# Neural Font Style Transfer

**Conference Paper** · November 2017

DOI: 10.1109/ICDAR.2017.328



**5 authors**, including:

Gantugs Atarsaikhan
Kyushu University


Brian Kenji Iwana
Kyushu University


Keiji Yanai
The University of Electro-Communications


Seiichi Uchida
Kyushu University

# Neural Font Style Transfer

Gantugs Atarsaikhan*, Brian Kenji Iwana*, Atsushi Narusawa†, Keiji Yanai†, Seiichi Uchida*
*Graduate School of Information Science and Electrical Engineering
Kyushu University, Fukuoka, Japan
Email: {gantugs.atarsaikhan,brian,uchida}@human.ait.kyushu-u.ac.jp
†University of Electro-Communication, Tokyo, Japan

***Abstract*—In this paper, we chose an approach to generate fonts by using neural style transfer. Neural style transfer uses Convolution Neural Networks (CNN) to transfer the style of one image to another. By modifying neural style transfer, we can achieve neural font style transfer. We also demonstrate the effects of using different weighted factors, character placements, and orientations. In addition, we show the results of using non-Latin alphabets, non-text patterns, and non-text images as style images. Finally, we provide insight into the characteristics of style transfer with fonts.**

## I. Introduction

Designing a digital font from scratch consumes lot of time, and requires professional skills. There are many elements to take into account, such as serifs, height, width, line width, etc. Moreover, the design process has many complicated steps, such as, sketching and digital vectorization. Due to that, automatic font generation would be an important tool for the future font design.

Recently, style transfer using neural networks has been an active field. There are numerous works on achieving style transfer between two image to generate new images. Gatys et al. [1] describes the neural style transfer method. Neural style transfer refers to the use of Convolutional Neural Networks (CNN) [2] to synthesize images using a content image and a style image. The texture and local features of the style image is applied to the structure of the content image. An example of neural style transfer is shown in Fig. 1. Johnson et al. [3] used a ConvDeconv style network to transfer styles of images in real time. Developed recently, Prisma[1] is a style transferring application. Isola et al. [4] use Generative Adversarial Networks (GAN) to map input images to synthesized output images.

The purpose of this paper is to propose a method of generating fonts using neural style transfer. The proposed method exploits the many already designed fonts in existence. By using neural style transfer between characters of input fonts, we explore the possibility of developing new fonts. We observe the basic performance of font generation by neural style transfer and evaluate the algorithm using various combinations of content fonts, style fonts, and parameters. We demonstrate the effects of using different weighted factors, character placements, and orientations. In addition, we show the results of using non-Latin alphabets, non-text patterns, and non-text images as style images. Finally, we provide insight into the characteristics of style transfer with fonts.

[1]https://prisma-ai.com/


Fig. 1. Neural style transfer example. Style of the style image has been transferred to the content image resulting transformed image.

The main contributions are summarized as follows.

1) This is the first trial of font generation using neural style transfer. Various results are shown for revealing the effect of style images.

2) Due to the use of simple black and white patterns, we can reveal characteristics of neural style transfer.

The remaining of this paper is organized as follows. Section II describes related work in font generation. In Section III, we describe neural style transfer and the underlying mechanism. The experiments and results are detailed in Section IV. Finally, Section V draws the conclusion and lays the future research.

## II. Related Work

There have been attempts for automatic font generation from literature. One approach is to use example fonts to determine predictive features [5], [6]. Another method is to generate fonts based on the interpolation of example fonts [7], [8]. Work has also been done by automatically generating new fonts from samples of user handwriting [9].

The idea of the proposed method is to use neural style transfer for the application of font generation. Neural style transfer [1] method is an active field and there have been many attempts on transferring the style of art to images [10]. Ulyanov et al. [11] refines neural style transfer to focus on only transferring the texture of the style image. Li and Wand [12] improves texture synthesis using Markovian GANs. In addition, there has been work to improve neural style transfer methods by combining semantic information [13], [14], preserving content image features [15], [16], [17], and increasing calculation speed [18].

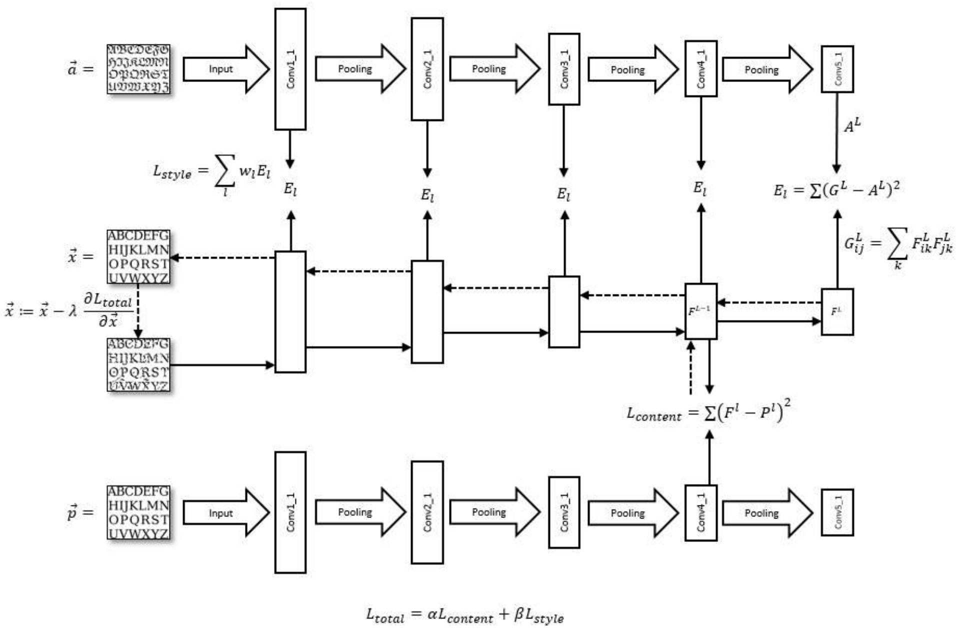


Fig. 2. The process flow of font generation using neural style transfer.

The advantage of neural style transfer is that no heuristic rules or handcrafted features are used. It has been used for a variety of domains, such as video [19], [20], sparse images to dense images [21], and portraits [22], [23]. The benefit of using neural style transfer for fonts is that the features of the style fonts are automatically learned and can be applied to other fonts.

## III. Neural Font Style Transfer

The basic principle of neural style transfer using a CNN based method to determine the content representation of a source image and feature maps of a style image to synthesize them using a loss function.

### A. Neural Style Transfer

In order to determine content representation and feature map of images, a deep CNN, such as Visual Geometry Group network (VGGNet) [24], is used. The VGGNet is pre-trained for natural scene object recognition. Pre-training deep neural networks has shown [25] to be useful for transferring learned features to other domains. In this paper, all the results are computed using VGG-16 [24], a 16-layer VGGNet.

The behaviour of font generation using neural style transfer is shown in Fig. 2. First, content image $\vec{p}$ runs through the neural network, on every layer the content representation $P^l$ is calculated. Then, generated image $\vec{x}$, initialized with content image $\vec{p}$, runs through the network and it's content features $F^l$ and style features $G^l$ are computed. Using content features $F^l$ and content representation $P^l$, loss of content on every layer is calculated as,

$$L_{content}(\vec{p}, \vec{x}, \vec{l}) = \frac{1}{2} \sum_{ij} (F^l_{ij} - P^l_{ij})^2. \quad (1)$$

Next, style image $\vec{a}$ runs through the neural network and its style representation $A^L$ is calculated. Using the style representation $A^L$ and style features $G^L$, style loss on each layer can be shown as,

$$E_l = \sum (G^L - A^L)^2. \quad (2)$$

Hence, the total style loss with weighting factors on each layer is calculated as,

$$L_{style}(\vec{a}, \vec{x}) = \sum_{l=0}^{L} w_l E_l. \quad (3)$$

### B. Font Generation using Neural Style Transfer

To transfer style of style image $\vec{a}$ to content image $\vec{p}$, the new generated image $\vec{x}$ is initialized with $\vec{p}$, and the total loss $L_{total}$ is minimized. Given $\alpha$ and $\beta$ are weighted factors for content and style reconstruction, the total loss is expressed as,

$$L_{total} = \alpha L_{content} + \beta L_{style}. \quad (4)$$

With regard to the weighted factors, to generate an image more similar to content image, $\alpha$ should be higher and $\beta$ should be lower. On the other hand, to generate an image more likely to style image, $\alpha$ should be lower and $\beta$ should be higher.

## IV. Experimental Results

In this section, we detail the experiments and results on various combinations of content images and style images. In addition, we observe the effects of the weighted factors, $\alpha$ and $\beta$.

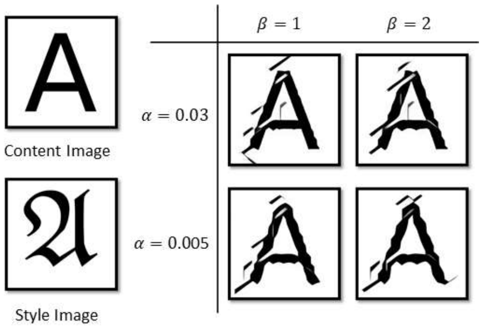


Fig. 3. Example of using a single character as the content image and style image. **(For more results, see our supplementary materials at https://gttugsuu.github.io/Neural-Font-Style-Transfer/)**

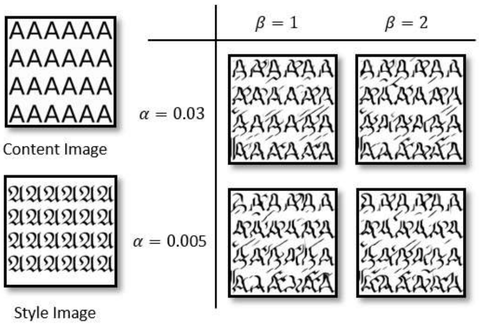


Fig. 4. Example of using multiple characters as the content image and style image. **(For more results, see our supplementary materials at https://gttugsuu.github.io/Neural-Font-Style-Transfer/)**

### A. Placement of Characters in Input Images

The placement of the characters in the content and style images can have an effect on the result image. To determine effective character placement for font style transfer, we compared single character style transfer and multiple character style transfer in Figs. 3 and 4 respectively. Figure 3 shows that there is little difference in increasing the weight of the style $\beta$. This indicates that there is insufficient data and features for the neural style transfer to learn. Neural style transfer works by generalizing common local features to the structure of the content image. When using single character input images, there are not enough common features from the style image to transform the content image. In addition, the style features are too large for the network to learn. On the other hand, when there are many character inside input image, such as Fig. 4, there is enough style to transfer. On the same $\alpha$ and $\beta$ combinations, there is a noticeable effect from the style image. Due to these factors, it is beneficial to work with multiple characters rather than a single character.

### B. Strength of Weighted Factors

The most important aspects of neural style transfer are weighted factors $\alpha$ and $\beta$. In this experiment, we will observe different combinations of those factors. As mentioned before, if $\alpha$ is large, then generated font is more similar to the content

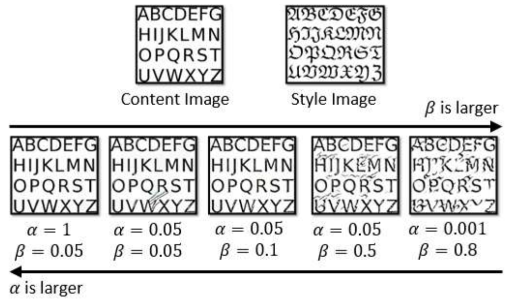


Fig. 5. The effect that the weighted factors $\alpha$ and $\beta$ has on style transfer.

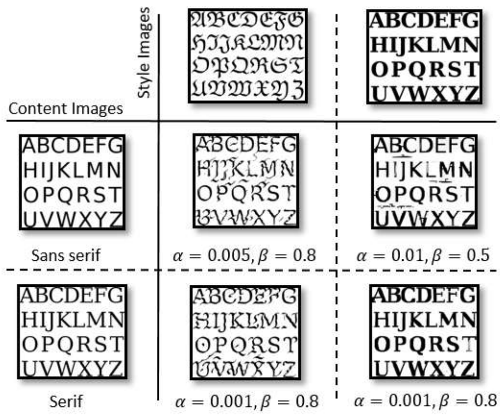


Fig. 6. The result of stylized fonts as the style images and regular fonts as the content images.

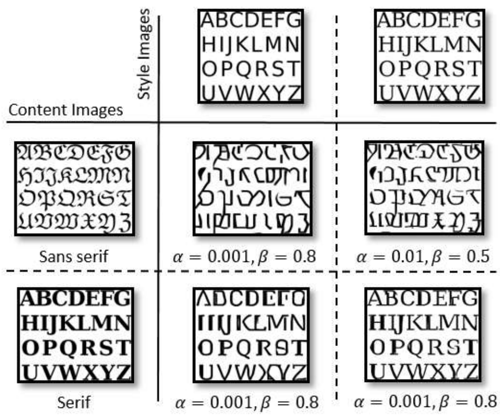


Fig. 7. The result of regular fonts as the style images and stylized fonts as the content images.

font, and if $\beta$ is large, then it is more similar to the style font. This can be observed in Fig. 5. In other words, the ratio between $\alpha$ and $\beta$ is important.

### C. Transferring Font Features

First, we explore the possibility of transferring the style of heavily stylized fonts to plain serif and sans serif fonts. As shown in Fig. 6, the rough structure of content fonts remain

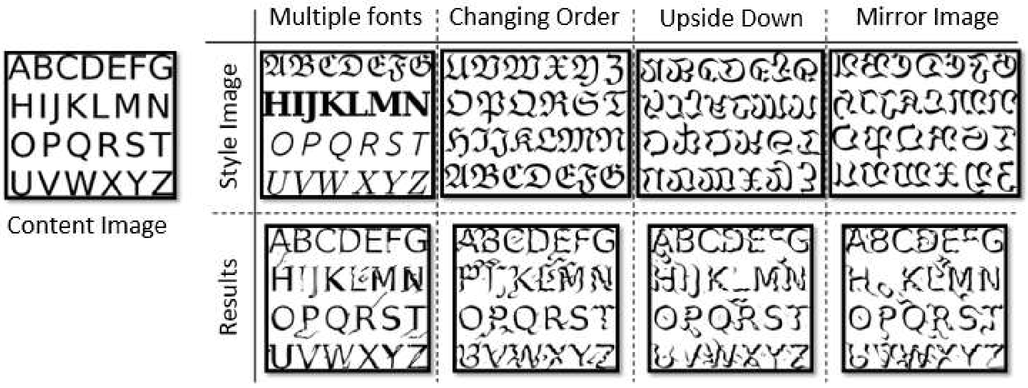


Fig. 8. The influence of the style image with multiple fonts and orientation transformations.

intact while the style from the style images is transferred. Additionally, the serif and sans serif features remain intact when the bold weight is applied.

Next, we observe the effect of applying the plain font styles to the complicated fonts in Fig. 7. The complicated fonts also maintained their overall structure, however in this trial, the font weight is normalized to same level as the style images. Again, by comparing the results of transferring the styles of the serif and sans serif fonts, we can observe the impact of the serifs. The characters resulting from the transferred serif style clearly contain serifs and the characters resulting from the transferred sans serif style do not.

### *D. Multiple Font Style Transfer*

When using multiple fonts in a single style image, the result, shown in Fig. 8, shows that insufficient style information was obtained by the neural style transfer network. The reason is that, since there is no dominant style, none of the styles are powerful enough to transform the content image. Thus, it is more effective to transfer a limited amount of style fonts to the content images.

### *E. Influence of Order and Orientation*

Figure 8 also shows the effects of changing the character order and orientation. While the results did show that the network was able to learn from the style images, the results from the upside down and mirrored style images created more illegible characters when compared to the ordered style characters in Fig. 6 and the misordered style characters in Fig. 8. This indicates that local features and individual character shapes are important for effective font style transfer.

### *F. Transferring Global Structure*

Common global structures, such as italics, can be transferred from style images. Figure 9 shows the effect of applying italicized versions of the respective serif and sans serif fonts. The results show that the neural style transfer was able to successfully transfer italics to some of the characters. However, there were some characters that degraded, such as the letters, ”G,” ”M,” ”Q,” and ”R” from the sans serif content font and ”N,” ”Q,” and ”R” of the serif content font.

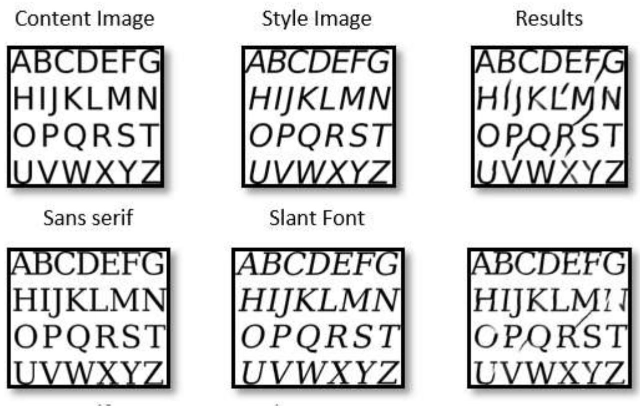


Fig. 9. The effect of transferring global structure.

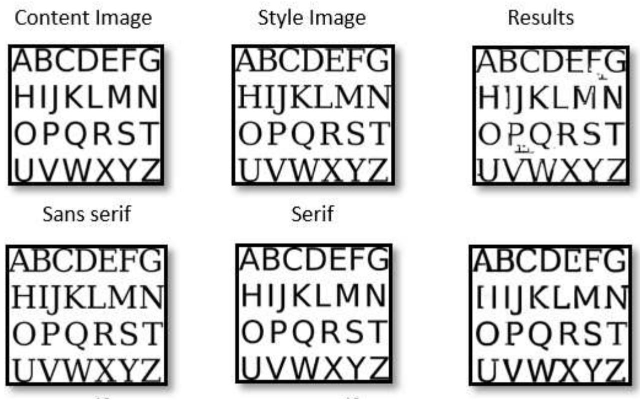


Fig. 10. The effect of transferring local structure.

### *G. Transferring Local Structures*

In addition, we explore the possibility of transferring local features, such as the presence of serifs. As opposed to global features, such as italics, which apply to the entire character, serifs exist only locally at particular stroke terminals. As shown in Fig. 10, when transferring a serif style to the sans

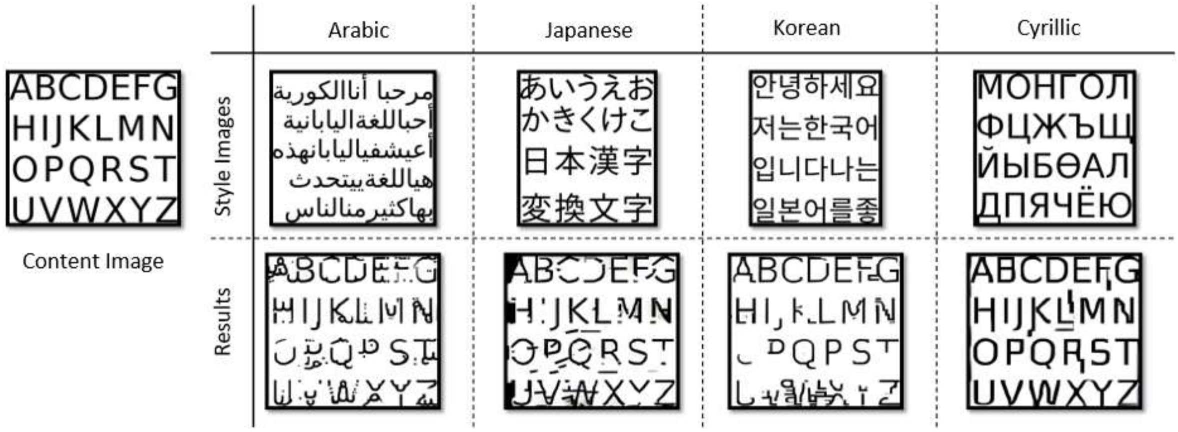

Fig. 11. Neural font style transfer from other languages to the Latin alphabet.

serif font, many of the characters picked up serifs. However, when applying the sans serif style to the serif font, the characters of the resulting image maintained the original serifs. As a result, we conclude that local features, such as serifs, are more easily applied than removed.

### *H. Transferring Styles from Other Languages*

Transferring style features from fonts with non-Latin character sets could be useful in order to discover new fonts. For this experiment, we attempted to transfer the styles from Arabic, Japanese, Korean, and the Cyrillic alphabet to the Latin alphabet. In the results of Fig. 11, Japanese and Korean had poor results. Some features of Korean can be seen in the ”W,” namely small circles and lines, however most characters are unchanged or heavily distorted.

Conversely, the style from Arabic characters and the Cyrillic alphabet were transferred to the Latin characters. Cyrillic is similar to the Latin alphabet, so it is unsurprising. However, Arabic is very different than Latin, but the results show many features from Arabic that were transferred. For example, dots and small curves. In particular, the Latin ”U” looks remarkably similar to Arabic. The possibility of using other alphabets as a source of style increases the number of potential new fonts for every language.

### *I. Using Styles from Non-Text Sources*

Font style transfer is not limited to learning from text sources. In order to discover new fonts, new sources of style can be used.

First, we show that simple patterns, such as tiled dots, can be used as styles to be transferred. In this experiment, dense and sparse dots are used as style images for the neural style transfer network. The results in Fig. 12 show the structures of the content font images being expressed using features from the dense and sparse dots. When the dots were sparse and sufficiently small, the characters became composed of the dots. And, when the style image had large dense dots, the result characters gained round dot-like features. However, the large dot experiment also yielded many illegible characters.

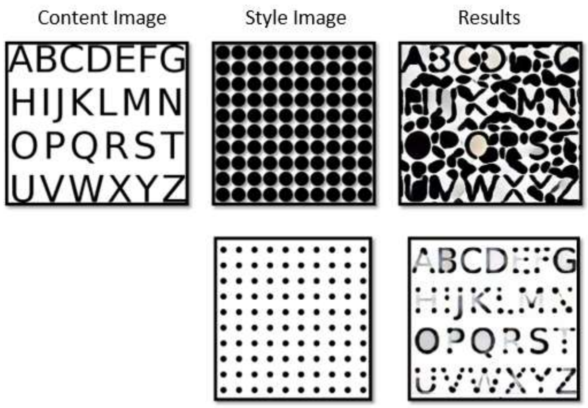

Fig. 12. Transferring styles from dot patterns.

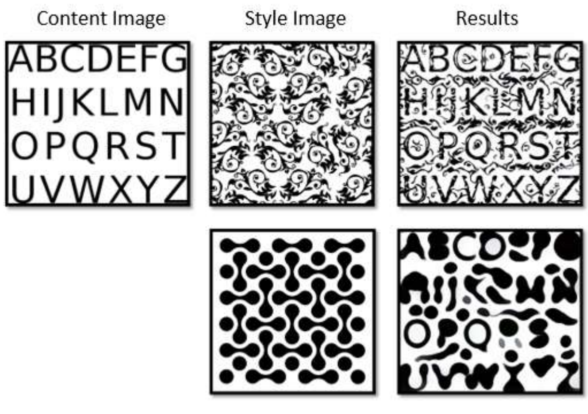

Fig. 13. Transferring styles from complex patterns.

Next, we experimented on complex patterns. In Fig. 13, we show the results of using a floral pattern and a bulbous pattern. The floral pattern resulted in characters with similar global structures to the font in the content image. However, the local features of the floral pattern, such as thin lines, were transferred to the resulting characters. In addition, a new

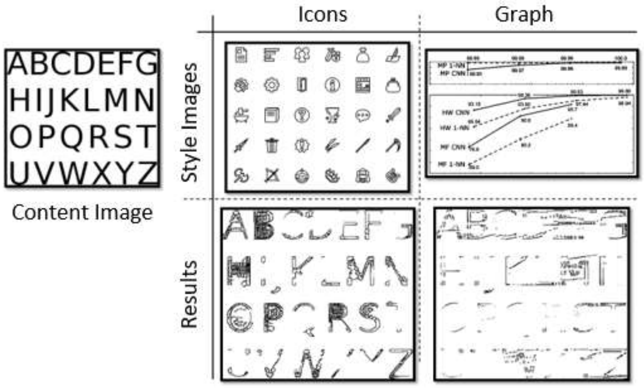


Fig. 14. Transferring styles from images.

floral pattern was injected into the whitespace between the characters. This shows that not just the target object features (i.e. the characters) is important, but the neural style transfer also considers the whitespace.

Finally, we used style images with non-text objects. We experimented on icons and graphs as style images. The results in Fig. 14 show characters with roughly similar structures as the content, but with the features of the style images. Both style images contain very thin lines and the results reflect it. When transferring graph style, there are some clear lines parallel to the graph generated, also there were small numbers from the graph's numbers.

## V. Conclusion

In this paper, we performed experiments and observed the performance of generating fonts using neural style transfer. We showed that it is possible to generate readable fonts using neural network approach. We observed that neural font style transfer is possible only if style images contain multiple characters. We demonstrated the effects of changing the direction of the style transfer, the orientation of the characters, and weighted factors. Also, we observed that both local and global character features can be transferred. However, it is easier to add local features, such as serifs, than it is to remove.

Finally, we showed that there is possibility to transfer styles from non-character patterns. For example, transferring icon styles would generate interesting fonts. When using patterns as style, if the difference between the content image and style image is large, many characters are illegible. However, the results showed that style features of non-character patterns were transferred successfully, and that using non-character style images can lead to the discovery of new fonts.

However, some of the results were illegible or heavily distorted. Future research includes reducing noise and deformations, exploring the transfer of style to single characters, using multiple style images to apply multiple sets of style features, and more experiments with non-character patterns. This research is useful for the discovery and design of new fonts.

## References


[1] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognition*, 2016, pp. 2414–2423.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conf. Comput. Vision*. Springer, 2016, pp. 694–711.

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

[5] R. Suveeranont and T. Igarashi, "Example-based automatic font generation," in *Int. Symp. Smart Graphics*. Springer, 2010, pp. 127–138.

[6] T. Tsuchiya, T. Miyazaki, Y. Sugaya, and S. Omachi, "Automatic generation of kanji fonts from sample designs," in *Tohoku-Section Joint Conv. Inst. Elect. and Inform. Engineers*, 2014.

[7] N. D. Campbell and J. Kautz, "Learning a manifold of fonts," *ACM Trans. Graphics*, vol. 33, no. 4, p. 91, 2014.

[8] S. Uchida, Y. Egashira, and K. Sato, "Exploring the world of fonts for discovering the most standard fonts and the missing fonts," *Int. Conf. Document Anal. and Recognition*, 2015.

[9] R. Suveeranont and T. Igarashi, "Example-based automatic font generation." in *Smart Graphics*. Springer, 2010, pp. 127–138.

[10] Y. Jing, Y. Yang, Z. Feng, J. Ye, and M. Song, "Neural style transfer: A review," *arXiv preprint arXiv:1705.04058*, 2017.

[11] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky, "Texture networks: Feed-forward synthesis of textures and stylized images." in *Proc. Int. Conf. Mach. Learning*, 2016.

[12] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *European Conf. Comput. Vision*. Springer, 2016, pp. 702–716.

[13] R. Yin, "Content aware neural style transfer," *arXiv preprint arXiv:1601.04568*, 2016.

[14] Y.-L. Chen and C.-T. Hsu, "Towards deep style transfer: A content-aware perspective." in *British Mach. Vision Conf.*, 2016.

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

[16] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman, "Controlling perceptual factors in neural style transfer," *arXiv preprint arXiv:1611.07865*, 2016.

[17] Y. Li, N. Wang, J. Liu, and X. Hou, "Demystifying neural style transfer," *arXiv preprint arXiv:1701.01036*, 2017.

[18] T. Q. Chen and M. Schmidt, "Fast patch-based style transfer of arbitrary style," *arXiv preprint arXiv:1612.04337*, 2016.

[19] A. G. Anderson, C. P. Berg, D. P. Mossing, and B. A. Olshausen, "Deepmovie: Using optical flow and deep neural networks to stylize movies," *arXiv preprint arXiv:1605.08153*, 2016.

[20] B. Joshi, K. Stewart, and D. Shapiro, "Bringing impressionism to life with neural style transfer in come swim," *arXiv preprint arXiv:1701.04928*, 2017.

[21] A. J. Champandard, "Semantic style transfer and turning two-bit doodles into fine artworks," *arXiv preprint arXiv:1603.01768*, 2016.

[22] A. Selim, M. Elgharib, and L. Doyle, "Painting style transfer for head portraits using convolutional neural networks," *ACM Trans. Graphics*, vol. 35, no. 4, p. 129, 2016.

[23] F. Luan, S. Paris, E. Shechtman, and K. Bala, "Deep photo style transfer," *arXiv preprint arXiv:1703.07511*, 2017.

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

[25] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in Neural Inform. Process. Syst.*, 2014, pp. 3320–3328.