This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Generate Like Experts: Multi-Stage Font Generation by Incorporating Font Transfer Process into Diffusion Models

Bin Fu[1], Fanghua Yu[1], Anran Liu[3], Zixuan Wang[1], Jie Wen[4], Junjun He[2], and Yu Qiao[1,2*]

[1]ShenZhen Key Lab of Computer Vision and Pattern Recognition,
Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[2]Shanghai Artificial Intelligence Laboratory [3] The University of Hong Kong
[4] Harbin Institute of Technology, Shenzhen
[1]{bin.fu, zx.wang8, yu.qiao}@siat.ac.cn, [2]{hejunjun, qiaoyu}@pjlab.org.cn
[3]liuar616@connect.hku.hk, [4]jiewen_pr@126.com

Abstract

Few-shot font generation (FFG) produces stylized font images with a limited number of reference samples, which can significantly reduce labor costs in manual font designs. Most existing FFG methods follow the style-content disentanglement paradigm and employ the Generative Adversarial Network (GAN) to generate target fonts by combining the decoupled content and style representations. The complicated structure and detailed style are simultaneously generated in those methods, which may be the sub-optimal solutions for FFG task. Inspired by most manual font design processes of expert designers, in this paper, we model font generation as a multi-stage generative process. Specifically, as the injected noise and the data distribution in diffusion models can be well-separated into different sub-spaces, we are able to incorporate the font transfer process into these models. Based on this observation, we generalize diffusion methods to model font generative process by separating the reverse diffusion process into three stages with different functions: The structure construction stage first generates the structure information for the target character based on the source image, and the font transfer stage subsequently transforms the source font to the target font. Finally, the font refinement stage enhances the appearances and local details of the target font images. Based on the above multi-stage generative process, we construct our font generation framework, named MSD-Font, with a dual-network approach to generate font images. The superior performance demonstrates the effectiveness of our model. The code is available at: https://github.com/fubinfb/MSD-Font .

*Corresponding author: Yu Qiao

1. Introduction

Designing new fonts is time-consuming and labor-intensive work for humans, especially for some glyph-rich scripts (e.g., Chinese and Korean). The few-shot font generation (FFG) task aims to automatically generate new stylized fonts with a few reference images, and it has received significant interest due to its academic and commercial values. As the requirement of this task is to render the character images into the target font style, current FFG methods [17, 27, 28, 39] usually employ the Generative Adversarial Network (GAN) to perform few-shot font generation under the style-content disentanglement paradigm. These models usually simultaneously generate the complicated structure and detailed style of target font images by combining the content features from source characters and the style representations from reference font images. Although many methods have been proposed to obtain better content and style representations, the generated font images still contain artifacts, inconsistent styles, and broken structures.

Inspired by the common practice of expert designers, we find the following characteristics that may be crucial for high-quality font generation: As font design requires preserving the global structure of the target characters, designers usually generate a new font by modifying from a similar existing font, which can be viewed as a transfer process from the source font to the target font. Thus, most manual font designs can be regarded as a multi-stage process, including template font generation for global structure, font transfer from the template font to the target font, and refinement of fine-grained details. In light of this, we think that the multi-stage generative process is an effective solution for high-quality font generation, since the problems of broken structures, inconsistent styles, and artifacts can be

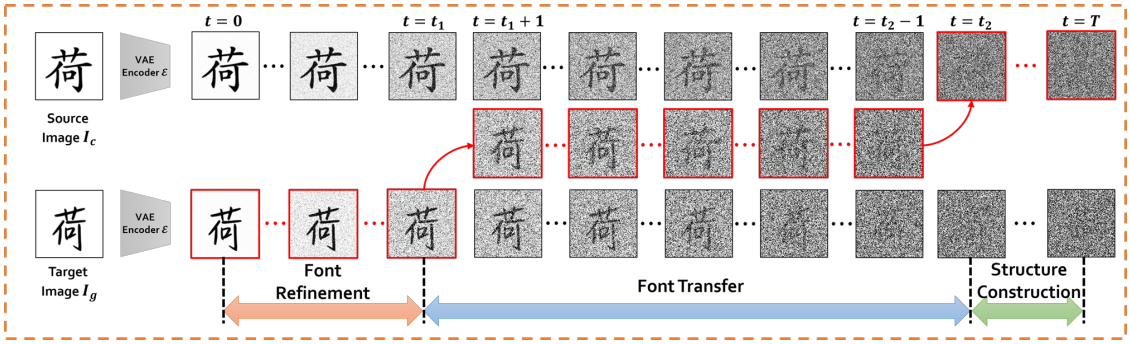

Figure 1. The forward process of our proposed model and the traditional diffusion model (DDPM). The first and third rows display the traditional forward process for the source image I_c and target font image I_g. The images with red boxes mark the path of our proposed diffusion process. In our model, we incorporate the font transfer process in time interval $(t_1, t_2]$, and align the distribution with the traditional diffusion model at the timesteps t_1 and t_2. With this forward process, the generative process of our model can be separated into three stages: the structure construction stage, the font transfer stage, and the font refinement stage, respectively. Note that the font images in this figure are visualized from the diffusion model in image space for better illustration[1], while we implement our model in latent space to reduce computational consumption.

better handled by the above three stages.

The diffusion model provides a powerful generative framework to achieve this multi-stage process, based on the following two features: (1). Since the injected noise and the data distribution in forward process can be well-separated into different sub-spaces, we are able to incorporate the font transfer process into diffusion models. (2). Recent studies [7, 18] have revealed that the different steps in the generative process have different functions for image synthesis. Therefore, modeling font generation as a transfer process and constructing a multi-stage generative process based on diffusion model is a promising direction for FFG task.

Motivated by the above observations, in this paper, we incorporate the font transfer process into the diffusion model and formulate font generation as a multi-stage generative process. To alleviate computational consumption, we construct this diffusion process in latent space. Specifically, in latent space, we first separate the injected noise and the data distribution into different sub-spaces, and then incorporate the font transfer process into the data sub-space in a predefined time interval $(t_1, t_2]$. Combining with the denoising process, the font generative process in our proposed diffusion model can be automatically separated into a multi-stage process (shown in Fig. 1): the structure construction stage, the font transfer stage, and the font refinement stage, respectively. The structure construction stage first generates the structure information for the target character based on the source image. Then, the font transfer stage gradually transforms the source font to the target font. Finally, the font refinement stage improves the appearances and local details of the generated font images. Under this scheme, the structure construction stage generates the global character structure from source images, while the font transfer stage provides a long generative path to produce detailed styles. Moreover, as the prediction network has different behaviors in font transfer stage and font refinement stage, we further propose a dual-network approach to predict the target latent features for different stages. Experiments demonstrate that our model achieves the superior performance on FFG tasks, verifying the effectiveness of our proposed approach.

In summary, our contributions in this paper are summarized as follows:

(1). Instead of simultaneously generating font images with complicated structures and detailed styles in most traditional FFG methods, we reformulate the font generation as a multi-stage generative process by incorporating font transfer process into the diffusion models.

(2). We construct the multi-stage font generation framework, named MSD-Font, with a dual-network approach to better deal with different behavious in different generative stages.

(3). Experimental results show that our method achieves superior performance on the FFG task, demonstrating the effectiveness of our proposed method.

2. Related Works

This section offers a concise review of recent development in font generation methods and diffusion models.

2.1. Many-shot Font Generation

As font generation can be formulated as mapping font images from the source domain to the target domain, early methods [4, 40, 41] usually utilize the image-to-image (I2I) methods [8, 14, 20, 50, 51] to learn the mapping function between different fonts. The mapping function will be fine-tuned on hundreds of reference samples (many-shot) to generate unseen font images. Based on the pix2pix [14] framework, Zi2zi [40] and Rewrite [41] utilize one-hot style

[1]Specifically, we replace z_0^c and z_0^g with I_c and I_g in Eq. (8)-(12) to visualize our proposed diffusion process in this figure.

labels to optimize font generation models in a supervised manner. HGAN [4] further extends Zi2zi model via a transfer network and the hierarchical adversarial discriminator. DC-font [15] utilizes a feature reconstruction network to embed style information for font synthesis. AGEN [23] employs the auto-encoder to transfer standard font to calligraphy font. Although these many-shot font generation methods can generate promising font images, collecting large numbers of reference samples to fine-tune the mapping function is still a difficult task for font generation.

2.2. Few-shot Font Generation

To relieve the weakness of the above models, few-shot font generation (FFG) methods have become popular solutions in recent years, which generate font images with a few samples. As font styles have complex structures and fine-grained details, the common-used statistics-based style transfer techniques are not suitable for font generation. Instead, recent FFG models [2, 3, 10, 17, 21, 26–28, 39, 45–47, 49] mainly follow the style-content disentanglement paradigm, which transfers the target font styles by combining the content representations of source characters with the style codes of reference samples. SA-VAE [38], EMD [49], and AGISNet [10] extract the style and content representations as global features to generate target font images. DM-Font [3], LF-Font [27], and MX-Font [28] decompose characters into components and learn component-wise representations to improve local details. CG-GAN [17] supervises the font generator to decouple content and style at the component level via a component-wise discriminator. Diff-Font [11] proposes a stroke-wise diffusion model to generate font images under the guidance of three character attributes (content, style, and strokes). FsFont [39] utilizes a cross-attention mechanism to aggregate style features into the fine-grained style representation. XMP-Font [21] develops a self-supervised cross-modality pre-training strategy and a cross-modality transformer-based encoder to model the style representations at all levels. DS-Font [12] proposes a cluster-level contrastive style loss to learn better style representations. DG-Font [45] and its extended version DG-Font++ [5] propose a feature deformation skip connection to learn geometric displacement maps between different fonts. CF-Font [43] further extends DG-Font to construct a robust content feature by projecting the content feature into a linear space defined by the content features of basis fonts. Besides, NTF [9] models font generation as a continuous transformation process with a neural transformation field.

2.3. Diffusion Probabilistic Models

Diffusion model (DM) provides a powerful probabilistic generative framework for high-quality image synthesis, which is first introduced in [33] and subsequently improved by Denoising Diffusion Probabilistic Model (DDPM) [13], Denoising Diffusion Implicit Model (DDIM) [34], score-based diffusion [35], and other techniques [25, 36, 37]. Early diffusion models construct the diffusion process in RGB space, which requires huge computation powers to generate high-resolution images. Latent Diffusion Model (LDM) [30] builds the diffusion process in a lower-dimensional latent space, significantly reducing the computation cost. The diffusion model has become the dominant solution for various computer vision tasks, such as image synthesis [29, 30, 32], image editing [1, 6, 24], and image-to-image translation [31, 44]. Large-scale text-to-image (T2I) diffusion models (e.g. DALL·E 2 [29], Imagen [32], Stable Diffusion [30]) have shown superior performance on high-quality image synthesis, which employ text embedding generated by large language models (LLM) as the condition to guide image generation. Recently, several works [19, 22, 52] modify the denoising diffusion process to model the residual between two given distributions. Different from these works, in this paper, we introduce the font transfer process into the diffusion model at a predefined region, forming a multi-stage diffusion process. Moreover, based on this process, a novel dual-network approach is proposed to predict latent features in different stages.

3. Methodology

In this section, we present our proposed method in detail, including a short background of the diffusion model (in Sec. 3.1), the proposed latent diffusion model with font transfer process (in Sec. 3.2), the multi-stage font generative process (in Sec. 3.3), and the overall framework (in Sec. 3.4).

3.1. Background and Preliminary

The diffusion model is a probabilistic generative model aiming to learn the data distribution by gradually denoising a variable sampled from a Gaussian distribution, which includes a forward process and a reverse process.

Given a data distribution $x_0 \sim q(x_0)$, **the Forward Process** (also named as the diffusion process) is fixed to a Markov chain that gradually injects Gaussian noise to the data at time t with a predefined variance schedule β_t according to the equations:

$$q(x_1, \ldots, x_T | x_0) = \prod_{t=1}^{T} q(x_t | x_{t-1}), \tag{1}$$

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

where $x_1, \cdots, x_T$ are latent variables. With the sufficiently large T, the distribution of x_T will become an isotropic Gaussian distribution. With the reparameterization trick

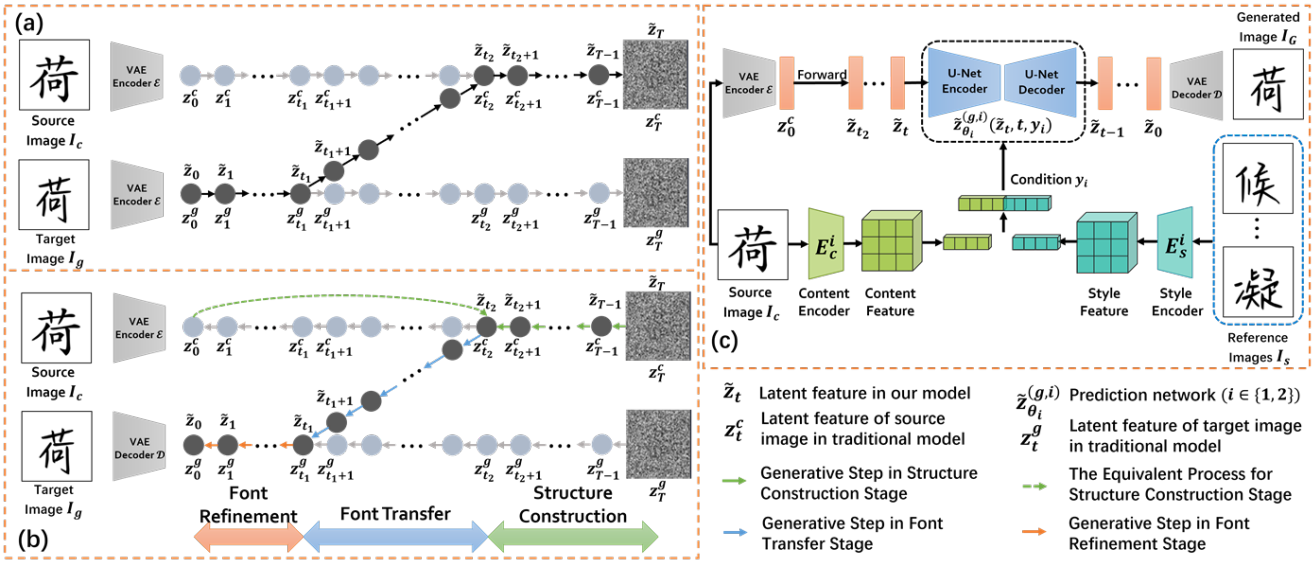

Figure 2. (a). The forward diffusion process of our proposed model. (b). The reverse process of our proposed model. Our proposed diffusion process is marked as black circles. The $\tilde{z}_t$ denotes the latent features in our diffusion process, while z_t^c and z_t^g denote the latent features of the source image and the target image in traditional diffusion process, respectively. As we have aligned our diffusion process with the traditional diffusion process of z^g and z^c at t_1 and t_2, our model shares the same process with the traditional diffusion process in the region $(0, t_1]$ and $(t_2, T]$. (c). Overall framework of our font generation model, including the VQ-VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$, the dual prediction networks $\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_t, t, y_1)$ and $\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_t, t, y_2)$, and the corresponding style and content encoders for each prediction network. Due to the same structure, in this figure, we only plot one prediction network $\tilde{z}_{\theta_i}^{(g,i)}$ ($i \in \{1, 2\}$) with its corresponding style encoder E_s^i and content encoder E_c^i, where i is determined by the generative stage of timestep t. We use I_g and I_G to denote the target image and generated image, respectively.

[16], the latent variable x_t can be written as

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\epsilon_{t-1}^*, \quad (3)$$

$$= \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_0, \quad (4)$$

where $\epsilon_{t-1}^*, \epsilon_0 \sim \mathcal{N}(0, I)$, $\alpha_t = 1-\beta_t$, and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. Thus sampling x_t at arbitrary timestep has a closed form $q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$.

The Reverse Process generates data from isotropic Gaussian noise x_T, and gradually recovers x_0 via the reverse distribution $p_\theta(x_{t-1}|x_t)$:

$$p_\theta(x_0) = \int p_\theta(x_{0:T}) dx_{1:T}, \quad (5)$$

$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \quad (6)$$

where $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}|\mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$ is a parameterized Gaussian transition network, and defined as

$$\mu_\theta(x_t, t) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\hat{x}_\theta(x_t, t)}{1-\bar{\alpha}_t},$$

and $\Sigma_\theta(x_t, t) = (1-\bar{\alpha}_{t-1})\beta_t I/(1-\bar{\alpha}_t)$. $\hat{x}_\theta(x_t, t)$ is a function approximator for predicting x_0 from x_t and t. The corresponding objective function can be simplified to:

$$L = \mathbb{E}_{x_0,\epsilon,t}\left[\|\hat{x}_\theta(x_t, t) - x_0\|_2^2\right]. \quad (7)$$

Latent Diffusion Model (LDM) [30] further employs a VQ-VAE [42] encoder $\mathcal{E}$ to project image x_0 into latent space z_0, and constructs the diffusion model in this lower-dimensional latent space, significantly reducing computational consumption. Once z_0 is sampled via the reverse process, the VQ-VAE decoder $\mathcal{D}$ will project it back to the image space.

Since LDM significantly reduces computation consumption and maintains high-quality image synthesis, in this paper, we construct our diffusion process in latent space.

3.2. Latent Diffusion Model with Font Transfer

As shown in Eq. (4), in the forward process, the data distribution x_0 and the injected noise can be well-separated in arbitrary timesteps. This characteristic enables us to construct a multi-stage font generative framework in latent diffusion model, by incorporating the font transfer process into the data sub-space in a predefined time interval $t \in (t_1, t_2]$. Based on this observation, we develop our latent diffusion model in this section, then discuss the multi-stage font generative process in the next section.

The Forward Process. After projecting the source images I_c and target images I_g into the latent features z_0^c and z_0^g by the VQ-VAE encoder $\mathcal{E}$, we implement our model in latent space. As shown in Fig. 2(a), our proposed model (marked as black circles and denoted as $\tilde{z}_t$) is constructed

by utilizing a font transfer process to connect the traditional diffusion processes (DDPM) of the latent features z_t^c and z_t^g. Therefore, our latent diffusion process $\tilde{z}_t$ can be divided into three regions: (1). In the region $t \in (0, t_1]$, only latent feature z_t^g evolved with noise. The forward latent $\tilde{z}_t$ in this region can be expressed as

$$\tilde{z}_t = z_t^g = \sqrt{\bar{\alpha}_t} z_0^g + \sqrt{1-\bar{\alpha}_t}\epsilon_0 \quad t \in (0, t_1]\,. \tag{8}$$

(2). In the region $t \in (t_1, t_2]$, together with the continuously injected noise, the latent feature z_t^g of the target font is gradually replaced by the latent feature z_t^c of the source character, and only z_t^c exists at t_2. After careful derivation (provided in Supplementary Materials), the forward latent $\tilde{z}_t$ in this region can be expressed as

$$\tilde{z}_t = \sqrt{\bar{\alpha}_t} z_0^g + \sqrt{1-\bar{\alpha}_t}\epsilon_0 - (z_0^g - z_0^c)\,\Psi\,\mathcal{H}_t, \tag{9}$$

for $t \in (t_1, t_2]$. The notations $\mathcal{H}_t$ and Ψ are coefficients obtained by aligning our model with traditional diffusion processes (DDPM) of z^g and z^c at timesteps t_1 and t_2, respectively. They are defined as

$$\mathcal{H}_t = 1 + \sqrt{\alpha_t} + \sqrt{\alpha_t \alpha_{t-1}} + \cdots + \sqrt{\alpha_t \cdots \alpha_{t_1+2}}\,, \tag{10}$$

$$\Psi = \frac{\sqrt{\tilde{\alpha}(1, t_2)}}{1 + \sum_{m=t_1+2}^{t_2} \sqrt{\tilde{\alpha}(m, t_2)}}\,, \tag{11}$$

where $\tilde{\alpha}(t_1, t_2) = \prod_{i=t_1}^{t_2} \alpha_i$. $\mathcal{H}_t$ can be further reduced to $\mathcal{H}_t = 1 + \sqrt{\alpha_t}\mathcal{H}_{t-1}$. (3). In the region $t \in (t_2, T]$, the injected noise finally disrupts the latent feature z_t^c to Gaussian noise, and the forward latent $\tilde{z}_t$ in this region can be expressed as

$$\tilde{z}_t = z_t^c = \sqrt{\bar{\alpha}_t} z_0^c + \sqrt{1-\bar{\alpha}_t}\epsilon_0 \quad t \in (t_2, T]\,. \tag{12}$$

The Reverse Process. Now we turn to discuss our choices for $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t) = \mathcal{N}(\tilde{z}_{t-1}|\mu_\theta(\tilde{z}_t, t), \Sigma_\theta(\tilde{z}_t, t))$. As shown in Eq. (8) and Eq. (12), in the time region $(0, t_1]$ and $(t_2, T]$, we have aligned the forward latent $\tilde{z}_t$ with the traditional diffusion processes (DDPM [13]), thus they share the same parameterized mean functions: for $t \in (0, t_1]$

$$\mu_\theta = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\tilde{z}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\tilde{z}_\theta^{(g)}(\tilde{z}_t, t)}{1-\bar{\alpha}_t}\,, \tag{13}$$

where $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t)$ is the prediction network for predicting z_0^g from noisy latent feature $\tilde{z}_t$ and time index t, while for $t \in (t_2, T]$,

$$\mu_\theta = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\tilde{z}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\tilde{z}_\theta^{(c)}(\tilde{z}_t, t)}{1-\bar{\alpha}_t}\,, \tag{14}$$

where $\tilde{z}_\theta^{(c)}(z_t, t)$ is the prediction network for predicting z_0^c from noisy latent feature $\tilde{z}_t$ and time index t. The parameterized mean function for the region $t \in (t_1, t_2]$ can be derivated via Bayes' theorem:

$$\begin{aligned}\mu_\theta \;=\; & \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\tilde{z}_t + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\Psi\tilde{z}_\theta^{(g)}(\tilde{z}_t, t) \\ & + \frac{\left[\mathcal{H}_{t-1}(1-\alpha_t) - \sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\right]}{1-\bar{\alpha}_t}\Psi z_0^c \\ & + \frac{(\sqrt{\bar{\alpha}_{t-1}} - \mathcal{H}_{t-1}\Psi)(1-\alpha_t)}{1-\bar{\alpha}_t}\tilde{z}_\theta^{(g)}(\tilde{z}_t, t).\end{aligned} \tag{15}$$

As we only incorporate the font transfer process in the data sub-space and keep the noise sub-space unchanged, the variance in the reverse process remains the same as DDPM:

$$\Sigma_\theta(\tilde{z}_t, t) = \sigma_t^2 I = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} I. \tag{16}$$

Therefore, the latent $\tilde{z}_{t-1}$ in reverse process can be obtained via the formula $p_\theta(\tilde{z}_{t-1}|\tilde{z}_t) = \mathcal{N}(\tilde{z}_{t-1}; \mu_\theta(\tilde{z}_t, t), \sigma_t^2 I)$, where the $\mu_\theta(\tilde{z}_t, t)$ is calculated from Eq. (13)-(15) based on different timesteps.

The Optimization Objective. As the forward behaviors keep the same as DDPM in the region $(0, t_1]$ and $(t_2, T]$, the optimization objectives of $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t)$ and $\tilde{z}_\theta^{(c)}(\tilde{z}_t, t)$ have the same expressions as DDPM in these regions. For the optimization objective of $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t)$ in the region $(t_1, t_2]$, it can be derived from the variational bound on negative log-likelihood [13], and only the coefficient is different with the DDPM's loss function. Moreover, as DDPM and LDM adopt the simplified training objectives in their implementations, we also simplify our loss functions and obtain the optimization objective as

$$L = L_g + L_c, \tag{17}$$

$$L_g = \mathbb{E}_{\mathcal{E}(x_0), \epsilon, t \sim U(0, t_2)}\left[\|\tilde{z}_\theta^{(g)}(\tilde{z}_t, t) - z_0^g\|_2^2\right], \tag{18}$$

$$L_c = \mathbb{E}_{\mathcal{E}(x_0), \epsilon, t \sim U(t_2, T]}\left[\|\tilde{z}_\theta^{(c)}(\tilde{z}_t, t) - z_0^c\|_2^2\right], \tag{19}$$

where $U(t_1, t_2]$ denotes the uniform distribution in region $(t_1, t_2]$.

The Conditioning Mechanism. Following the LDM [30], we generalize our method to model conditional distributions by implementing with the conditional prediction networks:

$$\tilde{z}_\theta^{(i)}(\tilde{z}_t, t, y), \;\; i \in \{g, c\}, \tag{20}$$

where y is the condition for controlling the synthesis process.

Extension to Conditional Font Generation. We further extend this conditional model to perform the font generation task. Since the few-shot font generation task aims to render the source character I_c into the target font style by a few reference samples I_s, we utilize the style encoder E_s and the content encoder E_c to extract the style code from the reference images I_s and the content representations from the source images I_c. The style and content features are flattened and concatenated to build condition y.

3.3. Multi-Stage Font Generative Process

Based on our proposed model, the font generation in the reverse process can be automatically separated into three different stages: the structure construction stage, the font transfer stage, and the font refinement stage.

As shown in Fig. 2(b), starting with the sampled Gaussian noise $\tilde{z}_T$, in the region $t \in (t_2, T]$, **Structure Construction Stage** generates the structure information for the target character via predicting the latent feature z_0^c of the source image I_c. However, since we can access the source image I_c during the inference, the iterative generative steps from T to t_2 can be replaced by a single forward step to generate the noised latent variable $\tilde{z}_{t_2}$ at time $t = t_2$ via Eq. (12). This will significantly reduce inference time and provide more accurate structure information. **Font Transfer Stage** performs font transformation from the timestep t_2 to t_1, where the noised latent feature $\tilde{z}_{t_2}$ is gradually transformed to the noised latent feature $\tilde{z}_{t_1}$ of target images via the $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t, y)$. To predict the latent feature z_0^g, the prediction network $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t, y)$ in this stage is optimized to not only denoise the latent features but also perform font transfer under the condition y. Finally, in the region $t \in (0, t_1]$, **Font Refinement Stage** improves appearances and local details for the generated font images via predicting z_0^g from the noisy latent feature $\tilde{z}_t$. Based on the above three stages, we can implement the reverse process to generate high-quality font images.

To better present this multi-stage font generative process, we summarize above steps and provide the pseudo-code in Supplementary Materials.

3.4. Overall Framework for Font Generation

As discussed in Sec. 3.3, the network $\tilde{z}_\theta^{(c)}(\tilde{z}_t, t, y)$ can be replaced by a single forward step. Moreover, instead of implementing a single prediction network $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t, y)$ to predict z_0^g, we adopt a dual-network approach in this paper, since the network $\tilde{z}_\theta^{(g)}(\tilde{z}_t, t, y)$ has different behaviors in the font transfer stage and the font refinement stage. The dual networks $\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_t, t, y_1)$ and $\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_t, t, y_2)$ share the same structure, but are trained to predict the latent feature z_0^g in different regions: one for font transfer stage and the other for font refinement stage. Therefore, as shown in Fig. 2(c), the overall framework of our font generation model contains the following parts: the VQ-VAE encoder $\mathcal{E}$, the VQ-VAE decoder $\mathcal{D}$, the dual prediction networks $\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_t, t, y_1)$ and $\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_t, t, y_2)$, and the corresponding style and content encoders for each prediction network $\tilde{z}_{\theta_i}^{(g,i)}(\tilde{z}_t, t, y_i)$. Finally, from Eq. (17), the optimization objective of our font generative model can be further reduced to

$$L = L_g^1 + L_g^2, \tag{21}$$

$$L_g^1 = \mathbb{E}_{\mathcal{E}(x_0),\epsilon,t\sim U(t_1,t_2)}\left[\|\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_t, t, y_1) - z_0^g\|_2^2\right], \tag{22}$$

$$L_g^2 = \mathbb{E}_{\mathcal{E}(x_0),\epsilon,t\sim U(0,t_1]}\left[\|\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_t, t, y_2) - z_0^g\|_2^2\right]. \tag{23}$$

4. Experiments

In this section, we conduct extensive experiments to evaluate our proposed model on the FFG task.

4.1. Implement Details

Our models are implemented on the Stable Diffusion [30] platform. We utilize $T = 1000$ steps to generate $128 \times 128 \times 3$ font images with 8 reference samples. The VQ-VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$, the dual-network $\tilde{z}_{\theta_1}^{(g,1)}(\tilde{z}_t, t, y_1)$ and $\tilde{z}_{\theta_2}^{(g,2)}(\tilde{z}_t, t, y_2)$ are initialized by the pre-trained Stable Diffusion models. The style encoder E_s and content encoder E_c are initialized by Kaiming initialization. We keep VQ-VAE encoder $\mathcal{E}$ and decoder $\mathcal{D}$ frozen and optimize the remaining networks.

4.2. Dataset and Evaluation Metrics

Datasets: As DM needs more training data to optimize, we collect 900 Chinese fonts to evaluate our method on FFG task. We randomly select 840 fonts as the training set (seen fonts) and the remaining 60 fonts as the testing fonts (unseen fonts). As each collected font covers different Chinese characters, we randomly sample 800 characters from the commonly-used Chinese characters (the first-level Chinese Character Table) to train our method. For the test set, we use 214 characters as the unseen contents and sample 200 characters in our training set as the seen contents.

Therefore, the training set contains 840 fonts with 800 characters, while the test set includes two parts: one is the Unseen Fonts Unseen Contents (UFUC) test set, containing 60 unseen fonts with 214 unseen characters; the other is the Unseen Fonts Seen Contents (UFSC) test set, containing 60 unseen fonts with 200 seen characters.

Evaluation Metrics: In this paper, we utilize four commonly used measurements to evaluate our model, including root mean squared error (RMSE), peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image path similarity (LPIPS) [48].

4.3. Ablation Study

With the above dataset and metrics, we conducted extensive experiments in different settings.

4.3.1 Ablation Study on the Overall Framework

We first construct a Base Model with traditional diffusion process (DDPM) to verify the effectiveness of our proposed

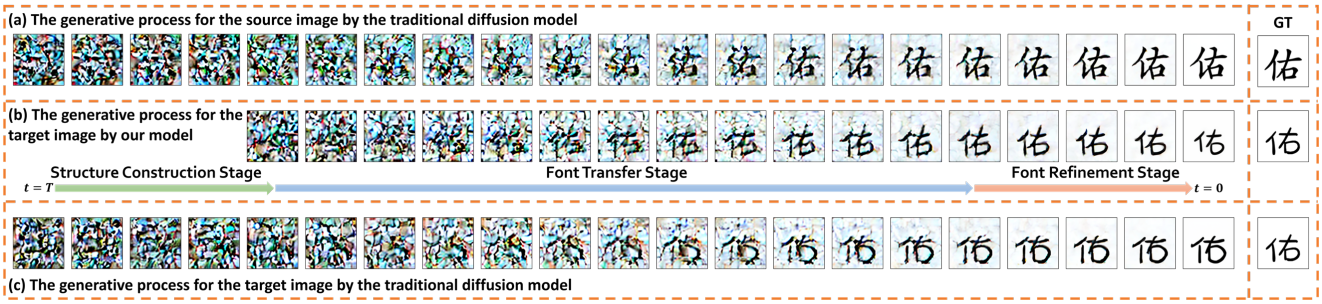

Figure 3. Visualization results of the intermediate results $\tilde{z}_t$ by projecting $\tilde{z}_t$ back to image space via VQ-VAE decoder $\mathcal{D}$.

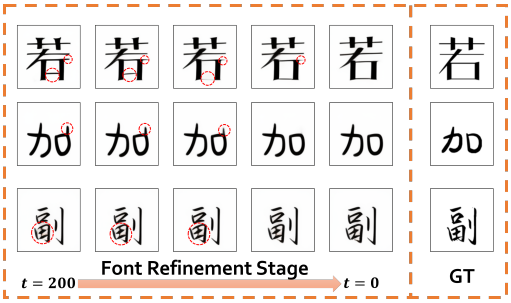

Figure 4. Visualization results of the predicted latent feature z_0^g in the intermediate timesteps by projecting z_0^g back to image space via VQ-VAE decoder $\mathcal{D}$.

Table 1. Ablation study for the overall framework on UFUC dataset. The bold number indicates the best.

Methods	RMSE↓	PSNR↑	SSIM↑	LPIPS↓
Base Model	0.2676	11.72	0.6961	0.1561
Single-Net	0.2601	12.02	0.7096	0.1531
Dual-Net	**0.2475**	**12.45**	**0.7189**	**0.1527**

latent diffusion model. Moreover, as discussed in Sec. 3.4, there are two different selections to obtain z_0^g in the reverse process: One utilizes a single network to predict z_0^g at all timesteps. The other adopts the dual networks to predict the latent feature z_0^g in font transfer stage and font refinement stage, respectively.

As shown in Tab. 1, compared with the Base Model, our Single-Net model improves the Base Model with 0.0075, 0.30, 0.0135 ,and 0.0030 in terms of RMSE, PSNR, SSIM, and LPIPS, verifying the effectiveness of our proposed latent diffusion model. Moreover, the Dual-Net approach further improves the Single-Net model with 0.0126, 0.43, and 0.0093 in terms of RMSE, PSNR, and SSIM, illustrating that the use of separate models is beneficial for predicting z_0^g in different stages.

4.3.2 Ablation Study on the Multi-Stage Font Generative Process

The selections of the region for different stages in the multi-stage generative process are hyper-parameters in our model, which significantly impact image quality. Therefore, we study the influence on model performance by selecting different t_1 and t_2. Due to the page limitation, the experimental results and the detailed discussions are provided in supplementary materials, and we briefly summarize the conclusion: The hyper-parameters t_1 and t_2 are trade-off selections, and the configuration $t_1 = 200$ and $t_2 = 800$ achieves the best record. Thus, we select this configuration for the following experiments.

4.3.3 Visualization Results

In this section, we provide the visualization results of our generative process to verify the effectiveness of our model.

As our diffusion process is implemented in latent space, we visualize the intermediate generative results $\tilde{z}_t$ by using the VQ-VAE decoder $\mathcal{D}$ to project them back into the image space. As shown in Fig. 3, the images in the first and third rows are the intermediate results for source image I_c and target image I_g generated by the traditional diffusion process (DDPM), while the images in the second row are generated by our multi-stage font generative process. From this figure, we can clearly observe a font transfer process from the source font to the target font, and the font refinement stage slightly improves style consistency by modifying local details.

To further evaluate the font refinement stage, we also visualize the predicted z_0^g from different timesteps in the font refinement stage. As shown in Fig. 4, from $t = 200$ to $t = 0$, the font refinement stage can gradually enhance local details by modifying anomalous pixels, recovering unwanted structures, and improving visual appearances.

4.4. Comparison with the State-of-the-art Methods

In this section, we compare our model with several state-of-the-art works, including LF-Font [27], DG-Font [45], MX-Font [28], FsFont [39], NTF [9], and Diff-Font [11]. LF-Font and MX-Font are component-related methods, which decompose characters into components and learn component-wise representations to improve local details. FsFont focuses on fine-grained local styles, which utilizes cross attention to aggregate the reference styles into target styles. DG-Font and NTF are shape transformation mod-

Figure 5. Qualitative comparisons of our proposed model with other state-of-the-art methods on UFUC dataset.

Table 2. Quantitative comparison of our proposed model with other state-of-the-art methods on few-shot font generation task. The bold number indicates the best.

Methods	RMSE↓	PSNR↑	SSIM↑	LPIPS↓
Unseen Fonts and **Seen** Contents				
LF-Font [27]	0.3025	10.66	0.6469	0.2787
DG-Font [45]	0.2833	11.24	0.6651	0.1479
MX-Font [28]	0.3007	10.75	0.6438	0.1651
FsFont [39]	0.2443	12.56	0.7105	0.1468
NTF [9]	0.2650	11.83	0.6896	0.1517
Diff-Font[2][11]	0.2811	10.95	0.6846	0.1557
MSD-Font	**0.2439**	**12.61**	**0.7110**	**0.1348**
Unseen Fonts and **Unseen** Contents				
LF-Font [27]	0.2910	11.09	0.6762	0.2564
DG-Font [45]	0.2750	11.54	0.6893	0.1534
MX-Font [28]	0.2845	11.29	0.6789	0.1635
FsFont [39]	0.2509	12.29	0.7185	0.1642
NTF [9]	0.2574	12.11	0.7108	0.1553
MSD-Font	**0.2475**	**12.45**	**0.7189**	**0.1527**

els. DG-Font models font generation as a shape deformable problem via the deformation skip connection, while NTF models font generation as a continuous transformation process. We train above methods on our dataset and use the same reference images to generate fonts in inference.

4.4.1 Quantitative Comparison

The quantitative results are shown in Tab. 2. From the table, we find that modeling font style in fine-grained manners by multiple localized encoders (MX-Font) or cross attention mechanism (FsFont) can improve the quality of generated font images. The shape transformation approaches (DG-Font and NTF) are also promising solutions for FFG task. Our MSD-Font achieves superior performance on both UFUC and UFSC datasets, which demonstrates the effectiveness of our proposed method. Specifically, compared with the NTF on the UFUC dataset, our method improves the generation performance with 0.0099, 0.34, 0.0081, and 0.0026 in terms of RMSE, PSNR, SSIM, and LPIPS.

[2]Diff-Font is one-shot font generation model, we randomly select one image from 8 reference samples to calculate style code.

4.4.2 Qualitative Comparison

To further evaluate our methods, as shown in Fig. 5, we visualize the generated font images in UFUC setting. LF-Font sometimes generates font images with some broken structures. The DG-Font keeps the global structure for the target character, but the generated images may contain some unwanted local structures and artifacts. FsFont utilizes several carefully selected reference characters to generate font images, thus sometimes it will fail to produce accurate structures for target images with randomly selected reference characters. The MX-Font and NTF also contain some missing strokes and distorted local details. Our MSD-Font is able to generate high-quality font images, which achieves higher visual quality than other state-of-the-art methods in terms of style consistency and structure completeness.

5. Conclusion

In this paper, we propose a multi-stage font generative process for few-shot font generation task, by incorporating font transfer process into the diffusion model. The proposed font generation model includes three stages: a structure construction stage to generate global structures from the source images, a font transfer stage to transform the source font into the target font, and a font refinement stage to improve the appearances and local details for the generated font images. Based on this multi-stage generative process, we construct our font generation framework, named MSD-Font, with a dual-network approach to generate high-quality font images using a few reference samples. The promising experimental results verify the effectiveness of our proposed model.

Limitations: As our MSD-Font based on diffusion models, there is an increase in the inference time and model size compared with previous GAN-based FFG methods. Many recent works have focused on improving the efficiency and model compactness of diffusion models, and most of these methods can be used in our model.

Acknowledgements: This work is supported by the National Key R&D Program of China (NO. 2022ZD0160102).

References

[1] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022. 3

[2] Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7564–7573, 2018. 3

[3] Junbum Cha, Sanghyuk Chun, Gayoung Lee, Bado Lee, Seonghyeon Kim, and Hwalsuk Lee. Few-shot compositional font generation with dual memory. In *European Conference on Computer Vision*, pages 735–751. Springer, 2020. 3

[4] Jie Chang, Yujun Gu, Ya Zhang, Yan-Feng Wang, and CM Innovation. Chinese handwriting imitation with hierarchical generative adversarial network. In *BMVC*, page 290, 2018. 2, 3

[5] Xinyuan Chen, Yangchen Xie, Li Sun, and Yue Lu. Dg-font++: Robust deformable generative networks for unsupervised font generation. *arXiv preprint arXiv:2212.14742*, 2022. 3

[6] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. *arXiv preprint arXiv:2108.02938*, 2021. 3

[7] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11472–11481, 2022. 2

[8] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8789–8797, 2018. 2

[9] Bin Fu, Junjun He, Jianjun Wang, and Yu Qiao. Neural transformation fields for arbitrary-styled font generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22438–22447, 2023. 3, 7, 8

[10] Yue Gao, Yuan Guo, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. Artistic glyph image synthesis via one-stage few-shot learning. *ACM Transactions on Graphics (TOG)*, 38(6): 1–12, 2019. 3

[11] Haibin He, Xinyuan Chen, Chaoyue Wang, Juhua Liu, Bo Du, Dacheng Tao, and Yu Qiao. Diff-font: Diffusion model for robust one-shot font generation. *arXiv preprint arXiv:2212.05895*, 2022. 3, 7, 8

[12] Xiao He, Mingrui Zhu, Nannan Wang, Xinbo Gao, and Heng Yang. Few-shot font generation by learning style difference and similarity. *arXiv preprint arXiv:2301.10008*, 2023. 3

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 3, 5

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[15] Yue Jiang, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. Dcfont: an end-to-end deep chinese font generation system. In *SIGGRAPH Asia 2017 Technical Briefs*, pages 1–4. 2017. 3

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 4

[17] Yuxin Kong, Canjie Luo, Weihong Ma, Qiyuan Zhu, Shenggao Zhu, Nicholas Yuan, and Lianwen Jin. Look closer to supervise better: One-shot font generation via component-based discriminator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13482–13491, 2022. 1, 3

[18] Mingi Kwon, Jaeseok Jeong, and Youngjung Uh. Diffusion models already have a semantic latent space. *arXiv preprint arXiv:2210.10960*, 2022. 2

[19] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. I^2sb: Image-to-image schrödinger bridge. *arXiv preprint arXiv:2302.05872*, 2023. 3

[20] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10551–10560, 2019. 2

[21] Wei Liu, Fangyue Liu, Fei Ding, Qian He, and Zili Yi. Xmp-font: Self-supervised cross-modality pre-training for few-shot font generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7905–7914, 2022. 3

[22] Ziwei Luo, Fredrik K Gustafsson, Zheng Zhao, Jens Sjölund, and Thomas B Schön. Image restoration with mean-reverting stochastic differential equations. *arXiv preprint arXiv:2301.11699*, 2023. 3

[23] Pengyuan Lyu, Xiang Bai, Cong Yao, Zhen Zhu, Tengteng Huang, and Wenyu Liu. Auto-encoder guided gan for chinese calligraphy synthesis. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, pages 1095–1100. IEEE, 2017. 3

[24] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 3

[25] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 3

[26] Wei Pan, Anna Zhu, Xinyu Zhou, Brian Kenji Iwana, and Shilin Li. Few shot font generation via transferring similarity guided global style and quantization local style. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19506–19516, 2023. 3

[27] Song Park, Sanghyuk Chun, Junbum Cha, Bado Lee, and Hyunjung Shim. Few-shot font generation with localized

style representations and factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2393–2402, 2021. 1, 3, 7, 8

[28] Song Park, Sanghyuk Chun, Junbum Cha, Bado Lee, and Hyunjung Shim. Multiple heads are better than one: Few-shot font generation with multiple localized experts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13900–13909, 2021. 1, 3, 7, 8

[29] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1 (2):3, 2022. 3

[30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 3, 4, 5, 6

[31] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022. 3

[32] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022. 3

[33] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 3

[34] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 3

[35] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019. 3

[36] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in neural information processing systems*, 33:12438–12448, 2020. 3

[37] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 3

[38] Danyang Sun, Tongzheng Ren, Chongxuan Li, Hang Su, and Jun Zhu. Learning to write stylized chinese characters by reading a handful of examples. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 920–927, 2018. 3

[39] Licheng Tang, Yiyang Cai, Jiaming Liu, Zhibin Hong, Mingming Gong, Minhu Fan, Junyu Han, Jingtuo Liu, Errui Ding, and Jingdong Wang. Few-shot font generation by learning fine-grained local styles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7895–7904, 2022. 1, 3, 7, 8

[40] Yuchen Tian. zi2zi: Master chinese calligraphy with conditional adversarial networks. 2

[41] Yuchen Tian. Rewrite: Neural style transfer for chinese fonts. 2016. 2

[42] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 4

[43] Chi Wang, Min Zhou, Tiezheng Ge, Yuning Jiang, Hujun Bao, and Weiwei Xu. Cf-font: Content fusion for few-shot font generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1858–1867, 2023. 3

[44] Tengfei Wang, Ting Zhang, Bo Zhang, Hao Ouyang, Dong Chen, Qifeng Chen, and Fang Wen. Pretraining is all you need for image-to-image translation. *arXiv preprint arXiv:2205.12952*, 2022. 3

[45] Yangchen Xie, Xinyuan Chen, Li Sun, and Yue Lu. Dg-font: Deformable generative networks for unsupervised font generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5130–5140, 2021. 3, 7, 8

[46] Zhenhua Yang, Dezhi Peng, Yuxin Kong, Yuyi Zhang, Cong Yao, and Lianwen Jin. Fontdiffuser: One-shot font generation via denoising diffusion with multi-scale content aggregation and style contrastive learning. In *Proceedings of the AAAI conference on artificial intelligence*, 2024.

[47] Mingshuai Yao, Yabo Zhang, Xianhui Lin, Xiaoming Li, and Wangmeng Zuo. Vq-font: Few-shot font generation with structure-aware enhancement and quantization. *arXiv preprint arXiv:2308.14018*, 2023. 3

[48] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 6

[49] Yexun Zhang, Ya Zhang, and Wenbin Cai. Separating style and content for generalized style transfer. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8447–8455, 2018. 3

[50] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017. 2

[51] Mingrui Zhu, Xiao He, Nannan Wang, Xiaoyu Wang, and Xinbo Gao. All-to-key attention for arbitrary style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23109–23119, 2023. 2

[52] Jianyi Wang Zongsheng Yue and Chen Change Loy. Resshift: Efficient diffusion model for image super-resolution by residual shifting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. 3