ICIC Express Letters
Part B: Applications
Volume **10**, Number **3**, March **2019**

ICIC International ©2019 ISSN 2185-2766
pp. **203–209**

CALLIGRAPHY FONTS GENERATION BASED ON GENERATIVE ADVERSARIAL NETWORKS

Guozhou Zhang, Wencheng Huang, Ru Chen, Jinyu Yang and Hong Peng*

School of Computer and Software Engineering
Xihua University
No. 999, Jinzhou Road, Jinniu District, Chengdu 610039, P. R. China
zhgzh931015@foxmail.com; *Corresponding author: ph.xhu@foxmail.com

Received August 2018; accepted November 2018

Abstract. *Style transfer is a hot research topic in the field of image processing in recent years, but the current studies on style transfer mainly focus on the oil paintings, landscape paintings and other images. This paper extends the study of style transfer to the calligraphy fonts, and proposes a method based on generative adversarial networks (GAN). It uses GAN to learn the mapping between two training sets (i.e., Chinese famous calligraphy fonts and printed fonts), and then any calligraphy fonts can be automatically generated. In our experiments, the Wang Xizhi's calligraphy fonts dataset is used to train the GAN, and the trained-well GAN automatically generates the corresponding calligraphy fonts with Wang Xizhi's style. The experimental results demonstrate the feasibility of the proposed method.*

Keywords: Style transfer, Generative adversarial networks, Calligraphy fonts

1. **Introduction.** For a long time, more than a quarter of the world's population have used Chinese as a daily language. The Chinese writing has always been one of the most basic skills in education and communication in China. However, as an ancient writing art, Chinese calligraphy, which evolved from oracle bone script to large seal script, small seal script, and was fixed to cursive script in Han Dynasty, semi-cursive script in Jin Dynasty, has been exuding the charm of art all the time. Today, the excellent calligraphy is not only an artistic expression of language, but also can better reflect the level of personal self-cultivation.

Although the calligraphy style transfer is not investigated as extensively as image style, there are still some methods to generate Chinese fonts. The generation of Chinese fonts can be mainly represented by art calligraphy, printing [1-4] or personal handwriting generation [5-7]. Most previous works relied on hierarchical representations of simple strokes [1,5] which was used as the basis for representing Chinese fonts. StrokeBank [8] decomposed Chinese fonts into component trees. FlexiFont [9] scanned and processed the handwritten character images captured by the camera, and then formatted the fonts into TrueType font files. Recently, the awesome printing technique [3] has discussed the problem of producing special effects for typography, and used statistical data with highly regular spatial distribution of text effects to guide the synthesis process. Zi2zi [2] learned to transform fonts style using pix2pix [10] method with paired fonts images as the training data. However, in the task of generating calligraphy fonts, even if the same person writes the same Chinese font, there will be some changes. Therefore, learning the style of the calligraphy fonts should focus on the overall style of fonts rather than writing changes of a single font. So, it should more focus on the overall style of fonts rather than writing changes of a single font. So, it is more appropriate to use unpaired Chinese font data to deal with the generation problem of calligraphy fonts. Chang et al. [22] proposed a

DOI: 10.24507/icicelb.10.03.203

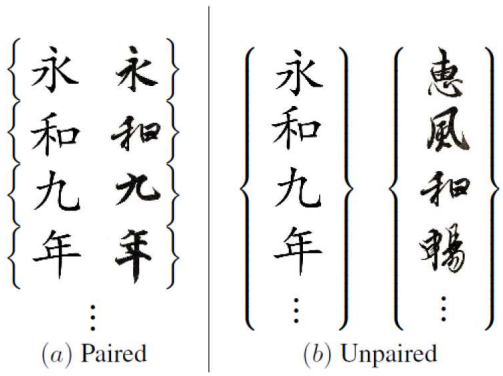

FIGURE 1. (a) Paired training data, and (b) unpaired training data

DenseNet CycleGAN [19,21] method based on unpaired image-to-image transfer to solve this problem. Their research focused on generating handwritten Chinese fonts. Figure 1 shows an example of paired training data and unpaired training data. The paired training data consist of the same printed font and calligraphy font, while unpaired training data does not necessarily include the same font.

Compared with handwritten Chinese fonts, calligraphy fonts have more complex features, such as joined-up writing, stroke weight difference and bifeng. Then, handwritten Chinese fonts are daily written fonts, and their style is very closer to the printed font. So the generation of handwritten fonts from printed fonts will be easier and it will be harder to extract calligraphy fonts style. In this work, we formulate the Chinese calligraphy fonts generation as a problem that learns a mapping from existing printed fonts to calligraphy fonts. We propose a style transfer method based on unpaired image to solve this problem.

2. **Preliminaries.**

2.1. **GANs.**

GAN [11,12] is a powerful generative model, and it has achieved amazing results on image generation [13,14], image editing [15], representation learning [16] and so on. Its main goal is to force the discriminative model D to help the generative model G generate fake data which is similar to the real data distribution. G and D are generally nonlinear mapping functions which are formalized by some network structures, such as multilayer perceptron (MLP) or convolutional neural networks (CNN) [17]. Given a random noise variable z following the simple distribution $p_z(z)$, the generative model G implicitly defines a generative distribution p_g by mapping z to $G(z)$ to fit the real sample distribution p_{data}. The discriminative model D, as a classifier, takes the real sample x and the generated sample $G(z)$ as input, and takes a scalar value as the probability output which indicates the confidence that the current input is real data or not. Thus, the quality of the data generated by G can be determined by D. When the input is a real training sample $x \sim p_{data}$, $D(x)$ expects to output a high probability. When the input is the generated sample $G(z)$, $D(G(z))$ expects to output a low probability. However, for G, it expects $D(G(z))$ outputs a high probability as much as possible so that D cannot distinguish between real data and the generated data.

The two models are alternately trained to form the competition and adversarial. The entire optimization process can be considered as follows:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{1}$$

2.2. **Cycle-consistent GANs (CycleGAN).**

The CycleGAN [19] method is based on pix2pix framework, which uses cGAN [18] to learn the mapping from input to output, but pix2pix relies on the paired training data. The innovation of the CycleGAN's is to

be able to realize this kind of transfer between the source domain and the target domain without establishing a one-to-one mapping between training data.

This method performs a two-step transformation on the original image: first, the original image is mapped to the target domain to get the first generated image, and then the first generated image is mapped from the target domain to source domain to obtain the second generated image, which eliminates the requirement for the paired images in the target domain. The generative network is used to map the original image to the target domain, and the quality of the generated image is improved by the discriminator.

3. **The Proposed Method.** Our task is to generate Chinese calligraphy fonts using unpaired source and target fonts. We collect the regular script font dataset $\{x_i\}_{i=1}^{N} \in X$ and Chinese calligraphy font dataset $\{y_j\}_{j=1}^{M} \in Y$. Our objective is consistent with CycleGAN, which is to learn style transfer between two domains without the paired training samples. Our model includes two mappings, G: $X \rightarrow Y$ and F: $Y \rightarrow X$, two discriminators D_x and D_y, D_x is used to distinguish the pictures $\{x\}$ and the transferred picture $\{F(y)\}$, while D_y is used to distinguish $\{y\}$ and $\{G(x)\}$. D_y encourages G to translate X into outputs indistinguishable from domain Y, and vice versa for D_x, F, and X. An example of the architecture of our method is shown in Figure 2.

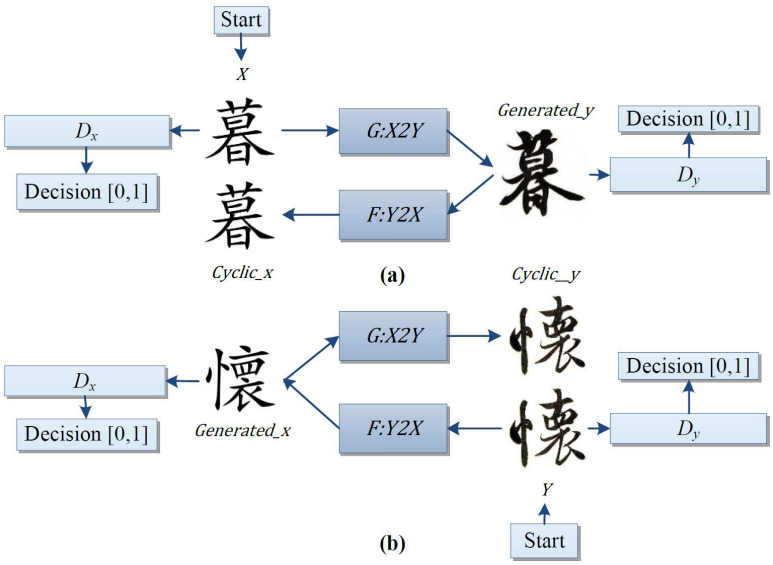

FIGURE 2. The model contains two mapping functions G: $X \rightarrow Y$ and F: $Y \rightarrow X$, and their adversarial discriminators D_y and D_x.

3.1. **Loss function.** Our total loss consists of two parts: adversarial loss, and cycle consistency loss. On the one hand, adversarial loss is applied in both mapping functions. For mapping G and its discriminator D_y, we use the following function:

$$l_{GAN}(G, D_y) = E_{y \sim p_{data}(y)}[\log D_y(y)] + E_{x \sim p_{data}(x)}[\log(1 - D_y(G(x)))] \tag{2}$$

where G tries to generate $G(x)$ that fits the image distribution in Y dataset, and D_y will try to distinguish the generated picture and the real picture. The objective function of the mapping G is:

$$G^* = \arg\min_G \max_{D_y} l_{GAN}(G, D_y) \tag{3}$$

We introduce a similar adversarial loss function for mapping F and its discriminator D_x:

$$F^* = \arg\min_F \max_{D_x} l_{GAN}(F, D_x) \tag{4}$$

On the other hand, theoretically, adversarial training can learn the mappings G and F respectively and generate the images matching the distribution of the target calligraphy

font data. However, because of using the mini-batch method, the extracted set is only a small part of the total dataset, and the two datasets are not paired at the same time. Therefore, the network can map the same set of input images to any arranged images in the target dataset. All the learned mappings can generate the images that match the target distribution. In other words, what the network learns by using a single adversarial loss is only a small part of the target dataset's data distribution rather than the entire target dataset. So using only one adversarial loss function cannot guarantee that any x_i can generate the $G(x_i)$ we want.

In order to reduce this limitation, CycleGAN added two "cycle consistency losses" that capture the intuition that if we translate from one domain to the other and back again we should arrive where we started: forward: $x \to G(x) \to F(G(x)) \approx x$ (picture x from data X), and backward: $y \to F(y) \to G(F(y)) \approx y$ (picture y from data Y). The structure is shown in Figure 2. The cycle consistency loss function is as follows:

$$l_{cycle}(G, F) = E_{x \sim p_{data}(x)} \left[||F(G(x)) - x||_1 \right] + E_{y \sim p_{data}(y)} \left[||G(F(y)) - y||_1 \right] \tag{5}$$

Finally, λ is added to control the relative importance of the two items, and the loss function of CycleGAN is given as follows:

$$l(G, D_x, F, D_y) = l_{GAN}(G, D_y) + l_{GAN}(F, D_x) + \lambda l_{cycle}(G, F) \tag{6}$$

3.2. **Implementation.** The generator in the CycleGAN has the layers that implement three stages of computation:

1) the first stage encodes the input via a series of convolutional layers that extract the image features;

2) the second stage then transforms the features from a style to another through one or more residual blocks [20];

3) the third stage decodes the transformed features using a series of convolutional layers to build an output image of the same size as the input.

The residual block used in the transformation stage consists of a convolutional layer, where the input is added to the output of the convolution. As a result, the characteristics of the output image (e.g., the shapes of objects) do not differ too much from the input. Table 1 provides the architecture of generator, and the proposed algorithm is given in Table 2.

TABLE 1. The architecture of the generator

Module	Specifications
Encoder	7*7 Convolution-batchNorm-ReLU, 32 filters, stride 1 3*3 Convolution-batchNorm-ReLU, 64 filters, stride 2 3*3 Convolution-batchNorm-ReLU, 128 filters, stride 2
Transfer	residual block residual block residual block residual block residual block residual block
Decoder	3*3 fractional-strided-Convolution-BatchNorm-ReLU, 64 filters, stride 1/2 3*3 fractional-strided-Convolution-BatchNorm-ReLU, 32 filters, stride 1/2 7*7 Convolution-BatchNorm-ReLU, 3 filters, stride 1

TABLE 2. The proposed algorithm

Algorithm Minibatch Adam training of CycleGAN.

(1) Draw a minibatch of samples $\left\{x^{(1)}, \ldots, x^{(n)}\right\}$ from domain X

(2) Draw a minibatch of samples $\left\{y^{(1)}, \ldots, y^{(m)}\right\}$ from domain Y

(3) Compute the discriminator loss on real images:

$$l_{real}^{(D)} = \frac{1}{n}\sum_{i=1}^{n}\left(D_x\left(x^{(i)}\right)-1\right)^2 + \frac{1}{m}\sum_{j=1}^{m}\left(D_y\left(y^{(j)}\right)-1\right)^2$$

(4) Compute the discriminator loss on fake images:

$$l_{fake}^{(D)} = \frac{1}{n}\sum_{i=1}^{n}\left(D_y\left(G\left(x^{(i)}\right)\right)\right)^2 + \frac{1}{m}\sum_{j=1}^{m}\left(D_x\left(F\left(y^{(j)}\right)\right)\right)^2$$

(5) Update the discriminators

(6) Compute the $Y \to X$ generator loss:

$$l^F = \frac{1}{m}\sum_{j=1}^{m}\left(D_x\left(F\left(y^{(j)}\right)\right)-1\right)^2 + l_{cycle}^{Y\to X\to Y}$$

(7) Compute the $X \to Y$ generator loss:

$$l^G = \frac{1}{n}\sum_{i=1}^{n}\left(D_x\left(G\left(x^{(i)}\right)\right)-1\right)^2 + l_{cycle}^{X\to Y\to X}$$

(8) Update the generators

4. **Experiments.** In this section, we evaluate our proposed method on Wang Xizhi calligraphy dataset. Main results are shown in this section.

4.1. **Environment.** This experiment is implemented by Torch 7 programming on Ubuntu 16.04 platform. The processor is Intel Core i7-6700, 3.4GHz, 8-core CPU, 16G memory, and the graphics card is NVIDIA GeForce GTX 1060 with 3G memory.

4.2. **Dataset.** In this paper, Wang Xizhi calligraphy fonts collected from the script of "lantingji xu" etc. are used as the calligraphy font dataset. The experiments also use the regular script font as the printed fonts dataset. Wang Xizhi set has 1000 training sample images and 100 test sample images. The regular script fonts dataset consists of a random input of 1000 Chinese characters. Both font datasets have undergone a unified standardization process. Each Chinese character image consists of 256*256 pixels. Figure 3 provides a part of dataset.

(a) (b)

FIGURE 3. (a) Regular script font dataset, and (b) Wang Xizhi's calligraphy font dataset

4.3. **Calligraphy fonts results.** The GAN learns the mapping between the Chinese regular script font image and the Wang Xizhi's calligraphy font image. After training, the test set of the regular script font image is input into the trained-well model. The calligraphy fonts results are shown in Figure 4 and it can be seen intuitively that the output fonts already have the style of calligraphy fonts. The generative model can learn the unique characteristics of the calligraphy style, such as strokes and joined-up, and these features can be transferred to the input fonts.

FIGURE 4. (a) The regular script font dataset as input, and (b) the output fonts which have Wang Xizhi's calligraphy fonts style

To further discuss the effect of the images generated by generative model, we compared these generated images with their corresponding real Wang Xizhi's calligraphy fonts, as shown in Figure 5. It can be seen that although overall the images generated by the generative model have good quality and the features of strokes and join-up, which means that it has largely learned the style of Wang Xizhi's fonts. However, after comparing with the original image, there are still many flaws in the details, such as the existence of noise, uneven thickness of strokes, incomplete strokes of individual image and strokes adhesions.

FIGURE 5. A comparison between the generative calligraphy fonts and Wang Xizhi's original calligraphy fonts for the same character: (a) and (c) are generative fonts, and (b) and (d) are Wang Xizhi's original fonts

5. **Conclusions.** GAN is a powerful unsupervised generative model, which satisfies the research and application requirements in many fields, and it has also injected new development impetus into these fields at the same time. This paper extends the research of GAN to the field of Chinese calligraphy. Some exploratory researches have been made in Chinese classical calligraphy, and some progress has been made, but the effect of individual generated font is not satisfactory. In the next step, we will further adjust and optimize the GAN network model to make the experimental results better. We will also explore GAN's research on ink painting in another aspect of Chinese classical art.

Acknowledgment. This work was partially supported by the Research Fund of Sichuan Science and Technology Project (No. 2018JY0083), Chunhui Project Foundation of the Education Department of China (Nos. Z2016143 and Z2016148), and Research Foundation of the Education Department of Sichuan province (No. 17TD0034), China.

REFERENCES

[1] S. Xu, F. C. Lau and W. K. Cheung, Automatic generation of artistic Chinese calligraphy, *Conference on Innovative Applications of Artificial Intelligence*, vol.20, no.3, pp.937-942, 2005.

[2] Y. Tian, *zi2zi: Master Chinese Calligraphy with Conditional Adversarial Networks*, https://github.com/kaonashi-tyc/zi2zi, 2017.

[3] S. Yang, J. Liu and Z. Lian, Awesome typography: Statistics-based text effects transfer, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.7464-7473, 2017.

[4] Z. Lian and J. Xiao, Automatic shape morphing for Chinese characters, *SIGGRAPH Asia 2012 Technical Briefs*, pp.1-4, 2012.

[5] S. Xu, T. Jin and H. Jiang, Automatic generation of personal chinese handwriting by capturing the characteristics of personal handwriting, *Conference on Innovative Applications of Artificial Intelligence*, pp.191-196, 2009.

[6] J. W. Lin, C. Y. Hong and R. I. Chang, Complete font generation of Chinese characters in personal handwriting style, *IEEE the 34th International Performance Computing and Communications Conference*, pp.1-5, 2016.

[7] D. Sun, T. Ren, C. Li et al., *Learning to Write Stylized Chinese Characters by Reading a Handful of Examples*, arXiv Preprint, arXiv:1712.06424, 2017.

[8] A. Zong and Y. Zhu, Strokebank: Automating personalized Chinese handwriting generation, *Association for the Advancement of Artificial Intelligence*, pp.3024-3029, 2014.

[9] W. Pan, Z. Lian and R. Sun, Flexifont: A flexible system to generate personal font libraries, *Proc. of the 2014 ACM Symposium on Document Engineering*, pp.17-20, 2014.

[10] P. Isola, J. Y. Zhu and T. Zhou, Image-to-image translation with conditional adversarial networks, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.5967-5976, 2017.

[11] I. Goodfellow, J. Pouget-Abadie and M. Mirza, Generative adversarial nets, *International Conference on Neural Information Processing Systems*, pp.2672-2680, 2014.

[12] J. Zhao, M. Mathieu and Y. LeCun, *Energy-Based Generative Adversarial Network*, arXiv preprint, arXiv:1609.03126, 2016.

[13] E. L. Denton, S. Chintala and R. Fergus, Deep generative image models using a Laplacian pyramid of adversarial networks, *Conference and Workshop on Neural Information Processing Systems*, pp.1486-1494, 2015.

[14] A. Radford, L. Metz and S. Chintala, *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*, arXiv preprint, arXiv:1511.06434, 2015.

[15] J. Y. Zhu, P. Krähenbühl and E. Shechtman, Generative visual manipulation on the natural image manifold, *European Conference on Computer Vision*, pp.597-613, 2016.

[16] T. Salimans, I. Goodfellow and W. Zaremba, Improved techniques for training gans, *Advances in Neural Information Processing Systems*, pp.2234-2242, 2016.

[17] J. Long, E. Shelhamer and T. Darrell, Fully convolutional networks for semantic segmentation, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.3431-3440, 2015.

[18] M. Mirza and S. Osindero, *Conditional Generative Adversarial Nets*, arXiv preprint, arXiv:1411.1784, 2014.

[19] J. Y. Zhu, T. Park and P. Isola, Unpaired image-to-image translation using cycle-consistent adversarial networks, *IEEE International Conference on Computer Vision*, pp.2242-2251, 2017.

[20] K. He, X. Zhang and S. Ren, Deep residual learning for image recognition, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.770-778, 2016.

[21] G. Huang, Z. Liu and K. Q. Weinberger, Densely connected convolutional networks, *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, vol.1, no.2, pp.1-3, 2017.

[22] B. Chang, Q. Zhang and S. Pan, *Generating Handwritten Chinese Characters Using CycleGAN*, arXiv preprint, arXiv:1801.08624, 2018.