Elegantly Written: Disentangling Writer and Character Styles for Enhancing Online Chinese Handwriting

Yu Liu[1,2], Fatimah binti Khalid[1], Lei Wang[1], Youxi Zhang[1], and Cunrui Wang[2]

[1] Faculty of Computer Science and Information Technology, University Putra Malaysia, 43400 UPM Serdang, Malaysia
{gs64481,fatimahk,gs68180,gs64680}@upm.edu.my
[2] Dalian Chinese Font Design Technology Innovation Center, Dalian Minzu University, 116600 Dalian, China
wcr@dlnu.edu.cn

Abstract. The electronic writing tools, while enhancing convenience, sacrifice the readability and efficiency of handwritten content. Balancing high efficiency with readable handwriting poses a challenging research task. In this paper, we propose a method sequence-based models to beautify user handwritten traces. Unlike most existing methods that treat Chinese handwriting as images and cannot reflect the human writing process, we capture individual writing characteristics from a small amount of user handwriting trajectories and beautify the user's traces by mimicking their writing style and process. We fully consider the style of radicals and components between the content and reference glyphs, assigning appropriate fine-grained styles to strokes in the content glyphs through a cross-attention mechanism module. Additionally, we find that many style features contribute minimally to the final stylized results. Therefore, we decompose the style features into the Cartesian product of single-dimensional variable sets, effectively removing redundant features with limited impact on the stylization effect while preserving key style information. Qualitative and quantitative experiments both demonstrate the superiority of our approach.

Keywords: Handwriting font optimization · Few-Shot generation · Calligraphy imitation · Vector quantization· Chinese character structure

1 Introduction

Handwritten traces are a unique way of personal expression and information transmission, with each person's writing style possessing irreplicable personal characteristics. With the popularization of digital tools, people not only pursue writing efficiency but also hope for clear and aesthetically pleasing results. Therefore, handwriting optimization becomes crucial, which is of practical significance for improving the user experience of modern electronic writing devices.

Font generation is considered as an image-to-image translation task. Some methods generate raster font images containing 9,169 characters in an end-to-end manner [21, 41, 42]. Additionally, some studies use unsupervised learning to learn separate representations for Chinese font style and content, generating raster font images for specific style-content combinations [18,23,33,38,47]. However, existing methods treat characters as images, often resulting in issues such as incorrect stroke connections and structural errors. Furthermore, some methods based on sequence generation models treat vector glyphs as sequences of drawing commands and use RNN [43] or LSTM [10] to encode and decode sequences. Although these methods focus on printing or typesetting fonts, they demonstrate great potential in gradually generating vector glyph drawing parameters, providing a new approach for optimizing handwritten traces in a sequence-based manner.

Unlike printed or typeset fonts, handwritten fonts contain natural irregularities of human writing, posing unique challenges. In recent years, although many handwriting generation technologies have been developed for alphabetic languages [6, 15, 28], the large number of Chinese characters and their complex structural and shape diversity pose additional challenges to handwriting generation. Most models treat handwritten characters as images for processing. However, handwritten traces typically contain more information [7, 25], such as stroke coherence, inclination, and stroke spacing. As shown in Figure 1, humans usually draw characters in a predetermined order step by step, rather than 'generating the image at once. Therefore, modeling handwritten traces using sequence models and representing strokes as continuous writing trajectories is a highly effective strategy [5, 20, 31, 44]. However, these studies randomly select one or more characters as style reference characters during training, limiting the model's ability to accurately extract style features when handling diversified style reference combinations. Additionally, these methods adopt decomposition and averaging strategies when handling extracted style features, weakening the style features due to feature separation and averaging processing.

Our objective is to correct and optimize Chinese handwritten traces while retaining text content and learning the writer's style. As shown in Figure 1, unlike raster images of characters, handwritten trajectories better reflect individual writing characteristics due to containing details of the writing process. Therefore, we separate character content and style features from handwritten trajectories and gradually generate handwritten trajectories by combining arbitrary styles with content. As shown in Figure 2, we match the best style reference characters for each character based on the structure of Chinese characters. During stylization, we first decompose style features into the Cartesian product of single-dimensional variable sets to sparsify style features. Then, through a style aggregation module based on cross-attention, the model effectively learns spatial correspondences at the component level between style and content glyphs, assigning the correct fine-grained style to each character. The contributions of this paper are summarized as follows:

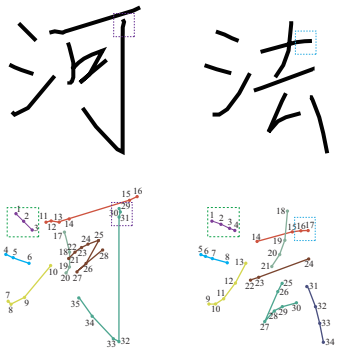

Fig. 1: Illustration of two online handwritten Chinese characters, with each color representing a stroke. The increasing numbers indicate the writing order from the start to the end.

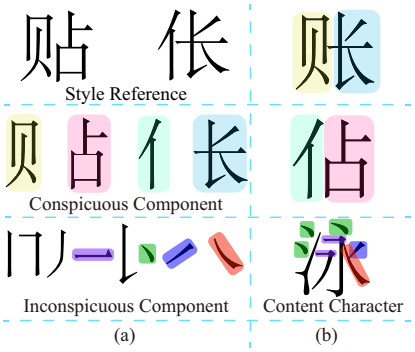

Fig. 2: (a) Structural decomposition of Chinese characters. (b) Example of style reference character selection.

– We propose a novel method to correct and optimize user handwritten traces. Users only need to write a small number of character samples, and the model optimizes user traces by mimicking the user's writing style and process. The code is public at: `https://github.com/ethanliuyu/ElegantlyWritten`
– We match the best style reference characters for each character based on the principle of maximizing Chinese character components and select a relatively small set of characters as the style reference set to allow the model to more effectively learn styles at the component level.
– By using a style aggregation module based on cross-attention, we learn spatial correspondences at the component level between style and content glyphs, transferring details from reference traces to target traces.
– Based on the sparsity of style features, style features are vector quantized into the cartesian product of single-dimensional variable sets, reducing redundant style features while retaining the main style features.

2 Related Work

2.1 Few-shot Font Generation

Font generation is a typical Image-to-image translation task [2, 4, 12, 13, 27], where the model learns the mapping function between the source font and the target font, allowing any character to be transformed into the target style style [8, 11, 14, 21, 36]. Global feature representation treats the entire character as an indivisible whole and directly learns the overall mapping relationship from the source font to the target font [9,33,38,45] . On the other hand, component-based feature representation methods decompose characters into smaller components (e.g., strokes, radicals, etc.) and then learn the styles of these components to

generate characters of the target style [24, 29, 37]. However, these methods tend to perform averaging operations on the style features extracted in the image, leading to significant weakening of local details due to untangling and averaging operations. Therefore, we propose representing handwritten text as a continuous writing trajectory and design a style aggregation module that aims to preserve the intricate details present in the reference handwriting, leveraging the details of the writing process to the fullest extent.

2.2 Generative Vector Graphics Model

Vector fonts precisely define glyphs through straight lines and Bezier curves, providing excellent scalability and editability, and are widely used in the field of digital media and type design. As sequence models like RNN [43], LSTM [10], and Transformer [32] continue to advance, vector fonts are modeled as drawing command sequences. DiffVG [16] makes the vector curve differentiable, establishes the relationship between vector graphics and bitmaps, and becomes the theoretical basis for vector shape generation. Im2vec [26] and DualVector [19] fit glyph contours by adjusting the control points on the contour. DeepVecFont [34] and DeepVecFont-v2 [35] jointly represent the features of font images and sequences, establish the relationship between vector graphics and bitmaps, and use Transformer's ability to model long sequences to generate drawing instructions. Although methods that focus on generating vector drawing parameters for printed or typesetting fonts show great potential for step-by-step generation of vector glyphs, they are largely limited to accurately reconstructing raster glyph images into vector images, without involving learning styles from the handwriting dynamics process.

2.3 Handwriting Generation

Handwritten fonts, characterized by curved lines and varied character sizes compared to printed or typographic fonts, pose additional challenges for handwritten font generation. Some of these approaches treat handwriting as an image, which captures the visual characteristics of handwritten text and thus mimics an individual's writing style to some extent [7, 25]. However, some important dynamic information will be lost. To overcome these limitations, studies have used sequence models to process handwritten handwriting, which not only take into account the visual features of the handwriting, but also include dynamic information during the writing process [1, 5, 30, 46]. However, the stylistic features extracted by these methods are independent of the character structure, which can over-smooth the styles of different characters and lose key details. We fully consider the structure of Chinese characters and capture the styles at the level of the writer's writing trajectory during the writing process, thus significantly improving the performance of handwriting trajectory generation.

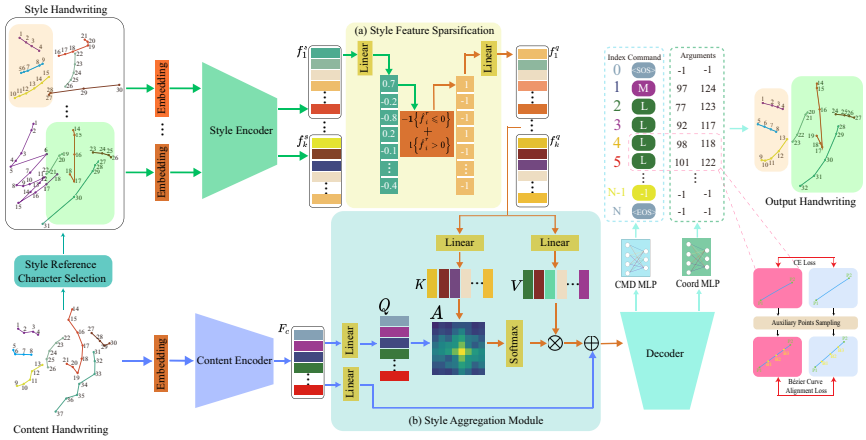

Fig. 3: Overview of the proposed method. The content encoder E_c extracts content features F_c from the input content handwriting trajectory sequence c. The style encoder E_r encodes R_s into reference features $F_s = \{f_i^s\}_{i=1}^K$. Considering the sparsity of style features F_s, discrete style features F_q are obtained through vector quantization. Then, using the style aggregation module based on cross-attention, we learn the spatial correspondence between the style F_c and content F_q at the component level, aggregating the style of the components into $F_{c,r}$. Finally, the decoder D decodes $F_{c,r}$ into handwriting trajectory sequence $\hat{Y}_c$,and optimizes it through cross-entropy loss.

3 Method Description

3.1 Problem Statement and Method Overview

Our goal is to learn the style of the writer from a small number of user handwriting trajectory samples and refine and beautify the writing trajectory. We employ the powerful sequence modeling architecture transformer [32] as our backbone network. Given the handwriting trajectories $C = \{c_i\}_{i=1}^L$ with content i, and K style reference handwriting trajectories $R_s = \{s_i\}_{i=1}^K \in S$ selected based on the component maximization principle, we stylize and optimize the user's handwriting based on the style reference traces. During training, we simultaneously perform style transfer and trace optimization tasks. The style transfer task enables the model to convert handwriting fonts from any user into the style of a specific user, while the trace optimization task focuses on optimizing and reconstructing the current user's handwriting trace.

3.2 Reference Selection

In many font generation methods, at each iteration, one or more characters are randomly selected from the training set as style reference. This makes it challenging for the model to accurately extract styles from diverse combinations of style reference sets. Therefore, we introduce a strategy by setting a fixed style

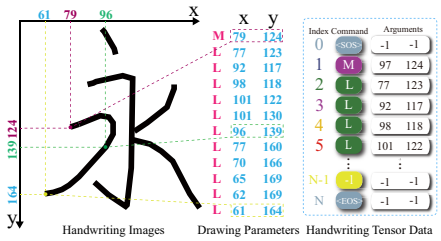

Fig. 4: Graphical parameter data structure visual description.<SOS> and <EOS> represent special markers for the beginning and end of the sequence.-1 represents a padding.

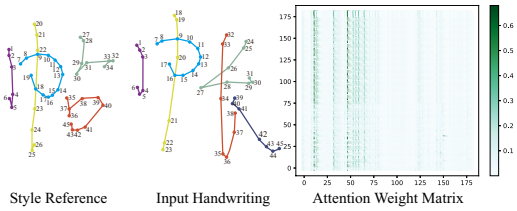

Fig. 5: The sparsity of the attention weight matrix reveals a key phenomenon: not all style features are equally important for the stylization result.

reference set S, composed of 500 characters. Additionally, we design a content-style reference mapping to select a fixed K style reference characters for each content character (see Appendix A.1 for more details). An example of style reference character selection is shown in Figure 2 .

3.3 Data Structure and Feature Embedding

We treat the user's handwriting trajectory as a vector image drawn by Bezier curves. Specifically, the instruction M (MoveTo) indicates when the user's pen leaves the writing surface and moves to a new position in the air, while the instruction L (LineTo) represents the action of the user's pen touching down and moving on the paper.

To enable parallel processing, we convert the non-structural drawing parameters into structural tensor data. Each handwriting trace consists of N drawing parameters V. Each drawing parameter $v_i = (h_i, p_i)$ consists of its drawing instruction type $h_i \in \{< \text{SOS} >, \text{M}, \text{L}, < \text{EOS} >\}$ and drawing coordinates $p_i = (x, y)$. An example of a handwriting vector image and tensor is shown in Figure 4 (see Appendix A.2 for more details).

Handwriting traces are discrete data, and we preprocess them using a feature embedding method similar to natural language processing. For each drawing parameter sequence V of a handwriting trace, we project it into a continuous embedding space $e \in \mathbb{R}^{N \times d}$ (see Appendix A.3 for more details).

3.4 Style and Content Encoder

As shown in Figure 3. The content encoder consists of six layers of Transformer modules with multi-head self-attention. For a content vector glyph V_c consisting of N drawing parameters, its feature embedding results in an embedding matrix $e_c \in \mathbb{R}^{d \times N}$. This embedding matrix is input to the sequence encoder to obtain the content feature $F_c = E_c\,(e_c)$.

For the K style reference glyphs V_s of the content character c, they are first embedded to obtain the embedding matrix $e_s \in \mathbb{R}^{d\times N\times K}$. This embedding matrix is then input to the style encoder, which consists of six layers of Transformer modules with multi-head self-attention, to extract the style feature $F_s = E_s\left(e_s\right)$.

3.5 Style Feature Sparsification

As shown in Figure 5, the sparsity of the attention weight matrix reveals a key phenomenon: not all style features are equally important for the stylization result. This implies that feature compression can be used to simplify the model [39,40]. With this sparsity, we adopt a method that decomposes the multi-dimensional space into a series of single dimensions.

Specifically, we define a latent space $\mathbb{Z} = \times_{i=1}^{\log_2 P} Z_i$, mapping the style reference features to this latent space. The latent space $\mathbb{Z}$ is constructed from the Cartesian product of sets of single-dimensional variables Z_i. Each Z_i is a set of single-dimensional variables. In information theory, $\log_2$ represents the two states of a binary system, and $\log_2 P$ represents how many single-dimensional variables Z_i are needed to represent P different states. Therefore, $\log_2 P$ represents the number of Z_i needed to construct $\mathbb{Z}$.

Given a style reference feature $f^s \in \mathbb{R}^{N\times d}$, its dimensionality is first reduced to $\hat{f}^s \in \mathbb{R}^{N\times \log_2 P}$ using a linear layer $L_q \in \mathbb{R}^{d\times \log_2 P}$. For any $\hat{f}^r = \{\hat{f}^s\}_{i=1}^{N}$, its quantized representation $q(\hat{f}^r)$ is obtained as follows:

$$q\left(\hat{f}_i^r\right) = Z_{i,j}, \text{ where } j = \arg\min_p \left\| \hat{f}_i^r - Z_{i,p} \right\|, \tag{1}$$

where, $Z_{i,j}$ is the j-th value in Z_i. When the elements of Z_i only contain -1 and 1, the quantization process can be simplified using the sign function. Specifically, for each feature $\hat{f}_i^r$, the sign (positive or negative) of $\hat{f}_i^r$ is used to determine whether to choose -1 or 1.

$$q\left(\hat{f}_i^r\right) = \operatorname{sign}\left(\hat{f}_i^r\right) = -\mathbb{1}\left\{\hat{f}_i^r \leqslant 0\right\} + \mathbb{1}\left\{\hat{f}_i^r > 0\right\}. \tag{2}$$

During training, we add entropy loss to encourage the model to output confident predictions,

$$\mathcal{L}_{\text{entropy}} = \mathbb{E}[H(q(\hat{f}^r))] - H[\mathbb{E}(q(\hat{f}^r))], \tag{3}$$

where $H(q(\hat{f}^r)) = \sum_{i=1}^{\log_2 P} H\left(q\left(\hat{f}_i^r\right)\right)$, $H(\cdot)$ denotes entropy. The first term $\mathbb{E}[H(q(\hat{f}^r))]$ encourages uncertainty in the quantized output to promote confident predictions. The second term $H[\mathbb{E}(q(\hat{f}^r))]$ encourages the model to utilize the codebook more comprehensively during the quantization process.

Additionally, commitment loss is added to reduce the gap between the encoder output and the quantized vector, thus improving the accuracy of the quantization process,

$$\mathcal{L}_{\text{commit}} = \left\| \hat{f}^r - q\left(\hat{f}^r\right) \right\|_2^2. \tag{4}$$

During inference, the index of each style reference feature is stored, allowing the style encoder computations to be skipped for efficiency. The token index t of $q(\hat{f}^r)$ is obtained by the following formula:

$$\text{Index}(\hat{f}^r) = \sum_{i=1}^{\log_2 P} \arg\min_p \left\| \hat{f}_i^r - Z_{i,p} \right\| \prod_{b=0}^{i-1} |Z_b| = \sum_{i=1}^{\log_2 P} 2^{i-1} \mathbb{1}\left\{ \hat{f}_i^r > 0 \right\}. \quad (5)$$

During inference, features are reconstructed based on the token index t,

$$q\left(\hat{f}_i^r\right) = \mathbb{1}\left\{ \left\lfloor \frac{t}{2^i} \right\rfloor \bmod 2 > 0 \right\}, \quad (6)$$

where $\left\lfloor \frac{t}{2^i} \right\rfloor$ represents the floor division of $\frac{t}{2^i}$, and $\bmod\ 2$ denotes the modulo operation. The style feature after vector quantization, $q\left(\hat{f}^s\right)$, is transformed back to the original dimensionality $F_s \in \mathbb{R}^{N \times d}$ using a linear layer $L_r \in \mathbb{R}^{\log_2 P \times d}$.

3.6 Style Aggregation Module

To aggregate style from K characters in the style reference set and incorporate it into the fine-grained style representation of a given content image, we employ M cross-attention blocks with M heads. We learn a Query map Q^m from the content feature F_c and a Key map K^m from the style feature F_s, resulting in a spatial correspondence matrix A^m between Q^m and K^m. This matrix A^m is then multiplied with the Value map V^m to aggregate local styles into the target style S^m.

For the m-th head, the content feature map F_c is first passed through a linear layer $L_q^m \in \mathbb{R}^{d \times d^m}$ to obtain the query matrix $Q^m \in \mathbb{R}^{d^m \times N}$. For the K style reference characters' features $F_q \in \mathbb{R}^{d \times N \times K}$, they are first reshaped to $\hat{F}_s \in \mathbb{R}^{d \times (K \times N)}$ and then passed through two linear layers $L_k^m, L_v^m \in \mathbb{R}^{d \times d^m}$ to obtain the key map K^m and the value map V^m.

Next, we compute the spatial correspondence matrix A^m, where each element $A^m(u, v)$ represents the pairwise feature correspondence between position u in the content features and position v in the reference features

$$A^m = \frac{Q^{m\top} K^m}{\sqrt{d^m}} \in \mathbb{R}^{N \times (K \times N)}, \quad (7)$$

where $1/\sqrt{d^m}$ is used to prevent the dot product from growing too large. By applying scaled dot-product attention, we obtain the aggregated style S^m

$$S^m = \text{softmax}\left(A^m\right) V^{m\top}, \quad (8)$$

where $S^m \in \mathbb{R}^{d^m \times d^m}$. After obtaining S^m for each head, they are concatenated along d^m and passed through a linear layer $L_s \in \mathbb{R}^{(d^m \times M) \times d}$ to get the aggregated style feature $S \in \mathbb{R}^{d \times N}$.

The content feature F_c is transformed to $\hat{F}_c$ through a linear layer, and then combined with the aggregated style feature S through another linear layer $F_{cs} = L(\hat{F}_c + S)$ to produce a feature F_{cs} that contains both content information and incorporates stylistic characteristics.

3.7 Decoder and Loss Functions

The aggregated style feature F_{cs} and the target handwriting trajectory feature Y are passed through a decoder consisting of six layers of Transformer modules with multi-head self-attention to obtain $\hat{F} \in \mathbb{R}^{d \times N}$,

$$\hat{F} = D\left(F_{cs}, F\right). \tag{9}$$

The predicted drawing instruction types $\hat{H} = \left[\hat{h_0}, \hat{h_1}, \ldots, \hat{h_N}\right]$ and drawing coordinate parameters $\hat{P} = [\hat{p_0}, \hat{p_1}, \ldots, \hat{p_N}]$ are then obtained through two MLP layers,

$$\mathit{\Gamma} = F_{mlp}(\hat{F}), \tag{10}$$

$$\hat{Y_c} = \left(\hat{H}, \hat{P}\right) \sim \mathit{\Gamma}. \tag{11}$$

During the training stage, the loss between the input drawing parameters and the generated drawing parameters is calculated, and the model is optimized by minimizing the difference between them. The loss between the generated drawing parameters and the corresponding ground truth is defined as:

$$\mathcal{L}_{rec} = \sum_{i=1}^{N} \lambda_{cmd} \ell_{CE}\left(h_i, \hat{h}_i\right) + \lambda_{coord} \ell_{CE}\left(p_i, \hat{p}_i\right), \tag{12}$$

where ℓ_{CE} denotes the cross-entropy loss, λ_{cmd} represents the loss weight for predicting command types, and λ_{coord} represents the loss weight for control point coordinate parameters. Unused parameters are masked.

We find that supervising the alignment of Bezier curves solely based on their control points is not sufficient. Therefore, we extract more points during each Bezier curve segment. For two control points p_0 and p_1, a Bezier curve of degree one is expressed as $B = (1 - r)p_0 + rp_1$. The Bezier curve alignment loss can be calculated as:

$$\mathcal{L}_{\text{bezier}} = \sum_{r \in r_p} \left(\hat{B}(r) - B(r)\right)^2, \tag{13}$$

where $r_p = \{0.25, 0.5, 0.75\}$ represents the parameters of the sampled auxiliary points. The overall training objective of this paper combines all four loss functions,

$$\mathcal{L}_{loss} = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{entropy}\mathcal{L}_{entropy} + \lambda_{commit}\mathcal{L}_{commit} + \lambda_{\text{bezier}}\mathcal{L}_{\text{bezier}}. \tag{14}$$

4 Experiments

4.1 Datasets and Evaluation Metrics

Handwriting dataset. We utilize the CASIA-OLHWDB (1.0-1.2) dataset [17] for model training and testing. The training set comprises approximately 3.7 million online handwriting Chinese characters from 1,020 writers. The test set contains characters from the 60 writers, each providing 3,755 of the most commonly used GB2312-80 characters (see Appendix A.4 for more details).
Data augmentation. Data augmentation is performed using two methods: interpolation and translation. Handwriting traces are considered as first-degree Bezier curves, with each segment having two control points p_0 and p_1. The segment of the handwriting trace is represented by the Bezier curve formula $B(t) = (1-t)p_0 + tp_1$. By fixing the control point p_0 and selecting a value $t \in \{0.9, 1.0, 1.1\}$, we calculate the value of $B(t)$ to replace p_1. This method slightly adjusts the end position of the original trajectory to produce a slightly changed writing trajectory.

In addition, we introduce offsets to the x and y coordinates of points p in the handwriting trace V. Specifically, we add an offset Δ_x to the x coordinate, where $\Delta_x \in [-5, 5]$. Similarly, we add offsets to the y coordinates. By adding these offsets to the drawing coordinates, the vector image of the handwriting trace is translated horizontally and vertically, thereby augmenting the data.
Evaluation metrics. The generated handwriting trajectory sequences are variable-length sequences. We use Dynamic Time Warping (DTW) [3] elastic matching technology to calculate the distance between generated and real handwriting trajectory sequences , allowing for nonlinear alignment. Content and style scores of the generated traces are evaluated using content recognizers and style recognizers, respectively [30]. Additionally, a user preference study is conducted to quantify the subjective quality of the output characters.

4.2 Comparison with Other Methods

We compare our proposed method with state-of-the-art handwriting trace generation methods, including Drawing [44], FontRNN [31], and WriteLikeYou [30], which model handwriting traces as sequential data. DeepImitator [46] and SDT [5] transform handwriting traces into images to generate stylized handwriting trace sequences. To ensure consistency among the compared methods, improvements are made to the Drawing [44] and FontRNN [31] models by adding a style encoder, enabling them to generate characters in arbitrary styles.

Qualitative comparison. The results generated by different methods are visualized, as shown in Figure 6. Since FontRNN [31] and Drawing [44] use a single style reference as input, they struggle to accurately identify and reproduce subtle features when the style features in the reference image are not prominent. DeepImitator [46], WriteLikeYou [30] , and SDT [5] learn styles from multiple images, which may reveal different aspects of style features, thus improving

Fig. 6: A qualitative comparison with the state-of-the-art online Chinese trace generation methods is presented. Green boxes indicate the details comparison between target characters and generated characters, while red boxes denote failures in style imitation.

Table 1: Comparisons with state-of-the-art methods on Chinese dataset.

Method	Style Score ↑	Content Score ↑	DTW ↓	User Prefer.(%) ↑
Drawing [44]	28.38	58.52	1.9331	3.03
FontRNN [31]	34.04	62.01	1.8217	6.07
DeepImitator [46]	46.88	81.28	1.6022	10.07
WriteLikeYou [30]	72.21	94.24	1.2623	41.32
SDT [5]	82.23	94.61	1.0089	58.24
Ours	87.50	95.04	0.8814	64.12

the model's understanding and reproduction capability of complex styles. Our proposed method extracts style features from K best style reference characters' handwriting trace sequences, utilizing a style aggregation module based on cross-attention to better leverage spatial correspondence at the character component level.

Quantitative comparison. Table 1 presents the performance of our proposed method compared to the contrasted methods. DeepImitator [46], WriteLikeYou [30], and SDT [5] , which learn style features from multiple images, outperform FontRNN [31] and Drawing [44] on all evaluation metrics. Our proposed method surpasses the contrasted methods, indicating outstanding style imitation performance of the proposed approach. However, human perception is sensitive to minor differences, and individuals may still perceive subtle distinctions between synthesized handwriting and real handwriting (see Appendix A.7 for more details).

Fig. 7: Generation results of distorted handwriting characters corrected by content-reference mapping.

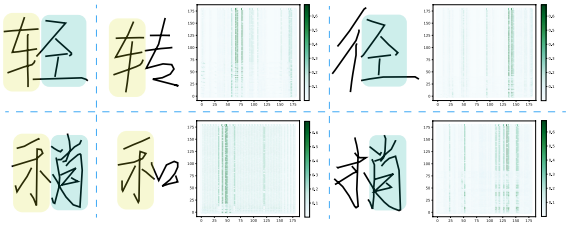

Fig. 8: Visualization of the style aggregation module. The bright spots in the figure indicate significant contributions of corresponding features in the reference feature map.

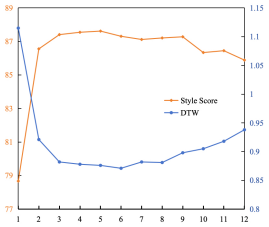

Fig. 9: Effect of the number of style reference characters on model performance.

4.3 Analysis

Handwriting correction. We distort handwritten characters to three different degrees to verify the ability of our method to correct handwriting. As depicted in Figure 7, the readability of characters is enhanced, eliminating some irregular distortions present in the original handwriting font while preserving the user's unique writing style (see Appendix A.6 for more details).

Visualization of style aggregation module. To demonstrate the effectiveness of the style aggregation module, attention feature maps are shown in 8. Specifically, given a spatial point q in the content feature map as a query, the relevance of the corresponding style feature from the spatial correspondence matrix A, denoted as A_q, is obtained, and attention maps are constructed through visualization. The proposed style aggregation module enables the model to focus on the correct component-level styles in style reference characters and extract fine-grained local style features for the handwriting content trace (see Appendix A.5 for more details).

Size of potential space. To assess the influence of the size of the potential space $\mathbb{Z}$ on the stylization of handwriting traces, five different sizes of potential

Table 2: Quantitative evaluation of the influence of the size of the potential space on the stylization of handwriting fonts.

Size	Style Score ↑	DTW ↓
N/A	76. 42	1.2123
2^{10}	80.63	1.0515
2^{11}	83.41	0.9218
2^{12}	87.50	0.8814
2^{13}	88.14	0.8791
2^{14}	88.65	0.8701

Table 3: Comparative quantitative analysis of the impact of fixed versus random selection of style reference characters on style learning capability.

Method	Random style reference characters		Fixed style reference characters	
	Style Score ↑	DTW↓	Style Score ↑	DTW ↓
Drawing [44]	28.18	2.0213	30.21	1.9132
FontRNN [31]	33.21	1.9721	35.68	1.7722
DeepImitator [46]	45.79	1.6180	51.64	1.3545
WriteLikeYou [30]	71.89	1.2911	75.21	1.0891
SDT [5]	81.13	1.1171	84.31	0.9912
Ours	82.34	1.0676	87.41	0.8871

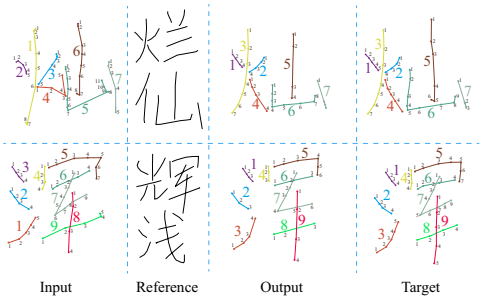

Fig. 10: The effect of incorrect stroke order on the beautification of handwriting traces. Colored numbers represent the sequence of strokes written, while black numbers indicate the writing order of each stroke.

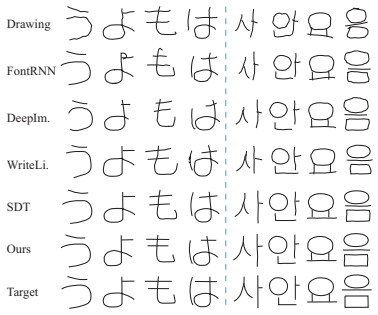

Fig. 11: Qualitative comparison in Japanese and Korean handwriting font generation.

space are set, $P \in \{2^{10}, 2^{11}, 2^{12}, 2^{13}, 2^{14}\}$. This simulates different granularities of feature mapping and corresponding information loss. As shown in Table 2, potential spaces of 2^{10} and 2^{11} levels result in less refined stylization due to significant information loss. In contrast, models in potential spaces of 2^{13} and 2^{14} levels can represent very fine-grained style features. However, compared to 2^{12}, larger potential spaces sacrifice computational efficiency for minimal performance gains. In addition, since the style character set is fixed, storing indexes incurs a lower memory cost than storing feature vectors.

Effect of different number of style inputs. The number of style reference characters K evidently affects the model's performance. As shown in Figure 9, with an increase in the number of input style reference characters, the style score also increases, indicating that the synthesized handwriting encompasses richer style information. However, when the number of style reference characters

is too large, the model struggles to capture consistent style features due to the instability of handwriting traces. Therefore, in other experiments, the number of style reference characters is set to $K = 3$.

Effect of incorrect stroke order. In handwriting recognition and correction techniques, considering the stroke order of Chinese characters is crucial. Ensuring accurate correction of characters even when the user's stroke order is incorrect is essential. We input handwriting with the stroke order shuffled into the model. Experimental results, as shown in Figure 10, demonstrate that even with incorrect stroke order, our method can identify the correct sequence of traces and adjust strokes accordingly to restore highly readable writing.

Effect of different style inputs. To validate the effectiveness of the proposed style-content reference mapping in improving style learning and content preservation capabilities, a quantitative comparison is conducted between fixed style reference characters and randomly selected style reference characters. As shown in Table 3, compared to randomly selected style reference characters, fixed style reference characters perform better in all evaluation metrics. The fixed style reference character set provides the model with a stable and representative set of style samples, facilitating more effective learning of style features by the model.

Applications to other languages. For Japanese handwriting font generation tasks, experiments were conducted on the TUAT HANDS [22] database. A Korean skeleton was extracted as a Korean handwriting sample using a skeleton extraction algorithm. Random style reference characters were chosen for each content character in the experiment. Figure 11 validate the effectiveness of our method in Japanese and Korean handwriting font generation, demonstrating outstanding performance in multilingual scenarios (see Appendix A.8 for more details).

5 Conclusion

In this paper, a novel method is proposed to enhance the handwriting trace of users. Users only need to provide a small amount of character samples, and the model beautifies the user's trace by mimicking their handwriting style and the human writing process. We maximally utilize the spatial correspondence between the components of style and content glyph shapes by adopting a style aggregation module based on cross attention. This transfers the details of the reference trace to the target trace. Additionally, style features are mapped to a latent space constructed by the Cartesian product of single-dimensional variables to achieve sparse processing of style features. Numerous experiments demonstrate that our method outperforms other methods significantly in both objective and subjective similarity aspects. **Negative Impact:** Although it is possible that this method can be used to mimic handwriting, human experts can still find differences between the generated and real handwriting.

Acknowledgements

This research was supported by Dalian Science and Technology Innovation Fund (No.2023JJGX026) and Key Laboratory of Informatization of National Education of Ministry of Education (No.EIN2024B002)

References

1. Aksan, E., Pece, F., Hilliges, O.: Deepwriting: Making digital ink editable via deep generative modeling. In: CHI. pp. 1–14. ACM (2018). https://doi.org/10.1145/3173574.3173779
2. Bhattarai, B., Kim, T.K.: Inducing optimal attribute representations for conditional gans. In: ECCV. pp. 69–85. Springer (2020). https://doi.org/10.1007/978-3-030-58571-6_5
3. Chen, Z., Yang, D., Liang, J., Liu, X., Wang, Y., Peng, Z., Huang, S.: Complex handwriting trajectory recovery: Evaluation metrics and algorithm. In: ACCV. pp. 1060–1076. Springer (2022). https://doi.org/10.1007/978-3-031-26284-5_4
4. Choi, Y., Uh, Y., Yoo, J., Ha, J.W.: Stargan v2: Diverse image synthesis for multiple domains. In: CVPR. pp. 8185–8194. IEEE (2020). https://doi.org/10.1109/CVPR42600.2020.00821
5. Dai, G., Zhang, Y., Wang, Q., Du, Q., Yu, Z., Liu, Z., Huang, S.: Disentangling writer and character styles for handwriting generation. In: CVPR. pp. 5977–5986. IEEE (2023). https://doi.org/10.1109/cvpr52729.2023.00579
6. Fogel, S., Averbuch-Elor, H., Cohen, S., Mazor, S., Litman, R.: Scrabblegan: Semi-supervised varying length handwritten text generation. In: CVPR. pp. 4324–4333. IEEE (2020). https://doi.org/10.1109/cvpr42600.2020.00438
7. Gan, J., Wang, W.: Higan: Handwriting imitation conditioned on arbitrary-length texts and disentangled styles. In: AAAI. pp. 7484–7492. AAAI Press (2021). https://doi.org/10.1609/aaai.v35i9.16917
8. Gao, Y., Wu, J.: Gan-based unpaired chinese character image translation via skeleton transformation and stroke rendering. In: AAAI. pp. 646–653. AAAI Press (2020). https://doi.org/10.1609/aaai.v34i01.5405
9. Gao, Y., Guo, Y., Lian, Z., Tang, Y., Xiao, J.: Artistic glyph image synthesis via one-stage few-shot learning. ACM Transactions on Graphics **38**(6), 1–12 (2019). https://doi.org/10.1145/3355089.3356574
10. Graves, A., Graves, A.: Long short-term memory. Supervised sequence labelling with recurrent neural networks pp. 37–45 (2012)
11. Hassan, A.U., Ahmed, H., Choi, J.: Unpaired font family synthesis using conditional generative adversarial networks. Knowledge-Based Systems **229**, 107304 (2021). https://doi.org/10.1016/j.knosys.2021.107304
12. Jeong, S., Kim, Y., Lee, E., Sohn, K.: Memory-guided unsupervised image-to-image translation. In: CVPR. pp. 6554–6563. IEEE (2021). https://doi.org/10.1109/CVPR46437.2021.00649
13. Jiang, L., Zhang, C., Huang, M., Liu, C., Shi, J., Loy, C.C.: Tsit: A simple and versatile framework for image-to-image translation. In: ECCV. pp. 206–222. Springer (2020). https://doi.org/10.1007/978-3-030-58580-8_13
14. Jiang, Y., Lian, Z., Tang, Y., Xiao, J.: Scfont: Structure-guided chinese font generation via deep stacked networks. In: AAAI. pp. 4015–4022. AAAI Press (2019). https://doi.org/10.1609/aaai.v33i01.33014015

15. Kotani, A., Tellex, S., Tompkin, J.: Generating handwriting via decoupled style descriptors. In: ECCV. pp. 764–780. Springer (2020). https://doi.org/10.1007/978-3-030-58610-2_45
16. Li, T.M., Lukáč, M., Gharbi, M., Ragan-Kelley, J.: Differentiable vector graphics rasterization for editing and learning. TOG **39**(6), 1–15 (2020). https://doi.org/10.1145/3414685.3417871
17. Liu, C.L., Yin, F., Wang, D.H., Wang, Q.F.: Casia online and offline chinese handwriting databases. In: ICDAR. pp. 37–41. IEEE (2011). https://doi.org/10.1109/icdar.2011.17
18. Liu, X., Meng, G., Chang, J., Hu, R., Xiang, S., Pan, C.: Decoupled representation learning for character glyph synthesis. IEEE Transactions on Multimedia **24**, 1787–1799 (2021). https://doi.org/10.1109/tmm.2021.3072449
19. Liu, Y.T., Zhang, Z., Guo, Y.C., Fisher, M., Wang, Z., Zhang, S.H.: Dualvector: Unsupervised vector font synthesis with dual-part representation. In: CVPR. pp. 14193–14202. IEEE (2023). https://doi.org/10.1109/cvpr52729.2023.01364
20. Liu, Y., binti Khalid, F., binti Mustaffa, M.R., bin Azman, A.: Dual-modality learning and transformer-based approach for high-quality vector font generation. Expert Systems with Applications **240**, 122405 (2024). https://doi.org/10.1016/j.eswa.2023.122405
21. Liu, Y., binti Khalid, F., Wang, C., binti Mustaffa, M.R., bin Azman, A.: An end-to-end chinese font generation network with stroke semantics and deformable attention skip-connection. Expert Systems with Applications **237**, 121407 (2024). https://doi.org/10.1016/j.eswa.2023.121407
22. Matsumoto, K., Fukushima, T., Nakagawa, M.: Collection and analysis of on-line handwritten japanese character patterns. In: Proceedings of Sixth International Conference on Document Analysis and Recognition. pp. 496–500. IEEE (2001). https://doi.org/10.1109/ICDAR.2001.953839
23. Pan, W., Zhu, A., Zhou, X., Iwana, B.K., Li, S.: Few shot font generation via transferring similarity guided global style and quantization local style. In: ICCV. pp. 19506–19516 (2023). https://doi.org/10.1109/iccv51070.2023.01787
24. Park, S., Chun, S., Cha, J., Lee, B., Shim, H.: Few-shot font generation with localized style representations and factorization. In: AAAI. pp. 2393–2402. AAAI Press (2021). https://doi.org/10.1609/aaai.v35i3.16340
25. Pippi, V., Cascianelli, S., Cucchiara, R.: Handwritten text generation from visual archetypes. In: CVPR. pp. 22458–22467. IEEE (2023). https://doi.org/10.1109/cvpr52729.2023.02151
26. Reddy, P., Gharbi, M., Lukac, M., Mitra, N.J.: Im2vec: Synthesizing vector graphics without vector supervision. In: CVPRW. pp. 7342–7351. IEEE (2021). https://doi.org/10.1109/cvprw53098.2021.00241
27. Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., Cohen-Or, D.: Encoding in style: a stylegan encoder for image-to-image translation. In: CVPR. pp. 2287–2296. IEEE (2021). https://doi.org/10.1109/CVPR46437.2021.00232
28. Sumi, T., Iwana, B.K., Hayashi, H., Uchida, S.: Modality conversion of handwritten patterns by cross variational autoencoders. In: ICDAR. pp. 407–412. IEEE (2019). https://doi.org/10.1109/icdar.2019.00072
29. Tang, L., Cai, Y., Liu, J., Hong, Z., Gong, M., Fan, M., Han, J., Liu, J., Ding, E., Wang, J.: Few-shot font generation by learning fine-grained local styles. In: CVPR. pp. 7895–7904. IEEE (2022). https://doi.org/10.1109/cvpr52688.2022.00774

30. Tang, S., Lian, Z.: Write like you: Synthesizing your cursive online chinese handwriting via metric-based meta learning. Computer Graphics Forum **40**(2), 141–151 (2021). https://doi.org/10.1111/cgf.142621
31. Tang, S., Xia, Z., Lian, Z., Tang, Y., Xiao, J.: Fontrnn: Generating large-scale chinese fonts via recurrent neural network. Computer Graphics Forum **38**(7), 567–577 (2019). https://doi.org/10.1111/cgf.13861
32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: NeurIPS. vol. 30 (2017)
33. Wang, C., Zhou, M., Ge, T., Jiang, Y., Bao, H., Xu, W.: Cf-font: Content fusion for few-shot font generation. In: CVPR. pp. 1858–1867. IEEE (2023). https://doi.org/10.1109/CVPR52729.2023.00185
34. Wang, Y., Lian, Z.: Deepvecfont: Synthesizing high-quality vector fonts via dual-modality learning. TOG **40**(6), 1–15 (2021). https://doi.org/10.1145/3478513.3480488
35. Wang, Y., Wang, Y., Yu, L., Zhu, Y., Lian, Z.: Deepvecfont-v2: Exploiting transformers to synthesize vector fonts with higher quality. In: CVPR. pp. 18320–18328. IEEE (2023). https://doi.org/10.1109/cvpr52729.2023.01757
36. Wen, C., Pan, Y., Chang, J., Zhang, Y., Chen, S., Wang, Y., Han, M., Tian, Q.: Handwritten chinese font generation with collaborative stroke refinement. In: WACV. pp. 3882–3891. IEEE (2021). https://doi.org/10.1109/WACV48630.2021.00393
37. Wen, Q., Li, S., Han, B., Yuan, Y.: Zigan: Fine-grained chinese calligraphy font generation via a few-shot style transfer approach. In: ACMMM. pp. 621–629. ACM (2021). https://doi.org/10.1145/3474085.3475225
38. Xie, Y., Chen, X., Sun, L., Lu, Y.: Dg-font: Deformable generative networks for unsupervised font generation. In: CVPR. pp. 735–751. IEEE (2021). https://doi.org/10.1109/cvpr46437.2021.00509
39. Yu, L., Cheng, Y., Sohn, K., Lezama, J., Zhang, H., Chang, H., Hauptmann, A.G., Yang, M.H., Hao, Y., Essa, I., et al.: Magvit: Masked generative video transformer. In: CVPR. pp. 10459–10469 (2023). https://doi.org/10.1109/cvpr52729.2023.01008
40. Yu, L., Lezama, J., Gundavarapu, N.B., Versari, L., Sohn, K., Minnen, D., Cheng, Y., Gupta, A., Gu, X., Hauptmann, A.G., et al.: Language model beats diffusion–tokenizer is key to visual generation. arXiv preprint arXiv:2310.05737 (2023). https://doi.org/10.48550/arXiv.2310.05737
41. Zeng, J., Chen, Q., Liu, Y., Wang, M., Yao, Y.: Strokegan: Reducing mode collapse in chinese font generation via stroke encoding. In: AAAI. pp. 3270–3277 (2021). https://doi.org/10.1609/aaai.v35i4.16438
42. Zeng, S., Pan, Z.: An unsupervised font style transfer model based on generative adversarial networks. Multimedia Tools and Applications **81**(4), 5305–5324 (2022). https://doi.org/10.1007/s11042-021-11777-0
43. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR. pp. 586–595. IEEE (2018)
44. Zhang, X.Y., Yin, F., Zhang, Y.M., Liu, C.L., Bengio, Y.: Drawing and recognizing chinese characters with recurrent neural network. PAMI **40**(4), 849–862 (2017). https://doi.org/10.1109/tpami.2017.2695539
45. Zhang, Y., Zhang, Y., Cai, W.: Separating style and content for generalized style transfer. In: CVPR. pp. 8447–8455. IEEE (2018). https://doi.org/10.1109/cvpr.2018.00881

46. Zhao, B., Tao, J., Yang, M., Tian, Z., Fan, C., Bai, Y.: Deep imitator: Handwriting calligraphy imitation via deep attention networks. Pattern Recognition **104**, 107080 (2020). https://doi.org/10.1016/j.patcog.2019.107080
47. Zhu, A., Lu, X., Bai, X., Uchida, S., Iwana, B.K., Xiong, S.: Few-shot text style transfer via deep feature similarity. IEEE Transactions on Image Processing **29**, 6932–6946 (2020). https://doi.org/10.1109/tip.2020.2995062