Multimedia Tools and Applications
https://doi.org/10.1007/s11042-020-09709-5

An end-to-end model for chinese calligraphy generation

Peichi Zhou[1] · Zipeng Zhao[1] · Kang Zhang[2] · Chen Li[1] · Changbo Wang[1]

Received: 2 July 2019 / Revised: 23 July 2020 / Accepted: 25 August 2020 /
Published online: 21 October 2020

Abstract

A Chinese calligraphy copybook usually has a limited number of Chinese characters, far from a whole set of characters needed for typesetting. Therefore, there is a need to develop complete sets of Chinese calligraphy libraries for well-known calligrapher styles. This paper proposes an end-to-end network for character generation based on specific calligraphy styles. Specifically, a style transfer network is designed to transfer the style of characters, and a content supplement network is designed to capture the details of stylish strokes. Our model can generate high-quality calligraphy images without manually annotating data. To verify the generated calligraphy styles, a new dataset is constructed for experimental comparison between our method and two other baseline methods. Moreover, a user study is conducted to evaluate our generated calligraphy from a visual perspective. When the experiment participants are asked to distinguish the real calligraphy from generated samples, the correct rate was 53.5%. The results show that the calligraphy styles generated by our model are almost indistinguishable from the original works.

Keywords Calligraphy · Generative models · Font style transfer · Deep learning

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s11042-020-09709-5) contains supplementary material, which is available to authorized users.

✉ Changbo Wang
cbwang@sei.ecnu.edu.cn

Peichi Zhou
52184501009@stu.ecnu.edu.cn

Kang Zhang
kzhang@utdallas.edu

[1] East China Normal University, Shanghai, China

[2] University of Texas at Dallas, Richardson, TX, USA

1 Introduction

In ancient China, Chinese characters gradually evolved into an art form called calligraphy. Unlike painting and music, calligraphy is both the art of a given form and a common medium of communication. Calligraphy is widely used in graphic design, stone inscriptions, and product design. Ideally, individual calligraphy characters could be scanned into a font library, making it easy for calligraphy enthusiasts to learn and for designers to use. It is, however, impossible in reality, as none of the original calligraphy works contains the entire set of commonly used characters, minimally over 3000. The current solution involves designers mimicking the style of the calligraphy in manually making the missing characters. This manual process is tremendously time-consuming, since each Chinese character is constructed by one or more radicals. There are several ways to combine these radicals into a character, such as left-right structure, up-down structure, as shown in Fig. 1. The position of a radical also influences the style.

The Chinese language has more than 80,000 logographic characters, in contrast to a phonological language that has a limited number of letters. Considering that the structure of Chinese characters has undergone great changes in the course of development, Chinese calligraphy has also been enriched and diversified significantly. Mastering Chinese calligraphy is difficult, demanding years of practicing. Therefore, few modern calligraphers are able to design font libraries by imitating well-known master styles.

Recently, there are several attempts in generating calligraphy automatically. One approach is to build a stroke library by decompose Chinese characters into strokes, or design a stroke library directly [2, 15]. Then new calligraphy characters can be assembled by the strokes in the library. However, such methods ignore the overall structure of characters and depend largely on the effect of stroke extraction. Many calligraphy styles are drastically different from the standard regular fonts, with strokes deformed or even omitted completely, making stroke extraction algorithms fail. Another general approach to generate calligraphy is to use deep learning methods. Yue et al. divided the generation of calligraphy into two separate procedures, their method writing trajectories synthesis and font style rendering

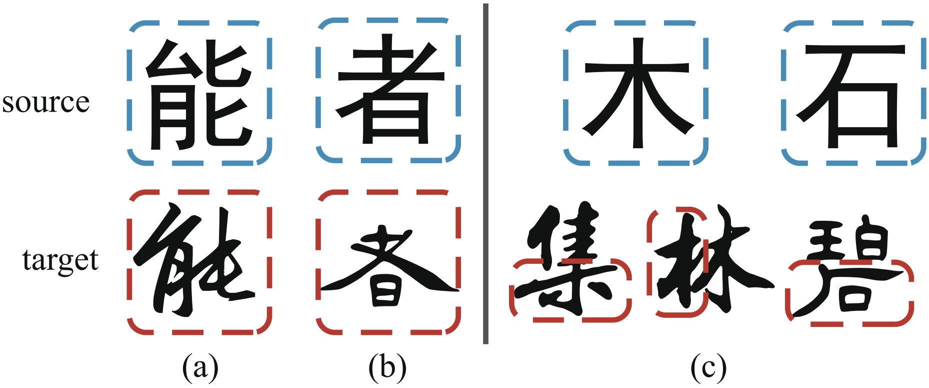


Fig. 1 By comparing the source (top) and target (bottom) characters in **a**, one may notice that the corresponding strokes have been greatly distorted, and several separate strokes are connected together. **b** shows another situation where the basic structures of the source (top) and target (bottom) are inconsistent. **c** shows that the structure in the blue box appears in the lower two characters. If the order of the radicals is different, the shape will be different

[11]. To be specific, the writing trajectories makes the character structure similar to the target style, and the font style rendering makes the character have the same strokes of the target style. That has achieved good results, but annotating the writing trajectories of character is very time-consuming.

This paper regards calligraphy generation as the problem of Image-to-Image Translation [10]. A model includes a content supplement network and a style transfer network is proposed. Specifically, the content supplement network extracts features of input images and generates content vectors, which contains the low-dimensional information of input images. And then, the style transfer network combines the features provided by content supplement network to convert the standard font into calligraphy. Since our model based on neural network, the parameters of the model are learned from the data by the network itself. The parameters of the loss functions are set by referring to [13]. Experiments demonstrate that the proposed content supplement network enriches the details of the generated calligraphy. Compared with previous methods [13, 33], our model minimizes manual handling while producing high-quality calligraphy images for multiple styles. The main contributions of our work are summarized as follows:

- An end-to-end model of calligraphy generation, that can transform characters from the standard printing form to a given calligraphy style directly without extracting individual strokes. Different from other methods of calligraphy generation, our model does not require paired data.
- Benefit from the content supplement network, the style and content of calligraphy do not need to be trained separately. Furthermore, this model can learn the writing trajectories and stroke style of complex calligraphy without extracting the writing method.

Our model and network approaches have been evaluated via an empirical study, participated by both professional calligraphers and ordinary people without much knowledge of calligraphy. The remaining part of the paper is organized as the following. Section 2 reviews the existing research on calligraphy generation related to our work. Section 3 introduces our model. In Sections 4 and 5, the generation effect is evaluated.

2 Related work

The following subsections first describe the relationship between style transfer and our approach. Image-to-Image Translation and several well-known generation models are discussed next. Finally, several calligraphy generation models are discussed.

2.1 Image style transfer

Style transfer works as the following. First, an image is input as content. Then another image is designated as the desired style. The output image is the result of transforming the input image's content into the desired style [12]. Several recent works have achieved good results in transferring colors [9], textures [5, 6], etc. yang et al. transfer the textures to plain characters [27], which can turn a character into an artistic one, while the shape of the character keep unchanged. As the transfer of calligraphy is accompanied by obvious deformation and movement, it is difficult to measure the content loss for different styles of calligraphy. Therefore this method of style transfer is not suitable for calligraphy generation.

2.2 Image-to-image translation

The process of transforming an image from a source to a target domain is called Image-to-Image Translation [10]. After the emergence of GAN [7], many classic problems used to be handled separately can be solved uniformly within a single model, such as image colorization [25], semantic segmentation [17, 28]. Different from the traditional GAN which takes noise z as input, for image translation, the generator's input is an image (noise may also be added according to specific tasks) [20]. The task of discriminator also changes. The discriminator should not only judge whether the distribution of generated images is consistent with the distribution in the target domain, but also determine whether the generated image corresponds to the input image.

$Pix2Pix$ [10] is suitable for many image translation problems. It extends the Image-to-Image problem to Pixel-to-Pixel format. However, in our task, the source font and target calligraphy are always different in positions. That is, the input and output are not identical in pixels. Experiments show that the characters generated using this method has many extra strokes. Therefore, the Pixel-to-Pixel format does not work for our purpose.

$CycleGAN$ [13] is proposed to solve the problems whose ground truth is not available. So this model is also called the unpaired version of Pix2Pix. Figure 2 explains paired and unpaired training data. The unpaired data does not require the standard font image and the calligraphy image to be the same character, while in paired data the characters must be the same. CycleGAN requires two sets of generators and discriminators that are structurally consistent. The two generators form the transfer of $X \rightarrow Y \rightarrow X$ and $Y \rightarrow X \rightarrow Y$ by a combination of different sequences, where X stands for the source and Y for the target. This model works well without annotating the original work of calligraphy, but the quality of generated calligraphy is unsatisfactory.

2.3 Calligraphy generation

Inspired by the success of deep learning in generative tasks, several researchers apply deep neural networks to calligraphy generation. "Rewrite" [24] adopts a top-down CNN structure to generate calligraphy. This model performs well in a limited number of styles but cannot handle any style with large deformation. As an upgraded version of "Rewrite", based on Pix2Pix, "Zi2Zi" [33] performs well for more complex calligraphy styles. Lyu et al. proposed a supervised network to guide the generation of calligraphy with realistic details [18]. Zheng et al. proposed a network to separate the style and content of calligraphy for training [21]. Wu et al. proposed an end-to-end network to transform the standard font to a desired style [31]. But all of these methods require paired data, and making pairs of training data is time-consuming and laborious. Different from these methods, our model could generate high-quality calligraphy without paired training data.

3 Proposed method

GAN consists of two parts, generator and discriminator, that compete with each other. The generator continuously improves the quality of the generated data to deceive the discriminator. The discriminator, on the other hand, continuously improves its discriminating capability to distinguish the generated data from the true data. Standard procedures often lead to the well-known problem of model collapse [7], where only a small number of real samples get represented in the output. In other words, all input images map to the same out-

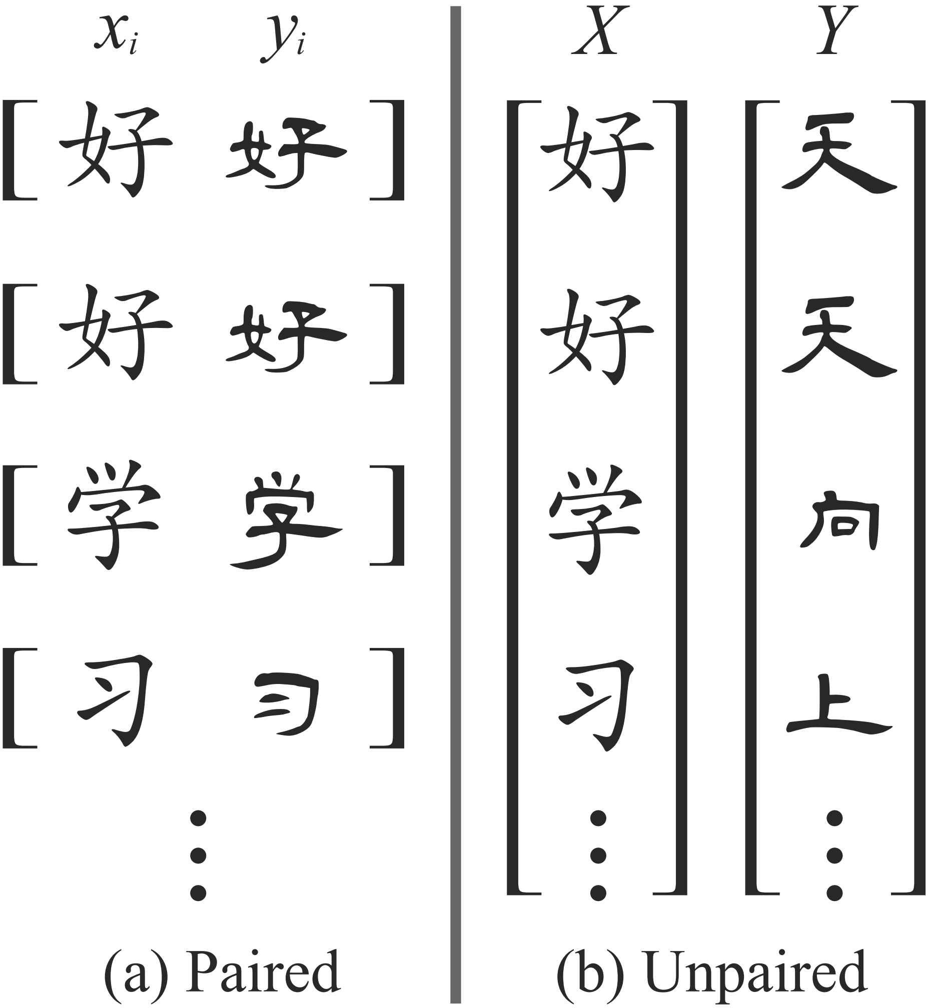


Fig. 2 **a** In the training set, one could pair data $(x_i, y_i)_{i=1}^{N}$ for N characters, where y_i is x_i's annotation. **b** In this paper, unpaired training data is considered as a source set X and a target set Y, with no matching for x_i and y_i

put image and optimization fails to make any progress. Therefore, it is necessary to maintain "cycle consistent", that is, a full cycle of forward and backward translations should return to the original state [13]. So that each input can get a corresponding output.

To ensure cycle consistency, our model consists of two sets of generator-discriminator networks of the same structure in the form of CycleGAN, that transform images between the standard font and calligraphy in opposite directions. The generator of the first set converts standard font images to calligraphy images, and the corresponding discriminator judges whether the generated calligraphy images have the same distribution of the real calligraphy images. The generator in the other set converts calligraphy images to standard ones, the discriminator judges whether the standard font images generated by the generator have the same distribution of the real standard ones. The PatchGAN is selected as the discriminator of the model [10], which is capable of learning the distribution of the target style, thereby guiding the generator to generate more realistic images.

The overall network architecture for calligraphy generation is shown in Fig. 3. The generator and discriminator are marked on the figure. Standard fonts and calligraphy represent the input and output of the model respectively. The next two subsection describes our design of the generator network (the blue block in the figure), and detailed function and working mechanism of the two sets of networks (between the blue and light-green blocks).

3.1 Generator design

This paper proposes an end-to-end network for calligraphy generation. Our model consists of two major components, i.e., contents supplement network (CSN) and style transfer network (STN). This section first introduces the network design and architecture, and then our model training method. Finally, the implementation details are discussed.

Firstly, STN is designed as the generator. In Image-to-Image Translation, a typical structure of the generator is encoder-decoder architecture [3]. Here, the encoder is a network that takes the input and outputs a feature vector representing the information in the input. The decoder is another network that takes the feature vector from the encoder, and gives the closest match to the intended output. In our approach, the output is different from the input both in position and shape after passing through the generator. Therefore, the simple encoder-decoder does not fulfill our requirements. A Residual Block [8] is added, allows a layer to feed into deeper layers and thus significantly enhance the feature processing capability of the network.

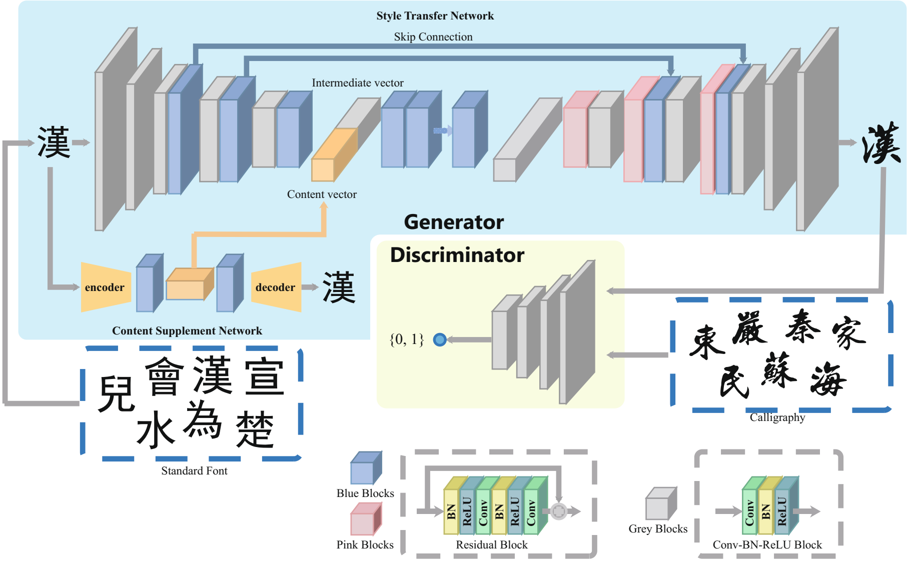


Fig. 3 Generator architecture for transferring characters from the reference font (Hei, regular typeset) into a desired calligraphy style. Specifically, a contents supplement network (CSN) is used to extract the content features of characters, and concatenate them with the features extracted by style transfer network (STN) to generate calligraphy. The pink and blue blocks in the figure are Residual Block [8], and the gray blocks are Convolution-BatchNorm-ReLU Block [18]

However, by simply adding the Residual Block in the intermediate layers only works well for texture conversion [18], hardly improves the transformation results from standard fonts to calligraphy styles. Further analysis reveals that the low-dimensional information of characters is lost before it enters the intermediate layers.

The former layers of the encoder extract low-dimensional features, while the latter layers extract high-dimensional features [19]. A Residual Block is applied into the encoding process, so that the network extracts the low-dimensional information fully before lost. It then transmits the encoded information to the decoder through Skip Connection [26], which could pass the intermediate layers and transfer low-dimensional information directly to the decoder. Via these operations, the generator is able to generate characters of calligraphy styles. One problem remains, i.e., some generated characters miss details, even small strokes.

To provide stroke details to STN, another network, CSN, a simple encoder-decoder network with a series of Residual Blocks in the intermediate layers is proposed. After experiment, this structure can roughly restore the input images to their original state, and the latent features in the intermediate layers contain the low-dimensional information that STN lacks. The image input to the STN is simultaneously input into the CSN, so that the features of the intermediate layers in CSN can contain the low-dimensional information that STN lacks at this time. The features of the intermediate layers in CSN are stored in the *content vector*, and those in STN in the *intermediate vector*. By concatenating the content vector with the intermediate vector, the details could be enriched.

As shown in Fig. 3, the input to our model includes two sets of characters, the standard set and the target styled set. In the encoder of STN, every gray block follows a blue block, except the first block. The blue one is a Residual Block, and the gray one is Convolution-BatchNorm-ReLU Block [18]. In the decoder of STN, the pink block means the Residual Block, the blue block is transferred from the encoder, and the gray block is Deconvolution-BatchNorm-ReLU block [16]. The intermediate layers of STN is a series of Residual Blocks. The encoder and decoder of CSN is also made up by gray blocks. The intermediate vector from the encoder of STN is concatenated with the content vector as the latent features. Having been processed by the intermediate layers and decoder, the latent features are finally decoded to a calligraphy image.

3.2 Loss functions

GAN can learn a mapping function between two domains X and Y with given samples x and y, which are sampled from domain X and Y respectively. The generator G aims to learn a mapping $G : X \rightarrow Y$ such that the images from $G(x)$ are indistinguishable from y, while D_Y aims to discriminate between y and $G(x)$. Formally, the objective between the generator G and the discriminator D_Y, which is called adversarial loss [7], can be expressed as:

$$\begin{aligned}\mathcal{L}(G, D_Y, X, Y) = & \ \mathbb{E}_{y\sim p_{data(y)}}[\log D_Y(y)] \\ & +\mathbb{E}_{x\sim p_{data(x)}}[\log(1 - D_Y(G(x))], \end{aligned} \tag{1}$$

G aims to minimize this objective while D_Y tries to maximize it, i.e. $min_G max_{D_Y} \mathcal{L}(G, D_Y, X, Y)$.

In our model, x is the standard font image sampled from domain X and y is the styled image sampled from domain Y during the training process.

Cycle consistency loss function [13] from CycleGAN is applied to ensure the learned function of STN could map each standard font image in X to its own Y. The schematic diagram of maintaining "cycle consistent" is shown in Fig. 4. In CycleGAN, to obtain a high-quality and well-trained generator G that can map X to Y, another generator that maps Y to X is also needed. Therefore a new generator F of the same structure as G is added. The cycle consistency loss is defined as:

$$\begin{aligned}\mathcal{L}_{cyc}(G, F) = & \mathbb{E}_{x\sim p_{data(x)}}[\|F(G(x)) - x\|_1] \\ & +\mathbb{E}_{y\sim p_{data(y)}}[\|G(F(y)) - y\|_1],\end{aligned} \tag{2}$$

where $G(x)$ converts the standard font to a calligraphy image, and $F(G(x))$ converts the generated calligraphy back to the standard font. For each standard font image x from domain X, G and F satisfy cycle consistency: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$. Similarly, for each calligraphy image y from domain Y, G and F should also satisfy cycle consistency: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$. $F(y)$ converts a calligraphy image to the standard font, and $G(F(y))$ converts the generated standard font back to a calligraphy image.

Corresponding to the two generators, there are two discriminators D_X and D_Y. They are used as the adversarial loss, that is defined as:

$$\begin{aligned}\mathcal{L}_{D_Y}(G, D_Y, X, Y) = & \mathbb{E}_{y\sim p_{data(y)}}[\log D_Y(y)] \\ & +\mathbb{E}_{x\sim p_{data(x)}}[\log(1 - D_Y(G(x))],\end{aligned} \tag{3}$$

$$\begin{aligned}\mathcal{L}_{D_X}(F, D_X, Y, X) = & \mathbb{E}_{x\sim p_{data(x)}}[\log D_X(x)] \\ & +\mathbb{E}_{y\sim p_{data(y)}}[\log(1 - D_X(F(y))].\end{aligned} \tag{4}$$

The discriminator D_Y judges whether the distribution of calligraphy images generated by the generator G is consistent with the distribution of the real calligraphy images according to the real calligraphy images in domain Y. Correspondingly, the discriminator D_X judges whether the distribution of the standard font images generated by the generator F is consistent with the distribution of the real standard font images in X.

Previous works [22] have proved that except the adversarial loss, the discriminator can also improve the quality of the generated image by calculating the L_1 distance between the input and target images. Since our data is unpaired, there no image y for image x to calculate the L_1 distance, so as image y. Thus L_{tid} [30] is used to calculate the L_1 distance between y and $G(y)$ (x and $F(x)$) as an alternative. The L_{tid} loss is defined as:

$$\begin{aligned}\mathcal{L}_{tid} = & \mathbb{E}_{y\sim p_{data(y)}}[\|y - G(y)\|_1] \\ & +\mathbb{E}_{x\sim p_{data(x)}}[\|x - F(x)\|_1],\end{aligned} \tag{5}$$

CSN is used to encode the input image to obtain content vectors. To ensure the content vectors to contain the information missing in STN, a decoder is added so that CSN could

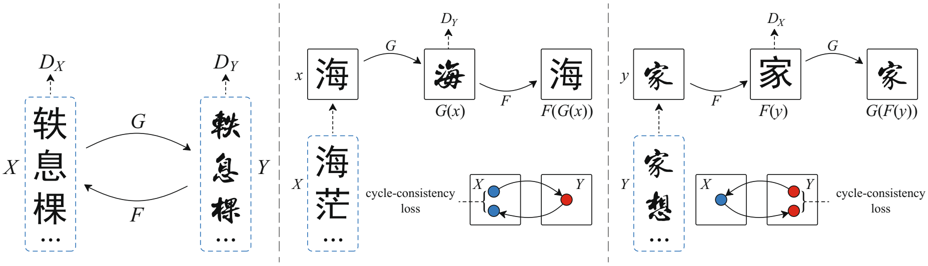


Fig. 4 Cycle-consistent loss function is applied to maintain cycle consistency, such that a full cycle of forward and backward transformations would return to the original state

be trained in the form of an auto encoder to restore the content vectors to the input image. The closer the reconstructed image is to the input image, the more complete the information content vectors contain [32]. Since there are two generators G and F, other two CSNs are needed to provide them with the low-dimensional information.

For generator G, the content loss of CSN is expressed as:

$$\mathcal{L}_{CSN_G} = \mathbb{E}_{x \sim p_{data(x)}}[\|x - AE(x)\|_1], \tag{6}$$

where $AE()$ stands for an operation that encodes and decodes the input like the auto encoder. $AE(x)$ is the standard font image reconstructed by CSN, L_{CSN_G} represents the L_1 distance between x and $AE(x)$.

For generator F, the content loss of CSN is expressed as:

$$\mathcal{L}_{CSN_F} = \mathbb{E}_{y \sim p_{data(y)}}[\|y - AE(y)\|_1], \tag{7}$$

where $AE(y)$ is the calligraphy image reconstructed by CSN, L_{CSN_F} represents the L_1 distance between y and $AE(y)$. The two models are pre-trained before being concatenated with STN.

The full objective is:

$$\begin{aligned} \mathcal{L}(G, F, D_X, D_Y) = & \ \mathcal{L}_{D_Y}(G, D_Y, X, Y) \\ & + \mathcal{L}_{D_X}(F, D_X, Y, X) + \lambda_1 \mathcal{L}_{cyc}(G, F) + \lambda_2 \mathcal{L}_{tid}. \end{aligned} \tag{8}$$

Here $\lambda_i (i = 1, 2)$ controls the weight of each item.

3.3 Implementation

All input images are scaled to 128 × 128 pixels before entering our model. The gray block in the encoder of STN has a dropout rate of 0.5 [29], and its ReLUs are leaky [23], with slope of 0.2. Except the first block whose kernel size is 7 × 7, all the remaining blocks have the kernel of 3 × 3. The first block has 64 filters, and each subsequent block doubles the number of filters over the previous block. The structure of decoder corresponds to the encoder in reverse order. The number of filters is halved per layer, to 64 for the last layer. The kernel size of layer i is the same as that of layer $n - i$, where n is the number of layers. The activation function of gray blocks is normal ReLU, while in the last layer the activation function is Tanh [1], which maps the resulting value between -1 to 1. The type of batch normalization is instance normalization [4]. CSN has the architecture similar to STN, but has no Residual Block in its encoder and decoder. Since the Residual Block in the intermediate layers retains the feature's size, its content vector has the same size as the intermediate vector. The discriminator is a 70 × 70 PatchGAN [10].

The model is implemented in Pytorch. In our experiment, the initial learning rate is 0.0002, batch size is 1 and the optimizer is Adam [14]. λ_1 and λ_2 in (8) are set at 10. All the experiments are performed on NVIDIA GeForce GTX 1080Ti.

4 Experiment

Each type of calligraphy has its own features that are difficult to quantify. Figure 5a, b, c and d illustrate three cases. Firstly, several subsequent tasks are generated by a dataset, and then a comparison between our model and the baseline model is performed. The efficiency of CSN for detail enhancement and the influence of the standard fonts are verified latter. More effects and failure cases are shown in the supplementary material.

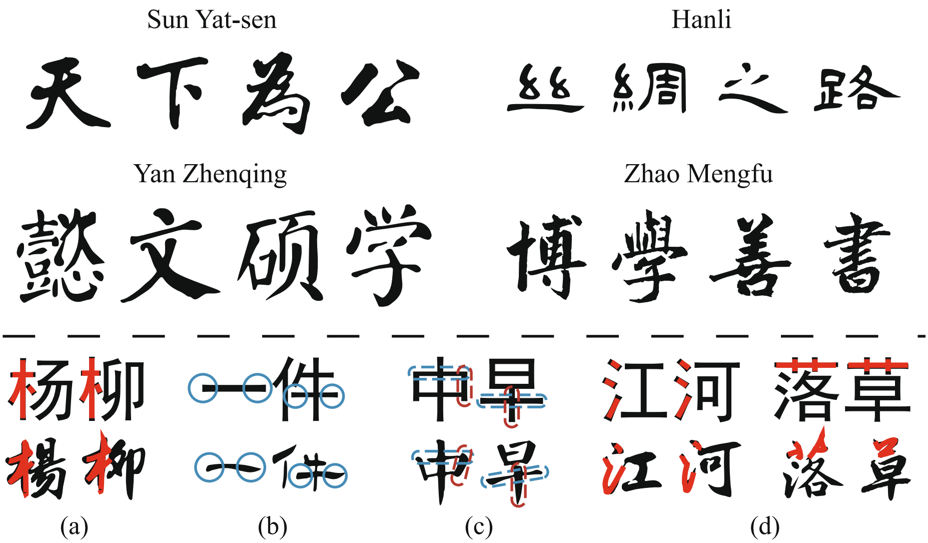


Fig. 5 Top: Several calligraphy images from the dataset, including 4 different styles. Bottom: Analysis of calligraphy writing styles: **a** the top and bottom characters have the structure, written in different styles, which is marked in red; **b** shows the unique brush stroke of Hanli, shown in circles; **c** shows the feature of Yan Zhenqing's calligraphy. The highlighted horizontal stroke is relatively thin, while the vertical stroke is relatively thick; The left side of **d** shows that some strokes in Sun Yat-sen's calligraphy are coherent, while the right side shows the same structure does not have fixed handwriting in Zhao Mengfu's calligraphy

4.1 Data preparation

The establishment of the dataset includes two aspects: structural features and usage scenarios. Chinese calligraphy has five major styles: clerical script (official script), semi-cursive script (running script), seal script (small seal), regular script (standard script) and cursive script (sloppy script). The structure of seal script is biased towards hieroglyphics, and only used in cut stone inscriptions. The cursive script is completely incomprehensible for amateurs. Also, this style has only the ornamental value rather than practical value. Therefore, the remaining three styles of calligraphy are chosen to build the dataset. For the clerical script, modern calligraphers tend to use Hanli, which is mature and resembles modern Chinese characters. So Hanli is chosen as a representative of clerical script. Regular script's structure is basically the same as modern Chinese characters, so it is widely used in daily life. The works of two famous calligraphers, Yan Zhenqing and Zhao Mengfu, are chosen as representatives of the regular script. Sun Yat-sen's calligraphy is used as the representative of semi-cursive script due to its clarity. Each style contains about 2200 images. The calligraphy works are divided into two sets: training set and validation set. 2000 images are randomly selected as the training set and the remaining images as the validation set, and these two sets have no overlap in characters. Several samples of 4 subsets are shown in Fig. 5(top).

4.2 Evaluation and discussions

Two representative methods are selected as the baseline of comparison against our method. The first is Zi2Zi, which is well-known in calligraphy generation. Another is CycleGAN, whose discriminator and part of loss functions are used in our model. Both the baseline method and our method are trained by the dataset. Samples generated by the baseline methods and ours are shown in Fig. 6. Having strokes of the same thickness without serif or any artistic styling, Hei is chosen as the standard font.

1) *Qualitative Results:* As shown in Fig. 6, CycleGAN method correctly captures the style of target characters. For example, the feature 'silkworm head and wild goose tail' in Hanli is captured well. But the result also reflects the downside of CycleGAN. It only learns the distribution of the target domain. In other words, CycleGAN cannot retain the structural information, many strokes are lost during the generation process. In addition, incorrect strokes are added, reducing the readability significantly.

   Compared to CycleGAN, Zi2Zi can retain more strokes, since it learns the global features of Sun Yat-sen's calligraphy, such as the order of strokes. Many features of Yan Zhenqing's calligraphy are also captured, but the edges of strokes are blurry, appeared to connect with each other. Many strokes are also unnaturally distorted, making the characters hard to recognize. The results of Hanli show that Zi2Zi adds extra strokes.

   In contrast, our model yields much better results. In Sun Yat-sen's calligraphy, our model captures the correct writing order of radicals. Connection strokes are also correctly rendered. In Yan Zhengqing's calligraphy, horizontal strokes are thinner than vertical ones, reflected by our model. On Hanli subset, our model could generate images of similar styles in both global structure and local stroke details. Even with more complex targets, our model performs satisfactorily.

   In addition to visual inspection, peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM) and fréchet inception distance (FID) are calculated to quantitatively evaluate the performance. The results are summarized in Table 1. As

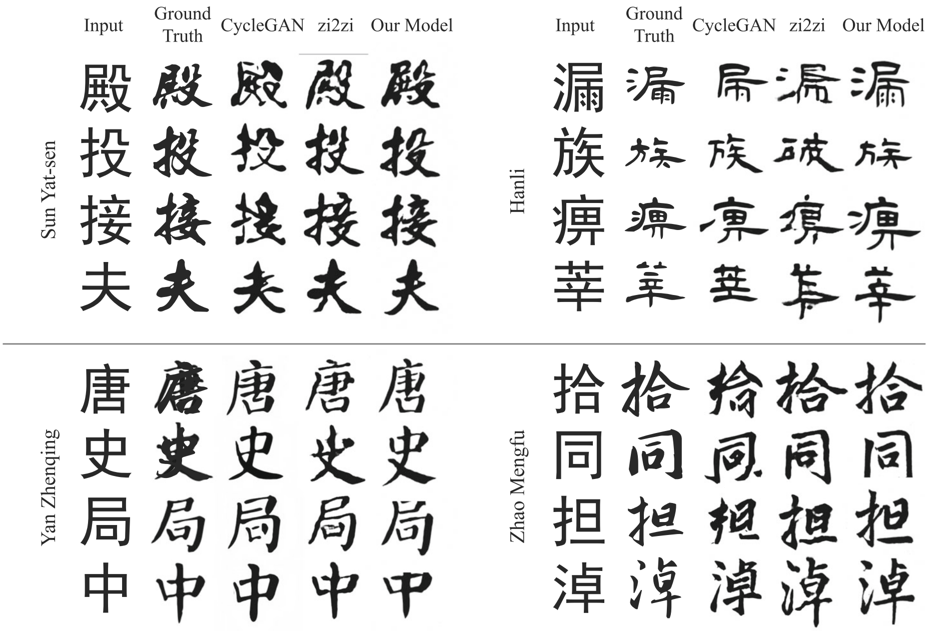


Fig. 6 The results of the baseline methods and our method on the same dataset

Table 1 Quantitative measurements for Fig. 6

	$PSNR$	$SSIM$	FID
$Hei \rightarrow HanLi$	23.4258	0.8341	15.1610
$Hei \rightarrow SunYat - sen$	25.3734	0.7908	23.9482
$Hei \rightarrow YanZhenqing$	23.5720	0.7746	23.8620
$Hei \rightarrow ZhaoMengfu$	24.9793	0.7884	20.8759

detailed in the table, generated results is similar to the ground truth in terms of image structure, visual similarity and data distribution.

2) *Effect of CSN:* The results of Sun Yat-sen style are selected to verify the effectiveness of CSN, as shown in Fig. 7. The calligraphy images generated by our model retain detailed features without blur. It can be concluded that the quality of generated details could be improved by adding the content supply network.

3) *Effect of Standard Fonts:* To evaluate the robustness of our model, the influence of the input fonts is also considered. Kai is chosen for comparison. Our model can generate high-quality calligraphy as shown in Fig. 8. For example, the calligraphy images generated for Zhao Mengfu's style are visually better than those using Hei as input. This is because Kai has several features similar to Zhao Mengfu's calligraphy, thus it is easy for the model to map the input to the target. It can be concluded that if the input fonts are selected by the features of the target calligraphy, one could obtain better quality results. This also proves the robustness of our model.

5 User study

Several experts with professional knowledge of calligraphy give their opinions about this work. They advised that the quality of calligraphy should not only be determined by whether the characters have the features of its original work's style. To objectively evaluate the quality of calligraphy generation, a user study was conducted. The study involves 45 participants

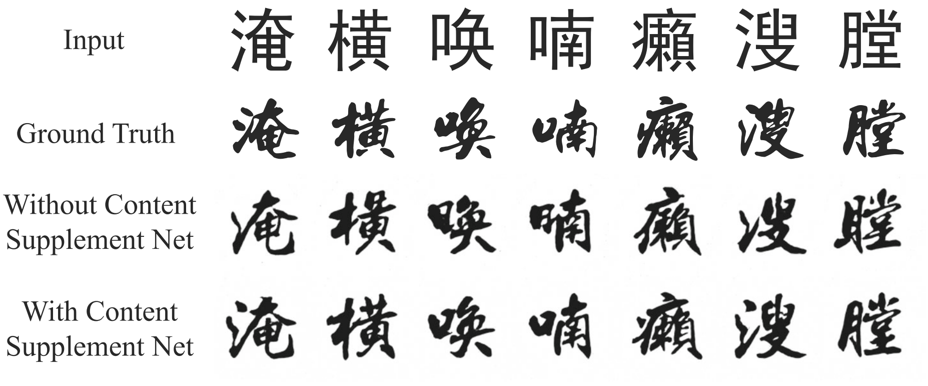


Fig. 7 The results of our model with/without content supplement network evaluated on Sun Yat-sen subset

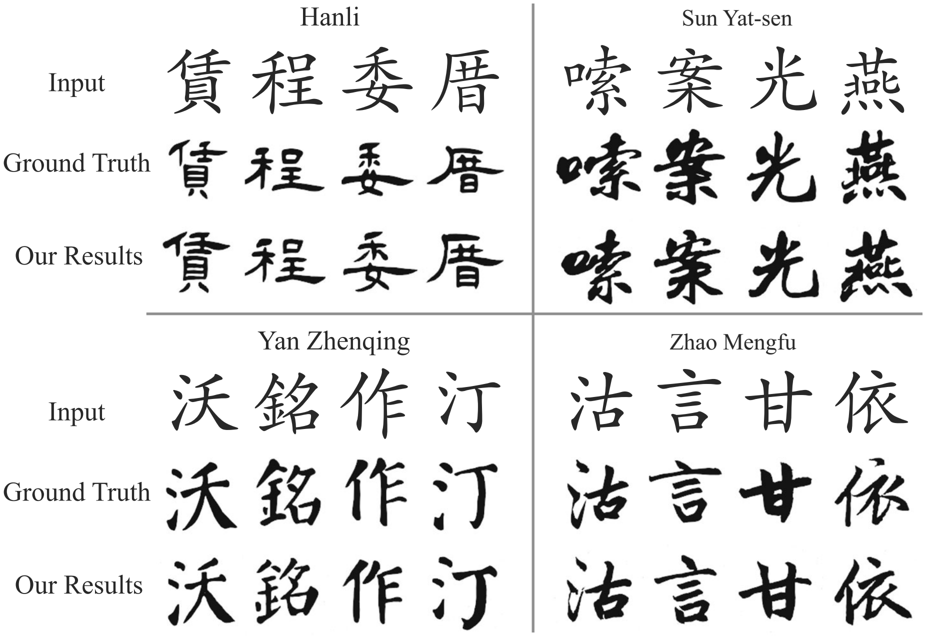


Fig. 8 The results of the baseline methods and our method on the dataset

(21 with professional knowledge of calligraphy, 24 without), who are students in calligraphy classes, researchers and people from various industries. The lowest education level of the participants is a bachelor degree.

5.1 Setup

Considering that even people with professional knowledge of calligraphy rarely learn all four styles, it is too difficult to evaluate the calligraphy's quality just by observing a single character. Furthermore, the evaluation of calligraphy should be at multiple levels, from the entire work to a individual character or radical. Therefore, the generated calligraphy are presented in the form of Chinese four-character idioms. These idioms are composed of original and generated characters. As shown in Fig. 9, the first character (red) in the first row is converted from the standard Hei character, the other three characters are selected from the original work. Participants could analyze each character or simply compare all the characters, then judge which appears unnatural. Having displayed these images, the participants are asked to answer the following questions:

(Q1) Find the character from the idiom that is not from the original work (the character that you consider a fake);
(Q2) Identify which of the two characters is original;
(Q3) Decide whether the displayed character styles are consistent.

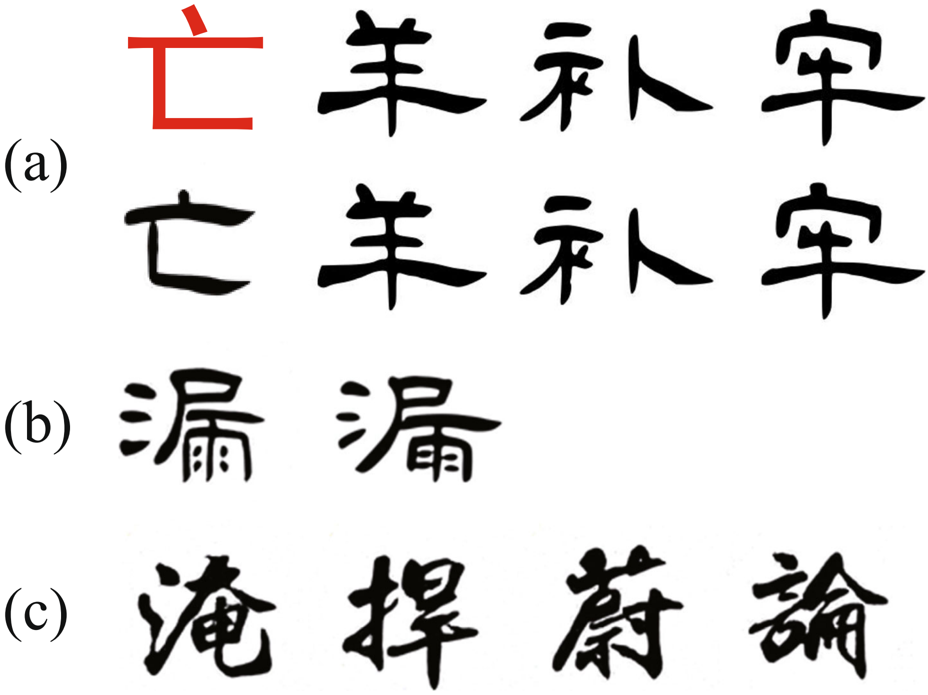


Fig. 9 The synthetic dataset used in user study. **a** In the form of four-character idioms, one of the characters is generated by our model from the standard Hei font (red). A participant's task is to find the character generated by our model. **b** One of the two characters is generated by our model, and the other is from the original work, the task is to select the original. **c** The task is to decide whether the displayed character styles are consistent

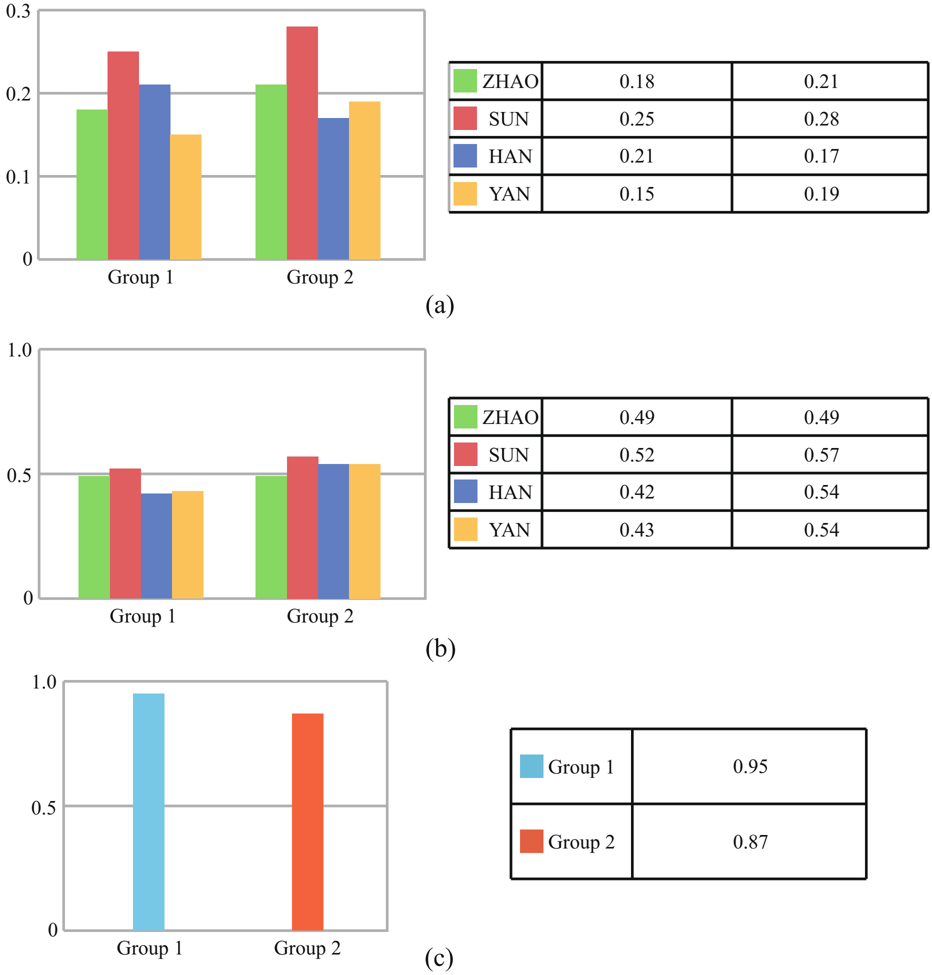
The first and second tasks evaluate whether the characters generated have the same style as that of the original work. The third task evaluates whether the generated characters have a consistent style. Question Q1 provides four choices, one for each character, for the participants to select one. Each of questions Q2 and Q3 provides two choices. Questions for all the four different styles are the same. To avoid interferences of different styles, each style of images are on a separate page, starting with the original work. The participants are asked to observe the original work for two minutes before answering the questions. They could compare the images in their questions with the calligrapher's original work at any time during each task.

If the generated characters are indistinguishable, the probability of choosing the correct answer for Q1 should be close to 25%, that for Q2 should be close to 50%, and that of choosing the style to be consistent for Q3 should be almost 100%. Participants are divided into two groups Group1 and Group2, according to their experiences in learning calligraphy, such that those in Group2 are experienced and those in G1 are not. The questions for all participants are the same. Before the study, it can be speculated that if the generated calligraphy was good enough, those with professional knowledge of calligraphy would answer questions more accurately than those without.

There is a 10-minute explanation before the experiment to let the participants know how the calligraphy images generated. There is no time limit for the participants, so they could take as long as they liked to view the images.

5.2 Results and analysis

The results on the three questions are shown in Fig. 10, that provide us interesting yet meaningful new findings. For each question, the results according to calligraphy learning experiences are compared. In Fig. 10a, the rate of identifying the generated characters is below 25%, except Sun Yat-sen's calligraphy, implying that our results are almost indistinguishable from the original works. In comparison to the other two styles, Sun Yat-sen's style involves much thicker and larger strokes, making the flaw due to automatic generation much easier to expose. The result of Hanli is different from our expectation, as the accuracy of identifying the generated characters by the participants without calligraphy knowledge is higher than that by professional calligraphers. Our explanation is that the structure of Hanli



🟩 ZHAO	0.18	0.21
🟥 SUN	0.25	0.28
🟦 HAN	0.21	0.17
🟨 YAN	0.15	0.19

(a)

🟩 ZHAO	0.49	0.49
🟥 SUN	0.52	0.57
🟦 HAN	0.42	0.54
🟨 YAN	0.43	0.54

(b)

🟦 Group 1	0.95
🟧 Group 2	0.87

(c)

Fig. 10 User study results including **a** The accuracy of identifying the generated character from the given four characters; **b** The success rate of comparing the character generated by our model and the character from the original work and select the original one; **c** Whether the generated calligraphy characters are consistent in style

is different from modern Chinese characters, professional calligraphers would consider the structural differences to be normal, without associating them to automatic generation.

Figure 10b shows that the generated calligraphy images are indistinguishable from the corresponding original works. Even participants with professional knowledge could only achieve an accuracy of about 50%. Figure 10c shows the generated calligraphy characters are consistent.

5.3 Interview with domain experts

Using the dataset described above, we consulted two experts in calligraphy, who are both familiar with the four styles used in our dataset, yet have different ways of evaluating calligraphy. One expert has a doctorate in fine arts, and is currently a university professor and a member of the Chinese Calligraphy Association. His calligraphy works have won the highest awards many times. His evaluation of calligraphy emphasizes its artistic expressions. The other expert is also a member of the Chinese Calligraphy Association, who often demonstrates his calligraphy to the public. His works have won numerous awards and participated in many exhibitions. He prefers to evaluate brushstrokes and forms of calligraphy.

Experts introduced the features of these four fonts as a criterion for judging the effect. To be specific, the horizontal strokes of Yan Zhenqing's calligraphy are thinner than the vertical strokes, and the font size is relatively uniform. The shape of Hanli is usually wide and flat, with long horizontal strokes and short vertical strokes. Long horizontal strokes are thin at the beginning and thick at the end. The calligraphy of Zhao Mengfu and Sun Yat-sen have something in common, i.e, the handwriting is coherent, connecting many strokes together and omitting some strokes for coherent needs. Sun Yat-sen's calligraphy strokes are usually thick, while Zhao Mengfu's calligraphy adjusts the thickness of the strokes to make the character more beautiful, and even if the same structure does not have fixed handwriting. Features mentioned above are illustrated in Fig. 5.

Both experts are interested in the techniques of automatic calligraphy generation. When referring to the significance of this work, one expert said that this idea was excellent, beneficial to the promotion of calligraphy and traditional writing skills. He is particularly interested in automatic generation of a digital calligraphy library, which was the original objective of our work. When referring to the quality of the generated calligraphy, he praised our techniques being magical. Apart from being able to transforming strokes to the corresponding styles, our model could also handle missing strokes well. They also made valuable comments that the typesetting of calligraphy and selection of the background were important to calligraphy evaluation. The darkness of ink and thickness of strokes should be adapted according to the usage scenario.

6 Conclusion and future work

This paper has presented a model for learning and generating Chinese calligraphy styles. The generator consists of a style transfer network and a content supplement network. The style transfer network can transfer the input standard font into the learned calligraphy style, with the content information extracted by the content supplement network. Compared with other recent calligraphy generation approaches, our model does not need paired data, so font designers do not need to annotate the original work. Results of our model with those of other baseline models are compared and presented. Our user study involving both professional calligraphers and non-professionals also shows encouraging results.

As a future direction, the darkness of ink will be taken into account when generating calligraphy to make the strokes more vivid. How to generate calligraphy in overall works rather than individual characters, and fine-tune our models for better calligraphy typesetting will be investigated. Finally, according to the calligraphy experts' suggestions, the style of calligraphy should be adjusted in conjunction with the scene.

References

1. Agostinelli F, Hoffman M, Sadowski P, Baldi P (2016) Learning activation functions to improve deep neural networks. In: 3rd international conference on learning representations
2. Alfred Z, Yuke Z (2014) Strokebank: automating personalized chinese handwriting generation. In: Proceedings of the AAAI conference on artificial intelligence, pp 3024–3029
3. Badrinarayanan V, Handa A, Cipolla R (2015) Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. arXiv:abs/1505.07293
4. Dmitry U, Andrea V, Victor L (2016) Instance normalization: The missing ingredient for fast stylization. arXiv:abs/1607.08022
5. Dmitry U, Vadim L, Andrea V, Victor SL (2016) Texture networks: Feed-forward synthesis of textures and stylized images. In: ICML vol 1, pp 4
6. Gatys L, Ecker AS, Bethge M (2015) Texture synthesis using convolutional neural networks. In: Advances in neural information processing systems, pp 262–270
7. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680
8. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
9. Hwang Y, Lee J-Y, So Kweon I, Joo Kim S (2014) Color transfer using probabilistic moving least squares. In: Proceedings of the IEEE Conference on computer vision and pattern recognition, pp 3342–3349
10. Isola P, Zhu J-Y, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1125–1134
11. Jiang Y, Lian Z, Tang Y, Xiao J (2019) scfont: Structure-guided chinese font generation via deep stacked networks. In: Proceedings of the AAAI conference on artificial intelligence, vol 33, pp 4015–4022
12. Johnson J, Alahi A, Fei-Fei L (2016) Perceptual losses for real-time style transfer and super-resolution. In: European conference on computer vision, pp 694–711
13. Jun-Yan Z, Taesung P, Phillip I, Alexei AE (2017) Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision, pp 2223–2232
14. Kingma DP, Ba J (2014) adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations
15. Li W, Chen Y, Tang C, Yu S (2018) Example-based chinese calligraphy synthesis. In 2018 international conference on advanced control, automation and artificial intelligence
16. Long J, Shelhamer E, Darrell T (2015) Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3431–3440
17. Luc P, Couprie C, Chintala S, Verbeek J (2016) Semantic segmentation using adversarial networks. In: NIPS workshop on adversarial training
18. Lyu P, Bai X, Yao C, Zhu Z, Huang T, Liu W (2017) Auto-encoder guided gan for chinese calligraphy synthesis. In: 2017 14th IAPR international conference on document analysis and recognition, vol 1, pp 1095–1100
19. Matthew DZ, Rob F (2014) Visualizing and understanding convolutional networks. In: European conference on computer vision, pp 818–833
20. Mehdi M, Simon O (2014) Conditional generative adversarial nets. arXiv:abs/1411.1784
21. Mirza M, Osindero S (2018) Coconditional autoencoding adversarial networks for chinese font feature learning. arXiv:abs/1812.04451
22. Pathak D, Krahenbuhl P, Donahue J, Darrell T, Efros AA (2016) context encoders: Feature learning by inpainting. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2536–2544

23. Radford A, Metz L, Chintala S (2015) unsupervised representation learning with deep convolutional generative adversarial networks. In: 4th international conference on learning representations
24. Rewrite. https://github.com/kaonashi-tyc/Rewrite
25. Richard Z, Phillip I, Alexei AE (2016) Colorful image colorization. In European conference on computer vision, pp 649–666
26. Ronneberger O, Fischer P, Brox T (2015) U-net: Convolutional networks for biomedical image segmentation. In: International conference on medical image computing and computer-assisted intervention, pp 234–241
27. Shuai Y, Jiaying L, Zhouhui L, Zongming G (2017) Awesome typography: Statistics-based text effects transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 7464–7473
28. Souly N, Spampinato C, Shah M (2017) Semi supervised semantic segmentation using generative adversarial network. In: Proceedings of the IEEE international conference on computer vision, pp 5688–5696
29. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from over-fitting. J Mach Learn Res 15(1):1929–1958
30. Taigman Y, Polyak A, Wolf L (2016) Unsupervised cross-domain image generation. In: 5th international conference on learning representations
31. Wu SJ, Yang CY, Jane YH (2020) calligan: Style and structure-aware chinese calligraphy character generator. arXiv:abs/2005.12500
32. Zhang J, Shan S, Kan M, Chen X (2014) Coarse-to-fine auto-encoder networks (cfan) for real-time face alignment. In: European conference on computer vision, Springer, pp 1–16
33. zi2zi. https://github.com/kaonashi-tyc/zi2zi

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.