Disentangling Writer and Character Styles for Handwriting Generation Supplementary Material

Gang Dai[1,*], Yifan Zhang[2*], Qingfeng Wang[1], Qing Du[1], Zhuliang Yu[1],
Zhuoman Liu[3], Shuangping Huang[1,4†]
[1]South China University of Technology, [2]National University of Singapore,
[3]The Hong Kong Polytechnic University, [4]Pazhou Laboratory
eedaigang@mail.scut.edu.cn, yifan.zhang@u.nus.edu, eehsp@scut.edu.cn

A. Overview

We organize our supplementary material as follows.

- In Sec. A.1, we provide more implementation details.
- In Sec. A.2, we provide more visualization examples for spectrum analysis of two style representations.
- In Sec. A.3, we describe additional related works about handwriting generation and review the works in font generation.
- In Sec. A.4, we provide qualitative results of offline Chinese handwriting generation with a comparison to previous state-of-the-art works.
- In Sec. A.5, we study the effect of the sampling ratio α and compare different combination strategies in decoder based on online Chinese handwriting dataset.
- In Sec. A.6, we provide the discussions on the format of style inputs.
- In Sec. A.7, we conduct failure case analysis.
- In Sec. A.8, we report more evaluation metrics on Japanese dataset.
- In Sec. A.9, we conudct more experiments on Indic dataset.
- In Sec. A.10, we give detailed data representations of online characters.
- In Sec. A.11, we describe more details about the pen moving prediction and pen state classification losses.
- In Sec. A.12, we show a large number of generated online samples, covering Chinese, Japanese, Indic and English scripts.

A.1. More Experimental Details.

A.1.1 Implementation Details of Metrics

DTW The lower DTW distance, the better quality of the generated characters. For a more robust evaluation metric, we normalize the DTW distance by the length of real online characters to eliminate the effects of different lengths.

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 [23], 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. 1 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 [34]. 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. 2.

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.1.2 Implementation Details of Robustness Training

After removing the redundant points of online characters, we follow [32] 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

*Authors contributed equally.
†Corresponding author

Table 1. Quantitative evaluations of four content recognizers on four datasets.

Datasets	Acc.(%)
Chinese [31]	95.43
Japanese [20]	93.61
Indic[1]	94.48
English [31]	80.12

Table 2. Quantitative evaluations of four writer identifiers on four datatsets.

Datasets	Acc.(%)
Chinese [31]	99.98
Japanese [20]	99.64
Indic[1]	72.54
English [31]	20.57

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 [9]. 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. Following [23], 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.1.3 Implementation Details of Baseline Methods

Drawing&FontRNN As mentioned in Sec. 4.1, we re-implement the variants of Drawing [32] and FontRNN [24] by adding a style branch proposed in DeepImitator [34]. Specifically, the variants first leverage the CNN encoder [34] to extract the style vector from the style images, then, following DeepImitator [34], they concatenate the obtained vector with the desired character embedding, which is finally fed into their decoder to generate stylized online handwritings.

WriteLikeYou-v1 adopts the CNN backbone [34] as its content and style encoder.

WriteLikeYou-v2 employs the CNN-Transformer architecture as its content and style encoder. Each encoder is a sequential combination of a standard Resnet18 [10] and a transformer consisting of 2 standard self-attentions layers [27].

A.1.4 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 distance between $\mathbf{a}^i$ and $\mathbf{b}^j$ as:

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

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:

$$c_{ij}=d_{ave}(\mathbf{a}^i,\mathbf{b}^j). \tag{2}$$

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.1.5 Dataset Details

Japanese Dataset TUAT HANDS [20] database 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 [3]. For a fair comparison with the previous RNN-based works [34], we drop characters with points more than 150, accounting for about 2% of the total datasets [23]. 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[1] 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 [32] 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[2] Format. Briefly, each character embedding is a latent vector embedded by a class label.

English Dataset In total, we have 53,248 English characters [18] written by 1,020 persons for training, and 3,120 characters [31] 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

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

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

coordinate points of online characters and employ printed English font as content images.

A.2. More Visualisations on Spectrum Analysis.

In Fig. 1, 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.

A.3. More Related Work

Additional Handwriting Generation Works Early traditional approaches are mainly designed to generate Latin characters. Two-step methods [17,28] 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 authentic handwritings [5, 8] conditioned on desired content labels. But these methods are unable to imitate the calligraphic styles of reference samples. DeepWriteSyn [26], Sketchformer [22] and CoSE [1] condition the generative process on online handwriting trajectories. Specifically, DeepWriteSYN [26] introduces the Variational Autoencoder to synthesize realistic forgeries based on given genuine handwritten signatures. Transformer-based methods (*e.g.*, Skecthformer [22] and Cose [1]) adopt the encoder-decoder architecture for reconstructing hand-drawn sketches. However, it is difficult to generalize these methods [1, 22, 26] to multi-style handwriting generation tasks. Specifically, they are confused about using which handwriting style to decorate the given textual content since they lack specific style guidance.

As for handwritten Chinese characters, some previous methods [15, 16, 35] 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.

Font Generation Generative Adversarial Networks (GANs) [6] open a new door for font generation and bring amazing performance gains. zi2zi [25] regards font generation as an image translation task and achieves diverse font style transfer via a condition GAN. MC-GAN [2] generates the whole set of letters with a consistent style by observing only a few examples via the proposed glyph generation network and texture transfer module. Later, EMD [33] and TET-GAN [30] 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 [12, 19, 21] or supervisions [11, 14]. However, all of the above works do not explicitly consider the geometric deformation of fonts. DG-font [29] 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. 3.

A.4. Offline Chinese Handwriting 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 [31], 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 [12]. In our experiments, we resize input images to 64×64. We insert an extra ornamentation network [29] following the proposed SDT to constitute our offline handwriting generation method. More specifically, our offline method adopts the following pipeline: first generating online handwritings with large shape changes conditioned on input images via the proposed SDT, and rendering offline character images by connecting coordinate points in generated handwriting trajectories, finally decorating the offline characters with realistic stroke width, ink-blot, etc. via the ornamentation network [29].

We compare our offline generation method with popular font generation and handwriting image generation works. Specifically, (1) font generation methods include zi2zi [25] and DG-FONT [29]. (2) handwriting image generation methods, such as GANWriting [13] and HWT [4] are considered compared methods.

Qualitative Comparison. Fig. 3 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 method (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 method, other methods struggle to synthesize the stroke width and ink-blot similar to the target characters.

Further, we provide more qualitative results with a com-

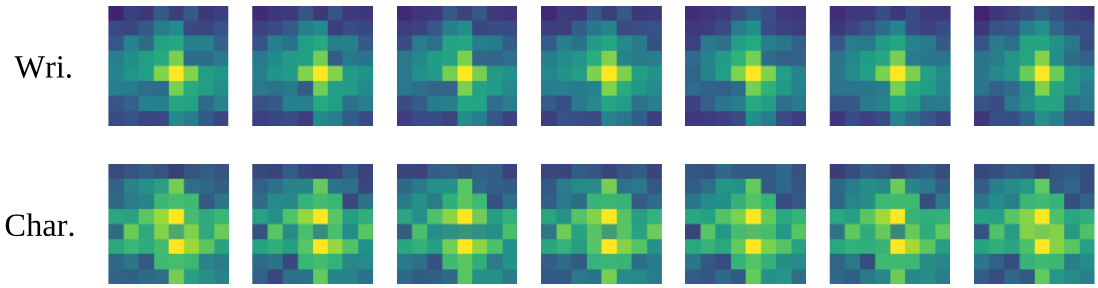

Figure 1. Frequency magnitude visualizations belong to 7 writers. Each spectrum map is averaged over 100 Chinese character samples.

Table 3. Effect of sampling ratio α. Our SDT works well with a low sampling ratio (25%).

Sampling Ratio	0.25	0.50	0.75	1.00
Style Score ↑	**94.50**	92.07	91.91	91.54

Table 4. 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(SDT)	**94.50**	**97.04**	**0.8789**

parison to GANWriting and DG-Font in Fig. 4-Fig. 7.

A.5. More Ablation Studies on Online Chinese Handwriting Dataset.

Effect of sampling ratio α. We conduct ablation studies to explore the effect of sampling ratio α on the test set. Tab. 3 summarizes the experimental results in terms of the Style Score. From these results, we observe that a relatively low sampling ratio α (i.e., 25%) achieves the best performance. The results indicate that the lower α guides our SDT to focus on the more fine-grained style pattern, which contributes to improving the generation performance of the proposed method in terms of the style imitation.

Evaluation of different combinations between the content feature q and previous points $\{y_j\}_{j=1}^{t-1}$. To evaluate the effect of different combination strategies between q and $\{y_j\}_{j=1}^{t-1}$, 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. 4, 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 [34] that suffer from the forgetting phenomenon [7], which demonstrates the effectiveness of the proposed method.

Evaluation of different combinations between the content feature q and style representations, i.e., E and G. To demonstrate the effectiveness of our attention-based combination strategy (as mentioned in Sec. 3.3) between q, E and H, we realize a new version of our SDT by directly concatenating q with E and H. Specifically, the new version takes previous points as the query vector, which then attends to combined content and style features. The experimental results are reported in Tab. 5. From these results, our method obtains better performance in terms of three quantitative metrics, especially a 16.38% improvement in Style Score, embodying the superiority of our SDT.

Table 5. Effect of different combinations between q, E and G.

Combination	Style Score ↑	Content Score↑	DTW ↓
Concat	78.12	96.88	0.8933
Ours(SDT)	**94.50**	**97.04**	**0.8789**

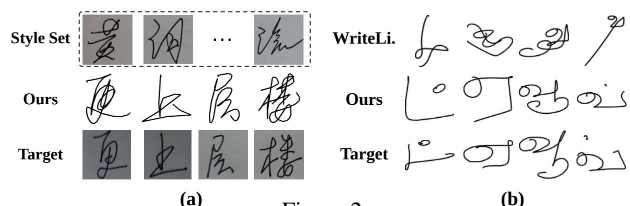

Figure 2

A.6. Discussions on the format of style inputs

As described in Sec.1, online trajectories contain more style information (*e.g.*, the order of writing). However, getting trajectories requires users to use specific equipment (*e.g.*, tablets and electric pens), making the methods (that use trajectories as inputs, *e.g.*, WriteLikeYou [23]) non-portable in real applications [41]. Instead, we explore offline images as inputs, which are easier for users to obtain (*e.g.*, through phones). As shown in Fig. 2 (a), by taking some pictures of the user's handwriting set as the style reference, our method can readily generate the target-stylized online characters. In addition, compared with original WriteLikeYou [23], our method receives less input information, but we still achieve comparable performance with it, in terms of Style Score (94.50% *vs*. 92.85%) and Content Score (97.04% *vs*. 97.92%). Overall, our method has better applicability in real scenarios.

Table 6. Additional quantitative evaluations of our SDT and competitors on Japanese dataset.

Methods	Style Score ↑
Drawing [32]	20.67
DeepImitator [34]	25.80
WriteLikeYou-v2 [23]	32.88
SDT(Ours)	**41.85**

A.7. Analysis of failure cases

We provide some failure cases in Fig. 2 (b), where our SDT fails to imitate the overall style (e.g., glyph slant and aspect ratios) for some Indic characters. However, thanks to our disentangling scheme, SDT can still capture their detailed style (e.g., stroke location and curvature), while previous methods (e.g., WriteLikeYou) cannot imitate the style of target characters at all.

A.8. More Evaluation metrics on Japanese Dataset.

We report the experimental results on Japanese dataset in Tab. 6, in terms of Style Score. From these results, our SDT outperforms the second best, i.e., WriteLikeYou-v2 [23], by a large margin (41.85% *vs*. 32.88%), which further demonstrates our method has a better imitation performance in respect of handwriting styles regardless of the script type.

A.9. More Experiments on Indic Dataset.

As described in Sec. 4.3, previous works [23,32] achieve very poor generation results (*e.g*., Content Score of 0.02) in Indic scripts, which means generating Indic handwritings with certain contents and specific styles may be too difficult for them. To this end, we reduce the difficulty of the Indic handwriting generation task and condition the generative process only on character contents. Quantitative comparison further demonstrates the superiority of our method. We detail the experimental setting and results below.

For this new task, we still conduct experiments on Tamil[1] dataset to compare our method with other competitors (i.e., Drawing [32] and WriteLikeYou [23]). For a fair comparison, without any style reference, all methods take input as the content reference (i.e., character embeddings [32]) and aim to synthesize handwritings consistent with the given content. Since Drawing [32] is initially designed for content-conditioned generation, we keep its original architecture. For WriteLikeYou [23] and our SDT, we remove their style and content encoder and directly input character embeddings into their decoder.

We provide the experiment results in Tab. 7. From these results, although Drawing [32] and WriteLikeYou [23] achieve the better performance than their content-style-conditioned generation settings (as shown in Sec. 4.3), our SDT still achieves the best results in terms of Content Score and DTW. Besides, compared with the content-conditioned generation, our content-style-conditioned results (as shown in Sec. 4.3) obtain higher Content Score (i.e, 97.22% *vs*. 68.50%) and better DTW (i.e., 0.7075 *vs*. 0.9748), which demonstrates that the extracted style representations by our method further improve the generation quality in terms of Content Score and DTW.

Table 7. Quantitative evaluations of our SDT and competitors on content-conditioned generation of Indic handwritings.

Methods	Content Score ↑	DTW ↓
Drawing [32]	26.07	2.7604
WriteLikeYou [23]	40.29	1.0503
Ours(SDT)	**68.50**	**0.9748**

A.10. Data Representations of Online Characters

Generally, each online character is composed of a sequence of points and can be mathematically represented as $Y=[y_1,...,y_L]$, where L is the length of Y. Following [32], 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.

A.11. More Details about the Pen Moving Prediction and Pen State Classification Losses

During training, we have the ground-truth point $y_t = (\Delta u, \Delta v, m_1, m_2, m_3)$ and the final output $O_t = (\left\{\hat{\pi}^i, \hat{\mu}_x^i, \hat{\mu}_y^i, \hat{\delta}_x^i, \hat{\delta}_y^i, \hat{\rho}_{xy}^i\right\}_{i=1}^R, \hat{m}_1, \hat{m}_2, \hat{m}_3)$ of our decoder at any time step t. Here, $\hat{\pi}^i$ is the component weight of different bivariate normal distributions, $\hat{\mu}_x^i$ and $\hat{\mu}_y^i$ are the means of distributions, $\hat{\delta}_x^i$, $\hat{\delta}_y^i$ denotes the standard deviations of distributions and $\hat{\rho}_{xy}^i$ is the covariance, as suggested in [23]. Then, the pen moving prediction loss for each time step can be formulated as:

$$\mathcal{L}_{pre} = \sum_{i=1}^{R} \hat{\pi}^i \mathcal{N}\left(\Delta u, \Delta v \mid \hat{\mu}_x^i, \hat{\mu}_y^i, \hat{\delta}_x^i, \hat{\delta}_y^i, \hat{\rho}_{xy}^i\right), \tag{3}$$

where $\mathcal{N}(\cdot)$ is the bivariate normal distribution function.

Regarding the pen state classification loss at any time step, we formulate it as follows:

$$\mathcal{L}_{cls} = \sum_{i=1}^{3} m_i \log \hat{m}_i. \tag{4}$$

A.12. More Online Generation Results.

Fig. 8-Fig. 11 show qualitative comparisons between our proposed SDT and the previous state-of-the-art work WriteLikeYou [23] 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 3. Qualitative comparisons between our proposed SDT with four competitors, including zi2zi [25], DG-FONT [29], GANWriting [13] and HWT [4], on offline handwritten Chinese character generation.

Figure 4. Qualitative comparisons between our proposed SDT with DG-FONT [29] and GANWriting [13], on offline handwritten Chinese character generation.

Figure 5. Qualitative comparisons between our proposed SDT with DG-FONT [29] and GANWriting [13], on offline handwritten Chinese character generation.

Figure 6. Qualitative comparisons between our proposed SDT with DG-FONT [29] and GANWriting [13], on offline handwritten Chinese character generation.

Figure 7. Qualitative comparisons between our proposed SDT with DG-FONT [29] and GANWriting [13], on offline handwritten Chinese character generation.

Figure 8. Additional generated online Chinese characters by our method and WriteLikeYou-v2 [23].

Figure 9. Additional generated online Chinese characters by our method and WriteLikeYou-v2 [23].

Figure 10. Additional generated online Chinese characters by our method and WriteLikeYou-v2 [23].

Figure 11. Additional generated online characters, covering Japanese, Indic and English scripts, by our method and WriteLikeYou-v2 [23].

References

[1] Emre Aksan, Thomas Deselaers, Andrea Tagliasacchi, and Otmar Hilliges. Cose: Compositional stroke embeddings. In *Advances in Neural Information Processing Systems*, pages 10041–10052, 2020. 3

[2] Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer. In *Computer Vision and Pattern Recognition*, pages 7564–7573, 2018. 3

[3] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, pages 157–166, 1994. 2

[4] Ankan Kumar Bhunia, Salman Khan, Hisham Cholakkal, Rao Muhammad Anwer, Fahad Shahbaz Khan, and Mubarak Shah. Handwriting transformers. In *International Conference on Computer Vision*, pages 1086–1094, 2021. 3, 6

[5] 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*, pages 4324–4333, 2020. 3

[6] 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*, pages 2672–2680, 2014. 3

[7] Alex Graves. *Long Short-Term Memory*. Springer Berlin Heidelberg, 2012. 4

[8] Alex Graves. Generating sequences with recurrent neural networks. *Arxiv*, 2013. 3

[9] David Ha and Douglas Eck. A neural representation of sketch drawings. In *International Conference on Learning Representations*, 2018. 2

[10] 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*, pages 770–778, 2016. 2

[11] 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*, pages 156–172, 2020. 3

[12] 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*, pages 4015–4022, 2019. 3

[13] 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*, pages 273–289, 2020. 3, 6, 7, 8, 9, 10

[14] 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*, pages 13482–13491, 2022. 3

[15] Zhouhui Lian, Bo Zhao, and Jianguo Xiao. Automatic generation of large-scale handwriting fonts via style learning. In *SIGGRAPH Asia Technical Briefs*, pages 1–4, 2016. 3

[16] 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*, pages 1–5, 2015. 3

[17] Zhouchen Lin and Liang Wan. Style-preserving english handwriting synthesis. *Pattern Recognition*, 40(7):2097–2109, 2007. 3

[18] Cheng-Lin Liu, Fei Yin, Da-Han Wang, and Qiu-Feng Wang. Casia online and offline chinese handwriting databases. In *International Conference on Document Analysis and Recognition*, pages 37–41, 2011. 2

[19] 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*, pages 7905–7914, 2022. 3

[20] 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*, pages 496–500, 2001. 2

[21] 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*, pages 2393–2402, 2021. 3

[22] Leo Sampaio Ferraz Ribeiro, Tu Bui, John Collomosse, and Moacir Ponti. Sketchformer: Transformer-based representation for sketched structure. In *Computer Vision and Pattern Recognition*, pages 14153–14162, 2020. 3

[23] 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. 1, 2, 4, 5, 11, 12, 13, 14

[24] 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. 2

[25] Yuchen Tian. zi2zi:master chinese calligraphy with conditional adversarial networks. `https://github.com/kaonashi-tyc/zi2zi`, 2017. 3, 6

[26] Ruben Tolosana, Paula Delgado-Santos, Andres Perez-Uribe, Ruben Vera-Rodriguez, Julian Fierrez, and Aythami Morales. Deepwritesyn: On-line handwriting synthesis via deep short-term representations. In *AAAI Conference on Artificial Intelligence*, pages 600–608, 2021. 3

[27] 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*, pages 5998–6008, 2017. 2

[28] 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*, pages 157–162, 2002. 3

[29] Yangchen Xie, Xinyuan Chen, Li Sun, and Yue Lu. Dg-font: Deformable generative networks for unsupervised font generation. In *Computer Vision and Pattern Recognition*, pages 5130–5140, 2021. 3, 6, 7, 8, 9, 10

[30] Shuai Yang, Jiaying Liu, Wenjing Wang, and Zongming Guo. Tet-gan: Text effects transfer via stylization and destylization. In *AAAI Conference on Artificial Intelligence*, pages 1238–1245, 2019. 3

[31] Fei Yin, Qiu-Feng Wang, Xu-Yao Zhang, and Cheng-Lin Liu. Icdar 2013 chinese handwriting recognition competition. In *International Conference on Document Analysis and Recognition*, pages 1464–1470, 2013. 2, 3

[32] 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. 1, 2, 5

[33] Yexun Zhang, Ya Zhang, and Wenbin Cai. Separating style and content for generalized style transfer. In *Computer Vision and Pattern Recognition*, pages 8447–8455, 2018. 3

[34] 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. 1, 2, 4, 5

[35] Alfred Zong and Yuke Zhu. Strokebank: Automating personalized chinese handwriting generation. In *AAAI Conference on Artificial Intelligence*, 2014. 3