GC-Font: Few-Shot Font Generation via Global Contextual Feature Modelling

Weiran Chen
wrchen2023@stu.suda.edu.cn
Guiqian Zhu
gqzhu@suda.edu.cn
Ying Li
ingli@suda.edu.cn
Yi Ji
jiyi@suda.edu.cn
Chunping Liu*
cpliu@suda.edu.cn

School of Computer Science and Technology, Soochow University, Suzhou, China

Abstract

Few-shot font generation aims to create new fonts with a limited number of glyph references. It can be used to greatly reduce the labour cost of manual font design. However, due to the variety and complexity of font styles, the results generated by existing methods often suffer from visible defects, such as stroke errors, blurriness or distorted shapes. To address these issues, we propose GC-Font, a novel framework which integrates a Global Contextual Feature Modelling (GCFM) module. Specifically, this module is inserted between the content encoder and decoder, where it fuses convolution and attention mechanisms to process intermediate feature maps and injects enhanced global contextual features into the decoder. Moreover, we apply adaptive convolutions to the low-level feature maps from the content encoder to strengthen contextual correlations. In addition, a skeleton consistency loss and an edge consistency loss are also designed to improve geometric alignment. Extensive experiments reveal that our GC-Font outperforms the state-of-the-art methods in both qualitative and quantitative evaluations, demonstrating its effectiveness on diverse font styles. Our source code can be found at https://github.com/wrchen2001/GC-Font.

1 Introduction

The task of few-shot font generation enables the transfer of font styles from a source domain to a target domain based on only a few reference images. It can significantly alleviate the burden of extensive and time-consuming manual design. Nowadays, with the rapid development of deep learning [2, 6], researchers have made great advances in creating gratifying fonts. Early approaches typically disentangle the content and style representations of fonts, then integrate them to directly synthesize new glyphs [29, 33]. In addition, some works adopt

* Corresponding author

structure-aware representations [19, 20], which mainly decompose characters into different components based on their structures and extra multiple style representations. Nevertheless, existing methods still have several drawbacks. The highly diverse and intricate nature of font styles often leads to obvious defects in synthesized results, like incomplete or extraneous strokes, anomalous blurriness and distorted shapes. Besides, while many approaches are concerned with the local patterns of glyphs, they usually ignore the importance of capturing global contextual dependencies within the feature maps. Considering that many logographic scripts rely on global structure to convey stylistic integrity, this neglect would make it difficult to maintain precise glyph shapes, especially for some complex characters.

To tackle the aforementioned challenges, we put forward a novel end-to-end few-shot font generation method called GC-Font. To be specific, we design an innovative Global Contextual Feature Modelling (GCFM) module, which is strategically placed between the content encoder and decoder. This module efficiently integrates convolutional operations with the attention mechanism, allowing the model to capture the global stylistic representations while maintaining the fidelity of intricate structural elements. In addition, we also employ adaptive convolutions on the low-level content feature maps to improve the model's capacity to encapsulate contextual relevance. Apart from these, to strengthen the geometric alignment, we bring in two specialized loss functions: skeleton consistency loss and edge consistency loss. Skeleton consistency loss ensures the accurate preservation of critical skeletons, while edge consistency loss focuses on retaining the sharp and smooth character boundaries. Consequently, our method yields superior and impressive stylization performance, highlighting its potential for practical applications.

To sum up, the major contributions of this paper are as follows:

- We devise an inventive Global Contextual Feature Modelling (GCFM) module, which consists of convolution operations and an attention mechanism to better augment the global contextual features. Based on this, we introduce GC-Font, a new font generation framework that facilitates accurate style transfer while preserving glyph integrity.

- We embed adaptive convolutions within the content encoder, enabling dynamic adjustments of convolutional filters according to the input data. Besides, we also design two auxiliary loss functions, skeleton consistency loss and edge consistency loss, to better improve the overall visual consistency.

- Extensive experimental results verify that our GC-Font surpasses the leading models in both qualitative and quantitative evaluations, which demonstrates its effectiveness and generalizability on diverse font styles and characters.

2 Related Work

Image-to-Image Translation. Image-to-image (I2I) translation is to convert an image from one domain to another while preserving its semantic content. Previous methods, such as Pix2pix [10] and CycleGAN [36], mainly leveraged Generative Adversarial Network (GAN) [6] for supervised or unsupervised settings. Later, several few-shot methods, including FUNIT [16] and ManiFest [21], were proposed to simultaneously generate various style outputs from the same input. Recently, diffusion models have shown great promise in I2I translation tasks. For instance, ILVR [4] generated high-quality images based solely on a pre-trained unconditional Denoising Diffusion Probabilistic Model (DDPM) [9] with one reference image.

SCDM [12] applied stochastic perturbations and conditioned I2I translation on the diffused labels. Notably, since font generation can be viewed as a special I2I translation task, many generic I2I translation approaches can be adaptively modified for font generation.

Few-Shot Font Generation. Few-shot font generation aims to generate a required font library with just a handful of glyph references. A classical idea involves extracting content and style features from the input source and reference images, respectively. EMD [33] and AGIS-Net [5] disentangled style and content as two domains, and modelled each font style as a universal representation. DG-Font [29] applied deformable convolutions to extract content features, while CF-Font [26] further extended it with a content fusion module. FontDiffuser [30] implemented a multi-scale content aggregation block along with a style contrastive refinement module to guide the transfer. To handle highly-structured characters, some researches utilized prior domain knowledge, like stroke decomposition, to optimize the final results. LF-Font [19] represented component-wise styles via low-rank matrix factorization. MX-Font [20] leveraged multiple experts with bipartite matching to extract localized features. CG-GAN [13] trained a component predictor to better supervise the generator during adversarial training, while FsFont [24] built a character-reference mapping relationship and employs cross attention to align the patch-level features. Diff-Font [7] infused predefined embedding tokens into the diffusion model to support the sampling process. DP-Font [47] adopted a multi-attribute guidance and a strict stroke order to direct the generation process. IF-Font [3] applied the ideographic description sequence to control the glyph semantics. However, these methods still demanded the component labels. In contrast, VQ-Font [17] employed a vector quantization-based encoder to automatically extract components. Despite the remarkable progress, existing approaches often neglect the necessity of capturing global contextual features within content representations, which would easily lead to degraded structural fidelity and distorted glyphs in the generated results.

3 The Proposed Method

3.1 Overall Scheme

Given a set of k reference images $x = \{x_1, x_2, \ldots, x_k\}$ and one content image I_c, our model aims to generate a target character I_o that retains the content of I_c while adopting the style of x. As illustrated in Fig. 1(a), the generator network mainly includes six parts. Among them, the reference encoder E_r extracts the style representations from x and encodes them into a vector $f_s = \{f_1, f_2, \ldots, f_k\}$. The content encoder E_c maps I_c to a feature representation f_c. It is pre-trained via a Residual-Quantized Variational Auto-Encoder (RQ-VAE) to acquire a component-wise codebook F_c as well. Next, the style-content fusion module integrates F_c and f_s to derive the style feature f_{sa}. Besides, the content alignment module re-weights and aggregates the feature vector f_s to get the aligned content feature f_{ca}. Furthermore, our proposed Global Contextual Feature Modelling (GCFM) module transfers global contextual features from the content encoder to the decoder to refine the generation quality. Ultimately, the decoder D_r generates the target image I_o by concatenating f_c, f_{sa}, and f_{ca} as inputs.

During the model's training process, a multi-task discriminator $D_{s,c}$ is employed to distinguish between the real and generated images, as shown in Fig. 1(b). Moreover, to ensure that the generated glyph can accurately reflect the reference style while preserving the input content structure, the discriminator also performs the classification on each character, identifying both its style and content categories.

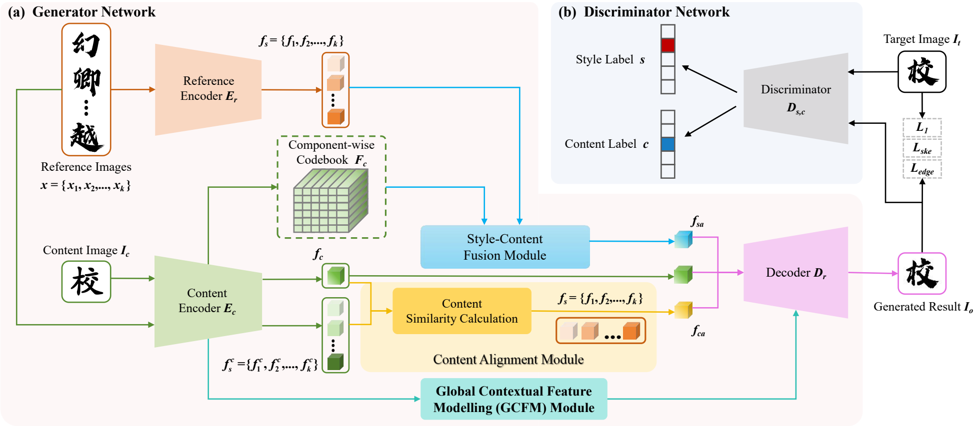


Figure 1: The framework of our proposed GC-Font model.

3.2 Glyph Feature Decomposition and Content Alignment Module

Glyph Feature Decomposition. In our method, the content encoder is pre-trained using a glyph feature decomposing network that maps each character into a component-wise codebook for image reconstruction. As illustrated in Fig. 2(a), the network comprises a content encoder E_c and a reconstruction decoder D_r, connected through a discrete space with a learnable codebook F_c of size U. Among them, adaptive convolution [1], employed in E_c, first applies a conditioned 2D style filter f to modulate feature channels based on the local neighbourhood $N(x)$ around each spatial location x, and subsequently expands the style parameters into a separable and pointwise convolution tensor $q \in \mathbb{R}^C$ for C input channels as follows:

$$\mathrm{AdaConv}(x) = \sum_{c \in C} q_c \left(\sum_{x_i \in N(x)} f_i \left(\frac{x_i - \mu_x}{\sigma_x} \right) + b \right) \tag{1}$$

where μ_x and σ_x are the channel-wise mean and standard deviation, while b denotes the bias.

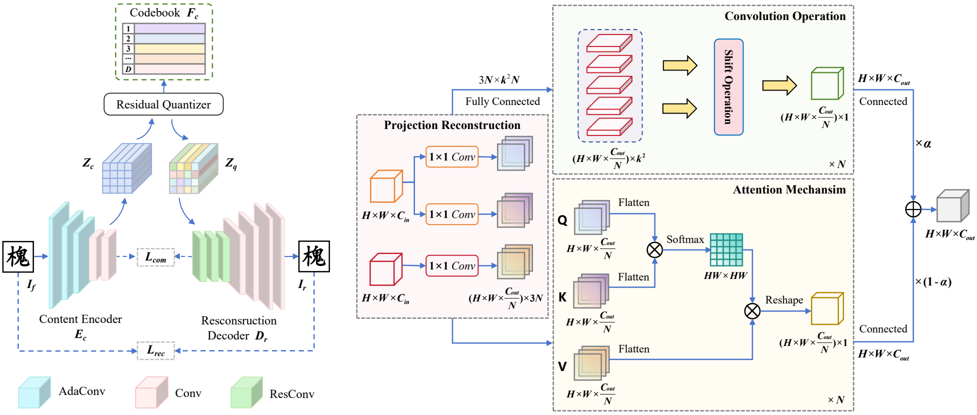
Given a character image I_f, the content encoder E_c first maps it into a continuous feature representation Z_c. Then, we adopt Residual Quantization (RQ) [14] to discretize Z_c. Explicitly, starting from the initial residual $r_0 = Z_c$ and a quantization depth D, at each depth $d \in \{1, 2, \dots, D\}$, RQ iteratively selects the index p_d that minimizes the distance between the current residual r_{d-1} and the embedding vector $e(p)$ in the codebook F_c as:

$$\begin{aligned} p_d &= \mathrm{argmin}_{p \in F_c} \| r_{d-1} - e(p) \|_2^2 \\ r_d &= r_{d-1} - e(p_d) \end{aligned} \tag{2}$$

The final quantized vector Z_q is obtained by accumulating all the D residual embeddings, which is fed into the reconstruction decoder D_r to generate the reconstructed glyph I_r.

During pre-training, E_c and D_r are optimized by minimizing an objective function $\mathcal{L}_{pre}$ that contains a reconstruction loss $\mathcal{L}_{rec}$ and a commitment loss $\mathcal{L}_{com}$, defined as:

$$\mathcal{L}_{pre} = \mathcal{L}_{rec} + \beta \mathcal{L}_{com} = \| I_r - I_f \|_1 + \beta \sum_{d=1}^{D} \| Z_c - \mathrm{sg}[Z_q^{(d)}] \|_2^2 \tag{3}$$



(a) Glyph Feature Decomposing Network (b) Global Contextual Feature Modelling Module

Figure 2: The architecture of glyph feature decomposing network and global contextual feature modelling module in GC-Font model.

where sg denotes the stop-gradient operator and β is a balancing hyper-parameter. Experimentally, we set β to 0.25, the quantization depth D to 3 and the codebook size U to 100.

Upon completing pre-training, we fix the content encoder E_c along with the codebook F_c to build the font generation model. To be noticed, although the reconstruction decoder D_r shares the same architecture with the generation model's decoder, the latter is re-initialized and trained from scratch during the main training phase.

Content Alignment Module. From a perceptual perspective, reference characters sharing similar elements with the input glyph should be more concerned during the style transfer process [35]. Therefore, in the content alignment module, we first extract the content features $f_s^c = \{f_1^c, f_2^c, ..., f_k^c\}$ from reference images via the content encoder E_c. Besides, to ensure dimension compatibility, both f_s^c and f_c are reshaped to $\overline{f_s^c}$ and $\overline{f_c}$. Then, the similarity values are calculated via the normalized cross-correlation measurement [23] as follows:

$$\Phi_{ia} = \frac{\langle \overline{f_i^{ca}}, \overline{f_c^a} \rangle}{\left\| \overline{f_i^{ca}} \right\| * \left\| \overline{f_c^a} \right\|}, a \in \{1, 2, \dots, B\} \tag{4}$$

where a refers to the position within the B-dimensional channel and Φ_{ia} represents the similarity between the a-th channel of the i-th reference image and the content image. Next, we normalize these values channel-wise into $\overline{\Phi}_{ia}$ using the softmax function, and apply them to weight the reference style feature representations to form the aligned content feature f_{ca} by:

$$f_{ca} = \mathrm{concat}_a \left(\sum_{i=1}^{k} \overline{\Phi}_{ia} f_i^a \right) \tag{5}$$

3.3 Style-Content Fusion Module

The style-content fusion module is built upon multi-head transformer [25] with m parallel attention heads. For the i-th head, we apply two linear projections to obtain the key matrix K_i and the value matrix V_i from the feature maps f_s, respectively. Meanwhile, the query matrix

Q_i is derived by projecting the codebook F_c with a learnable weight. Then, the attention output can be formulated as:

$$A_i = Attention(Q_i, K_i, V_i) = \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_i}}\right) V_i \tag{6}$$

where d_i denotes the dimension of K_i. The outputs from all the heads are concatenated and passed through a linear projection W_s to produce the final fused feature F_s by:

$$F_s = \text{concat}(A_1, A_2, ..., A_m) W_s \tag{7}$$

In our experiments, we set m to 8 in three stacked transformer layers. For each input glyph, its style feature f_{sa} can be aggregated from F_s by searching for the nearest index in F_c, as expressed in Eq. (2), and choosing the matching vector for substitution.

3.4 Global Contextual Feature Modelling Module

As depicted in Fig. 2(b), our proposed Global Contextual Feature Modelling (GCFM) module contains three parts: projection reconstruction, convolution operation and attention mechanism. Notably, a critical aspect of the GCFM module is the selection of feature levels for processing. Based on the observation that lower layers capture local patterns while higher layers encode global information [18, 22], we choose the feature maps from the last two layers of the content encoder, marked as $\Gamma_1, \Gamma_2 \in \mathbb{R}^{H \times W \times C_{in}}$, for the inputs.

In the projection reconstruction stage, we first apply 1×1 convolutions to Γ_1 and Γ_2 to obtain three feature maps, which are then split into N depth-wise subsets, yielding $I_1, I_2, I_3 \in \mathbb{R}^{H \times W \times \frac{C_{out}}{N}}$. During the convolution operation phase, I_1, I_2 and I_3 are reshaped and spliced, followed by a light fully connected layer that expands each spatial position into $k \times k$ patches to get $Y \in \mathbb{R}^{H \times W \times \frac{C_{out}}{N} \cdot k^2}$. Finally, Y is processed through a depth-separable convolution with $G = \frac{C_{out}}{N}$ groups to produce the output $F_{conv} \in \mathbb{R}^{H \times W \times C_{out}}$, as follows:

$$F_{conv} = \text{concat}[(C_{dev}(Y_1), C_{dev}(Y_2), ..., C_{dev}(Y_G)] \tag{8}$$

where $C_{dev}(\cdot)$ refers to the convolution calculation with a kernel size of 3.

Within the attention mechanism, similar to the style-content fusion module, we also adopt a multi-head transformer with n heads, where I_1, I_2 and I_3 serve as the query Q, key K and value V, respectively. After flattening, we can acquire $Q', K', V' \in \mathbb{R}^{H \cdot W \times \frac{C_{out}}{N}}$. Then, the outputs of the j-th head and the overall attention mechanism are calculated by:

$$\begin{gathered} S_j = Attention(Q_j^{'}, K_j^{'}, V_j^{'}) = \text{softmax}\left(\frac{Q_j^{'} K_j^{'T}}{\sqrt{d_j^{'}}}\right) V_j^{'} \\ F_{att} = \text{concat}[S_1, S_2, ..., S_n] W_g \end{gathered} \tag{9}$$

where $d_j^{'}$ represents the dimension of $K_j^{'}$ and W_g is a learnable linear projection.

Ultimately, the feature maps $F_{out} \in \mathbb{R}^{H \times W \times C_{out}}$ derived from the GCFM module are computed through a weighted sum of the convolutional and attention outputs:

$$F_{out} = \alpha F_{conv} + (1 - \alpha) F_{att} \tag{10}$$

where α is a tunable hyper-parameter. In our implementation, we set n to 4 and α to 0.5, respectively. After getting the output F_{out}, we add it to the first three layers of the decoder D_r, thereby elevating the decoder's ability to incorporate global context features.

3.5 Training Objective

The loss functions of our proposed model include five parts: adversarial loss, matching loss, style contrast loss, skeleton consistency loss and edge consistency loss.

Adversarial Loss. To ensure plausibility in both style and content, we utilize a multi-head discriminator $D_{s,c}$ and train it using the hinge GAN loss [31], as defined below:

$$\begin{aligned}\mathcal{L}_{adv}^{D} &= -\mathbb{E}_{I_t\sim p_{data}} min\left(0,-1+D_{s,c}\left(I_t\right)\right)-\mathbb{E}_{I_o\sim p_G} min\left(0,-1-D_{s,c}\left(I_o\right)\right)\\ \mathcal{L}_{adv}^{G} &= -\mathbb{E}_{I_o\sim p_G} D_{s,c}\left(I_o\right)\end{aligned} \tag{11}$$

where p_{data} and p_G denote the set of real images and generated images, respectively.

Matching Loss. To mitigate mode collapse and enforce alignment with the ground truth I_t at both pixel and feature levels, we apply an $\mathcal{L}_1$ loss on the image and its features, as follows:

$$\begin{aligned}\mathcal{L}_{img} &= \mathbb{E}_{I_t\sim p_{data}}\left[\|I_t-I_o\|_1\right]\\ \mathcal{L}_{feat} &= \mathbb{E}_{I_t\sim p_{data}}\left[\sum_{m=1}^{M}\left\|D_{s,c}^{(m)}\left(I_t\right)-D_{s,c}^{(m)}\left(I_o\right)\right\|_1\right]\end{aligned} \tag{12}$$

where M is $D_{s,c}$'s layer number and $D_{s,c}^{(l)}(\cdot)$ refers to the feature map in the l-th layer of $D_{s,c}$.

Style Contrast Loss. Since a character can appear in various styles, learning a unified character-level representation is essential [28]. Hence, we introduce a style contrastive loss to pull intra-class styles closer and push inter-class styles apart in the embedding space by:

$$\mathcal{L}_{cst} = -log\left(\frac{exp\left(\sum_{u=1}^{U} F_c^{u\top} F_{c+}^{u}\right)}{exp\left(\sum_{u=1}^{U} F_c^{u\top} F_{c+}^{u}\right)+\sum_{U_c^-} exp\left(\sum_{u=1}^{U} F_c^{u\top} F_{c-}^{u}\right)}\right) \tag{13}$$

For a given font codebook F_c, F_{c+} represents the positive pairs with the same style but different content, and F_{c-} denotes the negative pairs with the same content but different styles. Besides, U_c^- stands for the number of negative style samples.

Skeleton Consistency Loss. To reinforce structural continuity between the generated and target images, we bring in a skeleton consistency loss based on their morphological skeletons. By virtue of a classical index-table-based thinning algorithm [34], we extract precise skeletons and compute the difference between their normalized forms as follows:

$$\mathcal{L}_{ske} = \mathbb{E}_{I_t\sim p_{data}}\left[\frac{1}{hw}\sum_{i,j}||\varphi(I_t)_{ij}-\varphi(I_o)_{ij}||_1\right] \tag{14}$$

where $\varphi(\cdot)$ is the skeleton extraction algorithm, h and w denote the image height and width.

Edge Consistency Loss. To preserve sharper character edges, we propose an edge consistency loss function. Specifically, we adopt an adaptive edge detection strategy based on the Canny operator [15] to capture the edges of each character, which can keep the effectiveness of the traditional Canny operator while better restraining spurious edges, as expressed below:

$$\mathcal{L}_{edge} = \mathbb{E}_{I_t\sim p_{data}}\left[||\phi\left(I_t\right)-\phi\left(I_o\right)||_1\right] \tag{15}$$

where $\phi(\cdot)$ represents the corresponding edge detection strategy.

Overall Objective Loss. Finally, we optimize GC-Font by the following objective function:

$$\min_{G}\max_{D}(\mathcal{L}_{adv}^{D}+\mathcal{L}_{adv}^{G}+\lambda_1\mathcal{L}_{img}+\lambda_2\mathcal{L}_{feat}+\lambda_3\mathcal{L}_{cst}+\lambda_4\mathcal{L}_{ske}+\lambda_5\mathcal{L}_{edge}) \tag{16}$$

Here, $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ and λ_5 are the five weighting hyper-parameters. In our experiments, we empirically set them to 1, 1, 0.1, 0.5 and 0.5, respectively.

Dataset	Method	SSIM↑	RMSE↓	LPIPS↓	FID↓	L1↓	User study↑
UFUC	MX-Font [20]	0.6966	0.3159	0.2359	62.7500	0.1235	9.55%
	DG-Font [29]	0.6527	0.3238	0.2058	61.6022	0.1241	9.85%
	LF-Font [19]	0.6768	0.3110	0.2516	66.8840	0.1190	9.70%
	CF-Font [26]	0.6613	0.3102	0.2014	59.7644	0.1169	11.20%
	VQ-Font [17]	0.6776	0.3066	0.2157	58.2632	0.1175	13.15%
	FontDiffuser [30]	0.6390	0.3579	0.2816	79.4458	0.1530	10.65%
	IF-Font [3]	0.6891	0.3130	0.2173	59.6292	0.1038	14.50%
	GC-Font (Ours)	**0.7230**	**0.2748**	**0.1711**	**56.6042**	**0.0921**	**21.40%**
SFUC	MX-Font [20]	0.6733	0.3314	0.2197	61.0583	0.1321	8.90%
	DG-Font [29]	0.6478	0.3278	0.2006	58.3525	0.1277	9.80%
	LF-Font [19]	0.6140	0.3335	0.2726	69.2239	0.1520	9.35%
	CF-Font [26]	0.6569	0.3186	0.1974	57.9868	0.1202	11.65%
	VQ-Font [17]	0.6414	0.3304	0.2069	56.7299	0.1305	12.35%
	FontDiffuser [30]	0.6317	0.3691	0.2910	73.3052	0.1574	10.95%
	IF-Font [3]	0.6652	0.3280	0.2221	62.8246	0.1213	14.20%
	GC-Font (Ours)	**0.7166**	**0.2816**	**0.1781**	**54.5017**	**0.1142**	**22.80%**

Table 1: Quantitative comparison results on UFUC and SFUC datasets.

4 Experiments

4.1 Dataset and Evaluation Metrics

Dataset. We collect a Chinese font dataset with 575 fonts, each containing 3500 Chinese characters. Notably, the font *kai* is fixed as the content font throughout training and testing. It is also used to pre-train the feature decomposition network for codebook acquisition.

For the training set, we randomly select 550 fonts with 3000 characters per font, forming the Seen Fonts Seen Characters (SFSC) set. Our test set consists of two parts: 24 Unseen Fonts with 500 Unseen Characters per font (UFUC) and 550 Seen Fonts with 500 Unseen Characters per font (SFUC).

Evaluation Metrics. To comprehensively assess the quality of font generation, we utilize the following five metrics, including L_1, Root Mean Square Error (RMSE) [11], and Structural Similarity Index Measure (SSIM) [27], Learned Perceptual Image Patch Similarity (LPIPS) [32] and Frechet Inception Distance (FID) [8]. Furthermore, we also perform a user study to evaluate the subjective quality of generated results. In particular, we randomly select 10 font styles from each test set and 10 characters per font. Then, we invite 20 well-educated volunteers and ask them to choose the best generation result among the comparison methods. Here, all the samples are shuffled to avoid any potential bias.

4.2 Comparison with SOTA Methods

To verify the effectiveness of GC-Font, we compare it with seven state-of-the-art (SOTA) methods. To be noticed, for a fair comparison, we retrain all these models using their default settings on our training set.

The quantitative comparison results are presented in Tab. 1. It is evident that our GC-Font model achieves the best performance across all the evaluation metrics. Fig. 3 displays the qualitative comparison results. From the figure, we can observe that although MX-Font and LF-Font could maintain the overall character shapes, their outputs often appear blurry with indistinct textures, resulting in a loss of clarity and sharpness. Besides, DG-Font also struggles to capture fine-grained details in complex glyphs, which leads to incomplete com-

Figure 3: Qualitative comparison results on UFUC and SFUC datasets. Characters in blue boxes exhibit stroke errors (including missing or redundant strokes), red boxes indicate noticeable blurring, and purple boxes denote distorted glyph shapes. Zoom in to see details.

Block		SSIM↑	RMSE↓	LPIPS↓	FID↓
GCFM	AdaConv				
✗	✗	0.6705	0.3107	0.2086	60.7358
✗	✓	0.6829	0.2993	0.1895	59.3147
✓	✗	0.7168	0.2802	0.1964	57.8913
✓	✓	**0.7230**	**0.2748**	**0.1711**	**56.6042**

Table 2: Effectiveness of GCFM and AdaConv.

Loss function		SSIM↑	RMSE↓	LPIPS↓	FID↓
$\mathcal{L}_{ske}$	$\mathcal{L}_{edge}$				
✗	✗	0.6842	0.3044	0.2045	57.9573
✗	✓	0.6987	0.2958	0.1823	57.3486
✓	✗	0.7123	0.2815	0.1789	56.8921
✓	✓	**0.7230**	**0.2748**	**0.1711**	**56.6042**

Table 3: Effectiveness of $\mathcal{L}_{ske}$ and $\mathcal{L}_{edge}$.

ponents. While CF-Font generally preserves the correct glyph layouts, it usually exhibits noticeable artifacts and style inconsistencies. VQ-Font demonstrates good stability but lacks effective control over local features, resulting in glyph distortions and vague strokes. The performances of FontDiffuser and IF-Font are also outstanding, but their generated results sometimes suffer from obvious stroke errors. As a whole, characters generated by our GC-Font are of high quality in terms of style consistency, global structures and local details. More generation results can be found in the Supplementary Materials.

4.3 Ablation Study

In this section, we carry out a series of ablation experiments to assess the influence of each part in our model. These experiments are performed on the UFUC dataset. More analyses and results can be found in the Supplementary Materials.

The Effect of GCFM Module and AdaConv. To investigate the impacts of the GCFM module and adaptive convolution, we conduct ablation experiments by selectively enabling each block. As illustrated in Tab. 2, activating either block individually leads to prominent gains. When both blocks are included, the model achieves optimal results, highlighting the synergy between global context refinement and low-level content feature adaptation.

The Effect of Different Loss Functions. To validate the effectiveness of our introduced loss functions, we carry out ablation experiments on their different combinations. As shown in Tab. 3, incorporating either $\mathcal{L}_{ske}$ or $\mathcal{L}_{edge}$ alone could bring in varying degrees of improvement. Moreover, combining both the loss functions yields the best performance, indicating their complementary roles in optimizing local minutiae and geometric alignments.

5 Conclusion

In this paper, we propose GC-Font, a novel few-shot font generation framework to produce higher-quality font libraries. By incorporating a Global Contextual Feature Modelling (GCFM) module and adaptive convolutions, our method effectively enhances both the global structural integrity and stylistic consistency of the generated results. Additionally, the integration of skeleton and edge consistency losses further facilitates topological fidelity. Both quantitative and qualitative experimental results demonstrate that our proposed model exceeds other competitive methods on various fonts and characters.

In the future, we plan to extend our framework to multilingual font generation, enabling adaptation to diverse script systems beyond Chinese. Furthermore, exploring its applicability in personalized font creation scenarios may also yield interesting insights.

6 Acknowledgments

This work was partially supported by Project Funded by the Priority Academic Program Development of Jiangsu Higher Education Institutions, Postgraduate Research & Practice Innovation Program of Jiangsu Province KYCX24_3320, and National Natural Science Foundation of China (NSFC Grant No. 62376041).

References

[1] Prashanth Chandran, Gaspard Zoss, Paulo F. U. Gotardo, and et al. Adaptive Convolutions for Structure-Aware Style Transfer. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[2] Weiran Chen, Chunping Liu, and Yi Ji. Chinese Character Style Transfer Model Based on Convolutional Neural Network. In *Proceedings of the 31st International Conference on Artificial Neural Networks*, 2022.

[3] Xinping Chen, Xiao Ke, and Wenzhong Guo. IF-Font: Ideographic Description Sequence-Following Font Generation. In *Proceedings of the 38th Annual Conference on Neural Information Processing Systems*, 2024.

[4] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, and et al. ILVR: Conditioning Method for Denoising Diffusion Probabilistic Models. In *Proceedings of IEEE International Conference on Computer Vision*, 2021.

[5] Yue Gao, Yuan Guo an Zhouhui Lian, and et al. Artistic glyph image synthesis via one-stage few-shot learning. *ACM Trans. Graph.*, 38(6):1–12, 2020.

[6] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, and David Warde-Farley et al. Generative Adversarial Nets. In *Proceedings of the 28th Annual Conference on Neural Information Processing System*, 2014.

[7] Haibin He, Xinyuan Chen, Chaoyue Wang, Juhua Liu, and et al. Diff-Font: Diffusion Model for Robust One-Shot Font Generation. *International Journal of Computer Vision*, 132:5372—-5386, 2024.

[8] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, 2017.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Proceedings of the 34th Annual Conference on Neural Information Processing System*, 2020.

[10] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[11] Dulakshi Santhusitha Kumari Karunasingha. Root mean square error or mean absolute error? Use their ratio as well. *Inf. Sci.*, 585:609–629, 2022.

[12] Juyeon Ko, Inho Kong, Dogyun Park, and Hyunwoo J. Kim. Stochastic Conditional Diffusion Models for Robust Semantic Image synthesis. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

[13] Yuxin Kong, Canjie Luo, Weihong Ma, and et al. Look Closer to Supervise Better: One-Shot Font Generation via Component-Based Discriminator. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[14] Doyup Lee, Chiheon Kim, Saehoon Kim, and et al. Autoregressive Image Generation using Residual Quantization. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[15] Ersen Li, Baoming Zhang, Xiaoming Zhou, Haitao Guo, and et al. Study on the self-adaptive Canny edge detection algorithm. *Science of Surveying and Mapping*, 33(6): 119–121, 2008.

[16] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and et al. Few-Shot Unsupervised Image-to-Image Translation. In *Proceedings of IEEE International Conference on Computer Vision*, 2019.

[17] Wei Pan, Anna Zhu, Xinyu Zhou, and et al. Few shot font generation via transferring similarity guided global style and quantization local style. In *Proceedings of IEEE International Conference on Computer Vision*, 2023.

[18] Xuran Pan, Chunjiang Ge, Rui Lu, Shiji Song, Guanfu Chen, Zeyi Huang, and et al. On the Integration of Self-Attention and Convolution. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[19] 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 35th AAAI Conference on Artificial Intelligence*, 2021.

[20] 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 IEEE International Conference on Computer Vision*, 2021.

[21] Fabio Pizzati, Jean-François Lalonde, and Raoul de Charette. ManiFest: Manifold Deformation for Few-Shot Image Translation. In *Proceedings of the 18th European Conference on Computer Vision*, 2022.

[22] Ali Razavi, Aäron van den Oord, and Oriol Vinyals. Generating Diverse High-Fidelity Images with VQ-VAE-2. In *Proceedings of the 32nd Annual Conference on Neural Information Processing Systems*, 2019.

[23] Anima Singh, Trung Vu, Nikhil Mehta, and et al. Better Generalization with Semantic IDs: A Case Study in Ranking for Recommendations. In *Proceedings of ACM Conference on Recommender Systems*, 2024.

[24] Licheng Tang, Yiyang Cai, Jiaming Liu, Zhibin Hong, and et al. Few-Shot Font Generation by Learning Fine-Grained Local Styles. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, and et al. Attention is All you Need. In *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, 2017.

[26] Chi Wang, Min Zhou, Tiezheng Ge, Yuning Jiang, and et al. CF-Font: Content Fusion for Few-Shot Font Generation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2023.

[27] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Trans. Image Process.*, 13(4):600–612, 2004.

[28] Xudong Xie, Ling Fu, Zhifei Zhang, Zhaowen Wang, and Xiang Bai. Toward Understanding WordArt: Corner-Guided Transformer for Scene Text Recognition. In *Proceedings of the 18th European Conference on Computer Vision*, 2022.

[29] Yangchen Xie, Xinyuan Chen, Li Sun, and Yue Lu. DG-Font: Deformable Generative Networks for Unsupervised Font Generation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[30] Zhenhua Yang, Dezhi Peng, Yuxin Kong, Yuyi Zhang, and et al. FontDiffuser: One-Shot Font Generation via Denoising Diffusion with Multi-Scale Content Aggregation and Style Contrastive Learning. In *Proceedings of the 38th AAAI Conference on Artificial Intelligence*, 2024.

[31] Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. Self-Attention Generative Adversarial Networks. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[32] 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 IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[33] Yexun Zhang, Ya Zhang, and Wenbin Cai. Separating Style and Content for Generalized Style Transfer. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[34] Jie Zhou, Yefei Wang, Yiyang Yuan, and et al. SGCE-Font: Skeleton Guided Channel Expansion for Chinese Font Generation. *CoRR*, pages 1–12, 2022.

[35] Anna Zhu, Xiongbo Lu, Xiang Bai, and et al. Few-Shot Text Style Transfer via Deep Feature Similarity. *IEEE Trans. Image Process.*, 29:6932–6946, 2020.

[36] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *Proceedings of IEEE International Conference on Computer Vision*, 2017.