Style Transfer using Convolutional Neural Networks

Anish Shenoy
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
ashenoy3@illinois.edu

Dhruv Gelda
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
gelda2@illinois.edu

1. Introduction

The goal of this project is to generate synthetic images which smoothly blend the texture information of an artwork with the semantic content of a photograph. In recent years, the adoption of deep neural networks for supervised learning problems has led to significant breakthroughs, particularly in the field of image classification [8]. One of the primary reasons for this success is due to the large number of stacked layers [9]. It has been observed that the shallower layers are trained to identify low level features such as edges and gradients, whereas the deeper layers identify higher level features of the image [11]. Therefore, when an image is passed through a trained convolutional neural network, the output of deeper layers allows us to extract the semantic information of the image. It is also possible to extract the texture of an image from the correlations between the feature maps of a given layer [4]. The correlation from shallower layers give fine-scale textures whereas deeper layers provide large scale texture as shown in Fig 1. Here, we demonstrate a way to synthesize new images by smoothly blending the content and texture information.

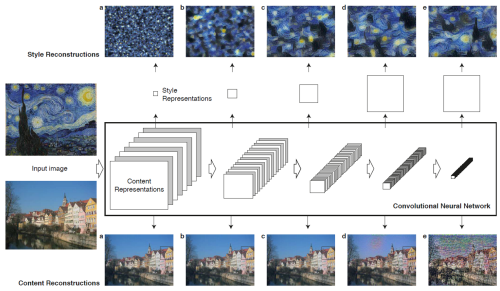

Figure 1. Style and feature reconstructions of an image using a pre-trained VGG model. The top row displays the textures extracted from the style image ('Starry Night' by Vincent Van Gogh) at various depths of the network, whereas the bottom row represents the semantic content extracted from the content image. As we move deeper into the network, we recover larger scale features. Image taken from [6].

2. Implementation

In this project, we implemented the style transfer method proposed by Gatys et al [6] using Tensorflow [1, 2, 7, 3]. We use a pre-trained VGG-19 network [9] to extract the texture from an artwork and the semantic information from a photograph. Each layer in the VGG-19 network defines a non-linear filter bank and a given input image is encoded in terms of the filter responses to that image. The activations from each layer can be stored in a matrix, $F^l \in \mathcal{R}^{N^l \times M^l}$, where N^L is the number of distinct filters and M^l is the size of feature maps. Along the depth of the network, these feature representations become increasingly sensitive to the actual content of the image. Therefore, we define content loss function, which captures how closely the synthesized image $\vec{x}$ matches the semantic information of the photograph $\vec{p}$ at a layer l sufficiently deep into the network:

$$\mathcal{L}_{content}(\vec{p}, \vec{x}, l) = \frac{1}{2N^l M^l} \sum_{i,j} (X_{ij}^l - P_{ij}^l)^2, \quad (1)$$

where, P^l and X^l are the feature representation of the photograph and the generated image from layer l.

We obtain the style representation of the image from the correlations of filter responses at layer l. The feature correlations are given by a Gram matrix, $\mathcal{G}^l \in \mathcal{R}^{N_l \times N_l}$, where $\mathcal{G}_{ij}^l$ is the dot product between the vectorized feature maps F_i^l and F_j^l in layer l:

$$\mathcal{G}_{ij}^l = \sum_k F_{ik}^l F_{jk}^l, \quad (2)$$

We use this Gram matrix to define the contribution to the style loss function from the layer l in the network:

$$E^l = \frac{1}{4(N^l M^l)^2} \sum_{i,j} ((\mathcal{G}^l)_{ij}^a - (\mathcal{G}^l)_{ij}^x)^2, \quad (3)$$

where, $(\mathcal{G}^l)^a$ and $(\mathcal{G}^l)^x$ are the style representations of the artwork and the generated image from layer l.

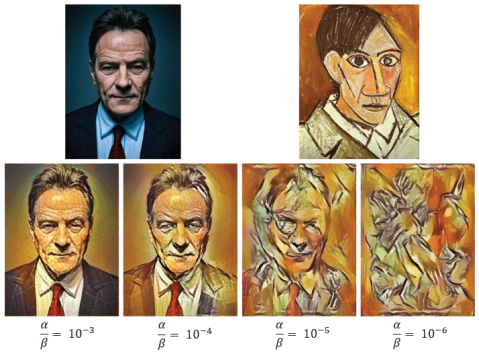

Figure 2. The effect of the α/β on the final synthesized image. The top row shows the content image (left) and the style image (right). On the bottom row, the ratio α/β for each synthesized images decreases from left to right. Thus the leftmost image has a dominating content component whereas the rightmost image has a dominating style component. The content image was taken from [10], while the style image is 'Self portrait', Pablo Picasso, 1907.

The style loss function is given by

$$\mathcal{L}_{style}(\vec{a}, \vec{x}, l) = \sum_{l} w_l E^l, \tag{4}$$

where w_l is the weighting factor given to each layer l in the network. The total loss function is given by

$$\mathcal{L}_{total}(\vec{p}, \vec{a}, \vec{x}) = \alpha \mathcal{L}_{content}(\vec{p}, \vec{x}) + \beta \mathcal{L}_{style}(\vec{a}, \vec{x}), \tag{5}$$

The ratio of hyper-parameters α and β control the relative contribution of the photograph and the artwork in the synthesized image as shown in Fig.2. For a large value of $\alpha/\beta = 10^{-3}$, the synthesized image preserves most of the semantic information of the photograph but the style representation of the artwork is not well-matched. On the other hand, for a smaller value of $\alpha/\beta = 10^{-6}$, the synthesized image loses the semantic information entirely and produces a texture that matches well with the style of the artwork. The ratio α/β therefore needs to be tweaked in order to produce a visually appealing image that simultaneously matches the photograph and artwork representation. The ideal value of α/β is not universal and needs to be adjusted for different style and photograph pairs, which was one of the primary challenges in this project.

We optimize the value of α and β for several artworks by doing a parametric sweep and judging based on visual appearance. The results are shown in Fig.3.

3. Color Preservation

With the method described above, we transfer both the color scheme and brush strokes pattern of the artwork onto the photograph. This does not preserve the color information of the original photograph. Therefore, we explored the

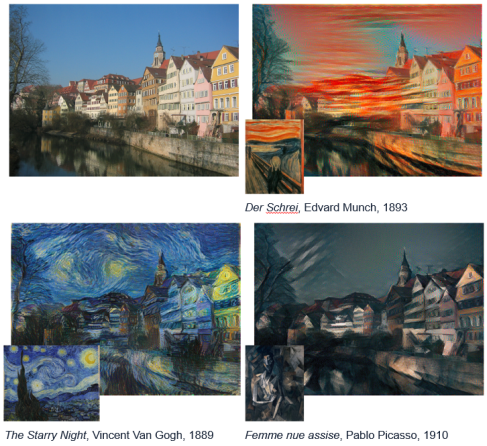

Figure 3. Application of three different styles to a photo of Neckarfront in Tübingen, Germany (top left). The top right photo shows the content rendered in the style of 'The Scream', the bottom left shows the content rendered in the style of 'Starry Night', and the bottom right shows the style from 'Femme nue assise' applied to the content image.

possibility of transferring just the pattern of brush strokes of the artwork onto the photograph. In order to do so, we implemented the color histogram matching procedure described in [5]. In this procedure, a linear transformation is applied onto the colors of the artwork so that the mean and covariance matrix of the modified artwork matches with respective values of the photograph. Since, there are many transformations that satisfy these constraints, a Cholesky decomposition is used to determine the transformation matrix. The results of the color preservation technique are shown in 4.

Figure 4. Style transfer while preserving color. The top row shows the style of 'Femme nue assise' applied to an input photograph of a city skyline. The bottom row shows the style of 'Starry night over the Rhône' applied to another input photograph. Both transfers preserve the color while transferring the style. The content photos were taken from [5].

4. Contributions

Both of us contributed equally to this project. We worked together while implementing the style transfer code and discussed the choice of various hyper-parameters, as well as which approach to use for color preservation. We split up the synthetic image generation tasks so that we could cycle through several hyper-parameter combinations.

5. Acknowledgement

We acknowledge the free computing credits on AWS made available by AWS Educate and the Github education pack.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016.

[2] M. Chang. neuralart-tensorflow. `https://github.com/ckmarkoh/neuralart_tensorflow`.

[3] L. Gatys. Pytorchneuralstyletransfer. `https://github.com/leongatys/PytorchNeuralStyleTransfer`.

[4] L. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 262–270, 2015.

[5] L. A. Gatys, M. Bethge, A. Hertzmann, and E. Shechtman. Preserving color in neural artistic style transfer. *arXiv preprint arXiv:1606.05897*, 2016.

[6] 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.

[7] J. Johnson. neural-style. `https://github.com/jcjohnson/neural-style`, 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[10] B. Trent. `https://damnuglyphotography.files.wordpress.com/2016/05/bryan_cranston_0095.jpg`, 2016. [Online; accessed May 3, 2017].

[11] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.