# Style Match

Yuqian Wang

*Cooper Union for the Advancement of Science and Art*
*wang2@cooper.edu*

**Abstract**

Style transformation tasks involve a content image and a style image. In recent years, explorations have been made to improve the quality of the generated image by achieving better resolutions [1]. We take this problem from a different perspective. Instead of focusing on the result image, we refine the choice of a style image by finding a best style image for a specific content image. The style of an arbitrary style image may not fit the content image. We use a style image which is most similar to the content image as the best fit. The system produces good output images during the test.

## 1 Introduction

Neuron networks have been used in artistic style transfer [1, 2, 4, 5, 6, 7]. Two images are fed into a convolutional neuron network and a new image is generated with the content of one image (known as the content image) and the style of the other (known as the style image). Usually, a photo is chosen to be a content image, and a painting is used for a style image. Researchers have worked on the improvement in resolutions of the generated images. An alternative approach to improve the performance of the style transfer system is to have a better style image, given a content image. A painting has mainly red color may not be a good fit for a photo of a beach. Hence, we want to find a best style match for a content image.

Because the idea of best match is subjective, we can only find possible best matches. A style image similar to the content image may work well in this case. We search for an image which looks like the selected content image in a candidate style image dataset. The chosen style image, together with the content image, is put into the style transfer network to generate a desired outcome. The code is on *https://github.com/yucy96/StyleMatch*.

# 2 Related work

## 2.1 Style transfer

A style transfer approach was first proposed in 2015 by L. Gatys, A. Ecker, and M. Bethge [2]. The method is based on VGG-19 network, which achieves a good accuracy in visual object recognition [3]. The paper introduces the loss function combining both content loss and style loss. By constraining both losses, the model retains the content of one image and the style of another. Researchers improve the style transformation task by increasing the training speed and the resolution of the generated image[1]. Perceptual loss functions are used in training feed-forward networks, which largely improves the training speed of a single style [1]. Another group extends the single style transfer to multiple styles, and argues that many styles share some degree of computation, which reduces the time training new styles [4]. The network in [2] uses Gram matrix to make the generated image similar to the style image. Other approaches have been tested on the job as well [5, 6, 7].

## 2.2 Similar image search

Similar image search has been used in searching engines to eliminate similar searching results. Perceptual hashing algorithms, including perceptive hashing, average hashing, and difference hashing, can fast transform the images to fingerprints and obtaining the similarity level through hamming distance computation [8]. Another algorithm to match an image with another is Scale-invariant feature transform using Gaussian convolution [9]. There are also approaches using convolutional neuron networks [10]. When applying a similar image search algorithm, we mainly considers the speed due to time constraints, so perceptual hashing is used in this project.

# 3 Dataset

The dataset is collected from the Painter by Numbers competition at Kaggle [11]. The original dataset tags all its images by their styles such as realism. Images tagged by "impressionism" are extracted from the original dataset and form the candidate dataset for the style matching system. Impressionism works are used as candidate style images due to their focus on color and texture instead of contours, which matches the expectations for style images. Other art styles are tested as well but fail to achieve desirable results. The dataset at Kaggle is divided into 10 subsets. During the test, impressionism works in subset 5, 6, 7, and 8 are used as potential style images.

# 4 Structure

We aim to find a painting similar to the input photo so that the system can generate a better output image. The style matching system consists of image similarity computation and style transfer. The most similar image is found in the dataset and fed into the style transfer network.

## 4.1 Image similarity

The image similarity model obtains the similarities between the input image and each painting in the candidate dataset. The most similar painting is chosen as the style image for the style transfer network.

Several image similarity models have been considered for this process. One approach is to use siamese network to extract the features of the images and calculate the similarity [10]. Because training convolutional neuron networks is time-consuming, we do not apply this algorithm to solve the image similarity problem. Instead, we use a much faster algorithm, perceptual hashing [8].

Perceptual hashing generates fingerprints of images and hashes the fingerprints into two-byte hexadecimal codes as features of the images. Before the feature extraction, all the images are reshaped into a smaller size in order to save future computation. We use $8 \times 8$ pixels as the universal size. The resized images are put into grayscale, so the three RGB channels are reduced to one channel. Discrete cosine transfer (DCT) is applied to the images and gets $8 \times 8$ DCT matrices. The matrices are resized to 1-D vectors. For values in a vector, if one value is larger than the vector's average, it returns one; if smaller, it returns zero. Hence, an image is turned into a 64 bit-code [8].

Both the input photo and the images in the dataset are transferred into two-byte hexadecimal codes. The similarity is obtained by computing the hamming distance between two image codes. The image with the smallest hamming distance from the input image is chosen to be the style image.

## 4.2 Style transformation

After learning the most similar style image, the style and content images are put into the style transfer network [2]. The style transfer model is based on the paper by Gatys, et. al [2]. It uses the convolutional layers of a VGG-19 network to extract features from the images. Only the convolutional layers of the original VGG network are applied to the style transfer model. As VGG-16 and VGG-19 have similar performance during tests [3], we use VGG-16 instead to have fewer parameters. According to the model performance test in the paper by Gatys, et. al, the style features from deeper layers do not necessarily achieve better results than shallower layers. Hence, we use the style features from the 4th layer instead of the deepest layer in the network.

The model is constrained by loss functions 1, 2, and 3. The loss of content is the mean square error (difference between the features of the content image, F, and the generated image, P) [2].

$$L_{content} = \frac{1}{2}\sum_{ij}(F_{ij} - P_{ij})^2 \tag{1}$$

The loss of style is the mean square errors (difference between the features of the style image, A, and the Gram matrix, G) multiplied by the weighing factor of each layer, (w). G is the sum of the inner products between two feature maps in each layer. The Ns are the multiplications of the feature maps numbers and sizes [2].

$$L_{style} = \sum_{l=0}^{L} \frac{1}{4N_l^2} w_l \sum_{i,j}(G_{ij}^l - A_{ij}^l)^2 \tag{2}$$

The total loss function is the summation of the two loss functions with hyperparameters $\alpha$ and $\beta$ [2]. The $\frac{\alpha}{\beta}$ ratio used in the test is 1/2000.

$$L_{total} = \alpha L_{content} + \beta L_{style} \tag{3}$$

Figure 1: The left is the resized content image. The middle is the resized style image found by the image similarity model. The right is the generated style image.

# 5 Discussion and future work

The test focuses on the similar paintings found. The paintings are not labelled, so there is no ground truth for the results of similar image search. Hence, we take a qualitative analysis instead of a quantitative evaluation. As in Figure 1, the building in the shape of a red triangle occupies more than half of the frame. The style image found also has a girl in a similar color and shape. Perceptual hashing generates the images' fingerprints as their features for similarity checking. Information, such as colors and strokes, is lost during the hashing. The fingerprints represent the structures of the images, for example, the positions of the objects in the frame. The test examples meet the expectation of similarity in image structure.

Another test is to input a black-and-white photo into the system, which leads to an unexpected result. All the images in the candidate dataset are gray scaled before hashing, so a gray image can still be matched with a colored style image. Although the color information is simplified to one channel, it still eliminates candidates with very different color combination. Dark colors will not be matched with light colors. In Figure 2, the style image provides not only the style but also its color to the content image. The system reconstructs the colors of the sky and the building behind the cube. This test gives a possibility in an alternative color reconstruction method.

However, the missing of color still has an impact on the final performance. We plan to change the color similarity algorithm to remain the color information of the images. It is possible that having three color channels instead of one will require more time and space resources during the image searching

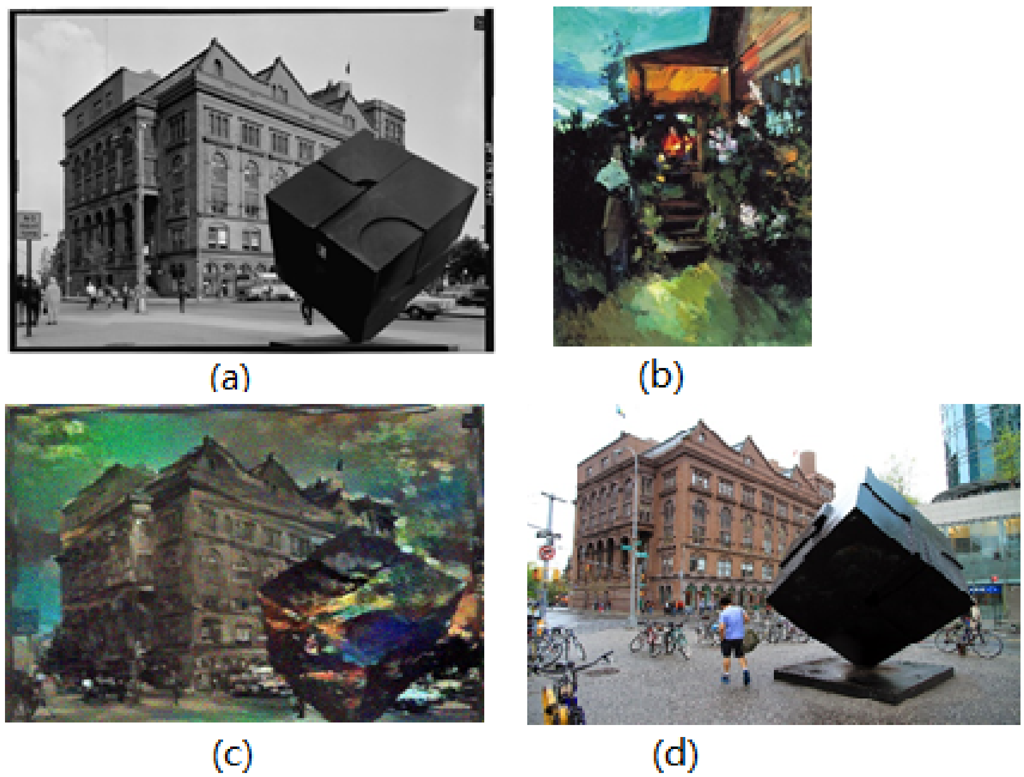


Figure 2: (a) is the content image taken in 1971 [12]. (b) is the style image found. (c) is the output image. (d) is a reference image taken in 2013.

process. Except the resource issue, deep neuron networks can be a better solution to extract features including colors and object types.

# References

[1] J. Johnson, A. Alahi, and F. Li, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," *ECCV*. 2016.

[2] L. A. Gatys, A. S. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," *arXiv:1508.06576v2*. 2015

[3] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556*. 2015.

[4] V. Dumoulin, J. Shlens, and M. Kudlur, "A Learned Representation for Artistic Style," *ICLR*. 2017.

[5] X. Huang, and S. Belongie, "Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization," *arXiv:1703.06868v2*. 2017.

[6] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," *ECCV*. 2016.

[7] P. Wilmot, E. Risser, and C. Barnes, "Stable and controllable neural texture synthesis and style transfer using histogram losses," *arXiv:1701.08893*. 2017.

[8] N. Krawetz, the "Hacker Factor" blog. Available at: *http://www.hackerfactor.com/blog/index.php?/archives/529-Kind-of-Like-That.html* [Accessed 20 Dec. 2018]

[9] D. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the International Conference on Computer Vision*. pp. 1150–1157. 1999.

[10] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese Neural Networks for One-shot Image Recognition," *Proceedings of the 32nd International Conference on Machine Learning*, Lille, France. 2015.

[11] Painter by Numbers: train. *https://www.kaggle.com/c/painter-by-numbers/data*. 2016.

[12] J. Boucher (1971). Cooper Union for the Advancement of Science and Art, Third and Fourth Avenues, Astor Place, Seventh Street 1971. Available at: *https://cdn.archpaper.com/wp-content/uploads/2016/11/120045pv.jpg* [Accessed 20 Dec. 2018].