VQ-Font: Few-Shot Font Generation with Structure-Aware Enhancement and Quantization

Mingshuai Yao[1], Yabo Zhang[1], Xianhui Lin[2] , Xiaoming Li[1]*, Wangmeng Zuo[1,3]

[1] Harbin Institute of Technology
[2] Institute for Intelligent Computing
[3] Peng Cheng Laboratory
{ymsoyosmy,hitzhangyabo2017,xhlin129,csxmli}@gmail.com, wmzuo@hit.edu.cn

Abstract

Few-shot font generation is challenging, as it needs to capture the fine-grained stroke styles from a limited set of reference glyphs, and then transfer to other characters, which are expected to have similar styles. However, due to the diversity and complexity of Chinese font styles, the synthesized glyphs of existing methods usually exhibit visible artifacts, such as missing details and distorted strokes. In this paper, we propose a VQGAN-based framework (*i.e.*, VQ-Font) to enhance glyph fidelity through token prior refinement and structure-aware enhancement. Specifically, we pre-train a VQGAN to encapsulate font token prior within a codebook. Subsequently, VQ-Font refines the synthesized glyphs with the codebook to eliminate the domain gap between synthesized and real-world strokes. Furthermore, our VQ-Font leverages the inherent design of Chinese characters, where structure components such as radicals and character components are combined in specific arrangements, to recalibrate fine-grained styles based on references. This process improves the matching and fusion of styles at the structure level. Both modules collaborate to enhance the fidelity of the generated fonts. Experiments on a collected font dataset show that our VQ-Font outperforms the competing methods both quantitatively and qualitatively, especially in generating challenging styles. Code is available at https://github.com/Yaomingshuai/VQ-Font.

Introduction

Font library elegantly represents text information in computer systems and has tremendous value in commercial and artistic applications (Liu et al. 2016; Liu, Chen, and Wong 2018). Manually designing such a library is highly resource-intensive and laborious, especially for logographic languages containing thousands of characters (*e.g.*, Chinese, Japanese, and Korean). However, each glyph is typically constructed using fundamental strokes, thereby making it feasible to create a new glyph by directly adopting styles from other reference glyphs at different levels of granularity, *e.g.,* structure and stroke.

Among these recent methods, few-shot font generation attracts significant attention, as it is effective in reducing the human labor required for designing a target font library.

*Corresponding Author

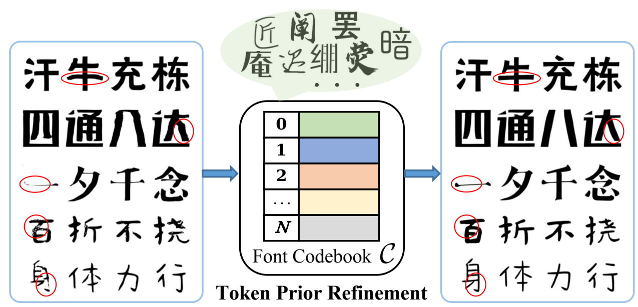

Figure 1: Left: the illustration of missing details and distorted strokes. Right: the refinement process through token prior encapsulated in our codebook.

Given limited target glyphs as style references, they usually translate a content glyph to a target one by adaptively querying fine-grained styles from references. FS-Font (Tang et al. 2022), one of the most representative methods, proposes a style aggregation module based on a cross-attention mechanism. It aggregates patch-grained styles to obtain target glyphs. However, it's noteworthy that the Chinese comprise over 3,000 characters, and subtle variations in strokes can give rise to completely different characters. Besides, nearly all the existing methods learn the representations of characters from scratch, inevitably leading to issues of missing or distorted strokes (see the left part in Fig. 1). Their performance encounters further degradation when handling complex styles, *e.g.*, serif and artistic fonts.

Furthermore, FS-Font takes the patch tokens from content glyphs as queries while those from reference glyphs act as keys and values. This approach focuses on the patch-level correspondence between content and reference glyphs, which neglects the inherent principles of character design. As an ideographic writing system, Chinese characters carry distinct structure information. As shown in Fig. 2, nearly all Chinese characters can be divided into 12 structures, such as top-bottom arrangement and left-right arrangement. Among them, most structures contain two or three components. According to the structure division of Chinese characters, when the radical or character component of the content glyphs appears in the reference glyphs, we aim to treat these structure components as unified entities rather than focusing only on patch-level correspondence.

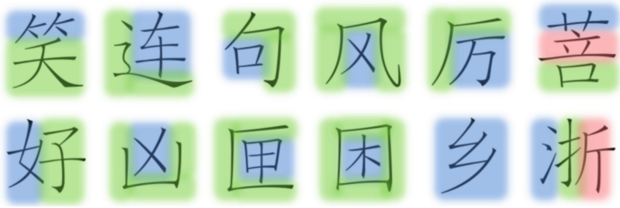

Figure 2: Structure component divisions in Chinese characters. Different colors represent different components.

To improve the fidelity of synthesized glyphs, we propose VQ-Font, a framework encompassing the structure-aware enhancement and token prior refinement. To be specific, we firstly pre-train a VQGAN model (Esser, Rombach, and Ommer 2021) on diverse and high-quality font images. This VQGAN model has the ability to generate font images aligning well with the real-world manifold. Then, we employ a Transformer (Vaswani et al. 2017) to globally model the synthesized font images and predict their corresponding indices within our pre-trained codebook, which can refine the font images by mapping into the token prior space. In addition, we propose a Structure-level Style Enhancement Module (SSEM) to explicitly incorporate Chinese character structure information. By establishing a correspondence between the structure components of the content and reference glyphs, it recalibrates the fine-grained styles derived from the references, thereby facilitating the accurate learning and matching process of glyph style transformation.

In summary, our work has three main contributions:

- We introduce a font codebook that encapsulates token prior to refine synthesized font images. By mapping the synthesized font into the discrete space defined by the codebook, our VQ-Font can effectively address the issues of missing details and distorted strokes.
- We explicitly incorporate the design criterion of Chinese characters by introducing structure-level correspondence. This promotes the model to better learn the styles of the reference glyphs at the structure level.
- Our VQ-Font outperforms these competing methods in both quantitative and qualitative evaluation. It is also capable of generating complex styles with better fidelity.

Related Work

Many-shot Font Generation

Early methods (Tian 2016, 2017; Jiang et al. 2017; Lyu et al. 2017; Chang et al. 2018; Sun, Zhang, and Yang 2018; Jiang et al. 2019; Yang et al. 2019a,b; Gao and Wu 2020; Wu, Yang, and Hsu 2020; Wen et al. 2021; Hassan, Ahmed, and Choi 2021) utilize Image-to-Image translation networks (Zhang et al. 2022) to achieve font generation by learning the mapping function between different fonts. Tian *et al*. presents zi2zi (Tian 2017) which modifies pixel2pixel (Isola et al. 2017) to make it suitable for font generation. AGEN (Lyu et al. 2017) proposes a model for synthesizing Chinese calligraphy images with specified style from standard font images. HGAN (Chang et al. 2018) proposes a Hierarchical Generative Adversarial Network consisting of a transfer network and hierarchical adversarial discriminator based on zi2zi. PEGAN (Sun, Zhang, and Yang 2018) employs cascaded refinement connections and mirror skip connections to embed a multiscale pyramid of downsampled input into the encoder feature maps. However, these methods can only transfer glyphs from one known domain to another one that has appeared in the training process, making them incapable of generalizing to new fonts.

Few-shot Font Generation

In comparison, few-shot font generation is more flexible, as it can obtain a new font library by utilizing a few reference glyphs. Current methods (Sun et al. 2017; Zhang, Zhang, and Cai 2018; Gao et al. 2019; Cha et al. 2020; Park et al. 2021a,b; Xie et al. 2021; Liu et al. 2022; Chen, Wang, and Liu 2022; Wang et al. 2023) disentangle font images into content features and style features to achieve few-shot font generation. SA-VAE (Sun et al. 2017), EMD (Zhang, Zhang, and Cai 2018), and AGISNet (Gao et al. 2019) learn global feature representation on font images, but they neglect the design criterion of characters, thereby easily resulting in local details missing. DM-Font (Cha et al. 2020), LF-Font (Park et al. 2021a) and MX-Font (Park et al. 2021b) explicitly decompose characters into components and learn component-wise feature representation to facilitate the learning of local details. XMP-Font (Liu et al. 2022) proposes a self-supervised cross-modality pre-training strategy and a cross-modality transformer-based encoder to model style representation of all scales. DG-Font (Xie et al. 2021) introduces a feature deformation skip connection to predict displacement maps from the content glyph to the target glyph and its improved version, CF-Font (Wang et al. 2023), expands the variety of content fonts and fuses multiple content features by CFM module. Besides, NTF (Fu et al. 2023) achieves few-shot font generation by modeling it as a continuous transformation process using a neural transformation field. Nevertheless, regardless of whether using global or component-wise disentanglement, an average operation is usually performed on the extracted features, which easily weakens the local information and results in the loss of fine-grained details. FS-Font (Tang et al. 2022) begins to avoid explicitly disentangling content and style features of font images. It utilizes a cross-attention mechanism to match the patch-level correspondence between content and reference glyphs, and then aggregate fine-grained styles for font generation. In this work, we mainly follow the settings of FS-Font and attempt to address the issues of missing details and distorted strokes.

Codebook for Encapsulating Token Prior

VQVAE (Van Den Oord, Vinyals et al. 2017) is an extension of the autoencoder (Hinton and Zemel 1993) that introduces the vector-quantized codebook for the first time. By converting the continuous features into discrete features within a limited space, it resolves the issue of "posterior collapse" in the autoencoder architecture. To achieve better self-reconstruction results, VQVAE2 (Razavi, Van den

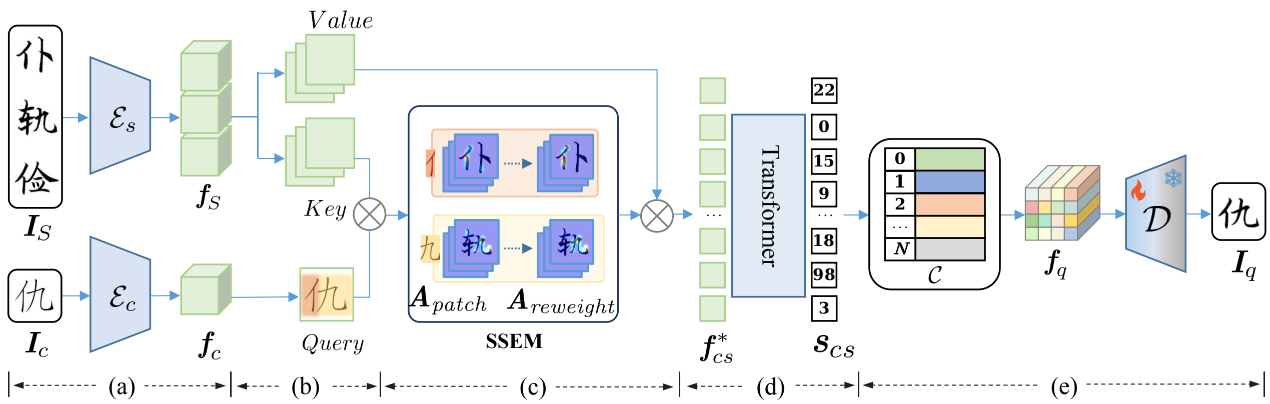

Figure 3: Overview of VQ-Font framework. VQ-Font mainly consists of the following components: a style encoder $\mathcal{E}_s$, a content encoder $\mathcal{E}_c$, a cross-attention module, a structure-level style enhancement module, a Transformer module, a pre-trained font codebook $\mathcal{C}$ and font decoder $\mathcal{D}$. (a) $\mathcal{E}_s$ and $\mathcal{E}_c$ first map $\boldsymbol{I}_S$ and $\boldsymbol{I}_c$ into style features $\boldsymbol{f}_S$ and content feature $\boldsymbol{f}_c$, respectively. (b) Then the cross-attention module is used to learn patch-level attention $\boldsymbol{A}_{patch}$ between the patches of content and references. (c) The structure-level style enhancement module further utilizes the structure information of Chinese characters to reweight $\boldsymbol{A}_{patch}$, in order to better learn the structure-level styles. (d) The Transformer module models the font images and predicts the target glyph indices $\boldsymbol{s}_{cs}$. (e) The obtained codebook indices $\boldsymbol{s}_{cs}$ is used to retrieve quantized token vectors from the font codebook, which is subsequently taken into the font decoder to generate the final image.

Oord, and Vinyals 2019) introduces a multi-scale codebook. VQGAN (Esser, Rombach, and Ommer 2021) enhances generation capabilities and further compresses the codebook size by introducing adversarial loss and perceptual loss. During the training phase, the codebook is continuously updated, thus encapsulating rich priors. Currently, the codebook has been used for many image restoration tasks to recover image details. RIDCP (Wu et al. 2023) utilizes a codebook that stores scene graphs without fog/rain information to achieve image dehazing, FeMaSR (Chen et al. 2022) utilizes a high-resolution prior codebook to achieve image super-resolution, Codeformer (Zhou et al. 2022) and VQFR (Gu et al. 2022) utilize a codebook that stores high-quality facial textures for blind face restoration. MARCONet (Li, Zuo, and Loy 2023) combines the codebook and StyleGAN (Karras et al. 2020) to generate the specific characters for providing detailed reference in text image super-resolution tasks. With the aid of the codebook which possesses rich priors, degraded images can be well restored to photo-realistic results. Currently, few-shot font generation tasks also suffer from stroke distortion, detail loss, and other related issues. Inspired by the benefits brought by VQGAN on the aforementioned methods, we introduce a font codebook with rich stroke priors for the first time. By using Transformer (Vaswani et al. 2017) to match corresponding code indices, the re-arranged tokens from the codebook can generate font images with clear strokes and realistic details.

Method

Few-shot font generation transfers a content glyph $\boldsymbol{I}_c$ to a new style described by several glyphs $\boldsymbol{I}_S = \{I_s^i\}_{i=1}^k$. It requires ensuring both the quality of synthesized glyphs and the fidelity of captured styles. In this section, we introduce a VQGAN-based framework (*i.e.*, VQ-Font) to improve them

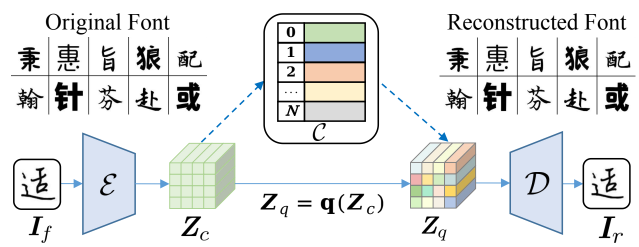

Figure 4: The details of our font VQGAN. The encoder $\mathcal{E}$ first maps the font image $\boldsymbol{I}_f$ into continuous feature space $\boldsymbol{Z}_c$. Then, $\boldsymbol{Z}_c$ is quantized into discrete feature $\boldsymbol{Z}_q$ with font codebook $\mathcal{C}$. Finally, the decoder maps $\boldsymbol{Z}_q$ back into the image space to generate result $\boldsymbol{I}_r$. Notably, the *out-of-domain* font can be also well reconstructed using the font VQGAN.

through token prior refinement and structure-level style enhancement. Firstly, VQ-Font encapsulates stroke priors by VQGAN-based self-reconstruction and then refines the synthesized strokes with the encapsulated priors. Secondly, it enhances the fine-grained styles using the inherent structure of Chinese characters. We present the details of our VQ-Font in the following sections.

Self-Reconstruction for Font Token Prior

To learn the token prior and incorporate it into the synthesized fonts, we pre-train a VQGAN by self-reconstructing font images with diverse styles and high quality. As shown in Fig. 4, VQGAN consists of an encoder $\mathcal{E}$, a learnable codebook $\mathcal{C} = \{c_k \in \mathbb{R}^d\}_{k=1}^K$, and a decoder $\mathcal{D}$. During self-reconstruction, $\mathcal{E}$ encodes a glyph image $\boldsymbol{I}_f \in \mathbb{R}^{H\times W\times 1}$ into continuous features $\boldsymbol{Z}_c \in \mathbb{R}^{h\times w\times d}$. Then, an element-wise quantization $\mathbf{q}(\cdot)$ is performed to replace each code in $\boldsymbol{Z}_c$ with its closest entry in codebook $\mathcal{C}$:

$$\boldsymbol{Z}_q = \mathbf{q}(\boldsymbol{Z}_c) := \left(\underset{c_k \in \mathcal{C}}{\arg\min} \left\| \boldsymbol{Z}_c^{(i,j)} - c_k \right\| \right) \in \mathbb{R}^{h \times w \times d} . \quad (1)$$

The final reconstruction result is obtained through:

$$\boldsymbol{I}_r = \mathcal{D}(\boldsymbol{Z}_q) = \mathcal{D}(\mathbf{q}(\mathcal{E}(\boldsymbol{I}_f))) \approx \boldsymbol{I}_f . \quad (2)$$

The indices sequence $s \in \{0, \ldots, |\mathcal{C}| - 1\}^{hw}$ of $\boldsymbol{Z}_c$ in codebook $\mathcal{C}$ is defined as:

$$s^{(i,j)} = k \quad \text{such that} \quad \boldsymbol{Z}_q^{(i,j)} = c_k . \quad (3)$$

During pre-training, the above three modules (encoder $\mathcal{E}$, codebook $\mathcal{C}$, and decoder $\mathcal{D}$) can be optimized in an end-to-end manner. We follow VQGAN and adopt L1 loss $\mathcal{L}_1$, perceptual loss $\mathcal{L}_{per}$ (Johnson, Alahi, and Fei-Fei 2016; Zhang et al. 2018), and adversarial loss $\mathcal{L}_{adv}$ (Esser, Rombach, and Ommer 2021) between the reconstructed image $\boldsymbol{I}_r$ and the input $\boldsymbol{I}_f$. $\mathcal{L}_{code}$ and commitment loss $\mathcal{L}_{comm}$ are used to update the codebook $\mathcal{C}$ and encoder $\mathcal{E}$, respectively:

$$\begin{aligned} \mathcal{L}_1 &= \|\boldsymbol{I}_f - \boldsymbol{I}_r\|_1 , \\ \mathcal{L}_{per} &= \|\Phi(\boldsymbol{I}_f) - \Phi(\boldsymbol{I}_r)\|_2^2 , \\ \mathcal{L}_{adv} &= -\log \mathfrak{D}(\boldsymbol{I}_r) , \\ \mathcal{L}_{code} &= \|\mathbf{sg}(\boldsymbol{Z}_c) - \boldsymbol{Z}_q\|_2^2 , \\ \mathcal{L}_{comm} &= \|\boldsymbol{Z}_c - \mathbf{sg}(\boldsymbol{Z}_q)\|_2^2 , \end{aligned} \quad (4)$$

where $\mathbf{sg}(\cdot)$ indicates the stop gradient operation. Φ denotes a pre-trained VGG16 model (Simonyan and Zisserman 2014) and $\mathfrak{D}$ represents the discriminator.

The overall training loss of VQGAN is summarized as:

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_{per} + \lambda_{adv}\mathcal{L}_{adv} + \mathcal{L}_{code} + \lambda_{comm}\mathcal{L}_{comm} . \quad (5)$$

λ_{comm} and λ_{adv} are the trade-off parameters and we set them to 0.5 and 0.8 in our experiment, respectively.

Notably, our VQGAN demonstrates remarkable generalization capabilities across out-of-domain glyphs and styles. As shown in Fig. 4, although these font styles and characters do not appear in the training process, nearly all stroke details are well reconstructed with the quantization process, showing the ability in generalizing to different font images.

Token Prior Refinement

To effectively integrate the token prior, VQ-Font leverages the well-trained codebook $\mathcal{C}$ and decoder $\mathcal{D}$. It casts font generation task into indices prediction task. This process involves aggregating fine-grained styles from reference glyphs and predicting codebook indices of ground-truth glyph $\boldsymbol{I}_g$.

Styles aggregation. Following FS-Font, our VQ-Font employs a cross-attention module to attentively capture fine-grained styles, where it takes a content glyph $\boldsymbol{I}_c$ as query, and k reference glyphs $\boldsymbol{I}_S$ as key and value. Specifically, we first use a content encoder $\mathcal{E}_c$ and a style encoder $\mathcal{E}_s$ to extract the feature maps from $\boldsymbol{I}_c$ and $\boldsymbol{I}_S$, *i.e.*, $\boldsymbol{f}_c \in \mathbb{R}^{hw \times c}$ and $\boldsymbol{f}_S \in \mathbb{R}^{khw \times c}$. Then, we calculate their attention weights as:

$$\boldsymbol{A}_{patch} = \frac{(\boldsymbol{W}^Q \boldsymbol{f}_c)(\boldsymbol{W}^K \boldsymbol{f}_S)^T}{\sqrt{c}} , \quad (6)$$

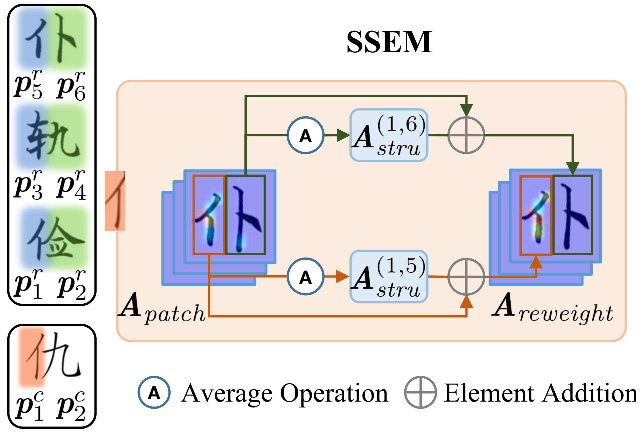

Figure 5: The overview of our Structure-level Style Enhancement Module (SSEM).

where $\boldsymbol{W}^Q$ and $\boldsymbol{W}^K$ are learnable parameters to project the extracted features into query and key, respectively. c is set to 256. With the above weights, VQ-Font attentively captures patch-level styles from reference features $\boldsymbol{f}_S$ to obtain the aggregated features $\boldsymbol{f}_{cs}$, which is formulated as:

$$\boldsymbol{f}_{cs} = \mathrm{Softmax}(\boldsymbol{A}_{patch}) \cdot (\boldsymbol{W}^V \boldsymbol{f}_S) , \quad (7)$$

In this way, we obtain patch-level aggregation features from the references, which may easily generate distorted strokes. Therefore, we need further fine-tuning for better quality.

Vector-Quantized font generation. To exploit the token prior in VQGAN, VQ-Font aims to quantize $\boldsymbol{f}_{cs}$ into $\boldsymbol{f}_q$ according to font codebook $\mathcal{C}$. However, due to the discrepancy between the feature spaces of $\boldsymbol{f}_{cs}$ and the VQGAN encoded, it is infeasible to compute the indices sequence of $\boldsymbol{f}_{cs}$ by nearest neighbor lookup. Inspired by (Zhou et al. 2022), we utilize a Transformer module to predict the indices $\boldsymbol{s}_{cs} \in \{0, \ldots, |\mathcal{C}| - 1\}^{hw}$ for all patch tokens of $\boldsymbol{f}_{cs}$. It employs 15 self-attention layers to globally model all tokens and an MLP to classify each token. Given the target glyph $\boldsymbol{I}_g$, we optimize this module from two aspects: (i) index prediction and (ii) image regression. Intuitively, the indices of $\boldsymbol{f}_{cs}$ can be approximated to those indices obtained by encoding and quantizing ground-truth font $\boldsymbol{I}_g$. Moreover, a glyph image $\boldsymbol{I}_q$ is projected from $\boldsymbol{f}_q$ with the VQGAN decoder and is combined with $\boldsymbol{I}_g$ to obtain reconstruction loss. To generalize the token prior to this task and preserve the effectiveness of prior knowledge, we fix the font codebook and only fine-tune the former layers of the decoder.

Structure-level Style Enhancement (SSEM)

Although leveraging the learned priors could promisingly improve the quality of synthesized glyphs, there are still inconsistent fine-grained styles between synthesized and reference glyphs, *e.g.*, the last second row in Fig. 7. Compared to style transfer in RGB images, few-shot font generation in single-channel glyphs is more likely to acquire irrelevant patch-level styles. The main reason is that the original attention weights $\boldsymbol{A}_{patch}$ are mainly based on geometry. To enhance the fidelity of captured styles, we propose to utilize the inherent structure (see Fig. 2) of Chinese characters to further recalibrate the attention weights.

Based on Fig. 2, we decompose the content glyph and reference glyphs into the structure components $\{\boldsymbol{p}_i^c\}_{i=0}^m$ and $\{\boldsymbol{p}_j^r\}_{j=0}^n$, where $\boldsymbol{p}_i^c$ and $\boldsymbol{p}_j^r$ denote the set of patch positions. As illustrated in Fig. 5, the attention weight $\boldsymbol{A}_{stru}^{(i,j)}$ between $\boldsymbol{p}_i^c$ and $\boldsymbol{p}_j^r$ is obtained by averaging their corresponding patch-level weights:

$$\boldsymbol{A}_{stru}^{(i,j)} = \frac{1}{|\boldsymbol{p}_i^c| \cdot |\boldsymbol{p}_j^r|} \sum_{x \in \boldsymbol{p}_i^c, y \in \boldsymbol{p}_j^r} \boldsymbol{A}_{patch}^{(x,y)} . \tag{8}$$

Finally, we reweight $\boldsymbol{A}_{patch}$ by adding $\boldsymbol{A}_{stru}$ to the corresponding patch positions:

$$\boldsymbol{A}_{reweight} = \boldsymbol{A}_{patch} \oplus \boldsymbol{A}_{stru} . \tag{9}$$

The new fusion feature $\boldsymbol{f}_{cs}$ in Eqn. (7) is reformulated as:

$$\boldsymbol{f}_{cs}^* = \mathrm{Softmax}(\boldsymbol{A}_{reweight}) \cdot (\boldsymbol{W}^V \boldsymbol{f}_S) . \tag{10}$$

In this way, our attention map can concentrate more on matching corresponding structure components, and reduce the adverse effect of other irrelevant strokes. Fig. 8 shows that after SSEM, our attention map $A_{reweight}$ has higher attention in corresponding structure components, thereby benefiting the following style transformation.

Training Objective

We train the content encoder $\mathcal{E}_c$, style encoder $\mathcal{E}_s$, cross-attention module, and Transformer module using cross-entropy loss $\mathcal{L}_{indice}$ while keeping font codebook fixed. We fine-tune the first four layers of the pre-trained font decoder using VQGAN-like losses, including L1 loss $\mathcal{L}_1$, perceptual loss $\mathcal{L}_{per}$, and adversarial loss $\mathcal{L}_{adv}$.

Cross Entropy loss. We first obtain the ground-truth codebook indices $\boldsymbol{s}_g$ using Eqn. (3) by taking the ground-truth font image $\boldsymbol{I}_g$ into the pre-trained VQGAN. To further improve the prediction performance, we follow FS-Font and design a self-reconstruction branch that uses $\boldsymbol{I}_g$ as the reference glyph. The code indices learning is defined as:

$$\mathcal{L}_{indice}^{main} = \mathrm{CE}(\hat{\boldsymbol{s}}_{cs}, \boldsymbol{s}_g); \quad \mathcal{L}_{indice}^{self} = \mathrm{CE}(\widetilde{\boldsymbol{s}}_{cs}, \boldsymbol{s}_g), \tag{11}$$

where $\hat{\boldsymbol{s}}_{cs}$ represents the indices predicted by the main branch and $\widetilde{\boldsymbol{s}}_{cs}$ represents the indices predicted by the self-reconstruction branch.

L1 loss. To maintain pixel-level consistency between the generated font images $\boldsymbol{I}_q$ and the ground-truth font images $\boldsymbol{I}_g$, we employ L1 loss as our reconstruction loss:

$$\mathcal{L}_1^f = \|\boldsymbol{I}_g - \boldsymbol{I}_q\|_1. \tag{12}$$

Adversarial loss and Perceptual loss. To further ensure that the generated font images have high visual quality, we additionally utilize adversarial loss and perceptual loss. Moreover, in our experiments, we employ a multi-head projection discriminator (Park et al. 2021a) and use the Unicode encoding of each Chinese character as the label:

$$\begin{aligned}
\mathcal{L}_{adv}^D &= -\mathbb{E}_{\boldsymbol{I}_g \sim p_{data}} \max(0, -1 + D_{uni}(\boldsymbol{I}_g)) \\
&\quad - \mathbb{E}_{\boldsymbol{I}_q \sim p_{gen}} \max(0, -1 - D_{uni}(\boldsymbol{I}_q)), \\
\mathcal{L}_{adv}^G &= -\mathbb{E}_{\boldsymbol{I}_q \sim p_{gen}} D_{uni}(\boldsymbol{I}_q), \\
\mathcal{L}_{per}^f &= \|\Phi(\boldsymbol{I}_g) - \Phi(\boldsymbol{I}_q)\|_2^2,
\end{aligned} \tag{13}$$

where Φ denotes VGG16 same as that in Eqn. (4).

Overall objective loss. To sum up, the final loss function for training our VQ-Font is formulated as:

$$\begin{aligned}
\mathcal{L}_{VQ\text{-}Font} =& \lambda_{self}\mathcal{L}_{indice}^{self} + \lambda_{main}\mathcal{L}_{indice}^{main} + \lambda_1^f \mathcal{L}_1^f \\
&+ \lambda_{adv}^f \mathcal{L}_{adv}^G + \lambda_{per}^f \mathcal{L}_{per}^f ,
\end{aligned} \tag{14}$$

where λ_{self}, λ_{main}, λ_1^f, λ_{adv}^f and λ_{per}^f are the trade-off parameters for balancing each loss item. In our experiments, we set $\lambda_{main} = \lambda_1^f = 2$, $\lambda_{self} = \lambda_{per}^f = 1$ and $\lambda_{adv}^f = 0.002$.

Experiments

Datasets and Evaluation Metrics

We follow previous works (Park et al. 2021a,b; Tang et al. 2022) and collect 382 fonts with various types to build our dataset. Each font contains 3499 Chinese characters and the resolution for each character is 128×128. We split these 3499 characters into 3 groups, *i.e.*, 2841 seen characters, 158 reference characters, and 500 unseen characters. For each character, we follow FS-Font (Tang et al. 2022) and select 3 reference characters from the reference set that can cover most of its structure components. We use Kai font as our default content font and train our model on 371 seen fonts, leaving 10 unseen fonts that do not appear in the training stage. In this way, our training set totally consists of 371 seen fonts, each of which has 2841 seen characters (SFSC). Our test set consists of two parts, *i.e.*, 10 seen fonts with 500 unseen characters (SFUC) and 10 unseen fonts with 500 unseen characters (UFUC), encompassing a diverse range of font types, such as handwriting, printing, and artistic styles.

To evaluate the performance, we follow these competing methods and report the L1, RMSE, PSNR, SSIM and LPIPS (Zhang et al. 2018), covering both pixel consistency and perceptual similarity. Additionally, we also conduct a user study to further assess the visual quality of the generated results from human perception.

Implementation Details

In the pre-training phase of our font codebook, we encode the font images into 16×16 features. The size of our font codebook is set to 1,024. At this stage, VQGAN is trained for $2e5$ iterations with a learning rate of $4e$-5. The number of attention heads in the cross-attention module is set to 8. We select 3 reference characters for each Chinese character. In the subsequent token prior refinement stage, we keep the pre-trained font codebook and the later layers of the decoder fixed, while concentrating on training the remaining layers of the VQ-Font for 300k iterations. Here, the learning rate is set to $2e$-4. We adopt Adam optimizer (Kingma and Ba 2014) with a batch size of 32 and rely on one A6000 GPU.

Method	L1 ↓	RMSE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	User (C)% ↑	User (S)% ↑
Seen Fonts Unseen Chars (SFUC)							
LF-Font (AAAI 2021)	0.0921	0.264	17.818	0.746	0.157	70.5	4.0
MX-Font (ICCV 2021)	0.1002	0.278	17.326	0.725	0.169	83.4	2.9
DG-Font (CVPR 2021)	0.0747	0.233	18.901	0.782	0.127	80.7	6.7
FS-Font (CVPR 2022)	0.0663	0.220	19.702	0.805	0.126	90.3	12.9
CF-Font (CVPR 2023)	0.0667	0.217	19.559	0.805	0.111	86.7	10.6
VQ-Font (Ours)	**0.0610**	**0.209**	**20.285**	**0.822**	**0.096**	**97.2**	**62.9**
Unseen Fonts Unseen Chars (UFUC)							
LF-Font (AAAI 2021)	0.0976	0.274	17.495	0.726	0.174	68.0	3.4
MX-Font (ICCV 2021)	0.1061	0.288	17.045	0.706	0.185	76.5	4.0
DG-Font (CVPR 2021)	0.0807	0.246	18.465	0.768	0.139	78.4	4.7
FS-Font (CVPR 2022)	0.0666	0.220	19.672	0.797	0.137	84.5	9.2
CF-Font (CVPR 2023)	0.0685	0.222	19.344	0.798	0.116	84.3	11.3
VQ-Font (Ours)	**0.0621**	**0.210**	**20.249**	**0.812**	**0.103**	**95.6**	**67.4**

Table 1: Quantitative comparison with state-of-the-art methods on SFUC and UFUC datasets.

Figure 6: Qualitative comparison with competing methods on UFUC dataset. Best view it by zooming in on the screen.

Comparison Methods

We compare the performance of our proposed VQ-Font with five state-of-the-art methods, including LF-Font (Park et al. 2021a), MX-Font (Park et al. 2021b), DG-Font (Xie et al. 2021), FS-Font (Tang et al. 2022), and CF-Font (Wang et al. 2023). To make a fair comparison, we retrain all these methods using their default settings on our dataset. More results and analyses can be found in our suppl.

Quantitative evaluation. Tab. 1 presents the comparison of our VQ-Font with other state-of-the-art methods. The results demonstrate that our approach outperforms others in terms of both pixel-based and perception-based metrics on UFUC and SFUC datasets. Specifically, in terms of L1, we obtain 7.99% improvement over the second-best result on SFUC dataset and 6.76% on UFUC dataset, respectively. As for LPIPS, our VQ-Font outperforms the second-best result with a large margin, *i.e.*, 13.51% improvement on SFUC dataset and 11.21% improvement on UFUC dataset. This indicates that the synthesized results of our VQ-Font have a better fidelity and also align better with human perception.

Qualitative evaluation. As shown in Fig. 6, we select various styles of fonts, including serif, artistic, and handwriting fonts, from UFUC dataset for qualitative comparison. We can observe that LF-font and MX-font struggle to capture the fine-grained styles, resulting in stroke artifacts (see redred boxes). DG-Font and CF-Font also exhibit missing strokes and distorted strokes in challenging cases (*e.g.*, 2∼3 columns in Fig. 6). Besides, FS-Font tends to produce blurry font and lose stroke details (*e.g.*, 5∼6 columns in Fig. 6). In comparison, our proposed VQ-Font can effectively capture and transfer the stroke-level and structure-level styles of the reference glyphs. The integration of token prior further contributes to the higher quality of the glyphs.

User study. To further compare the visual quality of different methods, we conduct a user study. A total of 30 volunteers with computer vision backgrounds participated in evaluating the experimental results. We utilize 10 fonts with challenging styles (*e.g.*, 2∼4 columns in Fig. 6) in UFUC and SFUC datasets respectively, and randomly generate 10 characters for each font using these methods. Here we consider two items, *i.e.*, 1) content accuracy which classifies

	L1 ↓	RMSE ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Baseline	0.0678	0.223	19.318	0.793	0.116
+C	0.0628	0.212	20.170	0.810	0.105
+CS	**0.0621**	**0.210**	**20.249**	**0.812**	**0.103**

Table 2: Quantitative results of different VQ-Font variants. C and S represent codebook and SSEM, respectively.

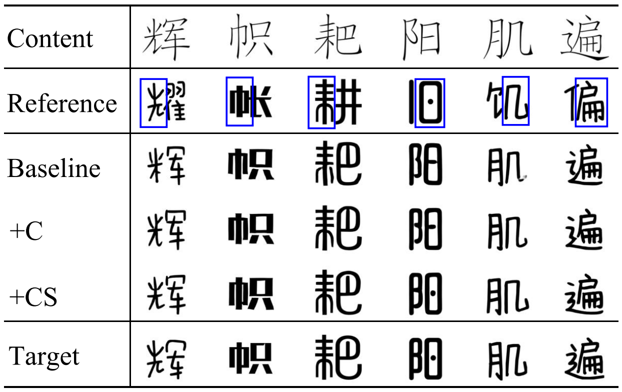

Figure 7: Qualitative results of different VQ-Font variants. The structure components appearing in the reference glyphs are highlighted with blueblue boxes.

whether the font images within each method have extra strokes and other anomalies or not, and 2) style consistency which concentrates on selecting the font images among all the methods that are closest to the reference in stroke style, stroke thickness, and edge details. The evaluation result on the right part of Tab. 1 demonstrates that on the one hand, our method has better content accuracy than others from human perception. On the other hand, users are more likely to select our results which have better style consistency, while other methods struggle with these challenging font styles.

Ablation Study

In this section, we mainly discuss the effectiveness of token prior refinement and structure-aware enhancement. We first train a baseline model by utilizing the cross-attention mechanism to learn the spatial correspondence at the patch level which is similar to FS-Font. Then, we gradually add the pre-trained font codebook and structure-level style enhancement module (SSEM) to the baseline for validation. Tab. 2 shows the quantitative results on UFUC. One can see that the pre-trained font codebook (+C) can obviously improve the SSIM and LPIPS performance, which indicates the better visual quality brought by our font codebook. When combining the codebook with the structure-level style enhancement module (+CS), our method has a further improvement, especially in PSNR. This indicates that SSEM can effectively capture the reference styles and contribute to higher fidelity.

The visual comparison in Fig. 7 demonstrates a noticeable reduction in distorted strokes and missing details when utilizing the font codebook (+C). Besides, our SSEM enhances the fidelity of the generated glyphs by aligning them more accurately with the corresponding structure components in the reference. These improved regions are highlighted within the blue box. Both the quantitative and qualitative evaluations demonstrate the improvements in fidelity and visual quality brought by our token prior refinement and structure-aware enhancement.

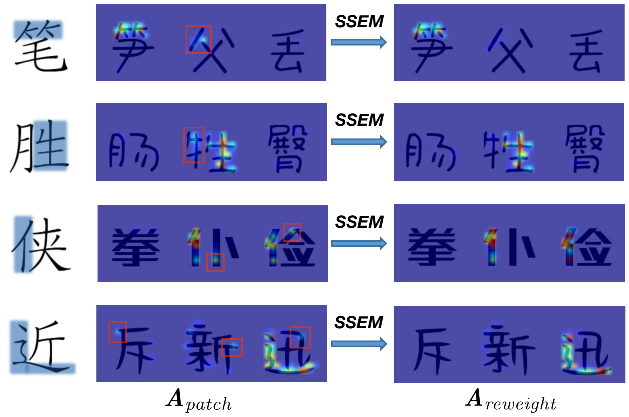

Figure 8: Attention maps w/o and w/ SSEM.

Decoder	L1 ↓	RMSE ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Fix	0.0648	0.216	20.069	0.805	0.121
Fine-tune	**0.0621**	**0.210**	**20.249**	**0.812**	**0.103**

Table 3: Comparison of fixed and fine-tuned decoders.

To further validate the effect of SSEM, we visualize the attention maps before and after the enhancement operation. As shown in Fig. 8, the leftmost column represents the content glyphs, and the visualized components are highlighted in blue. The visualization of different structure components demonstrates that our enhancement operation can eliminate the attention on irrelevant regions (see redred boxes) and concentrate more on the corresponding components of the reference. This helps to capture the structure-level styles and then boosts the subsequent style transformation.

Finally, we explore the effect of fine-tuning the pre-trained font decoder in our method. We conduct another experiment by freezing all parameters of the decoder. From Tab. 3 we can see that although the pre-trained decoder has the ability to generate photo-realistic font images, by fine-tuning the decoder with the end-to-end optimization, it can well generalize to our font generation task and bridge the domain gap between our synthetic and real-world fonts.

Conclusion

In this paper, we propose VQ-Font, a new few-shot font generation framework. It refines the font images by incorporating token prior encapsulated in a pre-trained font codebook. Additionally, the Structure-level Style Enhancement Module (SSEM) leverages the structure information of Chinese characters to recalibrate fine-grained styles from the references. This enhances the alignment of structure-level styles between content and reference glyphs. By combining the token prior refinement and SSEM, the results of our VQ-Font are more realistic and have higher fidelity in comparison with other start-of-the-art methods.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant No. U19A2073.

References

Cha, J.; Chun, S.; Lee, G.; Lee, B.; Kim, S.; and Lee, H. 2020. Few-shot compositional font generation with dual memory. In *European Conference on Computer Vision*, 735–751. Springer.

Chang, J.; Gu, Y.; Zhang, Y.; Wang, Y.-F.; and Innovation, C. 2018. Chinese Handwriting Imitation with Hierarchical Generative Adversarial Network. In *British Machine Vision Conference*, 290.

Chen, C.; Shi, X.; Qin, Y.; Li, X.; Han, X.; Yang, T.; and Guo, S. 2022. Real-world blind super-resolution via feature matching with implicit high-resolution priors. In *Proceedings of the 30th ACM International Conference on Multimedia*, 1329–1338.

Chen, Z.; Wang, G.; and Liu, Z. 2022. Text2light: Zero-shot text-driven hdr panorama generation. *ACM Transactions on Graphics (TOG)*, 41(6): 1–16.

Esser, P.; Rombach, R.; and Ommer, B. 2021. Taming transformers for high-resolution image synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition*, 12873–12883.

Fu, B.; He, J.; Wang, J.; and Qiao, Y. 2023. Neural Transformation Fields for Arbitrary-Styled Font Generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 22438–22447.

Gao, Y.; Guo, Y.; Lian, Z.; Tang, Y.; and Xiao, J. 2019. Artistic glyph image synthesis via one-stage few-shot learning. *ACM Transactions on Graphics (TOG)*, 38(6): 1–12.

Gao, Y.; and Wu, J. 2020. Gan-based unpaired chinese character image translation via skeleton transformation and stroke rendering. In *proceedings of the AAAI conference on artificial intelligence*, volume 34, 646–653.

Gu, Y.; Wang, X.; Xie, L.; Dong, C.; Li, G.; Shan, Y.; and Cheng, M.-M. 2022. VQFR: Blind face restoration with vector-quantized dictionary and parallel decoder. In *European Conference on Computer Vision*, 126–143. Springer.

Hassan, A. U.; Ahmed, H.; and Choi, J. 2021. Unpaired font family synthesis using conditional generative adversarial networks. *Knowledge-Based Systems*, 229: 107304.

Hinton, G. E.; and Zemel, R. 1993. Autoencoders, minimum description length and Helmholtz free energy. *Advances in neural information processing systems*, 6.

Isola, P.; Zhu, J.-Y.; Zhou, T.; and Efros, A. A. 2017. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1125–1134.

Jiang, Y.; Lian, Z.; Tang, Y.; and Xiao, J. 2017. DCFont: an end-to-end deep Chinese font generation system. In *SIGGRAPH Asia 2017 Technical Briefs*, 1–4.

Jiang, Y.; Lian, Z.; Tang, Y.; and Xiao, J. 2019. Scfont: Structure-guided chinese font generation via deep stacked networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 4015–4022.

Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 694–711. Springer.

Karras, T.; Laine, S.; Aittala, M.; Hellsten, J.; Lehtinen, J.; and Aila, T. 2020. Analyzing and improving the image quality of StyleGAN. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li, X.; Zuo, W.; and Loy, C. C. 2023. Learning Generative Structure Prior for Blind Text Image Super-resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Liu, W.; Chen, C.; and Wong, K.-Y. 2018. Char-net: A character-aware neural network for distorted scene text recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Liu, W.; Chen, C.; Wong, K.-Y. K.; Su, Z.; and Han, J. 2016. Star-net: a spatial attention residue network for scene text recognition. In *British Machine Vision Conference*, volume 2, 7.

Liu, W.; Liu, F.; Ding, F.; He, Q.; and Yi, Z. 2022. Xmp-font: self-supervised cross-modality pre-training for few-shot font generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 7905–7914.

Lyu, P.; Bai, X.; Yao, C.; Zhu, Z.; Huang, T.; and Liu, W. 2017. Auto-encoder guided GAN for Chinese calligraphy synthesis. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 1, 1095–1100. IEEE.

Park, S.; Chun, S.; Cha, J.; Lee, B.; and Shim, H. 2021a. Few-shot font generation with localized style representations and factorization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 2393–2402.

Park, S.; Chun, S.; Cha, J.; Lee, B.; and Shim, H. 2021b. Multiple heads are better than one: Few-shot font generation with multiple localized experts. In *IEEE International Conference on Computer Vision*, 13900–13909.

Razavi, A.; Van den Oord, A.; and Vinyals, O. 2019. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sun, D.; Ren, T.; Li, C.; Su, H.; and Zhu, J. 2017. Learning to write stylized chinese characters by reading a handful of examples. *arXiv preprint arXiv:1712.06424*.

Sun, D.; Zhang, Q.; and Yang, J. 2018. Pyramid embedded generative adversarial network for automated font generation. In *2018 24th International Conference on Pattern Recognition (ICPR)*, 976–981. IEEE.

Tang, L.; Cai, Y.; Liu, J.; Hong, Z.; Gong, M.; Fan, M.; Han, J.; Liu, J.; Ding, E.; and Wang, J. 2022. Few-shot font generation by learning fine-grained local styles. In *IEEE Conference on Computer Vision and Pattern Recognition*, 7895–7904.

Tian, Y. 2016. Rewrite: Neural style transfer for chinese fonts, 2016. *Retrieved Nov*, 23: 2016.

Tian, Y. 2017. Master Chinese calligraphy with conditional adversarial networks.

Van Den Oord, A.; Vinyals, O.; et al. 2017. Neural discrete representation learning. *Advances in neural information processing systems*, 30.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, C.; Zhou, M.; Ge, T.; Jiang, Y.; Bao, H.; and Xu, W. 2023. CF-Font: Content Fusion for Few-shot Font Generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1858–1867.

Wen, C.; Pan, Y.; Chang, J.; Zhang, Y.; Chen, S.; Wang, Y.; Han, M.; and Tian, Q. 2021. Handwritten Chinese font generation with collaborative stroke refinement. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 3882–3891.

Wu, R.-Q.; Duan, Z.-P.; Guo, C.-L.; Chai, Z.; and Li, C. 2023. RIDCP: Revitalizing Real Image Dehazing via High-Quality Codebook Priors. In *IEEE Conference on Computer Vision and Pattern Recognition*, 22282–22291.

Wu, S.-J.; Yang, C.-Y.; and Hsu, J. Y.-j. 2020. Calligan: Style and structure-aware chinese calligraphy character generator. *arXiv preprint arXiv:2005.12500*.

Xie, Y.; Chen, X.; Sun, L.; and Lu, Y. 2021. Dg-font: Deformable generative networks for unsupervised font generation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 5130–5140.

Yang, S.; Liu, J.; Wang, W.; and Guo, Z. 2019a. Tet-gan: Text effects transfer via stylization and destylization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 1238–1245.

Yang, S.; Wang, Z.; Wang, Z.; Xu, N.; Liu, J.; and Guo, Z. 2019b. Controllable artistic text style transfer via shape-matching gan. In *IEEE International Conference on Computer Vision*, 4442–4451.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, 586–595.

Zhang, Y.; Wei, Y.; Ji, Z.; Bai, J.; Zuo, W.; et al. 2022. Towards diverse and faithful one-shot adaption of generative adversarial networks. *Advances in Neural Information Processing Systems*, 35: 37297–37308.

Zhang, Y.; Zhang, Y.; and Cai, W. 2018. Separating style and content for generalized style transfer. In *IEEE Conference on Computer Vision and Pattern Recognition*, 8447–8455.

Zhou, S.; Chan, K.; Li, C.; and Loy, C. C. 2022. Towards robust blind face restoration with codebook lookup transformer. *Advances in Neural Information Processing Systems*, 35: 30599–30611.