ZiGAN: Fine-grained Chinese Calligraphy Font Generation via a Few-shot Style Transfer Approach

Qi Wen*
wenqijay@gmail.com
NetEase Fuxi AI Lab
Hangzhou, China

Shuang Li*
shuangli@bit.edu.cn
Beijing Institute of Technology
Beijing, China

Bingfeng Han
bfhan@bit.edu.cn
Beijing Institute of Technology
Beijing, China

Yi Yuan†
yuanyi@corp.netease.com
NetEase Fuxi AI Lab
Hangzhou, China

ABSTRACT

Chinese character style transfer is a very challenging problem because of the complexity of the glyph shapes or underlying structures and large numbers of existed characters, when comparing with English letters. Moreover, the handwriting of calligraphy masters has a more irregular stroke and is difficult to obtain in real-world scenarios. Recently, several GAN-based methods have been proposed for font synthesis, but some of them require numerous reference data and the other part of them have cumbersome preprocessing steps to divide the character into different parts to be learned and transferred separately. In this paper, we propose a simple but powerful end-to-end Chinese calligraphy font generation framework ZiGAN, which does not require any manual operation or redundant preprocessing to generate fine-grained target style characters with few-shot references. To be specific, a few paired samples from different character styles are leveraged to attain fine-grained correlation between structures underlying different glyphs. To capture valuable style knowledge in target and strengthen the coarse-grained understanding of character content, we utilize multiple unpaired samples to align the feature distributions belonging to different character styles. By doing so, only a few target Chinese calligraphy characters are needed to generated expected style transferred characters. Experiments demonstrate that our method has a state-of-the-art generalization ability in few-shot Chinese character style transfer.

CCS CONCEPTS

• **Computing methodologies** → **Image processing**.

KEYWORDS

Font Generation; Image-to-Image Translation; GANs

---

*Authors contribute equally.
†Dr. Yuan is the corresponding author.

1 INTRODUCTION

Chinese characters are an ancient and precious cultural heritage. In China, Chinese characters are called 'zi'. Since ancient times, countless outstanding calligraphers have left their valuable handwritings, which have become the brilliant achievements of human civilization. However, many valuable calligraphy works have been lost in the long history [35]. Unlike English, which has only 26 letters, there are tens of thousands of characters in Chinese characters, each of which has a different glyph and represents a different meaning. Furthermore, different calligraphers have their own writing styles with special overall structure and stroke details. Therefore, it is very meaningful and challenging to generate a complete personalized font library with only a few references.

Some specific fonts have a relatively complete font library, for example, the widely used Chinese font Sim Sun version 5.16 covers 28,762 Unicode characters. But for most calligraphy works, it is almost impossible to get enough authentic works. The automatic generation of glyph images can greatly reduce the labor cost of font designers. Meanwhile, it is very helpful for calligraphy beginners to emulate the masterpieces reproduced.

Early studies on Chinese character synthesis tend to decompose characters into different radicals and regions, and then reassemble them [43, 44]. But this kind of methods requires a lot of manual intervention and is inefficient. Additionally, they still produce undesirable results.

With the development of deep learning and computer vision, style transfer is discovered [9, 12, 17, 36], which is dedicated to transforming one style of artwork into another. It achieves success in texture features transfer tasks, but unable to adapt to the translation in large geometric variations. Subsequently, methods such as pix2pix [15] and CycleGAN [51] are proposed to solve image-to-image translation problem. But unlike photo-to-artwork task, Chinese characters are made up of pure black and white. More importantly, any lack of subtle structure or changes is unacceptable,

中国毛笔字手书

中国毛笔字手书

Figure 1: The upper row is the input standard characters in font style Song, and the following line is the generated characters with target style. The generated calligraphy font shown in the figure has been successfully implemented in the application scenario.

while the GAN-based methods [4, 5, 20, 45] often lead to minor inaccuracy or blur.

Recently, some studies have been conducted to generate fonts [16, 24, 37]. Zi2zi [34] is proposed based on the pix2pix framework, which results in good synthesizing performance in some specific font styles. On this basis, CalliGAN [41] further uses the prior information of Chinese character radicals to achieve better results. But this leads to a more complex and fragmented network structure. Furthermore, ChiroGAN [8] is committed to getting reasonable results without using paired data. But it cannot handle brush-written calligraphy images with complex skeletons. Moreover, all the above-mentioned methods require a large number of style reference glyphs to achieve acceptable results, which may be laborious or even impossible to obtain in real-world scenarios. RD-GAN [13] is committed to using only a few style references, but it still requires a lot of prior knowledge of radicals, which will be very troublesome to process. And it generates handwritten photo-style images, which is different from our calligraphy written in ink on a white background.

In this paper, we propose ZiGAN, a novel end-to-end framework for fine-grained Chinese calligraphy font generation with few-shot target references. Given a few calligrapher's characters of the expected style, we can easily obtain the corresponding standard font images of the same characters and get the well-aligned pairs. We leverage these small amounts of paired samples to attain fine-grained correlation between structures underlying different styles.

Meanwhile, brush-written calligraphic character images are much more irregular than font-rendered character images. Few existing papers use this type of images to conduct experiments. In order to deal with this situation, we pioneer the utilization of numerous other unpaired characters in the standard font library which can be easily rendered. Although the glyphs of these characters are different from the target, they contain rich structure and morphological information. To capture valuable style knowledge in target and strengthen the coarse-grained understanding of character content, we utilize multiple unpaired samples to align the feature distributions belonging to different character styles. Figure 1 shows a successful application case of our method.

To sum up, our major contributions are summarized as follows:

(1) We propose a simple but effective end-to-end framework that can generate fine-grained stylized calligraphy characters with only a few references. And it can easily adapt to a new handwriting style transfer task without tedious manual operations or prior knowledge.

(2) We innovatively learn the coarse-grained content knowledge of unpaired characters in the standard font library. To capture valuable structural knowledge, we map the features of the characters in different styles to Hilbert space and align the feature distributions. By doing so, we not only retain the semantic information of the character but also successfully translate the style from source to target while only a few target Chinese calligraphy characters are needed.

(3) Comprehensive experiments and analysis show that our approach can generate Chinese characters with state-of-the-art quality. More importantly, our method has been successfully implemented in actual application scenarios.

2 RELATED WORK

2.1 Generative Adversarial Networks

Generative Adversarial Networks (GAN) [10] has attracted a lot of interest since it was proposed. It has been successfully applied in many different fields and achieved impressive results, such as image generation [18, 19, 26], image completion [14, 46, 47], image editing [50], transfer learning [29, 31], image translation [4, 15, 20, 51], etc. The key to the success of GAN is that the discriminator tries to distinguish the generated images from the realistic images, while the generator tries to confuse the judgment of the discriminator. In this paper, our model is based on GAN and only uses a few reference data to learn the Chinese calligraphy character style translation.

2.2 Image-to-Image Translation

Image-to-image translation aims to learn a mapping function that can transform an image from the source domain to the target domain. It has been widely used in many applications, for example, for artistic style transfer [3, 17], semantic segmentation [23, 27, 28], photo enhancement or object replacement.

A great quantity of GAN-based methods have been proposed, quite a few of them condition on images [4, 5, 15, 38]. Pix2pix [15] is the pioneering method to figure out image-to-image translation. It follows the idea of conditional GAN, applying adversarial loss and L1-loss, and achieves impressive results. After that, high-resolution version is proposed to reinforce pix2pix in image synthesis and semantic manipulation [38]. But those paired training data are hard to obtain for some applications such as artistic style transfer. To alleviate this pain point, unpaired image-to-image translation frameworks have been proposed where no paired data are available anymore [21, 22, 51]. It is a remarkable fact that CycleGAN [51] proposes the cycle-consistent adversarial network, where two GANs interact in a cycle and learn source and target image distributions simultaneously. Based on CycleGAN, U-GAT-IT [20] proposes a novel method for unsupervised image-to-image translation, which incorporates a new attention module and a new learnable normalization function called AdaLIN in an end-to-end manner. It is effective in the task of animating faces. In summary, the aforementioned paired methods all require a lot of data for training, otherwise the results will be unsatisfactory. Meanwhile, the unpaired methods often cause missing or redundant construction. But in the task of Chinese calligraphy character style translation, calligraphers' handwriting is often difficult to obtain, and we cannot tolerate the inconsistency of character structure.

2.3 Chinese Font Generation

Chinese font generation has been studied for a long time [40, 42]. The image-based methods [6, 44] split and reorganize the corresponding strokes and radicals in the dataset to generate the characters we want. But these methods contain too much human intervention, which is very inconvenient. With the development of deep learning, people have paid more attention to GAN-based character translation. Since character translation requires higher accuracy according to its complex strokes and style, it is more difficult than classic image-to-image translation problems. Transferring the styles of the alphabet is quite helpful and efficient for English translation [1]. While it is not simple like this for Chinese character style transfer because each Chinese character has its own glyph shape and there is a large number of existed characters. The style of the strokes in a certain character may quite different from the same strokes in other characters [39], which makes the problem harder.

The first way to generate Chinese characters is following image-to-image methods, like zi2zi [34], an open-source project that was never published as a paper. It's based on pix2pix, trying to translate character images from source style to various target styles. Based on zi2zi, DCFont [16] and PEGAN [33] have made improvements and achieved better results. The second way to synthesize Chinese characters often separates a character into two parts, which are content and style [8, 39]. EMD [48] and SA-VAE [32] use two different encoders to process content and style respectively. After absorbing the advantages of the above methods, CalliGAN [41] adds an extra component code of the character to train a conditional GAN, exploiting prior knowledge to maintain the structure information. While it needs a dictionary for each Chinese character to save its component code, this is a complicated preprocessing work. Unlike the aforementioned methods, ChiroGAN [8] uses erosion and dilation operations to obtain the basic skeleton of characters, then transfers style from source to target at the skeleton level. The output of this module is the skeleton image so it has to use another network to render the skeleton into the target character. Moreover, it relies on the effects of corrosion and expansion algorithms so that it often crashes on complex characters with numerous strokes or irregular glyph styles.

In order to save the cost of multiple Chinese characters selection, several recent methods aim to generate new glyphs with few numbers style references. DMfont [2] disassembles Korean or Thai glyphs to stylize components and then reassembles them. But it cannot handle complex Chinese characters. RD-GAN [13] aims to generate unseen characters in the fixed style, but it still requires a lot of prior knowledge of radicals, which will be very troublesome to process. Other earlier few-shot methods also have fatal shortcomings, such as being unable to generate complex glyphs [1] or failing to capture local styles [7, 32].

To sum up, part of the methods require lots of data, but the handwritings of many ancient Chinese calligraphers are not handed down so we cannot obtain them. The other parts of the methods are doped with too much manual processing. Moreover, they utilize too much intricate prior knowledge, which makes the preprocessing work complicated and can only adapt to a single task. To overcome these challenges, in this paper we propose a novel ZiGAN that can learn an intact and delicate character style and structure when

Table 1: The architecture of encoder and decoder.

Layer	Encoder	Decoder
Input	$256 \times 256 \times 3$	$1 \times 1 \times 512$
L1	$128 \times 128 \times 64$	$2 \times 2 \times 1024$
L2	$64 \times 64 \times 128$	$4 \times 4 \times 1024$
L3	$32 \times 32 \times 256$	$8 \times 8 \times 1024$
L4	$16 \times 16 \times 512$	$16 \times 16 \times 1024$
L5	$8 \times 8 \times 512$	$32 \times 32 \times 512$
L6	$4 \times 4 \times 512$	$64 \times 64 \times 256$
L7	$2 \times 2 \times 512$	$128 \times 128 \times 128$
L8	$1 \times 1 \times 512$	$256 \times 256 \times 3$

only a few target characters are provided. ZiGAN is an end-to-end framework, which can be easily and conveniently applied to any character style translation task, and is capable of generating a complete and consistent font library.

3 METHOD

We distinguish each Chinese character based on structure, radicals and strokes. Therefore, each calligrapher writes the same content of Chinese characters but in different styles. The goal of our proposed method is to learn a way to generate Chinese character images with the expected style from only a small amount of given characters. Let s be the style we want, and y be a target image under the style s. We use TrueType fonts to render a source image x_p representing the same character as y in black with font style Song as the standard character image. Furthermore, we find that although we can only obtain a few target character images, we can render a mass of source character images from the TTF of font style Song. We randomly render images from font style Song, defined as x_r. In general, we leverage the paired image sets $\{x_p\}$ and $\{y\}$ to attain fine-grained correlation between structures underlying different glyphs. Moreover, the method we proposed learns the extra structural knowledge in the unpaired data $\{x_r\}$ simultaneously to strengthen the coarse-grained understanding of the character content. Our framework consists of two generators and two discriminators in two opposite directions. Here we only explain the direction of $x \rightarrow y$ as the vice versa should be straightforward.

3.1 Network architectures

Figure 2 shows the architecture of our network. We encode x_p and x_r into the feature space through an image encoder E_s, and then decode image features by an image decoder G_s to generate the stylized character images $\hat{y}_p$ and $\hat{y}_r$. After that, we model on CycleGAN [51] and set up a reversing generator, including encoder E_t and decoder G_t.

Image encoder and decoder. We use the encoder-decoder architecture as our generator, which is based on pix2pix [15] and zi2zi [34] with some improvements. Unlike zi2zi, we remove the category embedding vector because it is inapplicable for our task and will increase instability. The complete architecture of our generator is in Table 1. All convolution and deconvolution layers use 5-by-5 filters with stride size of 2, and apply batch normalization. The encoder layers actually use LeakyReLU for activation function with a slope of 2. While the decoder layers use the activation function ReLU. We use dropout with a rate of 0.5 only in L1 to L3 layers of the decoder.

Figure 2: Network architectures. ZiGAN is an end-to-end framework based on the encoder and decoder. The network can not only learn style information from a few target images but also learn structure and content information from numerous source images. An auxiliary classifier is added to the discriminator to force the model to focus on more important regions. ZiGAN has 4 losses: GAN loss (Eq. (3)), consistency loss (Eq. (6)), alignment loss (Eq. (9)), style loss (Eq. (10)).

CAM Discriminator. We add an auxiliary classifier η_{D_t} based on Class Activation Map (CAM) [49] to the discriminator so that the model can pay more attention to more important regions. For different calligraphy, there may be subtle but critical differences between the strokes and radicals. The local and global discriminator with CAM attention module can help the model distinguish better and generate finer characters of different styles. Unlike pix2pix [15] and zi2zi [34], we don't use conditional image knowledge to reduce complexity so the discriminator does not observe x. In Section 4, we demonstrate that the CAM attention module can learn the details successfully.

3.2 Loss Function

We define four losses in total. The loss items of $x \rightarrow y$ can be written as:

GAN loss. GAN loss is divided into main and auxiliary parts. In the main part, we impose adversarial loss to match the distribution of the translated images and target images. We use the Least Squares GAN [25] objective to train our model.

$$\mathcal{L}_{adv}^{x \rightarrow y} = \mathbb{E}_y[(D_t(y))^2]+ \\ \mathbb{E}_x[(1 - D_t(G_s(E_s(x))))^2]. \tag{1}$$

In addition, we add an auxiliary classifier η_{D_t} based on Class Activation Map(CAM) [49] to the discriminator D_t. Let $y \in \{Y\}$, $G_s(E_s(X))$

represent a sample from the target domain and the translated source domain. The discriminator D_t consists of an encoder E_{D_t}, a classifier C_{D_t}, and an auxiliary classifier η_{D_t}. The auxiliary classifier is trained to learn the weight of the k-th feature map for the target domain, w_t^k, by using the global average pooling and global max pooling, i.e., $\eta_{D_t}(y) = \sigma\left(\Sigma_k w_t^k \Sigma_{ij} E_{D_t}^{k_{ij}}(y)\right)$. By exploiting w_t^k, we can calculate a set of domain specific attention feature map $a_{D_t}(y) = w_{D_t} * E_{D_t}(y) = \left\{w_{D_t}^k * E_{D_t}^k(y) \mid 1 \le k \le n\right\}$, where n is the number of encoded feature maps. Then, our discriminator $D_t(y)$ becomes equal to $C_{D_t}(a_{D_t}(y))$. By doing so, the discriminator can better distinguish the differences in the details of different character styles, while E_s and G_s can make improvements in the most important regions.

$$\mathcal{L}_{cam}^{x \rightarrow y} = \mathbb{E}_y[(\eta_{D_t}(y))^2]+ \\ \mathbb{E}_x[(1 - \eta_{D_t}(G_s(E_s(x)))^2]. \tag{2}$$

On the whole:

$$\mathcal{L}_{GAN}^{x \rightarrow y} = \mathcal{L}_{adv}^{x \rightarrow y} + \mathcal{L}_{cam}^{x \rightarrow y}. \tag{3}$$

Consistency loss. We constrain the consistency of the model from two parts. First, the model must have the ability to cycle back. It means that after x is translated to $\hat{y}$, it must be successfully

translated back to the original domain:

$$\mathcal{L}_{cycle}^{x \to y} = \mathbb{E}_x[|x - G_t(E_t(G_s(E_s(x))))|_1]. \tag{4}$$

Second, identity loss is used to constrain the color and shape of the characters to not be distorted. Given an image y, after the translation of Es and Gs, it should be the same character in the same style.

$$\mathcal{L}_{identity}^{x \to y} = \mathbb{E}_y[|y - (G_s(E_s(y)))|_1]. \tag{5}$$

So the total consistency loss is:

$$\mathcal{L}_{consistency}^{x \to y} = \mathcal{L}_{cycle}^{x \to y} + \mathcal{L}_{identity}^{x \to y}. \tag{6}$$

Alignment loss We align the content and feature levels of x_p and y to leverage the paired samples to attain fine-grained structural correspondence. In the font style translation task, the job of the discriminator is still to distinguish which is generated or which is real, but the generator is tasked to not only fool the discriminator, but also to be as similar to the ground truth at the content level as possible. We use the L1 loss to constrain the output of paired data x_p,

$$\mathcal{L}_{L1}^{x \to y} = \mathbb{E}_{x,y}[|y - G_s(E_s(x_p))|_1]. \tag{7}$$

And in order to constrain the features of the generated image and the real image to the same space, we apply constancy loss:

$$\mathcal{L}_{constancy}^{x \to y} = \mathbb{E}_{x,y}[|E_t(y) - E_t(G_s(E_s(x_p)))|_2]. \tag{8}$$

Therefore, the total alignment loss can be formulated as:

$$\mathcal{L}_{alignment}^{x \to y} = \alpha \mathcal{L}_{L1}^{x \to y} + \mathcal{L}_{constancy}^{x \to y}. \tag{9}$$

where $\alpha = 5$.

Style loss For better understanding of coarse-grained character content and a maturer style translation, we have introduced style loss to take advantage of multiple unpaired samples x_r. Unlike paired data, unpaired data cannot simply be restricted by L1 or L2 losses. Therefore, with comprehensive consideration of time complexity and computational cost, we utilize MK-MMD [11, 30] to match the feature distributions to retain style information. Denote by $\mathcal{H}_k$ be the reproducing kernel Hilbert space (RKHS) endowed with a characteristic kernel k. The mean embedding of distribution p in $\mathcal{H}_k$ is a unique element $\mu_k(p)$ such that $\mathbf{E}_{\mathbf{x} \sim p} f(\mathbf{x}) = \langle f(\mathbf{x}), \mu_k(p) \rangle_{\mathcal{H}_k}$ for all $f \in \mathcal{H}_k$. The MK-MMD $d_k(x, y)$ between probability distributions x and y is defined as the RKHS distance between the mean embeddings of x and y. The squared formulation of style loss is defined as:

$$\mathcal{L}_{style}^2 = ||\mathbb{E}_y[\phi(E_t(y))] - \mathbb{E}_x[\phi(E_t(G_s(E_s(x_r))))]||_{\mathcal{H}_k}^2. \tag{10}$$

where ϕ is the corresponding feature map. And it's worth noting that when $x = y$, $\mathcal{L}_{style} = 0$. Here we choose Gaussian kernel function as the kernel function:

$$k_\sigma^{rbf}(P_s, P_t) = \exp\left(-\frac{1}{2\sigma^2}||x - y||^2\right). \tag{11}$$

Full objective Finally, the full objective function is defined as:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{GAN} + \lambda_2 \mathcal{L}_{consistency} + \lambda_3 \mathcal{L}_{alignment} \\ + \lambda_4 \mathcal{L}_{style}. \tag{12}$$

where $\lambda_1 = 5$, $\lambda_2 = 10$, $\lambda_3 = 10$, $\lambda_4 = 10$. Here $\mathcal{L}_{GAN} = \mathcal{L}_{GAN}^{x \to y} + \mathcal{L}_{GAN}^{y \to x}$ and the other losses are defined in the similar way.

Figure 3: Example characters in 9 different styles.(1-9 in order)

Table 2: Data sets used in our experiments. Dataset Name is the expert of the masterpiece and the sub-typeface for those created with the same master. Samples is the number of characters in each data set.

Style	Dataset Name	Samples
1	Chu Suiliang	7159
2	Liu Gongquan	6171
3	Ouyang Xun — Huangfu Dan Stele	6999
4	Ouyang Xun — Inscription on Sweet Wine Spring at Jiucheng Palace	6901
5	Yan Zhenqing — Stele of the Abundant Treasure Pagoda	6308
6	Yan Zhenqing — Yan Qinli Stele	7006
7	Yu Shinan	7008
8	XING	6800
9	CAO	6799

4 EXPERIMENT

4.1 Datasets

To better show our model's performance, we use the same datasets with CalliGAN [41]. The datasets could be downloaded from a Chinese calligraphy character website[1], where there are more than 20 kinds of brush-written calligraphy sets belonging to different Chinese ancient experts. And 7 styles belonging to regular script are used to complete our experiment. The 3rd and the 4th style sets are the same ancient calligraphic expert's masterpieces created in different periods of his life. They are treated as two different style sets due to the differences between them, which is also the rule of thumb in the Chinese calligraphy community. The 5th and the 6th style sets are the same situation as above. In addition to the above-mentioned dataset which is the same as CalliGAN, we also test our model in other more irregular and challenging Chinese character fonts like XING and CAO to prove that our method is highly adaptable and robust to any font style. So our data set consists of 9 fonts in total which are shown in Table 2. Figure 3 shows example characters in 9 different styles.

We collect 61151 target images that cover 6560 characters in the 9 styles in all. And we use TTF of font style Song to render source image x. To explore the ability of our few-shot method, we create two configurations for the dataset: 100-shot and 200-shot. Each style has 100 or 200 randomly selected training examples respectively as input y, while the remaining images as the test set. Such a training set size is much smaller than other methods which often require thousands of training images. Specific information is listed in Table

---

[1]http://163.20.160.14/~word/modules/myalbum/

Figure 4: Comparison of the results using each method in 9 different font styles. Characters in the purple box are generated with missing strokes, the yellow box means incorrect extra strokes translation, and the blue box indicates conspicuous blurred results.

Table 3: Statistics of our 100-shot and 200-shot configurations.

Style	1	2	3	4	5	6	7	8	9
100shot-Train	100	100	100	100	100	100	100	100	100
100shot-Test	7059	6071	6899	6801	6208	6906	6908	6700	6699
200shot-Train	200	200	200	200	200	200	200	200	200
200shot-Test	6959	5971	6799	6701	6108	6806	6808	6600	6599
Total	7159	6171	6999	6901	6308	7006	7008	6800	6799

3. Given y, we can easily get the same character image x_p in the source domain. In the meantime, we randomly sample and render 6000 unpaired images with font style Song which cover a large number of characters as input x_r.

The images in this repository have various shapes depending on the character's shapes. We follow the preprocessing steps of CalliGAN [41], but the only difference is that we process the images into three-channel RGB images. So we get $256 \times 256 \times 3$ images as our ground truth y. All images are converted to tensors linearly with a value range between -1 and 1 by our network.

4.2 Experiment Setup

We use standard methods to optimize our network: first optimize on D, then on E and G together. Similarly, we also alternate training on x_r and x_p. All models are trained using Adam with $\beta_1 = 0.5$, $\beta_2 = 0.999$. The learning rate is initialized to 0.0003 and drops by half every 500 epochs. Because the number of training samples is small, we train our model in 1500 epochs.

4.3 Qualitative Evaluation

To prove the advancement of our method in the field of few-shot font style transfer, we have extensively compared various methods. Six classic methods are used as the baselines, including zi2zi [34], pix2pix [15], U-GAT-IT [20], CycleGAN [51], StarGAN [4], CalliGAN [41]. Among them, zi2zi, pix2pix and CalliGAN need paired data for training. We use the same number of paired images as ours

Figure 5: The ablation experiment of ZiGAN. All characters are generated under style 1. The red rectangles mark the imperfect part of the character where some strokes are incomplete or fuzzy. It can be seen that every part of our method is beneficial to the result.

to train their model in corresponding configurations. CycleGAN, U-GAT-IT and StarGAN are unsupervised methods. We use 6200 images in font style Song as their source domain and 100 or 200 calligraphic images as their target domain for different configurations so that the size of their training set is not smaller than ours. Figure 4 shows the comparison of generation results.

CycleGAN not only did not fully learn the style of the characters but also lost some strokes. StarGAN has lost the structural information of the character and is completely unable to do this job. Pix2pix barely maintains the structure of the characters, but there are too many fuzzy and damaged places. U-GAT-IT seems to have learned the style of the font, but there are still many erroneous and missing strokes in the result. Although zi2zi and CalliGAN are professional in font style translation, they produce unsatisfactory results which

Table 4: IOU for difference font style translation mode. Higher is better. The seventh and eighth methods show the results of ablation experiments without $\mathcal{L}_{style}$ or $\mathcal{L}_{alignment}$.

		Intersection Over Union (IOU)									
Type	Method	Style									
		1	2	3	4	5	6	7	8	9	Mean
100 - shot	zi2zi	0.228	0.353	0.282	0.298	0.381	0.37	0.274	0.289	0.213	0.299
	pix2pix	0.259	0.374	0.303	0.314	0.416	0.404	0.281	0.305	0.21	0.318
	U-GAT-IT	0.234	0.315	0.261	0.264	0.346	0.342	0.243	0.292	0.211	0.279
	CycleGAN	0.24	0.241	0.259	0.28	0.379	0.369	0.254	0.26	0.2	0.276
	StarGAN	0.152	0.3	0.194	0.206	0.373	0.314	0.151	0.21	0.195	0.233
	CalliGAN	0.241	0.345	0.277	0.293	0.382	0.391	0.278	0.306	0.218	0.303
	ZiGAN w/o $\mathcal{L}_{style}$	0.236	**0.402**	0.308	0.332	**0.417**	0.403	0.291	0.313	0.203	0.323
	ZiGAN w/o $\mathcal{L}_{align}$	0.229	0.326	0.291	0.309	0.383	0.377	0.272	0.296	0.19	0.297
	ZiGAN(Ours)	**0.273**	0.389	**0.317**	**0.333**	0.408	**0.413**	**0.293**	**0.334**	**0.226**	**0.332**
200 - shot	zi2zi	0.257	0.395	0.308	0.324	0.426	0.407	0.274	0.319	0.233	0.327
	pix2pix	0.27	0.398	**0.321**	0.333	0.432	0.422	0.292	0.315	0.215	0.333
	U-GAT-IT	0.239	0.339	0.255	0.267	0.367	0.35	0.249	0.296	0.216	0.286
	CycleGAN	0.241	0.36	0.098	0.27	0.372	0.365	0.255	0.262	0.202	0.269
	StarGAN	0.2	0.331	0.26	0.221	0.374	0.359	0.21	0.221	0.235	0.268
	CalliGAN	0.267	0.347	0.319	0.324	0.414	0.404	0.289	0.327	**0.236**	0.325
	ZiGAN w/o $\mathcal{L}_{style}$	0.286	0.397	0.312	0.332	**0.438**	0.425	0.295	0.323	0.233	0.338
	ZiGAN w/o $\mathcal{L}_{align}$	0.284	0.379	0.302	0.303	0.394	0.403	0.287	0.333	0.23	0.324
	ZiGAN(Ours)	**0.290**	**0.407**	0.319	**0.357**	0.436	**0.427**	**0.316**	**0.344**	**0.236**	**0.348**

Table 5: The top-1 accuracy of generated characters. We train a Resnet18 model as a Chinese character recognizer on the ground truth of all styles. The recognition accuracy can show whether the characters retain the complete character structure and content. The seventh and eighth methods show the results of ablation experiments without $\mathcal{L}_{style}$ or $\mathcal{L}_{alignment}$.

		Top-1 Accuracy									
Type	Method	Style									
		1	2	3	4	5	6	7	8	9	Mean
100 - shot	zi2zi	0.075	0.075	0.094	0.101	0.105	0.129	0.094	0.186	0.114	0.108
	pix2pix	0	0	0.048	0.054	0	0.123	0.068	0.196	0.1	0.065
	U-GAT-IT	0.156	0.025	0.156	0.139	0.128	0.2	0.166	0.102	0.11	0.131
	CycleGAN	0.204	0.064	0.215	0.245	0.351	0.314	0.351	0.13	**0.16**	0.226
	StarGAN	0.003	0.002	0.001	0.001	0.007	0.007	0.002	0.001	0	0.003
	CalliGAN	0.091	0.083	0.089	0.099	0.108	0.184	0.104	0.104	0.11	0.108
	ZiGAN w/o $\mathcal{L}_{style}$	0.255	0.283	0.257	0.317	0.26	0.451	0.325	0.353	0.13	0.292
	ZiGAN w/o $\mathcal{L}_{align}$	0.199	0.16	0.195	0.237	0.294	0.334	0.288	0.336	0.12	0.24
	ZiGAN(Ours)	**0.567**	**0.740**	**0.647**	**0.625**	**0.592**	**0.733**	**0.694**	**0.404**	0.158	**0.573**
200 - shot	zi2zi	0.175	0.346	0.207	0.227	0.323	0.326	0.196	0.246	0.118	0.24
	pix2pix	0.154	0.17	0.151	0.149	0.213	0.294	0.149	0.339	0.101	0.191
	U-GAT-IT	0.236	0.268	0.131	0.132	0.283	0.246	0.184	0.32	0.12	0.213
	CycleGAN	0.211	0.304	0.19	0.234	0.347	0.321	0.385	0.14	0.16	0.255
	StarGAN	0.005	0.012	0.001	0	0.013	0.017	0.005	0.004	0.002	0.007
	CalliGAN	0.192	0.169	0.26	0.212	0.276	0.353	0.23	0.334	0.117	0.238
	ZiGAN w/o $\mathcal{L}_{style}$	0.586	0.614	0.528	0.577	0.62	0.637	0.593	0.403	0.16	0.462
	ZiGAN w/o $\mathcal{L}_{align}$	0.528	0.583	0.568	0.347	0.566	0.24	0.582	0.396	0.146	0.44
	ZiGAN(Ours)	**0.631**	**0.631**	**0.636**	**0.591**	**0.689**	**0.703**	**0.722**	**0.488**	**0.176**	**0.585**

contain too many meaningless blanks and blurs when few targets are referenced. Only ZiGAN has found the law of calligraphy from limited target characters.

4.4 Quantitative Evaluation

For quantitative evaluation, we evaluate it from two aspects: style and content. For the former, we use Intersection Over Union(IOU) to measure whether our results have completed the style translation. IOU calculates the ratio of the intersection and union between the generated character images and the real character images. The

Table 6: User study. We ask respondents to choose which generated character resembles the ground truth from these 7 methods.

Method	Vote Rate(%)	Method	Vote Rate(%)
zi2zi	1.52	CycleGAN	1.68
pix2pix	1.44	StarGAN	0.31
U-GAT-IT	2.27	CalliGAN	1.80
-	-	**ZiGAN(Ours)**	**90.98**

higher the value indicates that the distribution of the generated images is closer to the distribution of the real images, and the result

Table 7: Turing test samples. Each sample contains 6 fake glyph images generated by ZiGAN and 6 real glyph images. ZiGAN achieves an accuracy of 51.6%, which is very close to random selection.

Sample 1	Sample 2	Sample 3
馬 滇 柏 旦 詞 襠 翡 睚 妗 浣 翁 暄	駱 桂 訃 柩 孩 煨 俾 喁 博 徉 堨 凧	呾 � 洞 狅 焖 檻 孢 森 珂 半 殿 哲

is better. As above, we compare six classic methods, and Table 4 shows that ZiGAN achieves the highest IOU scores.

Similarly, for content evaluation, we train a Resnet18 model as a Chinese character recognizer on the ground truth of all styles. And test it on the images generated by our test set. As we can see from Table 5, the top-1 accuracy achieved by our method is significantly ahead of other methods, which proves that our method can effectively retain the structure and content information of the character.

4.5 User Study

We implement user study to verify that our results are not only better in the calculated indicators. 40 people who are familiar with Chinese characters participate in the experiment. We randomly select 65 characters in the test set, then use the compared methods and the proposed method to generate images. Therefore, the participants see a total of 520 images, including 390 images generated by the compared methods, 65 images generated by our ZiGAN, and 65 of ground truth. At each selection, participants will see 7 images generated by 7 different methods and ground truth. Overall, the participant's goal is to find the image that is most similar to ground truth. In detail, participants are asked to prioritize finding the images that are semantically consistent with the ground truth, which means that the generated characters cannot have wrong radicals or missing strokes. On this basis, consider which image style is closer to the ground truth and has better details. Table 6 shows the respondents' vote rates for each method.

Meanwhile, we build the Turing test set and make a Turing test. As shown in Table 7, each sample contains 6 fake glyph images generated by ZiGAN and 6 real glyph images. We ask 50 professional Chinese users to identify which images are generated in 30 sets of samples. ZiGAN achieves an accuracy of 51.6%, which is very close to random selection.

4.6 Empirical Analysis

Ablation Studies In order to verify that each step in our framework is beneficial, we did an ablation experiment. The cam loss helps the discriminator to better distinguish the differences in the details of different character styles, while the generator can make improvements in the most important regions. The style loss helps our model learn additional style and structural knowledge of unpaired data, and innovatively align the distribution of unpaired source and target data in the feature space. The alignment loss maintains the generated image with intact semantic information

Figure 6: Visualization of the attention maps:(a) Source style characters,(b) Generated target style characters,(c) Attention map of discriminator from source to target character,(d) The ground truth of target style characters.

Figure 7: Some unsatisfactory synthesis results.

from another level. The combination of these three forms our proposed method. Figure 5 displays the 200-shot image generation results without cam loss, style loss or alignment loss. Table 4 and Table 5 also present the complete quantitative results of the ablation experiment. These results show that every module of our method is critical.

Analysis of CAM Attention We visualize the local attention maps of the discriminator in Figure 6. It shows which regions the discriminator focuses its attention to determine whether the target image is real or generated. In row(c) of Figure 6, we can find that this attention module has successfully found the main body of the characters, and pay more attention to the sharp strokes and radicals with high recognition. This is consistent with our intuition, people also distinguish font styles in this way.

Failure cases As shown in Figure 7, for some extremely complex characters, there are still some subtle deficiencies in the generated results. The lack of training data leads to poor generalization performance in complex situations. For future work, We are planning to work on using fewer target references and get more robust and generalized model.

5 CONCLUSION

In this paper, we propose a novel ZiGAN, which can accomplish fine-grained Chinese calligraphy font generation with few-shot references. The main idea is that extra structural knowledge can be learned by utilizing numerous unpaired characters. We also groundbreakingly align the feature distribution of different font styles to capture valuable style knowledge in target and strengthen the coarse-grained understanding of character content. Besides, our method is an end-to-end framework that does not require any manual operation or redundant preprocessing. It can be easily and quickly adapted to new tasks.

REFERENCES

[1] Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. 2018. Multi-content gan for few-shot font style transfer. In *CVPR*. 7564–7573.

[2] Junbum Cha, Sanghyuk Chun, Gayoung Lee, Bado Lee, Seonghyeon Kim, and Hwalsuk Lee. 2020. Few-shot Compositional Font Generation with Dual Memory. *arXiv:2005.10510* (2020).

[3] Tian Qi Chen and Mark Schmidt. 2016. Fast patch-based style transfer of arbitrary style. *arXiv:1612.04337* (2016).

[4] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. 2018. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*. 8789–8797.

[5] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. Stargan v2: Diverse image synthesis for multiple domains. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8188–8197.

[6] Xueying Du, Jiangqin Wu, and Yang Xia. 2016. Bayesian relevance feedback based chinese calligraphy character synthesis. In *ICME*. IEEE, 1–6.

[7] Yue Gao, Yuan Guo, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. 2019. Artistic glyph image synthesis via one-stage few-shot learning. *ACM TOG* 38, 6 (2019), 1–12.

[8] Y. Gao and J. Wu. 2020. GAN-Based Unpaired Chinese Character Image Translation via Skeleton Transformation and Stroke Rendering. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 1 (2020), 646–653.

[9] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *CVPR*. 2414–2423.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*. 2672–2680.

[11] Arthur Gretton, Karsten Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alex J Smola. 2007. A kernel method for the two-sample-problem. In *NIPS*. 513–520.

[12] Xun Huang and Serge Belongie. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*. 1501–1510.

[13] Yaoxiong Huang, Mengchao He, Lianwen Jin, and Yongpan Wang. 2020. RD-GAN: Few/Zero-Shot Chinese Character Style Transfer via Radical Decomposition and Rendering. ECCV.

[14] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. 2017. Globally and locally consistent image completion. *ACM* 36, 4 (2017), 1–14.

[15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*. 1125–1134.

[16] Yue Jiang, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. 2017. DCFont: an end-to-end deep Chinese font generation system. In *SIGGRAPH Asia 2017 Technical Briefs*. 1–4.

[17] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*. Springer, 694–711.

[18] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv:1710.10196* (2017).

[19] Tero Karras, Samuli Laine, and Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *CVPR*. 4401–4410.

[20] Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwanghee Lee. 2019. U-gat-it: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. *arXiv:1907.10830* (2019).

[21] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. 2017. Unsupervised image-to-image translation networks. In *NIPS*. 700–708.

[22] Ming-Yu Liu and Oncel Tuzel. 2016. Coupled generative adversarial networks. In *NIPS*. 469–477.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *CVPR*. 3431–3440.

[24] Pengyuan Lyu, Xiang Bai, Cong Yao, Zhen Zhu, Tengteng Huang, and Wenyu Liu. 2017. Auto-encoder guided gan for chinese calligraphy synthesis. In *ICDAR*, Vol. 1. IEEE, 1095–1100.

[25] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *ICCV*. 2794–2802.

[26] SC Martin Arjovsky and Leon Bottou. 2017. Wasserstein generative adversarial networks. In *ICML*.

[27] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. 2015. Learning deconvolution network for semantic segmentation. In *ICCV*. 1520–1528.

[28] Li S., Xie B., Wu J., Zhao Y., Liu C., and Ding Z. 2020. Simultaneous Semantic Alignment Network for Heterogeneous Domain Adaptation. In *ACM MM*.

[29] Li S., Liu C., Xie B., Su L., Ding Z., and Huang G. 2019. Joint Adversarial Domain Adaptation. In *ACM MM*.

[30] Li S., Liu C., Lin Q., Wen Q., Su L., Huang G., and Ding Z. 2021. Deep Residual Correction Network for Partial Domain Adaptation. In *IEEE TPAMI*.

[31] Li S., Song S., Gao H., Ding Z., and Cheng W. 2018. Domain Invariant and Class Discriminative Feature Learning for Visual Domain Adaptation. In *IEEE TIP*.

[32] Danyang Sun, Tongzheng Ren, Chongxun Li, Hang Su, and Jun Zhu. 2017. Learning to write stylized chinese characters by reading a handful of examples. *arXiv:1712.06424* (2017).

[33] Donghui Sun, Qing Zhang, and Jun Yang. 2018. Pyramid Embedded Generative Adversarial Network for Automated Font Generation. In *ICPR*. IEEE, 976–981.

[34] Yuchen Tian. 2017. zi2zi: Master Chinese calligraphy with conditional adversarial networks. https://kaonashi-tyc.github.io/2017/04/06/zi2zi.html.

[35] Yu-ho Tseng and Youhe Zeng. 1993. *A history of Chinese calligraphy*. Chinese University Press.

[36] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor S Lempitsky. 2016. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images.. In *ICML*, Vol. 1. 4.

[37] Paul Upchurch, Noah Snavely, and Kavita Bala. 2016. From A to Z: supervised transfer of style and content using deep neural network generators. *arXiv:1603.02003* (2016).

[38] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*. 8798–8807.

[39] Chuan Wen, Jie Chang, Ya Zhang, Siheng Chen, Yanfeng Wang, Mei Han, and Qi Tian. 2019. Handwritten Chinese Font Generation with Collaborative Stroke Refinement. *arXiv:1904.13268* (2019).

[40] Helena TF Wong and Horace HS Ip. 2000. Virtual brush: a model-based synthesis of Chinese calligraphy. *Computers & Graphics* 24, 1 (2000), 99–113.

[41] Shan-Jean Wu, Chih-Yuan Yang, and Jane Yung-jen Hsu. 2020. CalliGAN: Style and Structure-aware Chinese Calligraphy Character Generator. *arXiv:2005.12500* (2020).

[42] Yingfei Wu, Yueting Zhuang, Yunhe Pan, and Jiangqin Wu. 2006. Web based chinese calligraphy learning with 3-d visualization method. In *ICME*. IEEE, 2073–2076.

[43] Songhua Xu, Hao Jiang, Tao Jin, Francis CM Lau, and Yunhe Pan. 2009. Automatic generation of chinese calligraphic writings with style imitation. *IEEE INTELL SYST* 2 (2009), 44–53.

[44] Songhua Xu, Francis CM Lau, William K Cheung, and Yunhe Pan. 2005. Automatic generation of artistic Chinese calligraphy. *IEEE INTELL SYST* 20, 3 (2005), 32–39.

[45] Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. 2017. Dualgan: Unsupervised dual learning for image-to-image translation. In *ICCV*. 2849–2857.

[46] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. 2018. Generative image inpainting with contextual attention. In *CVPR*. 5505–5514.

[47] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. 2019. Free-form image inpainting with gated convolution. In *ICCV*. 4471–4480.

[48] Yexun Zhang, Ya Zhang, and Wenbin Cai. 2018. Separating style and content for generalized style transfer. In *CVPR*. 8447–8455.

[49] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning deep features for discriminative localization. In *CVPR*. 2921–2929.

[50] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. 2016. Generative visual manipulation on the natural image manifold. In *ECCV*. Springer, 597–613.

[51] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*. 2223–2232.

ZiGAN: Fine-grained Chinese Calligraphy Font Generation via a Few-shot Style Transfer Approach

Qi Wen*
wenqijay@gmail.com
NetEase Fuxi AI Lab
Hangzhou, China

Shuang Li*
shuangli@bit.edu.cn
Beijing Institute of Technology
Beijing, China

Bingfeng Han
bfhan@bit.edu.cn
Beijing Institute of Technology
Beijing, China

Yi Yuan†
yuanyi@corp.netease.com
NetEase Fuxi AI Lab
Hangzhou, China

SUPPLEMENTARY

Fréchet Inception Distance For quantitative evaluation, we also use Fréchet Inception Distance to measure whether our results have completed the style translation. FID calculates the distance between the real image and the generated image in the feature space. And the feature representations are extracted from the Inception network. The lower the value indicates that the distribution of the generated images is closer to the real images distribution, and the result is better. As above, we compare six classic methods, and Table 1 shows that ZiGAN achieves the lowest FID scores.

Additional Experimental Results In addition to the results presented in the paper, we randomly select 50 common characters for each font with different styles and show supplement generation results for the datasets in Figure 1 and Figure 2.

---

*Authors contribute equally.
†Dr. Yuan is the corresponding author.

Table 1: Fréchet Inception Distance for difference font style translation mode. Lower is better. The seventh and eighth methods show the results of ablation experiments without $\mathcal{L}_{style}$ or $\mathcal{L}_{alignment}$.

		Fréchet Inception Distance (FID)							
Type	Method	Style							
		1	2	3	4	5	6	7	Mean
100 - shot	zi2zi	2.164	2.589	2.348	2.385	2.494	2.359	2.196	2.362
	pix2pix	1.789	1.739	2.335	2.322	2.208	1.574	2.318	2.041
	U-GAT-IT	1.222	0.904	1.273	0.920	**0.968**	1.218	1.288	1.113
	CycleGAN	1.608	2.743	1.480	1.659	2.425	1.230	1.707	1.836
	StarGAN	2.053	2.899	2.396	2.385	2.731	2.295	2.590	2.478
	CalliGAN	2.091	2.527	2.340	2.355	2.366	2.201	2.142	2.289
	ZiGAN w/o $\mathcal{L}_{style}$	1.254	1.842	1.827	1.779	2.759	1.914	1.6	1.854
	ZiGAN w/o $\mathcal{L}_{align}$	1.312	1.924	1.586	1.547	1.874	1.368	1.252	1.552
	ZiGAN(Ours)	**1.047**	**0.845**	**0.816**	**0.816**	1.189	**1.109**	**0.669**	**0.927**
200 - shot	zi2zi	1.380	0.756	1.649	1.010	1.270	1.119	1.359	1.220
	pix2pix	0.966	1.024	1.021	1.290	0.985	0.831	1.138	1.036
	U-GAT-IT	1.065	0.691	0.583	0.662	**0.667**	0.689	0.686	0.720
	CycleGAN	1.293	2.268	1.028	2.159	2.189	1.380	1.420	1.677
	StarGAN	2.456	2.757	2.373	2.497	2.370	2.180	2.342	2.425
	CalliGAN	1.331	1.808	1.615	0.916	1.423	1.001	1.208	1.329
	ZiGAN w/o $\mathcal{L}_{style}$	1.019	0.889	0.973	0.991	1.302	0.817	0.879	0.987
	ZiGAN w/o $\mathcal{L}_{align}$	0.517	**0.582**	0.616	**0.574**	0.883	0.720	0.729	0.660
	ZiGAN(Ours)	**0.279**	0.868	**0.476**	0.678	0.687	**0.663**	**0.654**	**0.615**

Figure 1: Visual comparison of the results of the first to fifth styles. The first row shows the source characters. The second row shows the characters generated by ZiGAN. And the third row shows the ground truth.

Figure 2: Visual comparison of the results of the sixth to ninth styles. The first row shows the source characters. The second row shows the characters generated by ZiGAN. And the third row shows the ground truth.