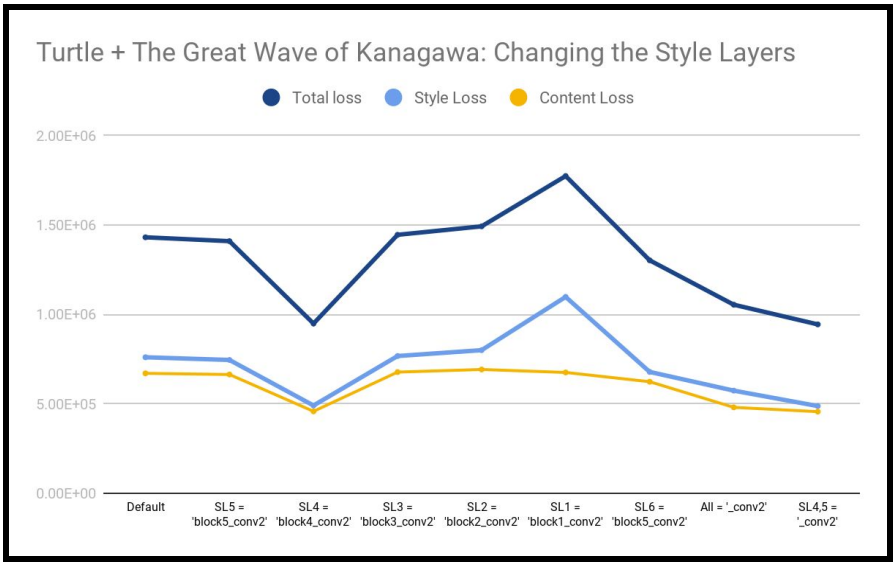


Fig. 4

Here, we observed that choosing convolutional layer 2 in blocks 4 and 5 reduces the best total loss. Based on this observation, we ran the algorithm for a few more combinations of content and style images to verify this hypothesis. The results in the below graph verify our findings, that choosing layer 2 instead of 1 in blocks 4 and 5, produces a lesser bess total loss.

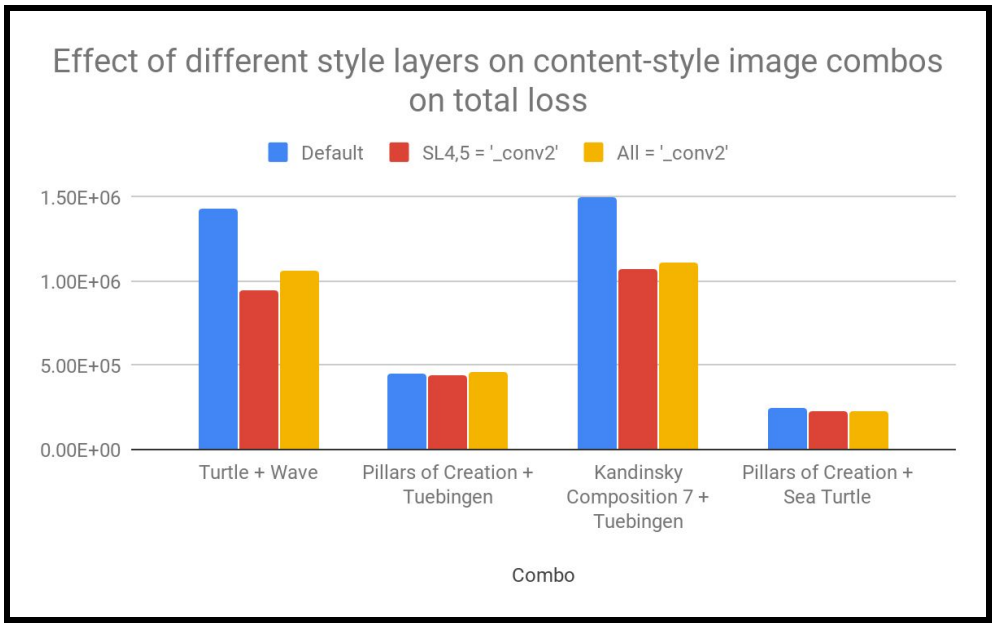


Fig. 5

5.1.2. Choosing a different pair of content and loss weights

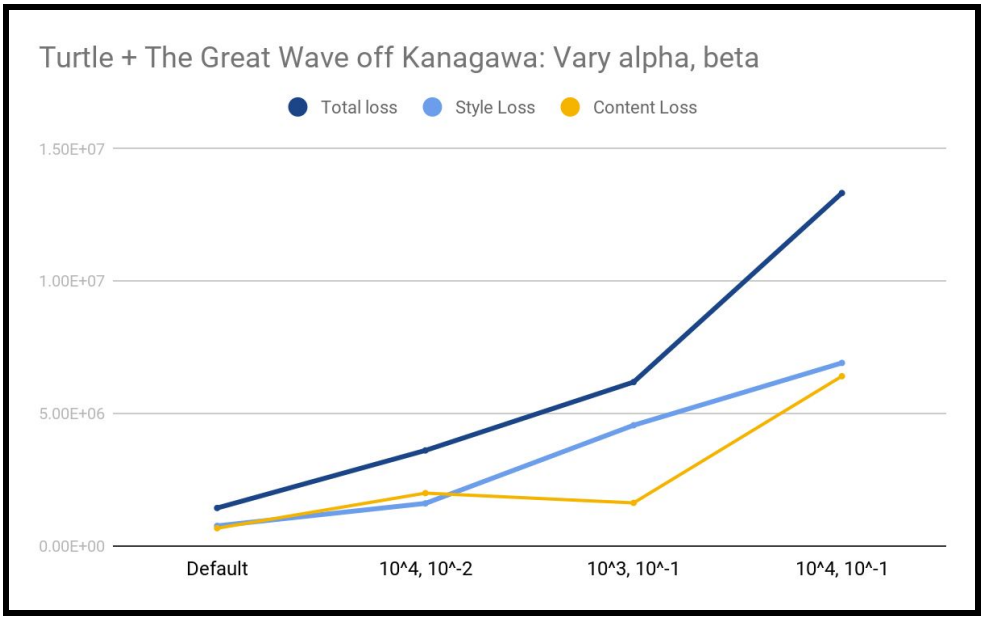


Fig. 6

These parameters were found to be quite optimised already. As was observed, by varying the content and style weights in either direction, by varying and keeping their ratio same, the best total loss quickly climbed.

5.2. Deep style transfer

To implement the technique described by Fujan et. al., we first attempted to do only image segmentation; using a proxy method. We directly provided our algorithm with single label images. As can be seen from figure 7, we provided our neural style transfer algorithm with a daytime sky image as the content, and an overcast night sky image as reference. This again resulted in a distorted indiscernible image (Fig. 8).

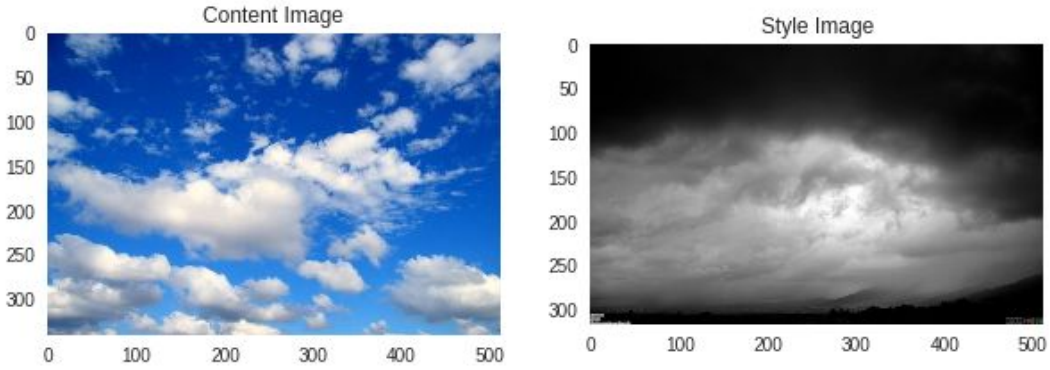


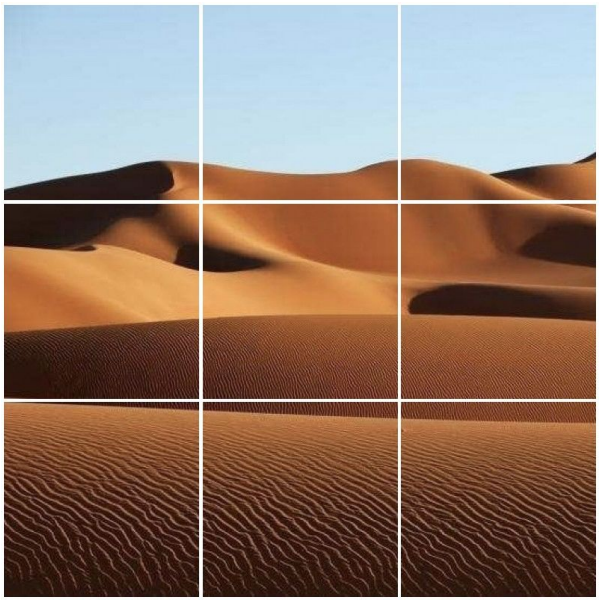
Fig. 7

Fig. 8

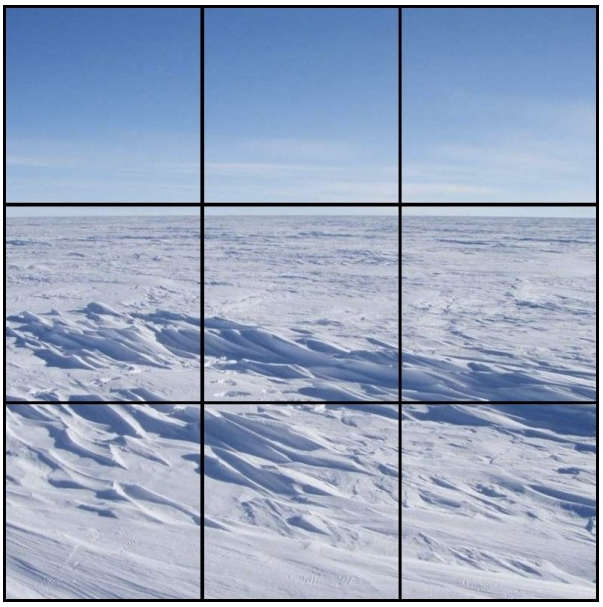
We further experimented on images with a two label set, by splitting the content and style images and doing the style transfer for individual parts.

We used an image of the sahara desert for content, and an antarctica landscape for style. Both these images only have two labels, the land and the

sky. We then split both of them into a 3x3 grid, ran a style transfer for each of these sub-parts with a one-to-one correspondence, and stitched the resulting sub-parts back together.

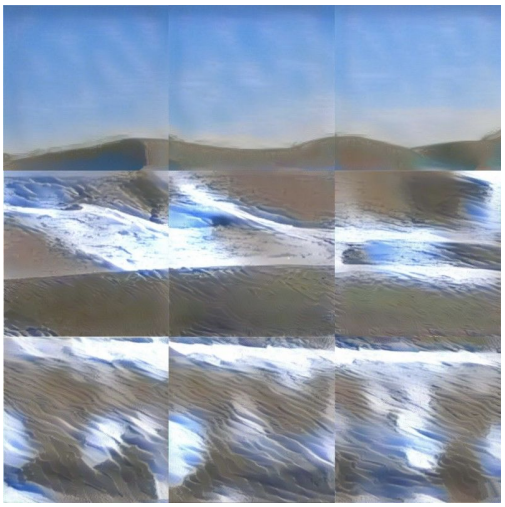


Content image

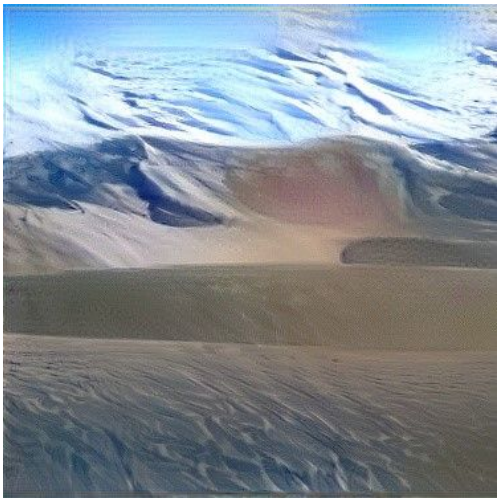


Style Image

We can observe that it has transferred the style (the icy surface) in parts onto the desert plain while still trying to maintain the texture of the sand. The colors have gotten all mixed up to produce a different sand color interspersed with hints of white ice; which we suspect would have been rectified with the use of image matting.



With Segmentation



No Segmentation

6. Conclusion

From our above described experiments, we realised that:

- Standard neural style transfer is not applicable to achieve photorealistic transfer.
- Performing only image segmentation, manual or automated, is not sufficient either. Although dividing the image into segments does help to keep the style transfers from spilling to other parts of the image (the sky does not have any snow), but quite evidently this is not enough to achieve photorealistic style transfer.
- Thus, including the matting laplacian term (photorealism regularisation) in the loss function is an integral process in order to get a better result, which would justify why it was used by the author in the first place.
- And finally it can be concluded that, to achieve deep photo style transfer, we need a combination of both image segmentation and masking, and photorealism regularisation to achieve a suitable result.

Given some more time, we would have liked to complete the implementation by incorporating both image segmentation and photorealism regularization. Studying about these concepts really augmented our knowledge about CNNs and image processing in general. In this project we tried to think independently of the original authors' work and attempted to arrive at our own conclusions, which unsurprisingly coincided with the previous research work to a large extent. As an ending note, we would like to appreciate all the past research that has been done in this field and has aided us in this enriching learning process.

7. Acknowledgements

7.1. https://github.com/LouieYang/deep-photo-styletransfer-tf
7.2. https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/

8. References

[1] L. A. Gatys, A. S. Ecker, and M. Bethge. Image style transfer using convolutional neural networks. In Proceedings of the IEEE Conference on

Computer Vision and Pattern Recognition, pages 2414–2423, 2016.
[2] Luan, Fujun et al., 2017 Deep Photo Style Transfer
[3] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. IEEE Transactions on Pattern Analysis and Machine Intelligence, 30(2):228–242, 2008.