DISENTANGLING WRITER AND CHARACTER STYLES FOR HANDWRITING GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Training machines for synthesizing diverse handwritings is an intriguing task. Recently, some RNN-based methods are proposed to generate stylized online Chinese characters. But these methods mainly focus on learning a person's overall writing style and hence neglect the detailed style inconsistencies between characters from the same writer. For example, one person's handwritings always appear an overall uniformity (e.g., character slant and aspect ratios) but there are still small style differences between local regions (e.g., stroke length and curvature) of characters. Motivated by this, in this paper, we propose to disentangle the style representations at both writer and character levels from individual handwritings. Specifically, we propose the style-disentangled transformer (SDT), equipped with two complementary contrastive objectives, to extract the overall writer-wise and detailed character-wise style representations, respectively, which boosts the generation quality of online handwritings. Extensive experiments on various language scripts verify the superiority of SDT. Particularly, we empirically find that the two learned style representations provide information with different frequency magnitudes, which demonstrates the necessity of separate style extraction.

1 INTRODUCTION

As the world's oldest writing system, Chinese characters are widely used in many Asian countries. Compared to Latin scripts, Chinese characters comprise an extremely large vocabulary (87,887 characters in GB18030-2022 charset) and have complex structures consisting of multiple strokes. Nowadays, the challenging and interesting Chinese character generation (Tian, 2017; Gao et al., 2019; Liu et al., 2022) has attracted intensive attention. For plausible handwriting synthesis, a promising strategy (Zhang et al., 2017) is to progressively generate online characters (i.e., the handwriting trajectory in a sequential format). As shown in Fig. 1, online characters carry richer information (e.g., the order of writing) and thus have wide application scenarios (e.g., writing robot (Yin et al., 2016)).

Our goal is to automatically generate online Chinese handwritings that both match a certain textual content and imitate the calligraphic style (character slant, shape, stroke length, curvature, etc.) of an exemplar writer. This task thus has many applications, such as font design and calligraphy education. A popular solution (Kang et al., 2020) for this task is to extract style information from the given stylized samples and combine it with the content reference. DeepImitator (Zhao et al., 2020) concatenates the style vector from a CNN encoder with a character embedding, which is then fed into the RNN to generate stylized online characters. Further, WriteLikeYou (Tang & Lian, 2021) modifies the large-margin softmax loss (Wang et al., 2018) to encourage discriminative learning of style features. However, these methods mainly focus on the overall writing style and neglect the detailed style inconsistencies (local regions in Fig. 2) between characters from the same writer.

The above observations motivate us to disentangle style representations at the writer and character levels from the stylized handwritings. However, capturing the two styles accurately is a non-trivial problem. To handle this, we propose a style-disentangled transformer (SDT) with a dual-head style encoder. We further adopt the contrastive learning framework (Hadsell et al., 2006) to guide the two heads to focus on the writer-wise and character-wise style, respectively. Specifically, considering the overall writer-wise style, we treat characters from a writer as positive instances, while characters from other writers are considered as negatives. Hence, the encoder learns the style commonalities between characters written by the same person. For the detailed character-wise style, we then

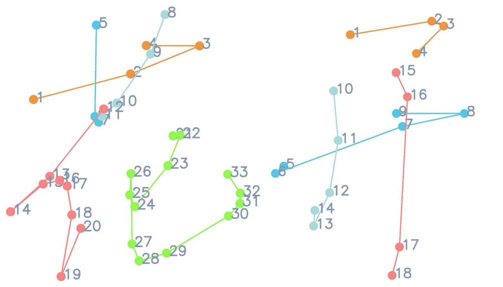

Figure 1: Illustration of two example online handwritten Chinese characters. Each color represents one stroke, and the increasing numbers on each stroke indicate the writing order from the starting to the end.

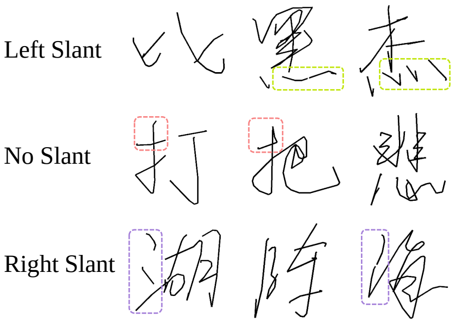

Figure 2: Each row shows handwritings from the same person. Although they have similar overall styles (e.g., character slant), there are still small style differences (e.g., stroke length, location and curvature) between them.

propose a novel patch-wise augmentation scheme, i.e., positive patches are independently sampled within a character while negatives are sampled from other characters. Aggregating the sampling results that represent distinct views of a character forces the encoder to focus on the detailed character style.

Furthermore, we introduce a content encoder to learn a textual feature with the global context. The above two style representations and the textual feature are then fed into a decoder that progressively generates online characters. Since the output characters are in a sequence form, we use Transformer (Vaswani et al., 2017), a powerful sequence modeling architecture, as our backbone. Moreover, our method can be extended to generate offline handwritten Chinese characters (i.e., character images with stroke-width, as shown in Fig. 10). Thus, we outline an offline-to-offline generation framework to improve the generation quality of offline characters. Specifically, we first generate online characters with large shape changes and then decorate them with stroke width, ink-blot etc., thus achieving authentic offline handwritings (see Fig. 10 and more details are put in Appendix A.3).

The key contributions are threefold. 1) We are the first to explore two style representations (i.e., writer-wise and character-wise) that exist in Chinese handwritten characters. Through our experimental analysis, the former contains more low frequencies and the latter mostly concentrates on high frequencies. (2) The proposed offline-to-offline framework narrows the gap towards plausible offline Chinese handwriting synthesis. (3) Extensive experiments on handwriting datasets in Chinese, English, Japanese, and Indic scripts demonstrate the effectiveness and superiority of our SDT.

2 Related Work

Handwriting Generation. Most of the early works are designed to generate Latin characters. Two-step methods (Wang et al., 2002; Lin & Wan, 2007) generate isolated letters, and then concatenate them to produce a whole word. These methods rely on handcrafted rules and only generate handwritings with limited variations. With the rapid development of deep learning, Recurrent Neural Networks (RNNs) and GANs are introduced to generate handwritings in a variety of styles (Graves, 2013; Fogel et al., 2020). After that, some methods (Kang et al., 2020; Kotani et al., 2020; Gan & Wang, 2021) that extract calligraphic styles from stylized samples with controllable styles are proposed. For instance, Kotani et al. (2020) segments the online handwritten word into the isolated letters, and encodes the whole word and each letter into global and letter-specific style vector, respectively, which are then combined with character embeddings for synthesizing stylized handwritten word images. Compared with it, our character-wise styles can explicitly pay attention to more fine-grained local details (e.g., stroke length, location and curvature) of each character. HWT (Bhunia et al., 2021) adopts a vanilla transformer encoder to extract rich style patterns from given style samples. Moreover, these methods rely on complex content references, such as recurrent embeddings and letter-wise filter maps. To address this issue, SLOGAN (Luo et al., 2022) proposes to extract textual content from easily obtainable printed images, but it's impractical to generalize to unseen handwriting styles due to the fixed writer ID. In contrast, our SDT obtains content and style information both from handwriting images and synthesizes characters with arbitrary styles.

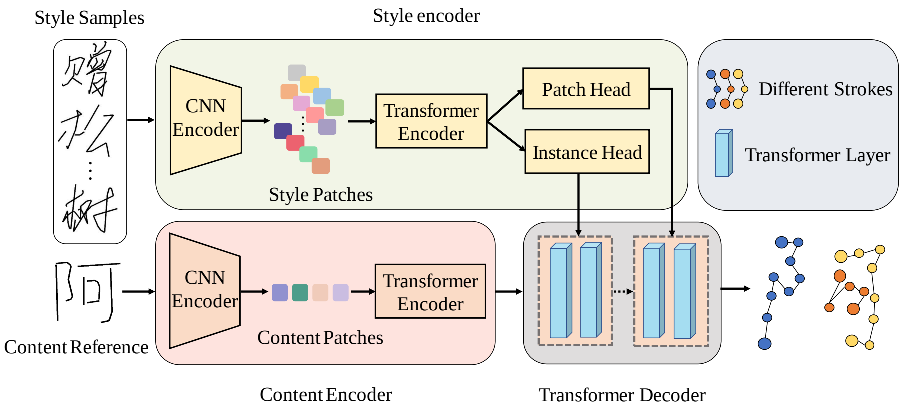

Figure 3: Overview of the proposed method. Our SDT consists of a dual-head style encoder, a content encoder and a transformer decoder. The writer-wise and character-wise style representations extracted by style encoder and the learned content features are fed into the transformer decoder to progressively generate online handwritings.

As for handwritten Chinese characters, some previous methods (Zong & Zhu, 2014; Lin et al., 2015; Lian et al., 2016) extract components (i.e., strokes and radicals) of characters via expert knowledge and then assemble them properly to generate the character. However, these methods rely on hundreds of references, which is labor-intensive. After that, several attempts (Kong & Xu, 2017; Chang et al., 2018) have been made to use GANs to directly generate Chinese handwriting images, but always result in characteristic artifacts. Zhang et al. (2017); Tang et al. (2019) adopt RNNs to generate online Chinese handwritings, but can only transfer to a fixed target style. Recently, Zhao et al. (2020); Tang & Lian (2021) propose to generate online Chinese handwritings with arbitrary styles from a few references. Unlike these methods that only extract an overall writer style, our SDT achieves style representations at both the writer and character levels, which significantly boosts the performance of handwriting imitation.

Contrastive Learning. Contrastive learning (Hadsell et al., 2006) aims to learn a discriminative representation by maximizing the mutual information between the input and output samples, which has been widely used in many fields (Tian et al., 2020; Gao et al., 2021; Ren et al., 2022). Specifically, some image translation works (Park et al., 2020; Han et al., 2021) employ InfoNCE (Oord et al., 2018) to bring together corresponding patches in the input and output, which contributes to retaining the semantic structure during the transfer. However, as mentioned in Zhang et al. (2022), the semantic similarity assumption does not hold for arbitrary style transfer tasks (e.g., stylized handwriting generation), which leads to unsatisfactory style representations.

3 METHOD

Problem statement. We aim to synthesize stylized online handwritings with conditional contents and styles. Let $X_s = \{x_s^i\}_{i=1}^K$ denote a subset of K randomly sampled Chinese handwriting images from a given writer w_s. Given a content image I and a set of style images X_s, our goal is to yield an online handwritten Chinese character $\hat{Y}_s$ presenting the calligraphic style of w_s and maintaining the same textual content with I. The key challenge of this task is to obtain discriminative style representations from limited stylized samples.

3.1 OVERALL SCHEME

According to our observations (see Fig. 2) where both the *overall uniformity* (i.e., writer-wise style) and *inconsistent details* (i.e., character-wise style) exist in individual calligraphy handwritings, we propose two contrastive objectives, namely, InstanceNCE and PatchNCE (Dou et al., 2019). Specifically, the InstanceNCE maximizes the mutual information between character instances belonging to the same writer, while the PatchNCE associates positive patches generated by independently sampling from the same character. Therefore, our style-decoupled transformer (SDT) can disentangle the above two styles from individual handwritings, which boosts the imitation performance.

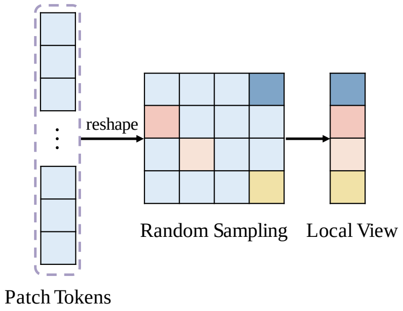

Figure 4: The proposed patch-sampling strategy for constructing the local view of a character. Specifically, we randomly select a small subset (e.g., 25%) of tokens from the patch head, following a uniform distribution.

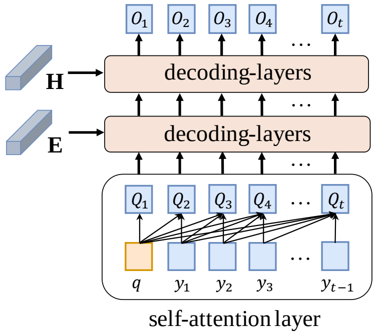

Figure 5: At each time step, the content feature and previous points are encoded to the query vector, which then successively attends to the writer-wise and character-wise style features for predicting the current-step point.

As shown in Fig. 3, the SDT consists of a content encoder, a dual-head style encoder and a transformer decoder. The content encoder uses a standard Resnet18 (He et al., 2016) as the CNN backbone to learn the compact feature map $q_{map} \in \mathbb{R}^{h \times w \times c}$ from a content reference I, and feeds the flattened feature patches into the transformer encoder to extract the textual content representation $q \in \mathbb{R}^{d \times c}$, where $d = h \times w$ and c is the channel dimension. Benefiting from the strong capability of transformer to capture long-range dependencies between feature patches, the content encoder expects an informative q with a global context. Similarly, the style encoder extracts rich calligraphic style patterns from reference style examples X_s via a sequential combination of a CNN and a transformer. Then, the instance and patch heads acquire discriminative writer-wise and character-wise style representations from the extracted style patterns, respectively. After the two encoders, a multi-layer transformer decoder is used to synthesize $\hat{Y}_s$ in an auto-regressive fashion, conditioned on the two style representations and q. We detail the style encoder and transformer decoder as follows.

3.2 Dual-head Style Encoder

As illustrated in Fig. 2, two distinguishing styles (i.e., writer-wise and character-wise) exist in handwritings written by a person. Inspired by this observation, we propose a dual-head style encoder to obtain the two style representations. Firstly, the given $X = \{x^i\}_{i=1}^K$ are encoded by the Resnet18 to obtain a sequence of feature maps $F_m = \{f_m^i\}_{i=1}^K \in \mathbb{R}^{K \times h \times w \times c}$, where we omit the style s of X for the sake of simplicity. Next, we flatten the spatial dimension of each feature map to obtain feature sequences $F = \{f^i\}_{i=1}^K \in \mathbb{R}^{K \times d \times c}$. Then, F are fed into a transformer encoder to extract the informative feature sequences $Z = \{z^i\}_{i=1}^K \in \mathbb{R}^{K \times d \times c}$. As opposed to HWT (Bhunia et al., 2021), where F is concatenated before fed into the transformer encoder, we handle each feature sequence $f \in F$ separately to avoid mutual interference within the set F and reduce the total computational complexity per F. Finally, the instance and patch heads, each of which comprises a standard self-attention layer (Vaswani et al., 2017), are employed to further separate the writer-wise style representations $E = \{e^i\}_{i=1}^K \in \mathbb{R}^{K \times d \times c}$ and the character-wise counterparts $H = \{h^i\}_{i=1}^K \in \mathbb{R}^{K \times d \times c}$ from Z, respectively. We provide the learning objectives of the instance head and patch head below.

3.2.1 Writer-wise Contrastive Learning

To explicitly encourage the instance head to learn the writer-wise style, we aim to map the style representations from the same writer to a similar point in the feature space. The intuition is that one person's handwritings always appear a similar style information, which can be used as an important clue for distinguishing writers. We propose the InstanceNCE to achieve this motivation, following the supervised contrastive framework (Khosla et al., 2020) that extends InfoNCE for multiple positives per anchor. Briefly, in mini-batch data, we take characters written by the same person as positive instances and those from different writers as negative instances. Formally, let $j \in M = \{1, ..., N\}$ be the index of an arbitrary element within a mini-batch and $A(j) = M \backslash \{j\}$ be other indices distinct from j, where N is the batch size. Given a writer-wise style feature e_j belonging to writer w_j as the anchor, we denote its in-batch positives as $P(j) = \{p \in A(j) : w_p = w_j\}$ and negatives as

$A\left(j\right)\backslash P\left(j\right)$. We further formulate the InstanceNCE loss as follows:

$$\mathcal{L}_{ins}= -\frac{1}{N}\sum_{j\in M}\frac{1}{\left|P\left(j\right)\right|}\sum_{p\in P(j)}\log\frac{\exp\left(f\left(e_j\right)^{\top}f(e_p)/\tau\right)}{\sum_{a\in A(j)}\exp\left(f\left(e_j\right)^{\top}f\left(e_a\right)/\tau\right)}, \tag{1}$$

where τ is a temperature parameter and $f\left(\cdot\right)$ is a multi-layer perceptron (MLP) that projects representations to the space that applies InstanceNCE. We assume the output of $f\left(\cdot\right)$ is ℓ_2-normalized.

3.2.2 Character-wise contrastive Learning

Compared with the overall writer-wise style, the character-wise style differences often exist in the stroke details of distinct characters. Inspired by this finding, we aim to maximize the mutual information between diverse local views of a character, which enforces the patch head to learn the detailed character-wise style. Instead of input images, we construct local representations of characters directly over sequential patch tokens from the patch head, as shown in Figure 4. Specifically, since strokes are distributed at arbitrary spatial locations in character images, we propose a sampling strategy to capture the stroke details by randomly selecting a small subset (i.e., 25%) of patches. This strategy samples local regions of a character following a uniform distribution, avoiding potential sampling biases (i.e., certain areas are oversampled). Formally, given the character-wise style representations $\{h_j\}_{j=1}^{B}$ extracted from B characters, we apply the proposed strategy to individually sample two positive local representations $o\in\mathbb{R}^{n\times c}$ and $o^{+}\in\mathbb{R}^{n\times c}$ from the same randomly selected h and $B{-}1$ negatives $\{o_j^{-}\}_{j=1}^{B-1}$ from the remaining $B{-}1$ style features, where $n{=}0.25d$ denotes the number of sampled patch tokens. Finally, the PatchNCE loss can be formulated as:

$$\mathcal{L}_{pat}= -\log\frac{\exp\left(g\left(o\right)^{\top}g(o^{+})/\tau\right)}{\exp\left(g\left(o\right)^{\top}g(o^{+})/\tau\right)+\sum_{j=1}^{B-1}\exp\left(g\left(o\right)^{\top}f\left(o_j^{-}\right)/\tau\right)}, \tag{2}$$

where $g(\cdot)$ is an MLP with the same structure as $f(\cdot)$, but does not share weights with each other.

3.3 Transformer Decoder for Handwritings

The goal of the proposed transformer decoder is to progressively generate the realistic online character $\hat{Y}$ using a few obtained style representations, i.e., $E=\{e^i\}_{i=1}^{K}$, $H=\{h^i\}_{i=1}^{K}$, and a global textual feature q. Generally, the ground-truth (GT) of $\hat{Y}$ is composed of a sequence of points and can be mathematically represented as $Y{=}\left[y_1,...,y_L\right]$, where L is the length of Y. Following Zhang et al. (2017), each point is a vector with 5 elements $y_t{=}\left(\Delta u_t,\Delta v_t,m_t^1,m_t^2,m_t^3\right)$, where $(\Delta u_t,\Delta v_t)$ are the relative offsets from the current point to the previous point and (m_t^1-down, m_t^2-up, m_t^3-end) are three types of pen states, which are mutually exclusive.

At decoding step t, instead of simply following previous RNN-based methods (Ha & Eck, 2018; Tang & Lian, 2021) concatenating q with each point $y_j\in\{y_j\}_{j=1}^{t-1}$, we take q as the initial point and apply a self-attention layer over a new point sequence $\left[q,y_1,...,y_{t-1}\right]$ to obtain the query vector $Q_t\in\mathbb{R}^{c}$ with past content context, as shown in Fig. 5. Next, Q_t successively attends to E and H over subsequent decoding layers for aggregating style information, which is then mapped to the final output $O_t\in\mathbb{R}^{6m+3}$ via an MLP. As suggested in Tang et al. (2019), O_t includes $6m$ parameters of Gaussian mixture model (GMM) for predicting $\left(\Delta\hat{u}_t,\Delta\hat{v}_t\right)$ and 3 logits used to generate $\left(\hat{m}_t^1,\hat{m}_t^2,\hat{m}_t^3\right)$. Correspondingly, the training loss comprises two parts, i.e., pen moving prediction loss $\mathcal{L}_{pre}\left(\Delta u,\Delta v|O\right)$ and pen state classification loss $\mathcal{L}_{cls}\left(m^1,m^2,m^3;O\right)$, following Zhao et al. (2020). During training, our decoder performs parallel prediction for all points, as shown in Fig. 5, superior to previous RNN-based methods (Tang & Lian, 2021) that are executed in a step-by-step manner. For testing, we take as input the generated points $\{\hat{y}_j\}_{j=1}^{t-1}$, then combine them with q, E, and H to predict the next point $\hat{y}_t$. This process repeats until a pen-end state ($\hat{m}_{t-1}^3{=}1$) is accepted.

3.4 Learning SDT with overall loss

In total, our method contains four loss functions, namely, InstanceNCE, PatchNCE, pen moving prediction and pen state classification loss, which can be formulated as:

$$\mathcal{L}=\mathcal{L}_{ins}+\mathcal{L}_{pat}+\mathcal{L}_{pre}+\lambda\mathcal{L}_{cls}, \tag{3}$$

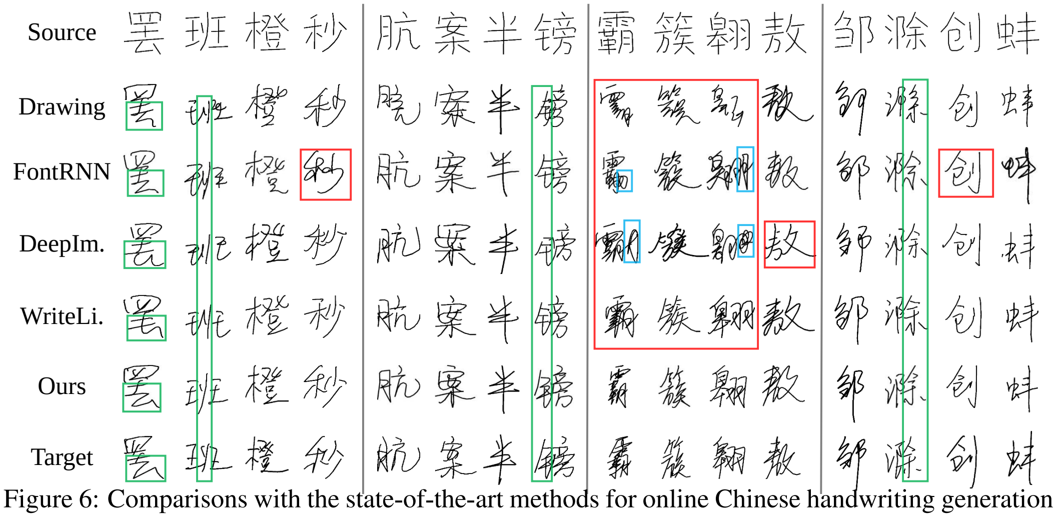

Figure 6: Comparisons with the state-of-the-art methods for online Chinese handwriting generation. The red and blue boxes highlight failures of style imitation and structure preservation, respectively. The green boxes highlight comparisons between the local details of targets and generated characters.

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

Method	Style Score ↑	Content Score ↑	DTW ↓	User Prefer. (%) ↑
Drawing (Zhang et al., 2017)	38.75	78.15	1.1813	4.73
FontRNN (Tang et al., 2019)	46.14	92.18	1.0448	9.93
DeepImitator (Zhao et al., 2020)	51.69	90.92	1.0622	10.27
WriteLikeYou (Tang & Lian, 2021)	73.07	93.89	0.9832	17.20
SDT(Ours)	**94.50**	**97.04**	**0.8789**	**57.87**

where λ is a trade-off factor, $\mathcal{L}_{ins}$ and $\mathcal{L}_{pat}$ correspond to the instance and patch head of the style encoder, respectively, while $\mathcal{L}_{pre}$ and $\mathcal{L}_{cls}$ are equipped on the transformer decoder.

4 Experiments

4.1 Chinese handwriting generation

Settings To evaluate SDT with the Chinese handwriting generation task, we use CASIA-OLHWDB (1.0-1.2) (Liu et al., 2011) for training and ICDAR-2013 competition database (Yin et al., 2013) for testing, following Tang & Lian (2021). The training set has about 3.7 million online Chinese handwritten characters produced by 1,020 writers, while the test set contains 60 writers and each writer covers 3,755 most frequently used characters set of GB2312-80. As suggested in Ha & Eck (2018), the Ramer–Douglas–Peucker algorithm (Douglas & Peucker, 1973) with a parameter of ϵ=2 is applied to remove redundant points of characters, leading to an average sequence length of 50. Following Zhao et al. (2020), we render offline style references using coordinate points of online characters. For content images, we use the popular average Chinese font (Jiang et al., 2019).

In all experiments, we use $K = 15$ style references, as in Zhao et al. (2020); Bhunia et al. (2021), and resize reference style and content images to 64×64. For architecture details, each transformer encoder employs 2 self-attentions layers while the transformer decoder adopts 4 layers for receiving style representations (2 for writer-wise and 2 for character-wise). Following the original transformer work (Vaswani et al., 2017), each transformer layer applies multi-headed attention with $c = 512$ dimensional states and 8 attention heads. We impose sinusoidal positional encoding (Vaswani et al., 2017) on the input tokens before feeding them to the transformer encoder and decoder. For training, we first pre-train the content encoder with 138k iterations (batch size is set to 256) for character classification over training samples and then train the whole model with 148k iterations (batch size is set to 128), on a single RTX3090 GPU. The optimizer is Adam (Kingma & Ba, 2015) with the learning rate of 0.0002 and gradient clipping of 5.0. As suggested in Tang & Lian (2021), we set λ=2. More implementation details are put in Appendix A.1.1.

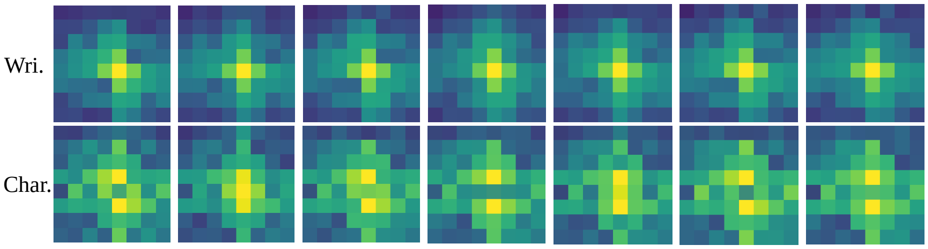

Figure 7: Frequency magnitude (8×8) belongs to 7 writers, where the top row shows the writer-wise style while the bottom represents the character-wise one. Each magnitude is averaged over 100 character samples written by the same person. A pixel that is closer to the center means a lower frequency. The brighter the color, the larger the magnitude.

Table 2: Effect of two style representations. $H-E$ denotes the transformer decoder first receives character-wise style features and then accepts writer-wise ones (and vice versa).

character-wise	writer-wise	$H-E$	$E-H$	Style Score↑
				85.52
✓				90.31
	✓			91.38
✓	✓	✓		93.72
✓	✓		✓	94.50

Table 3: Evaluation of different combinations between q and $\{y_j\}_{j=1}^{t-1}$.

Combination	Style Score ↑	Content Score↑	DTW ↓
Concat	91.61	96.95	0.8976
Ours	94.50	97.04	0.8789

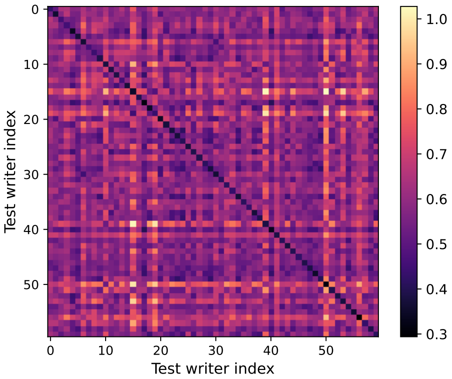

Figure 8: The heat map of the DTW matrix. The dark diagonal indicates that the generated characters still owns a higher similarity even using different X_s belonging to the same writer.

Quantitative Evaluation Metrics Dynamic time warping (DTW) (Berndt & Clifford, 1994), an elastic matching technique for aligning the given two sequences, is employed to calculate the distance between the generated and real characters. Content score (Zhao et al., 2020) is adopted to measure the structure correctness of generated characters. Style score (Tang & Lian, 2021) is employed to quantify the style similarity between the generated and real handwritings. User preference study is conducted to quantify the subjective quality of the output characters. More details are presented in Appendix A.1.2.

Comparison with State-of-the-Art Methods We compare our proposed SDT with the state-of-the-art online Chinese character generation methods, including Drawing (Zhang et al., 2017), FontRNN (Tang et al., 2019), DeepImitator (Zhao et al., 2020) and WriteLikeYou (Tang & Lian, 2021). For a fair comparison, we re-implement the variants of Drawing and FontRNN by adding a style branch proposed in DeepImitator (Zhao et al., 2020) using the PyTorch library, enabling them to achieve arbitrary stylized character generation.

Quantitative comparison The quantitative results are shown in Tab. 1. We observe that our SDT achieves the best performance on all the evaluation metrics. Particularly, SDT outperforms the second best with significant gaps in style score, i.e., a remarkably 21.43% gain. This demonstrates the strong generation performance of the proposed method in terms of the style imitation.

Qualitative comparison We illustrate the generated samples in Fig. 6 for each method, which intuitively explains the significant superiority of SDT in the user preference study. In Fig. 6, we observe that Drawing generates the worst results, as it often produces unreadable characters. FontRNN and DeepImitator occasionally synthesize unpleasant stroke paddings. WriteLikeYou performs not well on complex characters in terms of style mimicry. Compared to previous SOTA works, our method generates higher quality results, particularly recovering better local details of characters.

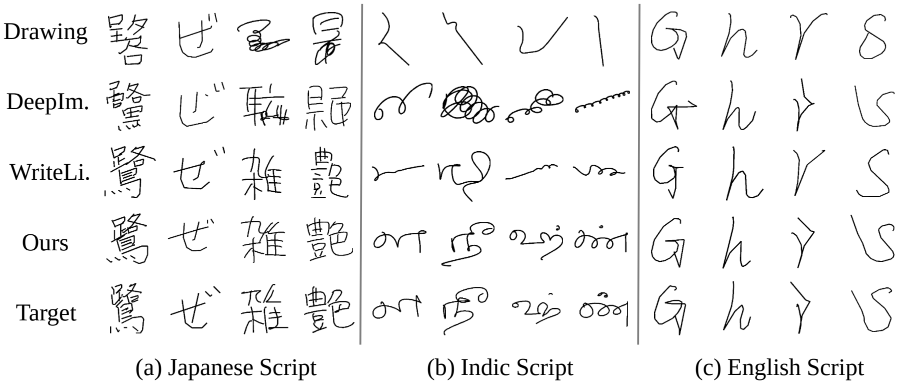

Figure 9: Comparisons with the competitors for online handwriting generation on various scripts.

4.2 Analysis

Effect of two style representations. To evaluate the effectiveness of the two extracted style representations, we conduct ablation studies on the test set and show the experimental results in Tab. 2. From these results, we have the following observations: (1) Both style representations contribute to improving the quality of the generated results in terms of style score. (2) Combining the two style features is able to further promote the model performance in terms of style score, which indicates the two extracted styles are complementary. Additionally, the order of the two style features entering transformer decoder has no significant impact in terms of style score. To further analyze the differences between the two styles, we resize the output patch tokens representing the two styles to feature maps, respectively, and visualize their frequency magnitudes in Fig. 7. A qualitative comparison indicates that character-wise style features contain more high-frequency information and writer-wise ones mainly focus on the low-frequency information. According to Cooley et al. (1969), the high-frequency information in an image usually captures local fine details while the low frequencies contain the global part of objects. This fact and the visualization strongly align with our motivation to separate the overall and detailed style representations from the style images.

Evaluation of different combinations between q and $\{y_j\}_{j=1}^{t-1}$. To evaluate the effect of different combination strategies, we re-implement a variant of our method by concatenating q with each point $y_j \in \{y_j\}_{j=1}^{t-1}$ and compare it with our method on the test set. As presented in Tab. 3, we find that our combination strategy improves the style consistency without decreasing content correctness of the generated results. This indicates that our method is able to draw global dependencies between q and $\{y_j\}_{j=1}^{t-1}$ unlike previous RNN-based methods (Zhao et al., 2020) that suffer from the forgetting phenomenon Graves (2012), which demonstrates the effectiveness of the proposed method.

Effect of using different style inputs X_s. As mentioned in Tang & Lian (2021), the imitation model may generate inconsistent characters if given different style inputs X_s belonging to the same writer w_s. To evaluate the effect of different style inputs, we conduct two independent experiments using different X_s on the same model. In each experiment, the model generates 200 characters for each writer in the test set, as in Tang & Lian (2021). We calculate the DTW distance between the corresponding character individually produced in the two experiments and then average them according to the writer index (see Appendix A.1.3 for more details of the calculation) to get a DTW square matrix, visualized as Fig. 8. The dark diagonal in Fig. 8 indicates that the generated characters still owns a higher similarity even using different X_s belonging to the same writer, which demonstrates that our SDT is able to generate comparable results from different style inputs.

4.3 Applications to other languages

Japanese handwriting generation. For Japanese handwriting generation task, we conduct experiments on TUAT HANDS databases (Matsumoto et al., 2001) to evaluate the superiority of our method (more dataset information can be seen in Appendix A.1.4). Tab. 4 and Fig. 9 (a) summarizes the results of Japanese handwriting generation with a comparison to previous works (i.e., Drawing, DeepImitator and WriteLikeYou). From these results, we observe that our SDT outperforms all compared methods on three quantitative metrics. In terms of style score, our SDT outperforms

Table 4: Quantitative evaluations of our SDT and competitors on Japanese datatset.

Datasets	Methods	Style Score ↑	Content Score↑	DTW ↓
Japanese	Drawing (Zhang et al., 2017)	20.67	50.74	1.4657
	DeepImitator (Zhao et al., 2020)	25.80	53.20	1.2564
	WriteLikeYou (Tang & Lian, 2021)	28.22	86.08	1.2388
	SDT(Ours)	**41.85**	**91.31**	**1.1289**

Table 5: Quantitative evaluations of our SDT and competitors on Indic datatset.

Methods	Content Score↑	DTW ↓
Drawing	2.34	9.8230
DeepImitator	4.13	6.7421
WriteLikeYou	11.61	4.7314
SDT(Ours)	**97.22**	**0.7075**

Table 6: Quantitative evaluations of our SDT and competitors on English dataset.

Methods	Content Score↑	DTW ↓
Drawing	79.14	1.8519
DeepImitator	76.53	1.6460
WriteLikeYou	84.54	1.6282
SDT(Ours)	**85.52**	**1.6048**

WriteLikeYou by a large margin (41.85% $vs.$ 28.22%), which further demonstrates our method has a better imitation performance in respect of handwriting styles regardless of the script type.

Indic handwriting generation. We evaluate our method with Indic handwriting generation task using Tamil dataset[1]. We compare our methods with Drawing, DeepImitator and WriteLikeYou on the test set of Tamil dataset (see more dataset information in Appendix A.1.4). As shown in Tab. 5, we report the experimental results in terms of content score and DTW, as it is intractable to train a high-performance writer recognizer (see details in Tab. 8) on Indic script due to the limited data. From these results, we find that our SDT surpasses the second best with significant gaps on the two quantitative metrics, i.e., achieving 85.61% higher content score and 4.02 lower DTW. This means our SDT is able to handle handwritten characters with large amount of points (with an average of 88) and ensure structure correctness of the generated samples, as shown in Fig. 9 (b).

English handwriting generation. To demonstrate the effectiveness of our method on English handwriting generation task, we collect all of the English samples from the symbol part of CASIA-OLHWDB(1.0-1.2) (Liu et al., 2011) and ICDAR-2013 competition database (Yin et al., 2013) (see more details in Appendix A.1.4). Similarly, due to the lack of high-performance writer identifier (see details in Tab. 8), we use content score and DTW as evaluation metrics . Fig. 9 (c) shows the qualitative results comparing our SDT with three competitors on the test set. From these results, we find that all methods achieve sound and comparable performance. One reason is that English script contains fewer character classes and a smaller number of points (with an average of 30), which makes their imitation easier compared to other scripts. Nevertheless, our SDT still outperforms other methods with a small margin both in content score and DTW, as shown in Tab. 6. Moreover, we observe that corresponding uppercase and lowercase letters sometimes have subtle inter-class differences, e.g., O $vs.$ o, which leads to our SDT achieving a relatively low content score.

5 Conclusion

In this paper, we propose a novel method named style-disentangled transformer (SDT) to synthesize realistic and diverse online handwritings. Our SDT improves imitation performance by disentangling the overall writer-wise and detailed character-wise style representations from the individual calligraphy handwritings. For the writer-wise style, we propose to group characters from a writer together and separate characters from different writers, encouraging SDT to learn the overall uniformity in individual handwritings. For the character-wise style, we propose maximizing the mutual information between sampling results that represent distinct views of a character. Moreover, we outline an offline-to-offline framework for improving the generation quality of offline handwritten Chinese characters. Promising results on various language scripts verify the effectiveness of our SDT. Though the SDT is currently designed for handwriting generation, it can be extended to other generation tasks in future works, e.g., font and artistic character generation.

[1] http://lipitk.sourceforge.net/datasets/tamilchardata.htm

References

Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer. In *Computer Vision and Pattern Recognition*, pp. 7564–7573, 2018.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, pp. 157–166, 1994.

Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining Workshop*, 1994.

Ankan Kumar Bhunia, Salman Khan, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, and Mubarak Shah. Handwriting transformers. In *International Conference on Computer Vision*, pp. 1086–1094, 2021.

Bo Chang, Qiong Zhang, Shenyi Pan, and Lili Meng. Generating handwritten chinese characters using cyclegan. In *Winter Conference on Applications of Computer Vision*, pp. 199–207, 2018.

James W Cooley, Peter AW Lewis, and Peter D Welch. The fast fourier transform and its applications. *IEEE Transactions on Education*, 12(1):27–34, 1969.

Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *Advances in Neural Information Processing Systems*, pp. 6447–6458, 2019.

David H Douglas and Thomas K Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: the international journal for geographic information and geovisualization*, pp. 112–122, 1973.

Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, and Roee Litman. Scrabblegan: Semi-supervised varying length handwritten text generation. In *Computer Vision and Pattern Recognition*, pp. 4324–4333, 2020.

Ji Gan and Weiqiang Wang. Higan: Handwriting imitation conditioned on arbitrary-length texts and disentangled styles. In *AAAI Conference on Artificial Intelligence*, pp. 7484–7492, 2021.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing*, pp. 6894–6910, 2021.

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

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Alex Graves. *Long Short-Term Memory*. Springer Berlin Heidelberg, 2012.

Alex Graves. Generating sequences with recurrent neural networks. *Arxiv*, 2013.

David Ha and Douglas Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations*, 2018.

Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer Vision and Pattern Recognition*, pp. 1735–1742, 2006.

Junlin Han, Mehrdad Shoeiby, Lars Petersson, and Mohammad Ali Armin. Dual contrastive learning for unsupervised image-to-image translation. In *Computer Vision and Pattern Recognition*, pp. 746–755, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Yaoxiong Huang, Mengchao He, Lianwen Jin, and Yongpan Wang. Rd-gan: few/zero-shot chinese character style transfer via radical decomposition and rendering. In *European Conference on Computer Vision*, pp. 156–172, 2020.

Yue Jiang, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. Scfont: Structure-guided chinese font generation via deep stacked networks. In *AAAI conference on Artificial Intelligence*, pp. 4015–4022, 2019.

Lei Kang, Pau Riba, Yaxing Wang, Marçal Rusinol, Alicia Fornés, and Mauricio Villegas. Ganwriting: content-conditioned generation of styled handwritten word images. In *European Conference on Computer Vision*, pp. 273–289, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, pp. 18661–18673, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Weirui Kong and Bicheng Xu. Handwritten chinese character generation via conditional neural generative models. In *Advances in Neural Information Processing Systems*, pp. 4–7, 2017.

Yuxin Kong, Canjie Luo, Weihong Ma, Qiyuan Zhu, Shenggao Zhu, Nicholas Yuan, and Lianwen Jin. Look closer to supervise better: One-shot font generation via component-based discriminator. In *Computer Vision and Pattern Recognition*, pp. 13482–13491, 2022.

Atsunobu Kotani, Stefanie Tellex, and James Tompkin. Generating handwriting via decoupled style descriptors. In *European Conference on Computer Vision*, pp. 764–780, 2020.

Zhouhui Lian, Bo Zhao, and Jianguo Xiao. Automatic generation of large-scale handwriting fonts via style learning. In *SIGGRAPH Asia Technical Briefs*, pp. 1–4, 2016.

Jeng-Wei Lin, Chian-Ya Hong, Ray-I Chang, Yu-Chun Wang, Shu-Yu Lin, and Jan-Ming Ho. Complete font generation of chinese characters in personal handwriting style. In *International Performance Computing and Communications Conference*, pp. 1–5, 2015.

Zhouchen Lin and Liang Wan. Style-preserving english handwriting synthesis. *Pattern Recognition*, 40(7):2097–2109, 2007.

Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. Casia online and offline chinese handwriting databases. In *2011 International Conference on Document Analysis and Recognition*, pp. 37–41, 2011.

Wei Liu, Fangyue Liu, Fei Ding, Qian He, and Zili Yi. Xmp-font: Self-supervised cross-modality pre-training for few-shot font generation. In *Computer Vision and Pattern Recognition*, pp. 7905–7914, 2022.

Canjie Luo, Yuanzhi Zhu, Lianwen Jin, Zhe Li, and Dezhi Peng. Slogan: Handwriting style synthesis for arbitrary-length and out-of-vocabulary text. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Kaoru Matsumoto, Takahiro Fukushima, and Masaki Nakagawa. Collection and analysis of on-line handwritten japanese character patterns. In *International Conference on Document Analysis and Recognition*, pp. 496–500, 2001.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *Arxiv*, 2018.

Song Park, Sanghyuk Chun, Junbum Cha, Bado Lee, and Hyunjung Shim. Few-shot font generation with localized style representations and factorization. In *AAAI Conference on Artificial Intelligence*, pp. 2393–2402, 2021.

Taesung Park, Alexei A Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. In *European Conference on Computer Vision*, pp. 319–345, 2020.

Xuanchi Ren, Tao Yang, Yuwang Wang, and Wenjun Zeng. Learning disentangled representation by exploiting pretrained generative models: A contrastive learning view. In *International Conference on Learning Representations*, 2022.

Shusen Tang and Zhouhui Lian. Write like you: Synthesizing your cursive online chinese handwriting via metric-based meta learning. *Computer Graphics Forum*, 40(2):141–151, 2021.

Shusen Tang, Zeqing Xia, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. Fontrnn: Generating large-scale chinese fonts via recurrent neural network. *Computer Graphics Forum*, 38(7):567–577, 2019.

Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2020.

Yuchen Tian. zi2zi:master chinese calligraphy with conditional adversarial networks. https://github.com/kaonashi-tyc/zi2zi, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *Computer Vision and Pattern Recognition*, pp. 5265–5274, 2018.

Jue Wang, Chenyu Wu, Ying-Qing Xu, Heung-Yeung Shum, and Liang Ji. Learning-based cursive handwriting synthesis. In *International Workshop on Frontiers in Handwriting Recognition*, pp. 157–162, 2002.

Yangchen Xie, Xinyuan Chen, Li Sun, and Yue Lu. Dg-font: Deformable generative networks for unsupervised font generation. In *Computer Vision and Pattern Recognition*, pp. 5130–5140, 2021.

Shuai Yang, Jiaying Liu, Wenjing Wang, and Zongming Guo. Tet-gan: Text effects transfer via stylization and destylization. In *AAAI Conference on Artificial Intelligence*, pp. 1238–1245, 2019.

Fei Yin, Qiu-Feng Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Icdar 2013 chinese handwriting recognition competition. In *2013 12th international conference on document analysis and recognition*, pp. 1464–1470, 2013.

Hang Yin, Patrícia Alves-Oliveira, Francisco S Melo, Aude Billard, and Ana Paiva. Synthesizing robotic handwriting motion by learning from human demonstrations. In *International Joint Conference on Artificial Intelligence*, pp. 3530–3537, 2016.

Xu-Yao Zhang, Fei Yin, Yan-Ming Zhang, Cheng-Lin Liu, and Yoshua Bengio. Drawing and recognizing chinese characters with recurrent neural network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):849–862, 2017.

Yexun Zhang, Ya Zhang, and Wenbin Cai. Separating style and content for generalized style transfer. In *Computer Vision and Pattern Recognition*, pp. 8447–8455, 2018.

Yuxin Zhang, Fan Tang, Weiming Dong, Haibin Huang, Chongyang Ma, Tong-Yee Lee, and Changsheng Xu. Domain enhanced arbitrary image style transfer via contrastive learning. In *SIGGRAPH Conference*, pp. 12:1–12:8, 2022.

Bocheng Zhao, Jianhua Tao, Minghao Yang, Zhengkun Tian, Cunhang Fan, and Ye Bai. Deep imitator: Handwriting calligraphy imitation via deep attention networks. *Pattern Recognition*, 104:107080, 2020.

Alfred Zong and Yuke Zhu. Strokebank: Automating personalized chinese handwriting generation. In *AAAI Conference on Artificial Intelligence*, 2014.

A Appendix

We organize our supplementary material as follows.

- In section A.1, we describe more experimental details.
- In section A.2, we review the works in font generation.
- In section A.3, we provide additional qualitative results of offline Chinese handwriting generation with a comparison to previous state-of-the-art works.
- In section A.4, we show a large number of generated online samples, covering Chinese, Japanese, Indic and English scripts.
- In section A.5, we provide more visualization examples for spectrum analysis of two style representations.

A.1 More Experimental Details.

A.1.1 Implementation Details of Robustness Training

After removing the redundant points of online characters, we follow Zhang et al. (2017) to normalize the absolute coordinates of points into a standard interval. As mentioned in Sec. 3.3, we define three states "pen-down", "pen-up" and "pen-end" respectively, which are denoted as m^1, m^2, m^3. Specifically, pen-down means that the pen is touching the paper now, and the current and following points will be connected by strokes. Pen-up indicates that the pen has just finished a stroke and is to be lifted up. Pen-end means that the pen has finished writing a completed character. It is obvious that pen-end data points are much less than the other two classes. To solve the biased dataset issue, we pad each online character $Y = [y_1, ..., y_L]$ to a fixed length N_{max}, where N_{max} is the length of the longest character in our training dataset and L is the length of Y, following (Ha & Eck, 2018). As L is usually shorter than N_{max}, we set y_i to be $(0, 0, 0, 0, 1)$, for $i > L$. During training, we set the temperature τ=0.07 both in Eqn.1 and Eqn.2. Following Tang & Lian (2021), we use the Gaussian mixture model (GMM) with m=20 bivariate normal distributions, i.e., the final output $O_t \in \mathbb{R}^{123}$.

A.1.2 Implementation Details of Metrics

DTW The lower DTW distance, the better quality of the generated characters. Following Tang & Lian (2021), we normalize the DTW distance by the spatial size and length of real handwritings.

Content and Style score We use the content recognizer and writer identifier to evaluate the content and style score of generated handwritings, respectively. We give the implementation details of the two recognizers below. For the content recognizer (Tang & Lian, 2021), we train it on the training set. The optimizer is Adam with the learning rate of 0.001 and the batch size is set to 256. In total, we train four content recognizers on four training sets, i.e., Chinese, Japanese, Indic and English datasets, respectively. Tab. 7 summarizes their recognition results on the corresponding test sets. For the writer identifier, we train it on the handwritings belonging to the test writers. Different from the content recognizer receiving a character once, the writer identifier takes 15 characters written by the same person as one input set (Zhao et al., 2020). Similarly, we use the Adam optimizer to train four writer identifiers with the batch size of 128, learning rate of 0.001. We report their recognition accuracy in Tab. 8.

User preference study At each time, given a style reference along with several candidates generated by different methods, participants are required to pick up the most similar candidate with the reference. We finally collect 1500 valid responses contributed by 50 volunteers.

A.1.3 Implementation Details of DTW Matrix

As mentioned in Sec. 4.2, we generate two groups of characters $\{\mathbf{a}^i\}_{i=1}^T$ and $\{\mathbf{b}^j\}_{j=1}^T$ using different style inputs, where T is the number of test writers, $\mathbf{a}^i = [a_1, ..., a_M]$ and $\mathbf{b}^j = [b_1, ..., b_M]$ denote the M characters belonging to the writer w_i and w_j, respectively. Next, we formulate the average DTW

Table 7: Quantitative evaluations of four content recognizers on four datasets.

Datasets	Acc.(%)
Chinese (Yin et al., 2013)	95.43
Japanese (Matsumoto et al., 2001)	93.61
Indic[4]	94.48
English (Yin et al., 2013)	80.12

Table 8: Quantitative evaluations of four writer identifiers on four datatsets.

Datasets	Acc.(%)
Chinese (Yin et al., 2013)	99.98
Japanese (Matsumoto et al., 2001)	99.64
Indic[4].	72.54
English (Yin et al., 2013)	20.57

distance between $\mathbf{a}^i$ and $\mathbf{b}^j$ as:

$$d_{ave}(\mathbf{a}^i, \mathbf{b}^j) = \frac{1}{M} \sum_{m=1}^{M} d(a_m, b_m), \tag{4}$$

where $d(\cdot, \cdot)$ is the DTW distance between two characters. Finally, we denote the DTW matrix $\mathbf{C} = (c_{ij}) \in \mathbb{R}^{T \times T}$, where c_{ij} can be formulate as:

$$c_{ij} = d_{ave}(\mathbf{a}^i, \mathbf{b}^j). \tag{5}$$

In particular, when $i = j$, c_{ij} indicates the average DTW distance between generated characters using different style references belonging to the same person.

A.1.4 Dataset Details

Japanese Dataset TUAT HAND (Matsumoto et al., 2001) contains about 3 million online handwritten Japanese characters belonging to 271 writers. We randomly select 216 writers for training and 55 writers for testing. Similarly, we use the Ramer–Douglas–Peucker algorithm (ϵ=2) to preprocess the online characters. After simplification, the maximum sequence length of characters reaches 770, which is a trouble for training RNN (Bengio et al., 1994). For a fair comparison with the previous RNN-based works (Zhao et al., 2020), we drop characters with points more than 150, accounting for about 2% of the total datasets (Tang & Lian, 2021). After that, the average length of characters is shortened to 68. We render style images from processed online characters and use easily obtainable printed font as content references.

Indic Dataset Tamil dataset[2] consists of samples of 156 Indic character classes written by 169 people, which offers an official train set and test set, i.e., 117 writers for training and 52 writers for testing. Similarly, we remove the redundant points of characters via Ramer–Douglas–Peucker algorithm (ϵ=2) and discard characters with points more than 150. After that, the average sequence length of characters are reduced to 88. We use online Indic characters to render style images. As for content references, we use character embeddings instead of offline images. This is because Tamil encodes characters to special indexes that can not be directly matched with the printed font in UTF-8[3] Format.

English Dataset In total, we have 53,248 English characters (Liu et al., 2011) written by 1,020 persons for training, and 3,120 characters (Yin et al., 2013) from 60 writers for testing, where the characters written by each writer cover 52 classes. Similarly, the Ramer–Douglas–Peucker algorithm ($\epsilon = 2$) is adopted to remove redundant points of characters, leading to an average sequence length of 30. We render style images using coordinate points of online characters and employ printed English font as content images.

A.2 Font Generation

Generative Adversarial Networks (GANs) (Goodfellow et al., 2014) open a new door for font generation and bring amazing performance gains. zi2zi (Tian, 2017) regards font generation as an image translation task and achieves diverse font style transfer via a condition GAN. MC-GAN (Azadi et al., 2018) generates the whole set of letters with a consistent style by observing only a few examples

[2] http://lipitk.sourceforge.net/datasets/tamilchardata.htm

[3] https://www.utf8.com

[4] http://lipitk.sourceforge.net/datasets/tamilchardata.htm

via the proposed glyph generation network and texture transfer module. Later, EMD (Zhang et al., 2018) and TET-GAN (Yang et al., 2019) learn the disentangled representations for contents and styles, and thus achieve the unseen style transfer. To further generate high-quality characters, some component-based methods are proposed to take auxiliary annotations (e.g., stroke and radical decomposition) as inputs (Jiang et al., 2019; Park et al., 2021; Liu et al., 2022) or supervisions (Huang et al., 2020; Kong et al., 2022). However, all of the above works do not explicitly consider the geometric deformation of fonts. DG-font (Xie et al., 2021) introduces a feature deformation skip connection to conduct spatial deformation, thus performing better on cursive characters. Nonetheless, the advanced DG-font struggles to address the large geometric variations, as shown in Fig. 10.

A.3 Offline Chinese handwritten characters generation.

Experimental Setting. To demonstrate the superiority of the proposed offline-to-offline handwriting generation framework, we use the offline character images of ICDAR-2013 competition database (Yin et al., 2013), which contains 60 writers and 3755 different Chinese characters for each writer. We randomly select 80% of the entire dataset as the training set, and the remaining 20% as the test set. As for content images, we use the popular average Chinese font (Jiang et al., 2019). In our experiments, we resize input images to 64×64. We insert an extra ornamentation network (Xie et al., 2021) behind our method and compare it with font generation and handwriting image generation methods. Specifically, (1) font generation methods include zi2zi (Tian, 2017) and DG-FONT (Xie et al., 2021). (2) handwriting image generation methods such as GANWriting (Kang et al., 2020) and HWT (Bhunia et al., 2021) are considered compared methods.

Qualitative Comparison. Fig. 10 shows qualitative comparison between our method with four competitors. To ensure fair comparisons, we randomly select source and target characters with the same textual contents. The rows of "Source" present standard characters with different content. Each row of "Target" presents characters belonging to the same writer. We can observe that the handwritten characters generated by our SDT (rows of "Ours") yield the most similar styles to target images in terms of geometric shape and ink-blot. Besides, serious artifacts (e.g., blur and collapsed character structure) appear on the handwritings generated by zi2zi (rows of "Zi2zi") and HWT (rows of "HWT"). There are different degrees of stroke missing in the handwritings generated by GANWriting (rows of "GANW.") and DG-Font ("rows of DG-F."). Moreover, except our SDT, other methods struggle to synthesize the stroke width and ink-blot similar to the target characters. Further, we provide more qualitative results with a comparison to GANWriting and DG-Font in Figs. 11-14.

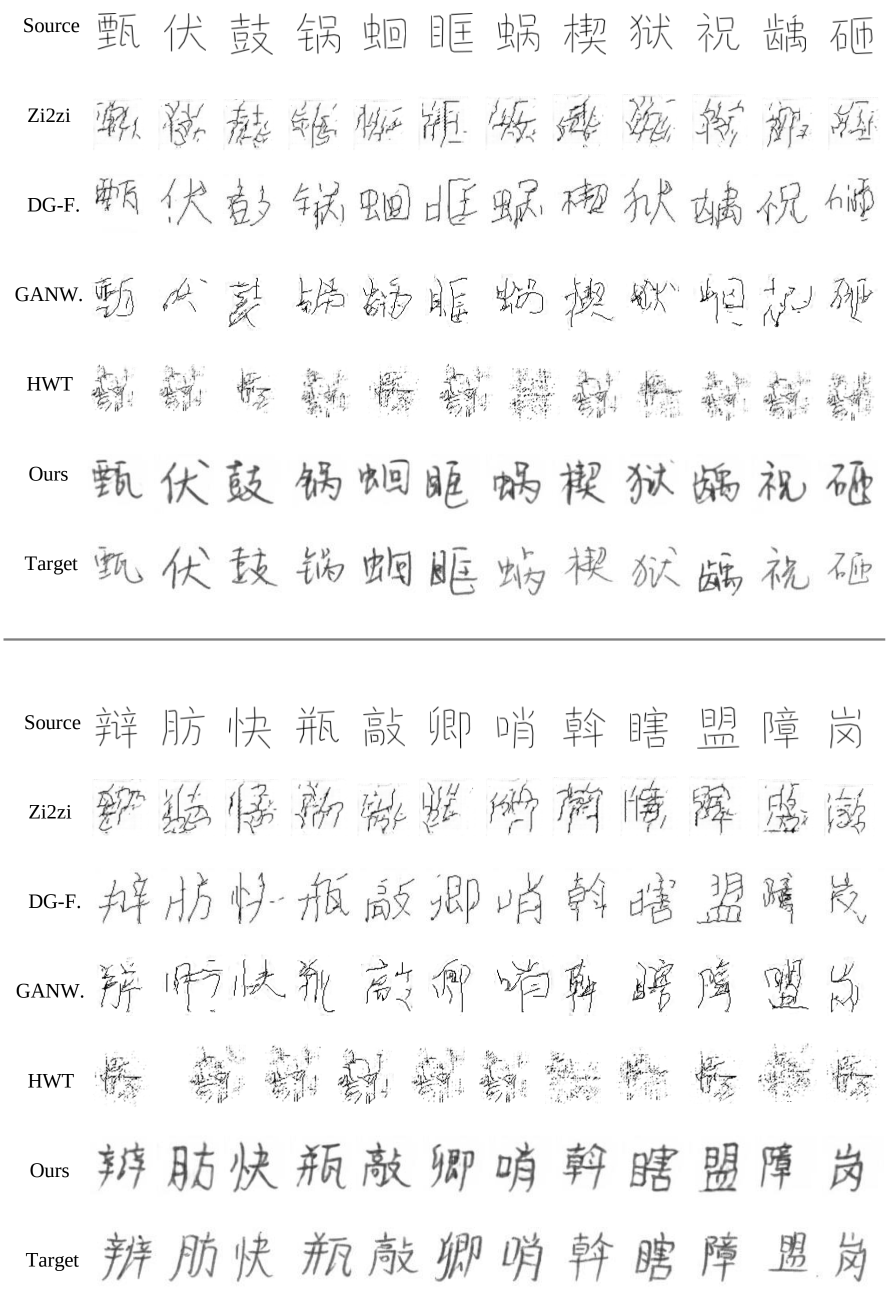
Source
甄 伏 鼓 锅 蛔 眶 蜗 楔 狱 祝 龋 砸
Zi2zi
DG-F.
GANW.
HWT
Ours
Target
Source
辩 肪 快 瓶 敲 卿 哨 斡 瞎 盟 障 岗
Zi2zi
DG-F.
GANW.
HWT
Ours
Target

Figure 10: Additional qualitative comparisons between our proposed SDT with four competitors, including zi2zi (Tian, 2017), DG-FONT (Xie et al., 2021), GANWriting (Kang et al., 2020) and HWT (Bhunia et al., 2021), on offline handwritten Chinese character generation.

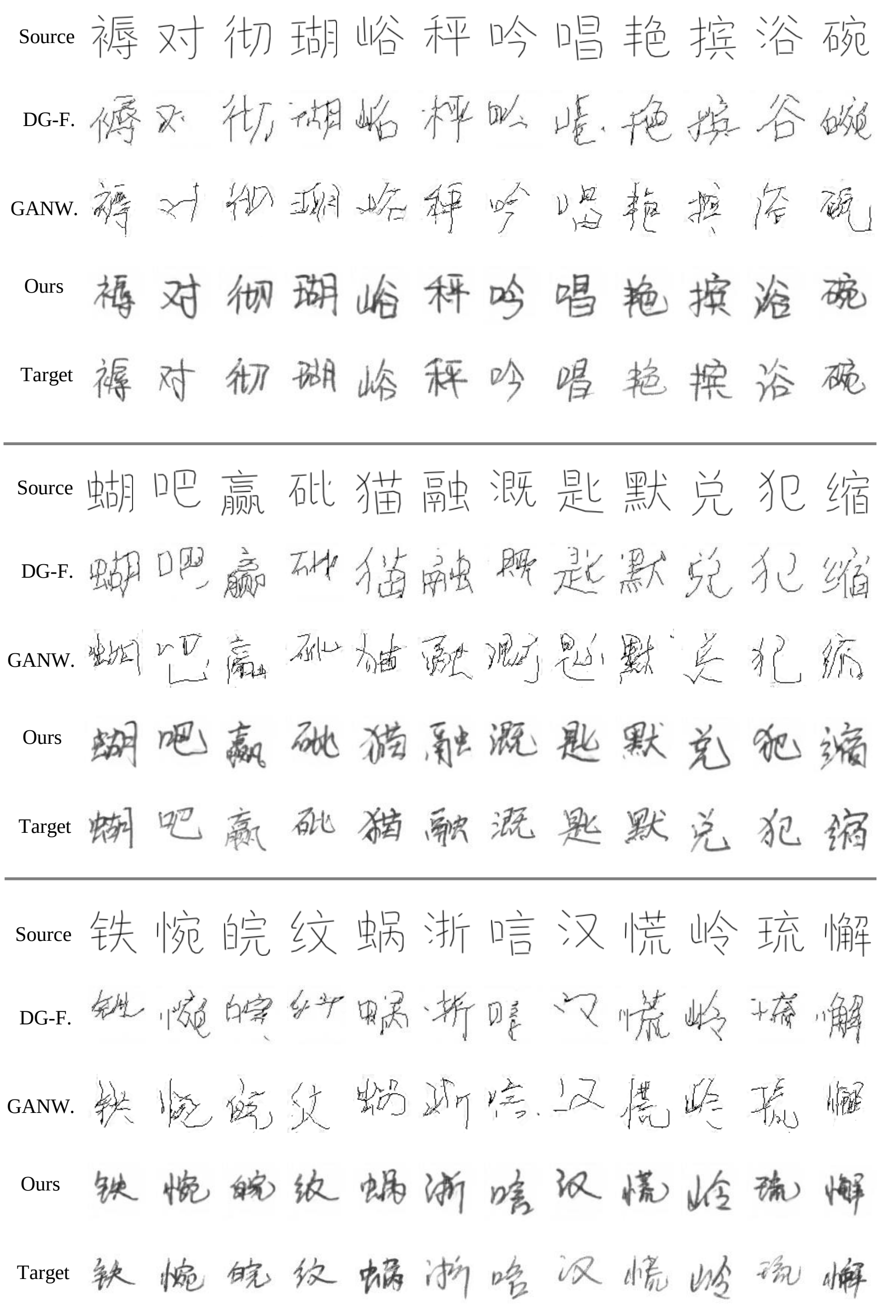
Source
DG-F.
GANW.
Ours
Target
褥 对 彻 瑚 峪 秤 吟 唱 艳 摈 浴 碗
蝴 吧 赢 砒 猫 融 溉 匙 默 兑 犯 缩
铁 惋 皖 纹 蜗 浙 唁 汉 慌 岭 琉 懈

Figure 11: Additional qualitative comparisons between our proposed SDT with DG-FONT (Xie et al., 2021) and GANWriting (Kang et al., 2020), on offline handwritten Chinese character generation.

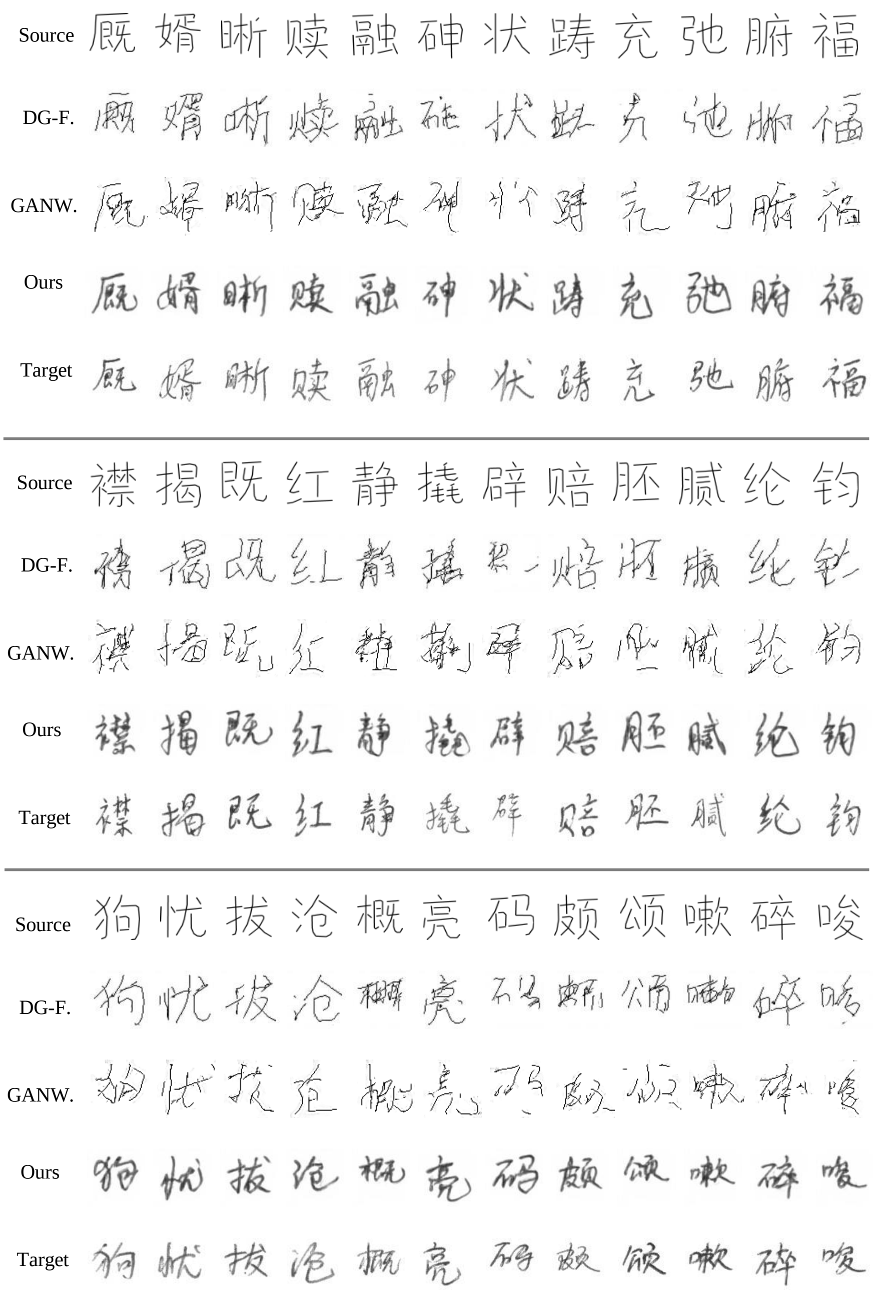
Source
厩 婿 晰 赎 融 砷 状 踌 充 弛 腑 福
DG-F.
GANW.
Ours
Target
Source
襟 揭 既 红 静 撬 辟 赔 胚 腻 纶 钧
DG-F.
GANW.
Ours
Target
Source
狗 忧 拔 沧 概 亮 码 颇 颂 嗽 碎 唆
DG-F.
GANW.
Ours
Target

Figure 12: Additional qualitative comparisons between our proposed SDT with DG-FONT (Xie et al., 2021) and GANWriting (Kang et al., 2020), on offline handwritten Chinese character generation.

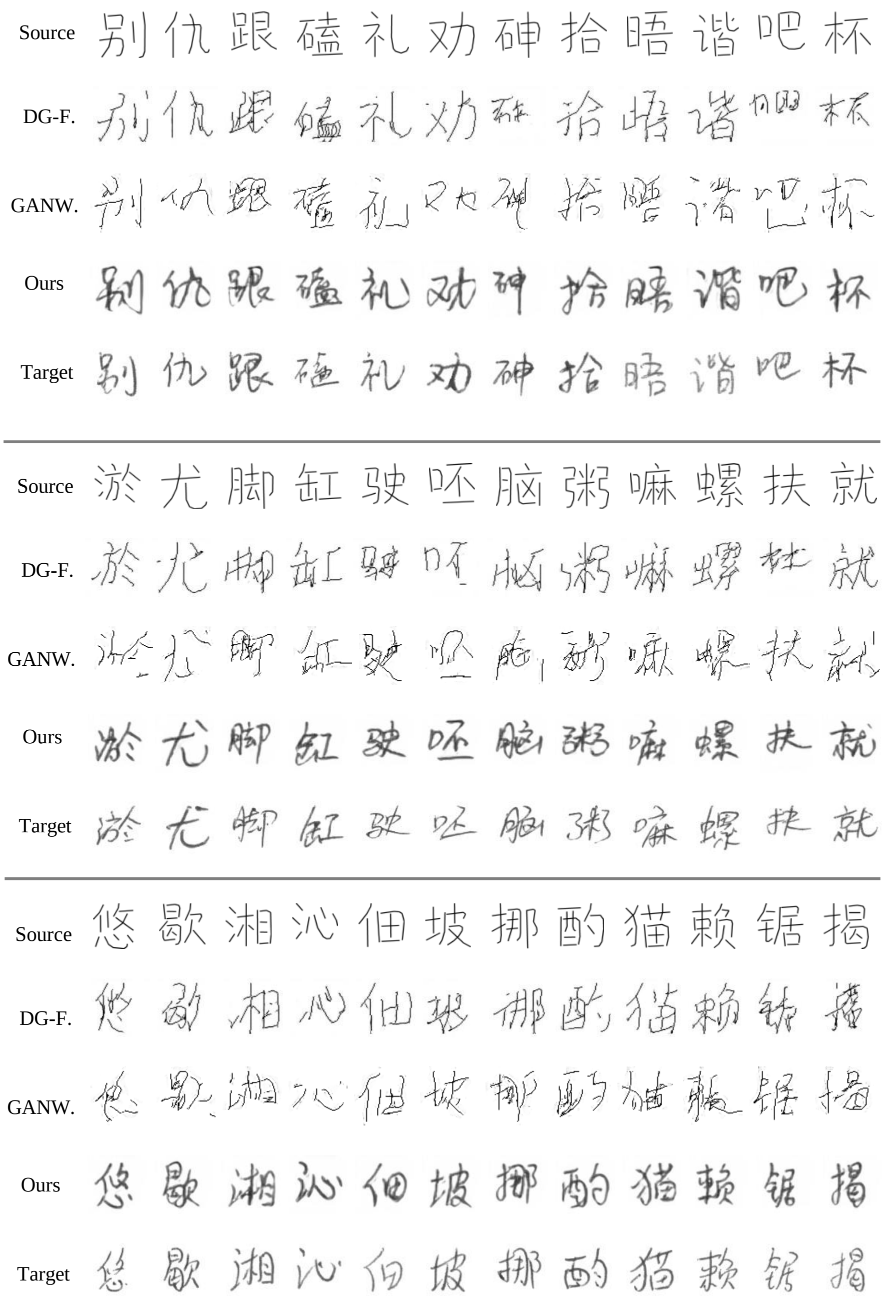
Source 别 仇 跟 磕 礼 劝 砷 拾 晤 谐 吧 杯
DG-F.
GANW.
Ours
Target
Source 淤 尤 脚 缸 驶 呸 脑 粥 嘛 螺 扶 就
DG-F.
GANW.
Ours
Target
Source 悠 歇 湘 沁 佃 坡 挪 酌 猫 赖 锯 揭
DG-F.
GANW.
Ours
Target

Figure 13: Additional qualitative comparisons between our proposed SDT with DG-FONT (Xie et al., 2021) and GANWriting (Kang et al., 2020), on offline handwritten Chinese character generation.

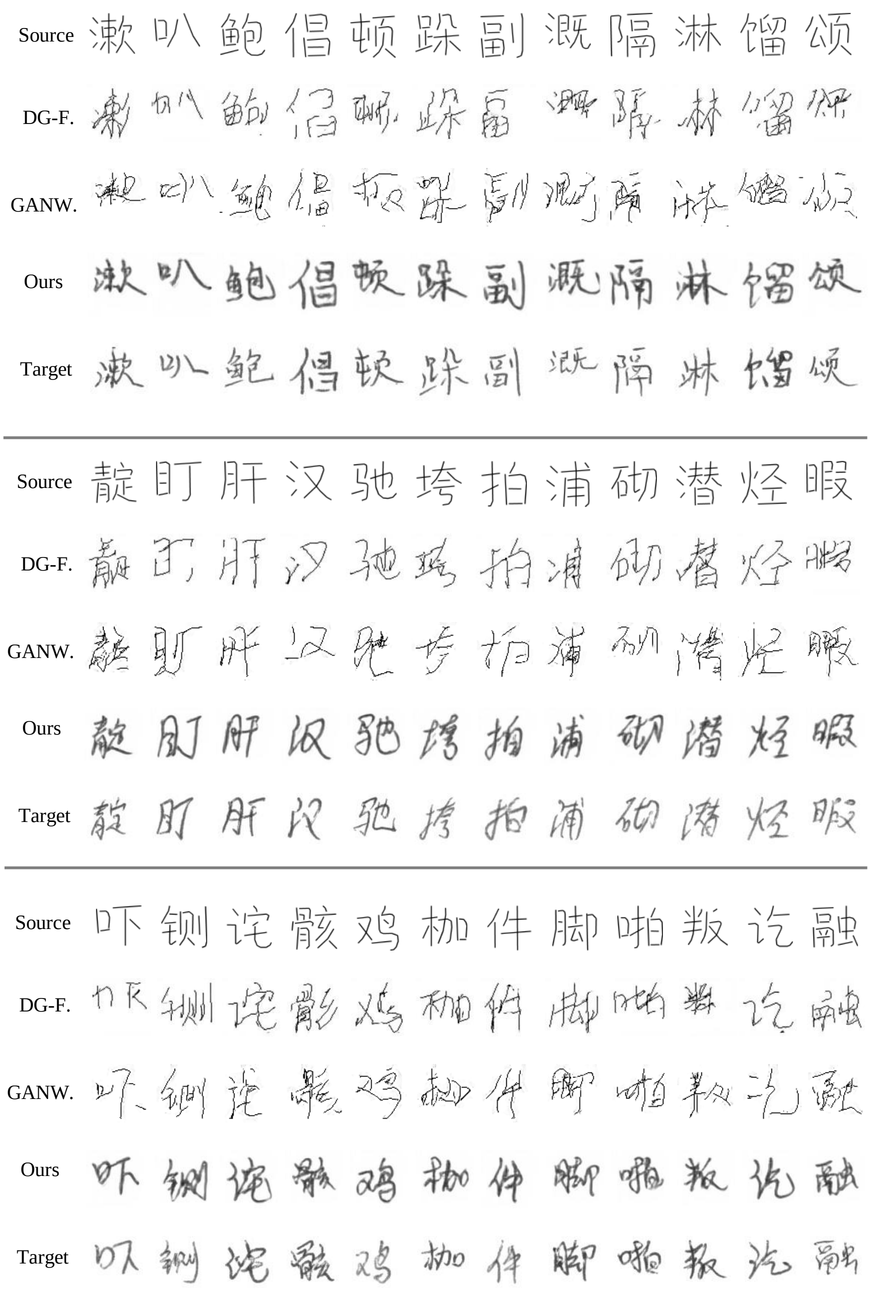
Source 漱 叭 鲍 倡 顿 跺 副 溉 隔 淋 馏 颂
DG-F.
GANW.
Ours
Target
Source 靛 盯 肝 汉 驰 垮 拍 浦 砌 潜 烃 暇
DG-F.
GANW.
Ours
Target
Source 吓 铡 诧 骸 鸡 枷 件 脚 啪 叛 讫 融
DG-F.
GANW.
Ours
Target

Figure 14: Additional qualitative comparisons between our proposed SDT with four competitors DG-FONT (Xie et al., 2021) and GANWriting (Kang et al., 2020), on offline handwritten Chinese character generation.

A.4 Online handwriting generation.

Figs. 15-18 show qualitative comparisons between our proposed SDT and the previous state-of-the-art work WriteLikeYou (Tang & Lian, 2021) on online multilingual characters generation (e.g., Chinese, Japanese, Indic and English scripts). The results suggest that our method is more competitive in both style imitation and structure preservation of generated multilingual characters.

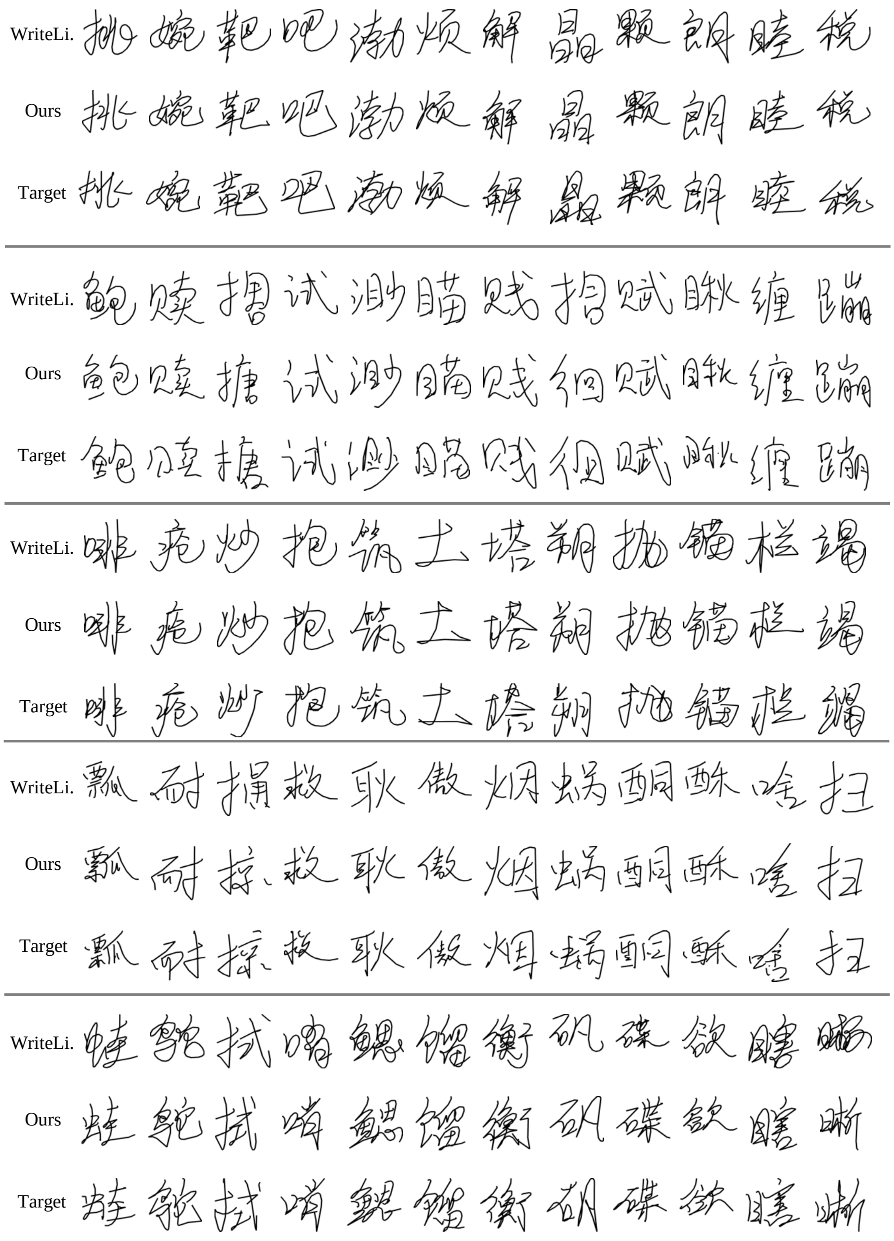

Figure 15: Additional generated online Chinese characters by our method and WriteLikeYou (Tang & Lian, 2021).

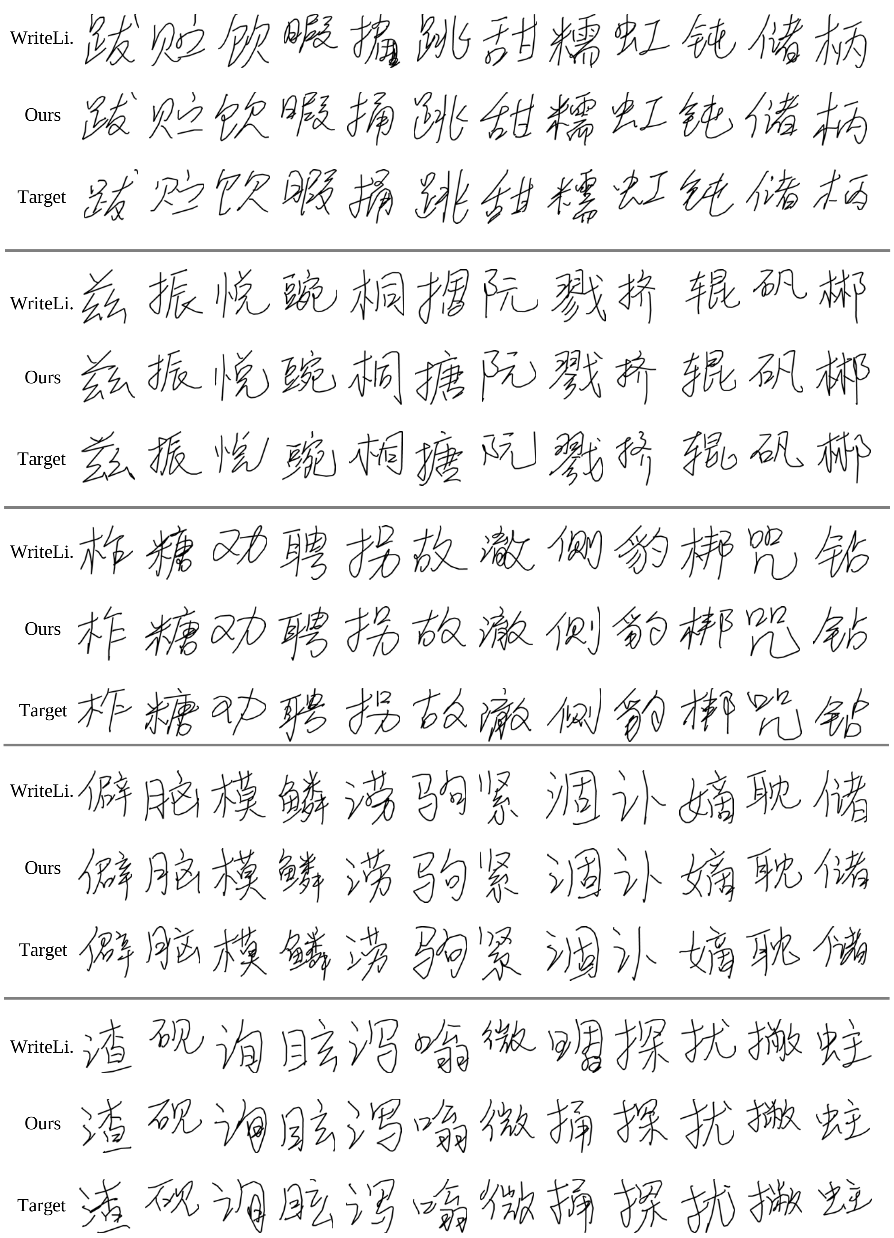

Figure 16: Additional generated online Chinese characters by our method and WriteLikeYou (Tang & Lian, 2021).

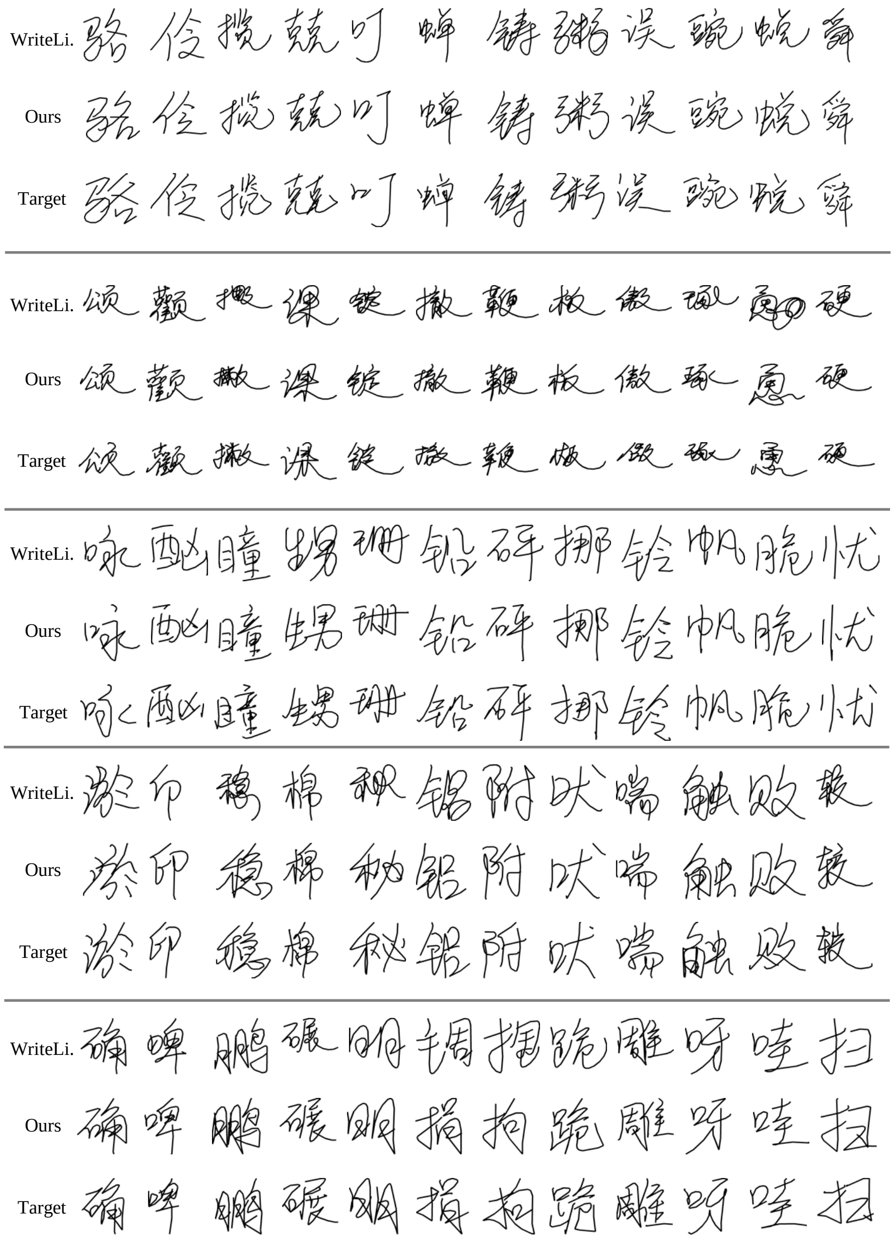

Figure 17: Additional generated online Chinese characters by our method and WriteLikeYou (Tang & Lian, 2021).

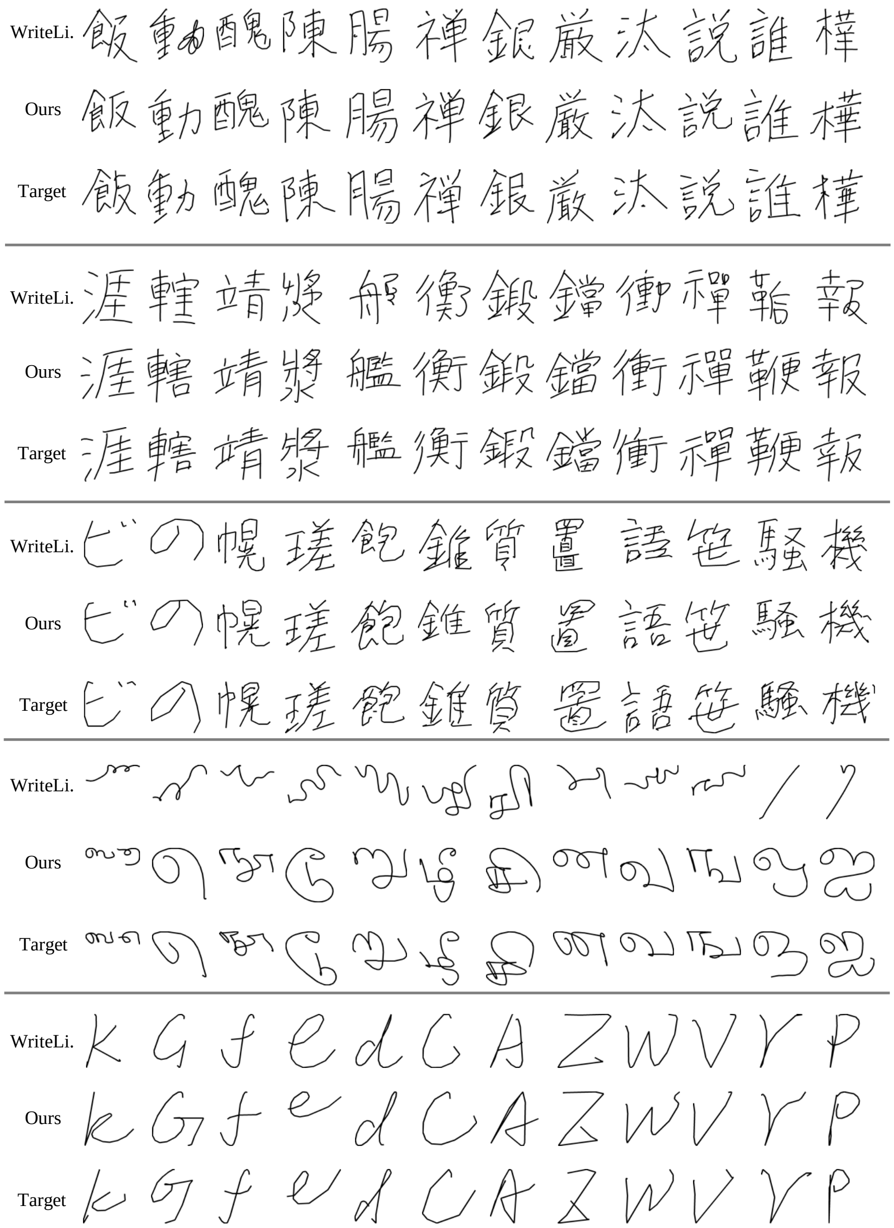

Figure 18: Additional generated online characters, covering Japanese, Indic and English scripts, by our method and WriteLikeYou (Tang & Lian, 2021).

A.5 More Visualisations on Spectrum Analysis.

In Fig. 19, we provide additional frequency magnitude visualizations for writer-wise and character-wise style representations, respectively. Clearly, the results indicate that character-wise styles focus on more high-frequency information, while writer-wise styles mainly pay attention to low-frequency information.

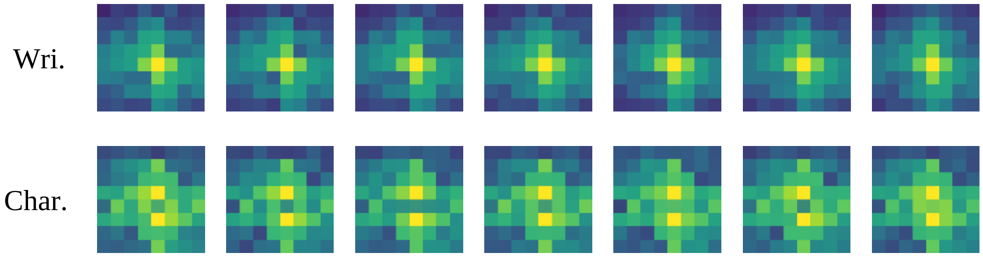

Figure 19: Frequency magnitude (8×8) belongs to 7 writers. The magnitude is averaged over 100 samples.