Few-shot Font Generation based on SAE and Diffusion Model

Yizuo Shi [1, a], Wenxia Yang [1, *], Mengxu Yuan [2, b], Zeping Yi [2, c]

[1] School of Science, Wuhan University of Technology, Wuhan, China

[2] Faculty of Science, Wuhan University of Technology, Wuhan, China

* Corresponding Author Email: wenxiayang@whut.edu.cn, [a]yii@whut.edu.cn, [b]addisonymx@gmail.com, [c]310192@whut.edu.cn

Abstract. Generating Chinese characters via few-shot font generation is an intriguing and important challenge in recent years, primarily due to the intricate and unique nature of Chinese fonts. However, the conventional GAN-based model for font generation has encountered issues such as unpredictable training and inaccurate generation. Simultaneously, in the realm of image generation, diffusion models have demonstrated remarkable success, even garnering application in AI painting commercials. Some studies have endeavored to integrate diffusion models into Few-shot Font Generation (FFG). In this paper, we present a straightforward, few-shot font generation framework utilizing a conditional diffusion model. We generate conditional embedding tokens using three encoders, which extract essential character information such as content and style. By combining these conditions into the diffusion process, we can effectively model these three pieces of information. Our model possesses three key features: i) Our model attains disentanglement of all encoders and the diffusion model. The content encoder focuses solely on extracting the content or the relative position of strokes, the style-coding provides only style features, and the diffusion model is limited to generating the target image without obscuring any content or style information. This enhances the model's interpretability and makes the addition of new functionalities a simpler process. ii) For different fonts, our model requires fewer training steps due to the use of pre-training. We only train the style-coding on a small scale, bypassing the need for extensive training of the large-scale diffusion model. iii) Our model achieves two types of "Few-shot" training. The first type involves the same style but different characters, requiring only a few characters for training. The second type pertains to different styles, needing only a few style fonts for training. Experimental results reveal that our model outperforms previous few-font generation models in terms of quality, generation speed, and the scale of well-trained training datasets.

Keywords: Chinese font, Image generation, Few-shot font generation, Diffusion model.

1. Introduction

Chinese calligraphy has fostered the unique artistic beauty of Chinese characters through the diversity of its writing styles, and Chinese font generation technology has a wide range of applications, including the creation of personal calligraphy font libraries, signature verification, restoration of ancient texts, and so on.

Early approaches mainly relied on manually extracting features [1] [2]. The performance was often unsatisfactory [3] due to the complexity and vast number of unique characters, and it required expert knowledge in calligraphy, typography, and graphic design.

With the development of deep learning, Chinese font generation is often framed as image-to-image translation or image generation problems [4] [5] [6]. Zi2zi [7] learns a mapping from one source-style font to another target-style font, facilitating the transformation of source-style font images into images of a different target style font. CalliGAN [8] introduces Chinese character component labels and style tags to generate various styles of Chinese calligraphy characters. However, these methods utilize paired data for supervised learning, which limits its practicality due to the difficulty in collecting paired data for Chinese calligraphy fonts.

Few-shot Font Generation (FFG) [9] [10] has garnered widespread attention since FFG can generate target fonts with minimal reference font images. FFG aims to use a small number of reference font images to generate new fonts based on the source font. Few-shot font style transfer

involves training a model on a large number of existing fonts, allowing it to learn how to extract font style features. He et al. [11] proposed three challenges associated with GAN-based models for font generation: i) Unstable training dynamics and difficulty achieving convergence; ii) The inherent difficulty in separating font content and style; iii) The preservation of the structural integrity of complex characters poses a significant challenge. Diff-font [11] first presents a network that uses a conditional diffusion model instead of GAN. While it exhibits commendable performance, certain highly intricate characters still exhibit issues related to stroke amalgamation.

In our previous work, we proposed Stroke GAN Autoencoder (SAE) [12] to extract the morphological information of Chinese characters from stroke images, which can be applied to the generation of stylish Chinese fonts. The SAE models reportedly outperformed several few/zero-shot methods and some of the character-based methods. The paper's conclusions assert the effectiveness of the proposed SAE models in dealing with the challenge of recognizing unseen (zero-shot) and seen Chinese characters. Furthermore, SAE highlight the potential for the approach to be extended towards improving the understanding of the structural and semantic properties of Chinese characters.

In this study, we introduce a model based on conditional diffusion for FFG. Comprising three key components, the proposed model firstly utilizes an 'anchor character' image directly sourced from a standard font library. Each character corresponds to a single anchor image. Utilizing this anchor image, we employ the encoder in the SAE [12] to extract stroke and relative position information. Furthermore, we encode the strokes of each character for additional content information. Finally, we implement an autonomous 'style-coding' layer specifically for style encoding. Except for style-coding, the other two encoders remain unaltered. During the training phase, we alternately train the style-coding layer and the diffusion model. Thanks to these three encodings, the unique features of each character can be comprehensively extracted with an independent correlation.

Our main contributions are as follows:

- We propose a generative diffusion network for few-font generation framework. Compared to GAN-based methods and the recent advanced Diff-font, our model performs better in training and the effectiveness of generation.

- We managed to engineer three distinctive encoders capable of executing the disentanglement of the diffusion model. The primary task of the content encoder is to pinpoint the substance or the relative placement of strokes, while the unique responsibility of the style-coding lies in offering the style attribute. Conversely, the diffusion model is solely commissioned with the production of the target picture, making certain that it does not obscure any content or style details. This methodology not only elucidates the model's interpretation but also eases the integration of novel characteristics.

- Our model necessitates minimal training on distinct fonts, gaining higher training efficiency. This is owing to the fact that post the initial pre-training phase, we focus on small-scale training of the style-coding, which effectively eliminates the requirement for large-scale training of the diffusion model.

- Our model manages to achieve two types of 'Few-shots' in our research. The first type involves the same style but varying characters, where training necessitates merely a handful of characters. The second type concerns differing styles, requiring only a small number of style fonts for training. This approach overcomes distinct characteristics and results in heightened training efficiency.

2. Related Works

2.1. Image-to-Image Translation

Image-to-image translation tasks involve learning a mapping function that can transform source domain images into corresponding images in the target domain while maintaining the original content and adopting the target style [13]. This concept can be applied to font generation by using image-to-image translation models to create any desired font styles from a given content font image. Image-to-image translation via generative adversarial networks (GANs) [14] is a classic problem in computer vision. In Pix2Pix [13], the generator takes an input image from one domain and generates a corresponding output image in the desired target domain, which requires paired data to guide the

generation process. To remove the need for paired data, unsupervised methods have been developed. BicycleGAN [15] enables one-to-many domain translation by building a bijection between latent coding and output modes. For many-to-many domain translation, methods such as MUNIT [16], CD-GAN [17] and FUNIT [18] disentangle the content and style representations of images, which allow the models to manipulate the content of an image while preserving its style, or vice versa. However, existing image-to-image translation methods generally focus on transforming object pose, texture, color, and style while preserving the content structure, which may not be directly applicable to font generation. Unlike natural images, font styles are primarily defined by variations in shape and specific stroke rules rather than texture and style information. Therefore, the direct application of image-to-image translation methods may not yield satisfactory results for font generation.

2.2. Few-Shot Font Generation

Few-shot font generation aims to create an entire font library with thousands of characters using just a few reference-style images. Current methods primarily use the image-to-image translation framework to transfer the source style of content characters to the reference style. Several methods have been proposed to incorporate font-specific prior information or labels, showing that such integration can enhance the quality and diversity of generated fonts. DGFont [19] uses deformable convolutional blocks in an unsupervised framework to enable effective style transfer. ZiGAN [20] employs Hilbert space to learn coarse-grained content knowledge by projecting the same character features of different styles. SC-Font [21] and DM-Font [22] utilize additional information like strokes and components to improve training. SC-Font uses stroke-level data to enhance the structural accuracy and reduce stroke errors in generated images, while DM-Font uses a dual-memory architecture to disassemble glyphs into stylized components and reassemble them into new glyphs. However, all these methods are based on GANs, which are known for their instability during training due to their adversarial objective. This often leads to mode collapse, resulting in suboptimal outcomes, especially for font styles with significant or subtle variations.

Diff-Font [11] is a one-shot font generation framework that uses a diffusion model to address challenges in generating complex characters, particularly in languages like Chinese and Korean. Unlike traditional GAN-based methods, Diff-font provides stable training and improved font style replication by managing character content through predefined embedding tokens and extracting font style from a reference image. It has demonstrated superior performance in replicating high-fidelity font styles and generating intricate characters. Unlike Diff-Font, our model decouples Chinese characters to three encodings, causing easier in component reuse and strong generalization without any re-training.

2.3. Diffusion Model

Diffusion models have witnessed significant development in the way we generate and manipulate data. Denoising diffusion probabilistic models (DDPMs) [23] consist primarily of two processes: the forward process (diffusion process) which gradually adds Gaussian noise to the original data in a Markov chain fashion, and the reverse process which takes random noise and generates the target data. This strategy has proved to be more effective in generating high-quality data. Building upon DDPM, Denoising Diffusion Implicit Models(DDIM) [24] addressed the issue of efficiency by reducing the number of steps required in the reverse process for image generation. DDIM views the diffusion process as implicit and performs sampling by solving corresponding differential equations. By utilizing advanced differential equation solvers for efficient sampling, DDIM minimizes approximation errors even with fewer sampling steps. Latent Diffusion Model (LDM [24]) shifts the diffusion process to a lower-dimensional latent space, which significantly reduces computational requirements and enhances the efficiency of image generation.

Dhariwal et al. [25] introduced a classifier-guidance mechanism that uses a pre-trained classifier to provide gradients as guidance toward generating images of the target class. Song et al. [24] extended the original model to non-Markovian cases, reducing sampling steps significantly. Nichol et al.[26],

Ramesh et al. [27] introduced a pre-trained text encoder to generate semantic latent spaces, achieving exceptional results in text-to-image tasks. Diffusion models have shown amazing results in image generation, often generating specific categories of objects or concept-driven generation guided by text prompts. They can generate high-quality data, avoid mode collapse, and have record-breaking performance in many applications. They are trained using maximum likelihood estimation, a well-understood optimization problem, making them a promising area of research in deep generative models.

2.4. StrokeGAN Autoencoder

In our previous work, we introduced a research study focusing on the development of stroke-based autoencoders (SAEs) [12] to learn the structural information of Chinese characters for improved character recognition and semantic enhancement. The study explores two distinct architectures of SAEs for different forms of datasets, aimed at addressing both zero-shot and seen Chinese character recognition tasks. The SAE models have been pre-trained on a small, focused dataset, yet they demonstrate superior performance on larger and more complex datasets when benchmarked against existing methods. Additionally, these models show promise in contributing to the creation of word embeddings that could benefit a variety of Chinese Natural Language Processing (NLP) applications, such as text classification and named entity recognition. The models are able to learn the morphological information of the characters, which has potential applications in generating new Chinese font styles and enhancing NLP tasks such as Chinese Named Entity Recognition.

In this proposed framework, based on SAE's ability of fully extracting Chinese character features, we use the SAE as an encoder in our architecture, which is able to extract more precise content information, specifically strokes with its relative position features.

3. Method

3.1. Architecture

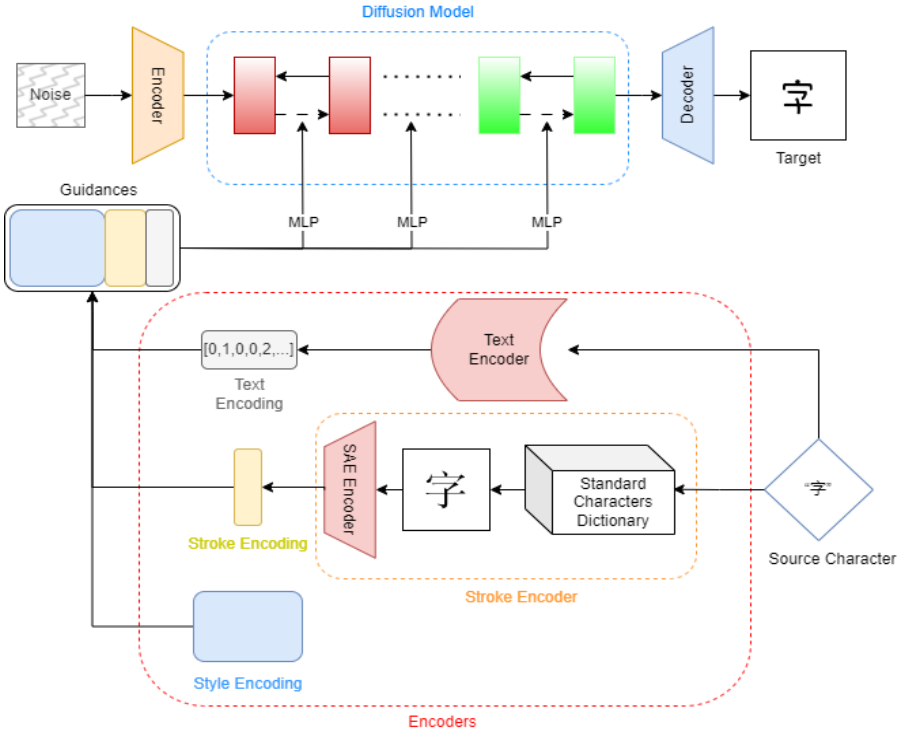

Figure 1. The framework of our model. It contains 2 parts, the encoding part and the diffusion part. In the encoding part, there are 3 encoders, namely the stroke encoder, the text encoder, and the style encoding learned from datasets.

Fig.1 illustrates the proposed framework. It consists of two parts: three character encoders for guidance, encoding the content or style of a character into a latent variable, and a diffusion generation model, which uses the latent variable as a condition to generate the character image from Gaussian noise. The character attributes encoder is designed to process the attributes of a character image separately. The guidance consists of three parts: text guidance, regarding different content characters as different tokens; image guidance, which extracts from the same character in standard characters; stroke guidance, encoding the standard image before by SAE, which can extract the stroke features of character. We combine these three encodings as a whole guidance, and then embed them as guidance for the diffusion process.

Text encoder E maps the source character (denoted as c) to text encodings (denoted as $t := E(c)$). With the help of standard characters pool (denoted as p), source character is also mapped to source image: $i = I_p(c)$. Images encodes as stroke guidance $s = S(i)$ by SAE,. The style encoding is denoted by sty. Then merge these 3 encodings into one guidance:

$$g = \mathrm{Com}(t, sty, s) = \mathrm{Com}\left(E(c), sty, S\left(I_p(c)\right)\right)$$

Which utilizes for diffusion model as condition.

3.2. Text Encoder

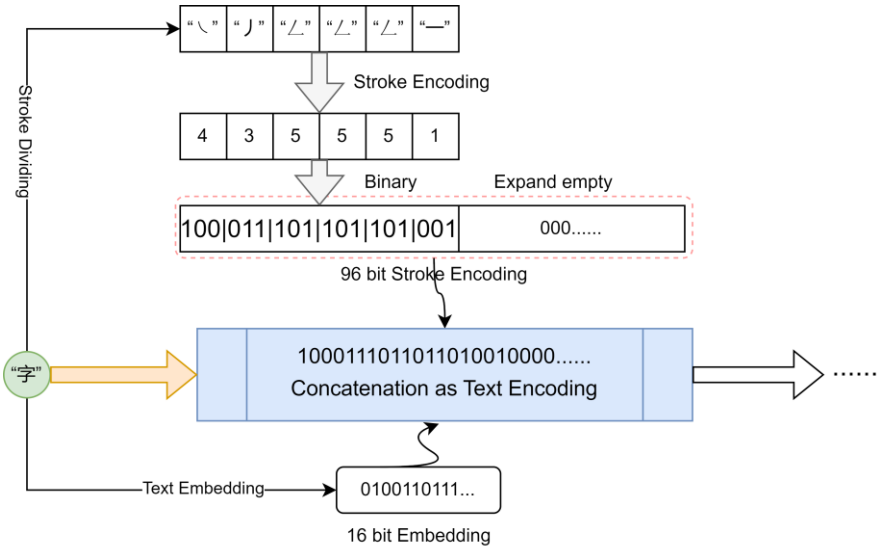

Figure 2. Text Encoder Architecture

The architecture of our text encoder (Fig. 3) in our network is designed to process Chinese characters directly. The encoder is specifically structured to handle stroke-based text encoding of individual characters. A Chinese character is first input to the *Input Layer* , and then transformed into a stroke encoding in the *Stroke Encoding Layer*. Each Chinese character is viewed as a combination of five basic strokes: the horizontal stroke , the vertical stroke, the left falling stroke, the right falling stroke, and the hook stroke as shown in Fig. 3. Considering that the number of strokes in about 6000 Chinese characters is mostly up to 30, we use 32 3-bit numbers (totally 96 bits) to encode these strokes. Each stroke is encoded as a multi-dimensional vector, and the sequence of these vectors forms the stroke encoding for the character. In the Self-Embedding Encoding Layer, the encoder generates a self-embedding encoding for the character. This layer encodes the 6000 Chinese characters using 16-bit numbers (which can encode $2^{16} = 65536$ characters). The self-embedding encoding captures the intrinsic features and semantic information of the character, mapping it to a dense vector of fixed size. The stroke encoding (96 bits) and the self-embedding encoding (16 bits) are then concatenated together in the *Concatenation Layer*. This forms a unified encoding for the character, totally 112 bits, which captures both the visual (stroke) and semantic (self-embedding) aspects of the character. The output of the text encoder is the combined encoding, and this 112-bit encoding serves as a comprehensive

representation of the Chinese character and is used as input to the subsequent layers of our network. This text encoding will be combined with other types of encoding in later stages of the network.

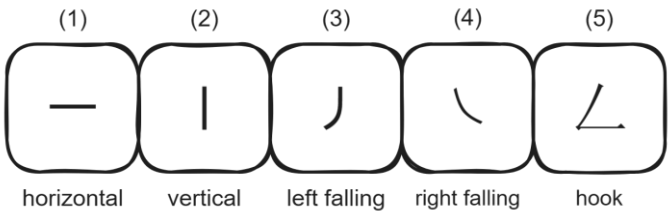

Figure 3. Five strokes in Chinese character

With this designed architecture, the text encoder can generate a rich and versatile encoding for each Chinese character, effectively supporting further processing and analysis in our network.

3.3. SAE Encoder

The SAE (Stroke AutoEncoder, Fig. 4) is a critical component that extracts rich stroke information from Chinese character images. To do this, we first convert the text into an image using a standard Chinese character library. The SAE Encoder then processes this image and generates a 128-bit encoding. With the help of SAE's ability to extract relative position and stroke features of characters, SAE encoding can correctly lead the generation process in terms of stroke features.

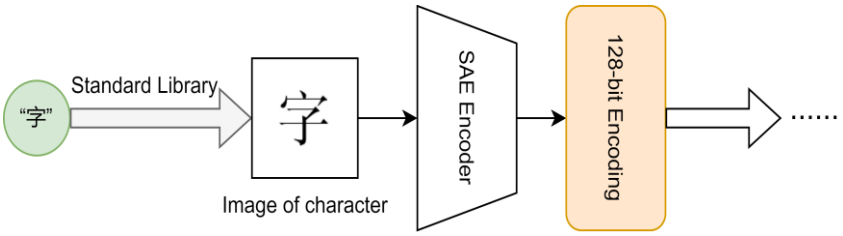

Figure 4. SAE Encoder

3.4. Style Encoding and Combining Encoding

The style encoding as depicted in Fig. 5 is a distinct network. It does not stem from any specific encoders but self-adjusts according to the network's backward propagation. Comprising 272 bits, the style encoding separates style and content. The style isn't extracted from a particular Chinese character text, rather it's derived from the image features of a group of characters. We then merge three encodings - the text encoding, SAE encoding, and style encoding - into an integrated encoding (called combining encoding). Amounting to a total of 512 bits (272 for style, 128 for SAE, and 112 for text), this combined encoding is utilized in the subsequent diffusion process. This consolidated encoding encompasses all vital information from the separate encodings, paving the way for advanced stages in the deep learning network.

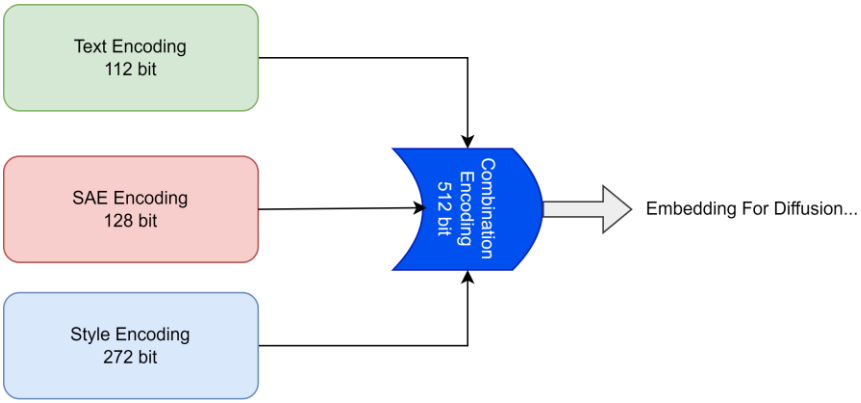

Figure 5. Style Encoding With Combination

3.5. Conditional Diffusion Model

In our method, we regard each raw image of the character attributes as a sample in the whole training data distribution, and denote the sample as $x_0 \sim q\big(x_0|\mathrm{Com}(t, sty, s)\big)$. Like the thermal motion of molecules, we add random Gaussian noise to the image thousands of times to gradually transform it from a stable state to a chaotic state. This process is called diffusion process and can be defined as:

$$q(x_{1:T}|x_0) = \prod_{t=1}^{T} q\,(x_t|x_{t-1}) \tag{1}$$

Where

$$q(x_t|x_{t-1}) = N\big(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\big), t = 1, \dots, T \tag{2}$$

And $\beta_1 < \cdots < \beta_T$ is a variance schedule. According to the (2), x_t can be rewritten as:

$$\begin{aligned} x_t &= \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\epsilon_{t-1}, \quad \epsilon_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &= \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, \bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i\,, \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ &\sim \mathcal{N}\big(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\big) \end{aligned}$$

Where $\alpha_t = 1 - \beta_t$, and α_t is negatively correlated with β_t, therefore $\alpha_1 > \cdots > \alpha_T$. When the $T \to \infty$, $\bar{\alpha}_T$ close to 0, x_T nearly obeys $N(0, I)$ and the posterior $q(x_{t-1}|x_t)$ is also Gaussian. So in the reverse process, we can sample a noisy image x_T from an isotropic Gaussian and generate the designated character image by denoising x_T in the long Markov chain with a multi-attributes condition z = f (c, s) (if using the optional condition, then, z = f (c, s, op)) that contains the semantic meaning of character. Since the posterior $q(x_{t-1}|x_t)$ is hard to estimate, we use p_θ to approximate the posterior distribution which can be denoted as

$$p_\theta(x_{0:T} \mid z) = p(x_T) \prod_{t=1}^{T} p_\theta\,(x_{t-1} \mid x_t, z),$$

$$p_\theta(x_{t-1} \mid x_t, z) = \mathcal{N}\big(\mu_\theta(x_t, t, z), \Sigma_\theta(x_t, t, z)\big),$$

Following DDPM Ho et al. (2020), we set $\Sigma_\theta(x_t, t, z)$ as constants and the diffusion model $\theta(x_t, t, z)$ learns to predict the noise added to x_0 in diffusion process from x_t and condition z for easier training. Through these simplified operations, we can adopt a standard MSE loss to train our multi-attributes-conditional diffusion model:

$$L = \mathbb{E}_{x_0 \sim q(x_0), \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), z}\big[\|\epsilon - \epsilon_\theta(x_t, t, z)\|^2\big]$$

4. Experiment

4.1. Experiment

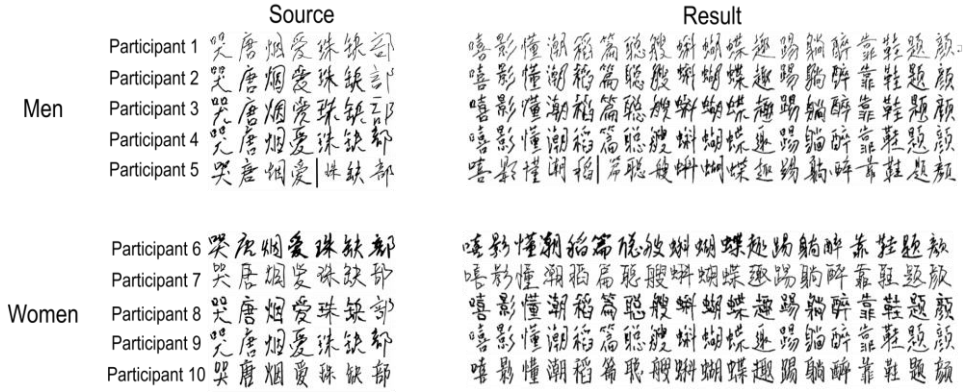

Figure 6. Different gender generating results with 10 participants.

We collected 200 fonts as entire dataset. Each font has 6,625 Chinese characters which cover almost all commonly used Chinese characters. Specifically, we use 60 % (120) fonts to train our model, and in each font we only take 10 % (about 600) of the Chinese characters for training. We conduct testing in two dimension. One is testing for the effectiveness of trained font generation; that is, for every font in the trained 120 fonts, we test the performance of generating the rest of 90 % Chinese characters; the other is testing for the effectiveness of generating untrained fonts. We froze all models' parameters except for the style-coding, then input only 100 Chinese characters for simple training; before testing the generation performance of the rest of the 6,525 Chinese characters.

4.2. Evaluation Metrics

To make a quantitative comparison between our method and other advanced approaches, we employ commonly used evaluation metrics in image generation tasks. These metrics include SSIM (Structural Similarity) [30], RMSE (Root Mean Square Error) [31], LPIPS (Learned Perceptual Image Patch Similarity) [32], and FID (Fréchet Inception Distance) [33]. SSIM mimics the human visual system and assesses the structural similarity between two images by considering aspects such as luminance, contrast, and structure. RMSE, on the other hand, calculates the root mean square error of pixel values to evaluate the similarity between two images at a pixel-level. LPIPS operates at a perceptual level and measures the distance between images in a deep feature space. FID measures the difference between generated and real images in terms of their distributions. Additionally, we conduct a user study following a similar approach to [34] to obtain human feedback and evaluation of our method. This user study provides valuable insights through human testing.

4.3. Setup

Encoders for guidance. Our model has three parts of encodings. Since SAE performs well on the feature extraction for strokes of characters, we use an encoder in SAE to encode the image of the original character, which encodes a 128*128 image to a 1-d 64-bit encoding. We freeze this encoder to get stable encoding. Then, we use the strokes encoding to encode characters into a 128-bit encoding. These two encoders can be seen as a whole content encoder. The last style coding is not an encoder but just a network layer in 256-bit. It learns the style of each font in training steps. These three encodings are combined as a whole embedding with a size of 448-bit. This embedding is directly inserted into every diffusion process as a condition.

Conditional Diffusion Model. Our diffusion model is based on DDPM architecture. Table 1 is the hyper-parameters of our model.

Table 1. Hyper-parameters of Diffusion model.

Parameter	Value
Batch size	64
Channels	128
Diffusion Steps	1000
Learning rate	1e-4
Optimizer	Adam
Loss	MSE

4.4. Comparison

We choose some GAN-based Chinese Font Generation models and Diff-font to make comparisons with our model. MX-Font [34] extracts information by using multi-headed encoders. DG-Font [19] uses deformable convolution to replace traditional convolution. Diff-font uses a diffusion process and simple encoders. ZiGAN [20] is a powerful end-to-end Chinese calligraphy font generation framework that can generate fine-grained target style characters with few-shot references.

Quantitative comparison. As Table 2 shows, in comparison between our model and other models, our model performs the best on all metrics (LPIPS, RMSE, FID, SSIM).

Table 2. Quantitative comparison of our model against other state-of-the-art font generation models. ↓ indicates that a smaller value is better.

Model	LPIPS(↓)	RMSE(↓)	FID(↓)	SSIM(↑)
MX-Font	0.174	0.192	31.4	0.703
DG-Font	0.169	0.187	38.7	0.709
ZiGAN	0.203	0.311	52.1	0.689
Diff-Font	0.133	0.176	29.2	0.738
Our model	0.124	0.169	28.6	0.744

Qualitative comparison. We compare the generation results of all these 5 model showing in Fig. 7. It shows that MX-Font has poor quality of generation. ZiGAN and DG-Font work well for most characters; but have a joined-up situation in some characters. Diff-Font generates very well, but some generated words have not studied the style of reference. Our model can not only generate given content precisely, but also maintain the features of the font style.

Figure 7. Qualitative comparison among 5 models.

4.5. Ablation Studies

Evaluating the effectiveness of encodings. Embedding within our proposed model are three pivotal encodings that significantly influence the generation process. One may naturally question, are these encodings uniformly effective? To ascertain the answer, we conducted three distinct ablation studies. In these studies, we systematically omitted each encoding and subsequently collated and compared the resultant findings. Please note that the process of removing an encoding implies that the associated encoding is set to zero, while the original dimension remains unaltered. The quantitative findings gleaned from this multifaceted analysis are visually represented in Table 3.

Table 3. The effectiveness of encodings in ablation study.

Experiment	LPIPS(↓)	RMSE(↓)	FID(↓)	SSIM(↑)
Remove Stroke Enc.	0.451	0.340	59.8	0.211
Remove Text Enc.	0.257	0.201	33.7	0.686
Remove Style Enc.	0.388	0.279	52.2	0.342
Original	0.124	0.169	28.6	0.744
Our model	0.124	0.169	28.6	0.744

The data, as scrutinized from the results, dictates that all three encodings play vital roles in the model's operation. However, the Stroke Encoder emerged as the most consequential followed by the Style Encoder, with the Text Encoder bringing up the rear. This sequential order of importance elucidates the operational efficiencies of all the encodings. The Stroke Encoding was identified as the most influential, while the text encoding was found to be the least effective.

Assessing the impact of number of training data. It's indisputable that a more substantial pool of training data will yield more precise and accurate results when generating data. However, in reality,

users may often only provide a limited supply of handwriting characters. In light of this, to optimize the balance between the quality of generated data and the volume of training data, we embarked on an experiment aimed at identifying the optimum number of training data. The experiment involved testing several training scale limits, specifically: 30, 50, 80, 100, 150, 200, and 500 units of characters. Our chosen method of evaluation was SSIM, and the associated trend of SSIM concerning the training data scale is graphically depicted in Fig. 6.

Analyzing the impact of the number of characters in training strokes on the model's effectiveness. The cardinality of characters found within the training strokes can profoundly influence the quality of the data generated by the model. In order to gauge the extent of this impact, we carried out an experiment wherein we varied the number of characters in training strokes and measured the subsequent effect on the model's performance, using SSIM as our yardstick. The results of this investigative analysis are presented in Fig. 8.

Our findings suggested that a low number of characters in the training strokes could cause underfitting, leading to a noteworthy decline in the SSIM. Conversely, when the count of characters in the training strokes was significantly high, overfitting ensued, which also detrimentally affected the SSIM, albeit to a lesser degree than underfitting. The data generated by our experiment indicated that the SSIM experienced a substantial elevation when the range of strokes was between 1 to 10. However, as the range of strokes extended from 10 to 15, the SSIM increased only marginally, with the improvements plateauing beyond this range. Strikingly, when the total number of strokes was beyond the 15-stroke threshold, the SSIM experienced a gradual diminishing trend.

We also composed a histogram illustrating the typical quantity of strokes in characters, as seen in Fig. 9, which revealed an average of approximately 10.25 strokes per character. Given this average, and to facilitate user convenience, we chose to implement 10 strokes per character in our training dataset. For each set of training data, we randomly selected characters with 10 strokes. This methodology strikes a balance between underfitting and overfitting, ensuring a high standard of quality in our generated data, while simultaneously maintaining user convenience.

In examining the graphical data, it is readily evident that there is a marked increase in the SSIM correlated with an increase in the count of characters, but only up until approximately 100 characters. Beyond this number, the SSIM began to plateau, reaching a stable value. This observation suggests that an excess of 100 training data offers minimal incremental benefit to the SSIM. We assume that a greater number of characters would generate redundant strokes or components. In view of this, we selected 100 as the optimum count of training data.

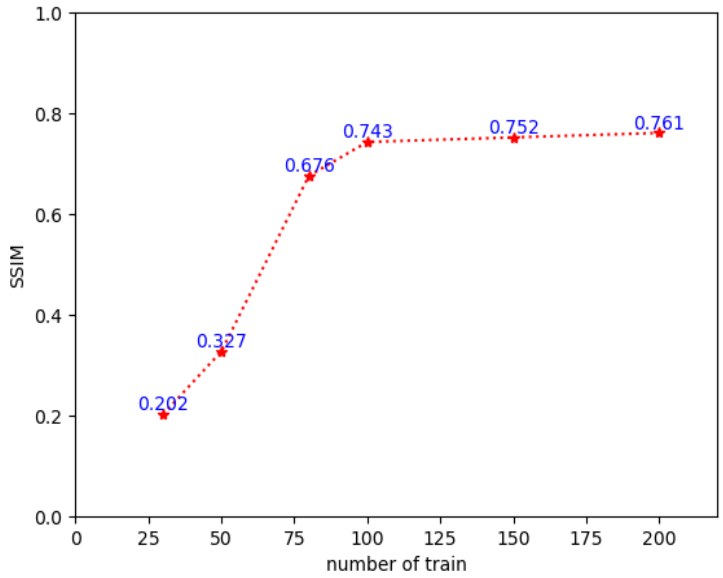

Figure 8. Ablation for number of trainings. SSIM index was used for measuring effectiveness of training.

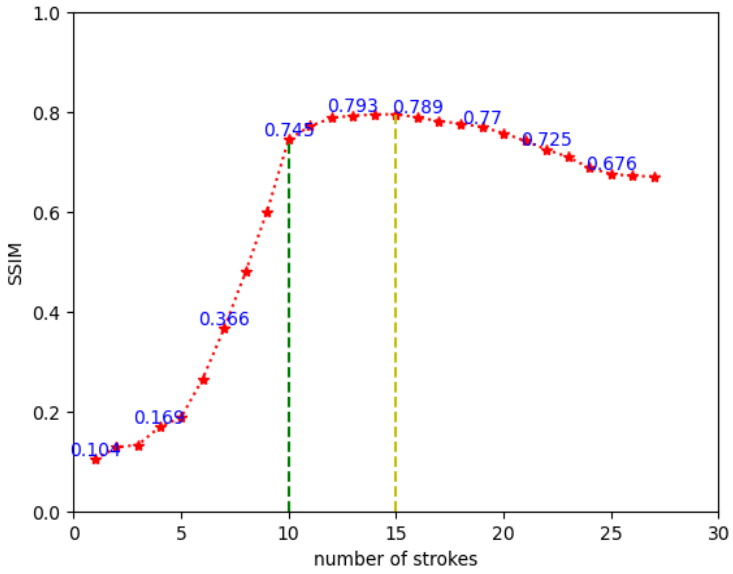

Figure 9. The effectiveness of the number of characters in training strokes in ablation study

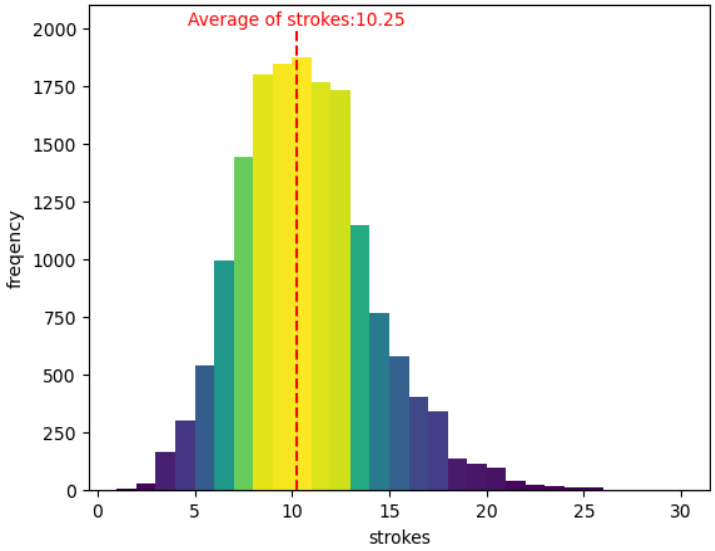

Figure 10. Analysis of the impact of number of Chinese strokes. Average number of strokes of Chinese characters was labelled.

4.6. Other research

Behaviors on handwriting and calligraphy fonts. The training and testing sets utilized so far primarily come from open-source computer-designed vector fonts. These fonts are characterized by their stable structure, uniform thickness, distinct style, but without noticeable random style fluctuations. However, the font structure of people's daily handwriting is more disorganized compared to the aforementioned fonts, the thickness is uneven, and the handwriting style can vary randomly due to various external factors in daily writing. As for calligraphic fonts, the thickness, brush strokes and other brushstroke changes are even more pronounced. Both of these are different from the typical computer vector fonts. Therefore, it's essential to study these two types of fonts separately. This model is based on the Diffusion process for generation, during which random sampling is implemented, leading to different results each time; hence, it is more suitable for generating fonts with certain random style adjustments.

We separately trained the model on daily handwritten fonts and calligraphy fonts, and the generation results are as Fig. 10.

The quantitative results (SSIM) are as follows: handwriting font 0.738, calligraphy font 0.722. As can be seen from Fig.10, the generated results basically imitate the style of the original font, while adding random style fluctuations. At the same time, the model also effectively captures the font's features such as connected strokes, brush tips, and thickness. Therefore, this model performs well in generating handwriting and calligraphy fonts, with good representations of the fonts' random style and local fine-grained features.

Comparison of Masculine and Feminine Handwriting Fonts. In this section, we differentiate and compare the generation effects of masculine and feminine fonts. The reason for distinguishing between these two genders is that, from an objective and universal perspective, there are significant differences in the structure, morphology, thickness, etc., of male and female fonts. As most of the fonts in the training set are male fonts, we separately compare them to test the generalization ability of this model. We found 5 males and 5 females to conduct font tests, each writing 120 fixed characters (100 for the training set, 20 for the test set). Each model is trained and generates the corresponding 20 characters for the test set and a new 500 characters, and then the results are quantitatively and qualitatively evaluated. The results obtained are seen in Fig. 11.

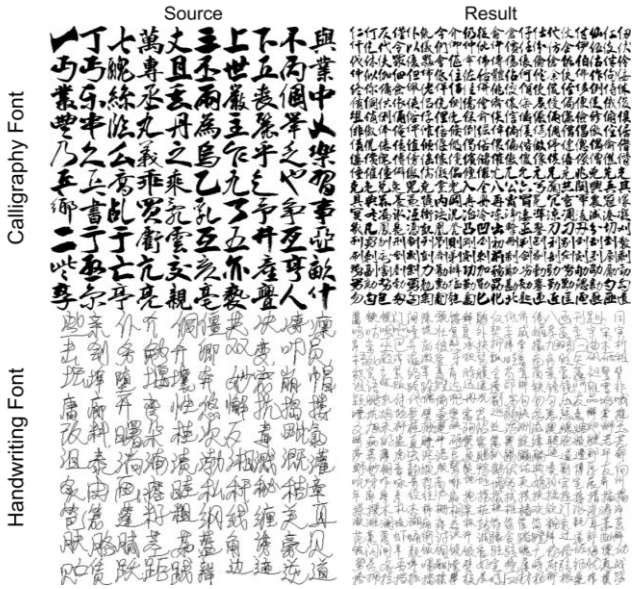

Figure 11. Comparation of handwriting and calligraphy fonts generating results

Firstly, for the input fonts, male fonts are more straight and powerful, with distinct thickness and clearly defined brush strokes; female fonts are more delicate and smooth, with a more balanced overall structure. Looking at the generated results, the model has effectively captured the differences in male and female fonts while maintaining consistency with the basic style and font itself. That is, it has strong generalization ability in terms of stroke thickness, structure control, and the curvature of the strokes.

Also, for the results of the above test, a quantitative detection was performed using the SSIM evaluation for comparison, with the results showing in Fig. 12.

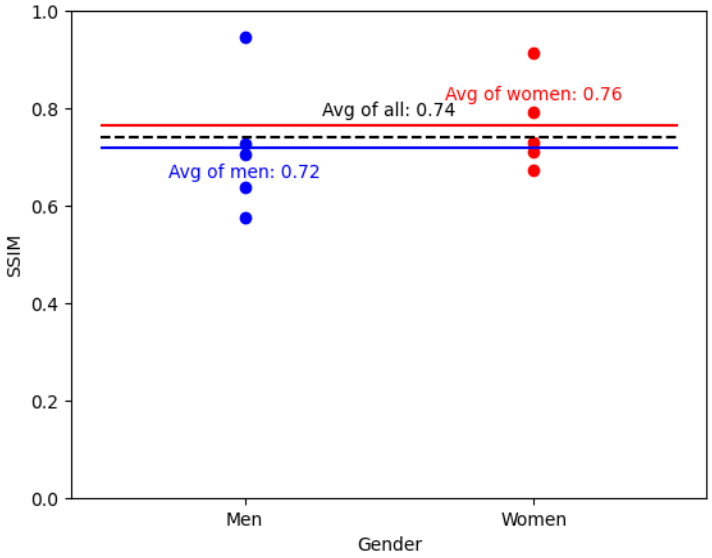

Figure 12. Analysis of SSIM evaluation in different genders

It can be seen that the average quality of the generated results for females is slightly higher than that for males. However, after performing a t-test (p-value is 0.5634) on the two samples, it can be reasonably assumed that the results of the two sample groups are consistent. This indicates that the model has a strong generalization ability. Similarly, it can be observed from the chart that the sample shift is not large (with a variance of approximately s=0.11), meaning that the model is quite stable in creating a variety of generated results.

5. Conclusion

In conclusion, few-shot Chinese character generation presents a significant and intriguing challenge, given the complexity and specificity of Chinese fonts. Traditional GAN-based models for font generation suffer from issues such as unstable training and imprecise generation. However, the diffusion model has shown excellent performance in image generation, even finding applications in AI painting commercials. Some prior work has attempted to apply the diffusion model to few-shot font generation. In this paper, we propose a simple yet effective few-shot font generation framework using a conditional diffusion model. Our framework leverages three encoders to generate condition embedding tokens, extracting key information such as content and style. By incorporating these conditions into the diffusion process, our model effectively captures and models the three types of information. Notably, our model achieves disentanglement of all encoders and the diffusion model, enhancing model interpretability and facilitating the addition of new functionalities. Additionally, our approach requires fewer training steps for different fonts, as we only train the style-coding with smaller scales after pre-training, avoiding the need for large-scale diffusion model training. We achieve two forms of few-shot generation: one for the same style but different characters, requiring only a few characters for training, and another for different styles, needing only a few style fonts for training. Experimental results demonstrate that our model outperforms previous few-shot font generation methods in terms of quality, generating speed, and scalability of well-trained training datasets. Overall, our proposed framework shows promise in addressing the challenges of few-shot Chinese character generation and opens up avenues for further research and improvement in this domain.

References

[1] D. Lu, “Contemporary Font Design Method Inspired by Chinese Character Tradition,” Packaging Engineering, vol. 44, no. 4, pp. 248 – 254, 2023.

[2] W. Shi, “Research on The Evolution of Visual Expression Form of Chinese Character Font,” Design in Digital Age, vol. 35, no. 19, pp. 44 – 46, 2022.

[3] L. Yan, "Discussing the complete set of Chinese font design in the course of font design.," Packaging World, no. 3, pp. 64 – 66, 2017.

[4] Z. Lai, C. Tang, and J. Lv, "Arbitrary Chinese Font Generation from a Single Reference," in 2020 International Joint Conference on Neural Networks (IJCNN), Jul. 2020, pp. 1 – 7.

[5] C. Wen, Y. Pan, J. Chang, Y. Zhang, S. Chen, Y. Wang, M. Han, and Q. Tian, "Handwritten Chinese Font Generation with Collaborative Stroke Refinement," in 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), Jan. 2021, pp. 3881 – 3890.

[6] J. Zeng, Q. Chen, Y. Liu, M. Wang, and Y. Yao, "Strokegan: Reducing mode collapse in Chinese font generation via stroke encoding," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, 2021, pp. 3270 – 3277.

[7] Y. Tian, "Zi2zi: Master Chinese Calligraphy with Conditional Adversarial Networks," Apr. 2017. [Online]. Available: https://kaonashi-tyc.github.io/2017/04/06/zi2zi.html.

[8] S.-J. Wu, C.-Y. Yang, and J. Y.-j. Hsu, "CalliGAN: Style and Structure-aware Chinese Calligraphy Character Generator," May 2020. [Online]. Available: http://arxiv.org/abs/2005.12500.

[9] M. Yao, Y. Zhang, X. Lin, X. Li, and W. Zuo, "VQ-Font: Few-Shot Font Generation with Structure-Aware Enhancement and Quantization," Aug. 2023. [Online]. Available: http://arxiv.org/abs/2308.14018.

[10] X. He, M. Zhu, N. Wang, X. Gao, and H. Yang, "Few-shot Font Generation by Learning Style Difference and Similarity," Jan. 2023. [Online]. Available: http://arxiv.org/abs/2301.10008

[11] H. He, X. Chen, C. Wang, J. Liu, B. Du, D. Tao, and Y. Qiao, "Diff-Font: Diffusion Model for Robust One-Shot Font Generation," May 2023. [Online]. Available: http://arxiv.org/abs/2212.05895.

[12] Z. Chen, W. Yang, and X. Li, "Stroke-Based Autoencoders: Self-Supervised Learners for Efficient Zero-Shot Chinese Character Recognition," Applied Sciences, vol. 13, no. 3, p. 1750, Jan. 2023. [Online]. Available: https://www.mdpi.com/2076-3417/13/3/1750.

[13] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," Nov. 2018. [Online]. Available: http://arxiv.org/abs/1611.07004.

[14] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio, "Generative Adversarial Nets," in NIPS, Dec. 2014. [Online]. Available: https://www.semanticscholar.org/paper/Generative-Adversarial-Nets-Goodfellow-Pouget-Abadie/54e325aee6b2d476bbbb88615ac15e251c6e8214.

[15] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward Multimodal Image-to-Image Translation," Oct. 2018. [Online]. Available: http://arxiv.org/abs/1711.11586.

[16] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal Unsupervised Image-to-Image Translation," Aug. 2018. [Online]. Available: http://arxiv.org/abs/1804.04732.

[17] J.-J. Wang, N. Dobigeon, M. Chabert, D.-C. Wang, T.-Z. Huang, and J. Huang, "CD-GAN: A robust fusion-based generative adversarial network for unsupervised remote sensing change detection with heterogeneous sensors," Nov. 2023. [Online]. Available: http://arxiv.org/abs/2203.00948.

[18] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz, "Few-Shot Unsupervised Image-to-Image Translation," Sep. 2019. [Online]. Available: http://arxiv.org/abs/1905.01723.

[19] Y. Xie, X. Chen, L. Sun, and Y. Lu, "DG-Font: Deformable Generative Networks for Unsupervised Font Generation," Apr. 2021. [Online]. Available: http://arxiv.org/abs/2104.03064

[20] Q. Wen, S. Li, B. Han, and Y. Yuan, "ZiGAN: Fine-grained Chinese Calligraphy Font Generation via a Few-shot Style Transfer Approach," in Proceedings of the 29th ACM International Conference on Multimedia, Oct. 2021, pp. 621 – 629. [Online]. Available: http://arxiv.org/abs/2108.03596.

[21] Y. Jiang, Z. Lian, Y. Tang, and J. Xiao, "SCFont: Structure-Guided Chinese Font Generation via Deep Stacked Networks," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, no. 01, pp. 4015 – 4022, Jul. 2019. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/4294.

[22] J. Cha, S. Chun, G. Lee, B. Lee, S. Kim, and H. Lee, "Few-shot Compositional Font Generation with Dual Memory," Jul. 2020. [Online]. Available: http://arxiv.org/abs/2005.10510.

[23] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," Dec. 2020. [Online]. Available: http://arxiv.org/abs/2006.11239.

[24] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," Oct. 2022. [Online]. Available: http://arxiv.org/abs/2010.02502.

[25] P. Dhariwal and A. Nichol, "Diffusion Models Beat GANs on Image Synthesis," Jun. 2021. [Online]. Available: http://arxiv.org/abs/2105.05233.

[26] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models," Mar. 2022. [Online]. Available: http://arxiv.org/abs/2112.10741.

[27] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, "Hierarchical Text-Conditional Image Generation with CLIP Latents," Apr. 2022. [Online]. Available: http://arxiv.org/abs/2204.06125.

[28] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes, T. Salimans, J. Ho, D. J. Fleet, and M. Norouzi, "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding," May 2022. [Online]. Available: http://arxiv.org/abs/2205.11487.

[29] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," Apr. 2022. [Online]. Available: http://arxiv.org/abs/2112.10752.

[30] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600 – 612, Apr. 2004. [Online]. Available: http://ieeexplore.ieee.org/document/1284395/.

[31] A. Botchkarev, "Performance Metrics (Error Measures) in Machine Learning Regression, Forecasting and Prognostics: Properties and Typology," Interdisciplinary Journal of Information, Knowledge, and Management, vol. 14, pp. 045 – 076, 2019. [Online]. Available: http://arxiv.org/abs/1809.03006.

[32] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The Unreasonable Effectiveness of Deep Features as a Perceptual Metric," Apr. 2018. [Online]. Available: http://arxiv.org/abs/1801.03924.

[33] M. Heusel, H. Ramsauer, T. Entertainer, B. Nessler, and S. Hochreiter, "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium," Jan. 2018. [Online]. Available: http://arxiv.org/abs/1706.08500.

[34] S. Park, S. Chun, J. Cha, B. Lee, and H. Shim, "Multiple Heads are Better than One: Few-shot Font Generation with Multiple Localized Experts," Apr. 2021. [Online]. Available: http://arxiv.org/abs/2104.00887.