Look Closer to Supervise Better: One-Shot Font Generation via Component-Based Discriminator

Supplementary Material

A. Implementation

A.1. Training details

The model is optimized using Adam with the settings of β_1=0.5 and β_2=0.999. All modules are trained from scratch with a learning rate of 0.0001. We initialize the weights of convolutional and linear layers with a Guassian distribution $\mathcal{N}$(0,0.02). The batch size is set to 16 in all experiments. Our method is implemented in PyTorch and all experiments are conducted on a single NVIDIA 1080Ti GPU.

Chinese font generation All the images are resized to 128×128 pixels. The learning rate is initially set to 0.0001 and linearly decreased to zero after 40 epochs.

Handwriting generation The images are resized to a height of 64 pixels, and the width is calculated with the original aspect ratio (up to 384 pixels). We keep the learning rate as 0.0001 for the first 15 epochs and linearly decay the rate to zero over the next 30 epochs.

Scene text editing For the scene text editing task, the model is trained with synthetic data and evaluated on real-world scene text image data. Specifically, we generate 1.4M synthetic data $(I_s, I_s^{'})$ with the synthesizing engine SynthTiGER [10], where I_s and $I_s^{'}$ have different textual content $(T, T^{'})$ respectively but other image properties such as background, font, *etc*. remain the same. In the training process, we use I_s as the style reference input and meanwhile, render the textual content $T^{'}$ into a content reference image, which is used as the content reference input. Since the scene text image lacks a style label, we set the style retention loss to zero and add the perceptual loss [1] and the spatially-correlative loss [12] on the basis of the original training objectives. The test set is sampled from regular and irregular scene text datasets, including IIIT5k [5], SVT [9], IC03 [4], IC13 [3], SVT-P [6], CUTE80 [7] and IC15 [2], with a total of 9,350 real-world scene text images. All the images are resized to 64×256 pixels and the model is trained for 20 epochs with a learning rate of 0.0001.

A.2. Network architectures

Generator architecture Our generator is built upon the ResNet architecture of [11], and is further extended with our proposed changes. The original generator of [11] is an encoder-decoder architecture. In order to obtain the font generator, we adopt the original encoder architecture as our style encoder and content encoder, while using the original decoder architecture as our mixer, with a channel multiplier $ch = 64$. Specifically, the style encoder and the content encoder have the same architecture, consisting of five ResNet down-sampling blocks with a total down-sampling rate of 32. In the mixer, the encoded features are upsampled via five ResNet up-sampling blocks until the original image resolution is reached. To produce the $3 \times H \times W$ output image, an InstanceNorm-ReLU-conv2d block with output channel 3 is additionally appended as the last layer of the mixer. We remove the self-attention layer in all ResNet blocks and add AdaIN operation as the normalization layer in every up-sampling block of the mixer.

CAM architecture The proposed CAM aims to supervise the generator at the component level. The detailed architecture of the CAM is shown in Table 1.

Discriminator architecture For the discriminator networks, we adopt a U-Net based discriminator [8]. Specifically, We adopt the U-Net discriminator architecture of the 128×128 resolution with a channel multiplier $ch = 16$.

B. Additional qualitative results

In this section, We present more qualitative results and ablation study results to better validate the effectiveness of our proposed method.

B.1. One-shot font generation

In Figure 1 and Figure 2, we present more generated samples in two scenarios: seen styles and unseen styles, respectively. Specifically, we randomly select 30 seen fonts and 20 unseen fonts from the two Chinese glyph test sets, and randomly sample 10 unseen target glyphs for each font to carry out the qualitative evaluation. Note that all the generated glyphs are tested in a one-shot setting, with one single style reference image provided. The results show that CG-GAN can generate high-quality glyph images in both scenarios, suggesting the superior one-shot font generation ability. Figure 3 shows that our model is able to extend to cross-lingual font generation. The model is trained on Chinese fonts but is able to generate a complete Korean font library in inference.

B.2. Latent space interpolations

In Figure 5, we perform a linear style interpolation between two random styles on the IAM dataset. We can observe that the generated image can smoothly change from one style to another, while strictly preserving its textual content. The results indicate that CG-GAN can generalize in the style latent space rather than memorizing some specific style patterns. Besides, we present some synthetic

Table 1. **CAM architecture**. BN denotes the batch normalization, and IN denotes the Instance normalization

	Operation	Kernel size	Resample	Padding	Feature maps	Normalization	Nonlinearity
Feature encoder	Convolution	7	MaxPool	3	96	BN	PReLU
	Convolution	3	MaxPool	1	128	BN	PReLU
	Convolution	3	MaxPool	1	160	BN	PReLU
	Convolution	3	-	1	256	BN	PReLU
	Convolution	3	MaxPool	1	256	BN	PReLU
Attention decoder			256 hidden units, 256 GRU units				
Style classifier	Convolution	3	MaxPool	1	256	IN	PReLU
	Convolution	3	-	1	512	IN	PReLU
	Convolution	3	MaxPool	1	512	IN	PReLU
	Convolution	3	-	1	n styles	-	-
Component-wise discriminator	Convolution	3	MaxPool	1	128	IN	PReLU
	Convolution	3	MaxPool	1	64	IN	PReLU
	Convolution	3	-	1	16	IN	PReLU
	Convolution	3	-	1	1	-	-

word images with various calligraphic styles in Figure 4, where each row presents diverse generated samples in the same calligraphic style.

B.3. Scene text editing

In Figure 6, we present more scene text editing results. As we can observe, our model can robustly edit textual contents with different lengths, and achieve promising results even in challenging cases, such as complex backgrounds or slanted or curved texts.

B.4. Additional ablation results

Influence of the style latent vector In this part, we trained a variant where the AdaIN operation including the style latent vector f_s is removed. Results are shown in Table 2. It is noted that there is only a slight drop in performance, indicating that the style latent vector f_s is not that necessary. Such results partly reflect our primary purpose, that is, the performance improvement is mainly gained by providing more effective supervision for the generator, not by struggling to increase the complexity of the generator.

Influence of the U-net Discriminator We further investigate the influence of the U-net architecture of the discriminator. Specifically, we trained a variant where the U-net architecture of the discriminator is removed, only the encoder part D_{enc} is preserved. For font generation and handwriting generation tasks, we set the channel multiplier ch of the discriminator to 16 and 64, respectively. As shown in Table 3, the performance of the variant is comparable to our current approach on the font generation task, which still outperforms all the other baselines in all metrics. And the variant is also competitive on handwriting generation task, as shown in Table 4. The results indicate that decoder part D_{dec} has no significant effect on the performance. This may be due to the simple background of the dataset, which contains a lot of pixels with values (255,255,255), thus the pixel-level discrimination performed by D_{dec} may not be so effective.

Table 2. The impact of the style latent vector on the Chinese font generation task.

Method	SSIM↑	RMSE↓	LPIPS↓	FID↓
CG-GAN (ours)	**0.7568**	**0.0218**	**0.2058**	**17.94**
w/o style latent	0.7549	0.0225	0.2193	18.73

Table 3. The impact of the U-net architecture of the discriminator on the Chinese font generation task.

Method	SSIM↑	RMSE↓	LPIPS↓	FID↓
Seen styles and Unseen contents				
CG-GAN (ours)	0.7703	0.0212	0.1919	**6.54**
w/o D_{dec}	**0.7795**	**0.0207**	**0.1821**	7.14
Unseen styles and Unseen contents				
CG-GAN (ours)	0.7568	0.0218	0.2058	**17.94**
w/o D_{dec}	**0.7603**	**0.0214**	**0.1967**	19.07

Table 4. The impact of the U-net architecture of the discriminator on the writer-relevant handwriting generation task.

	IV-S	IV-U	OOV-S	OOV-U
CG-GAN (ours)	102.18	**110.07**	104.81	113.01
w/o D_{dec}	**101.48**	111.29	**102.67**	**112.77**

Figure 1. Seen styles and unseen contents in Chinese one-shot font generation.

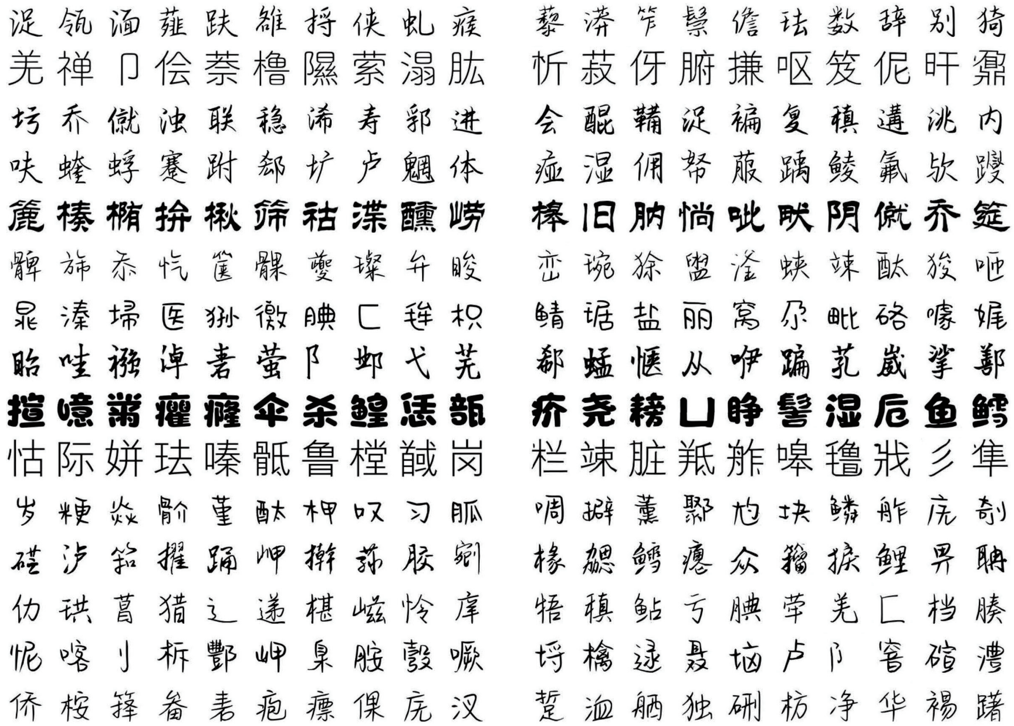

Figure 2. Unseen styles and unseen contents in Chinese ont-shot font generation.

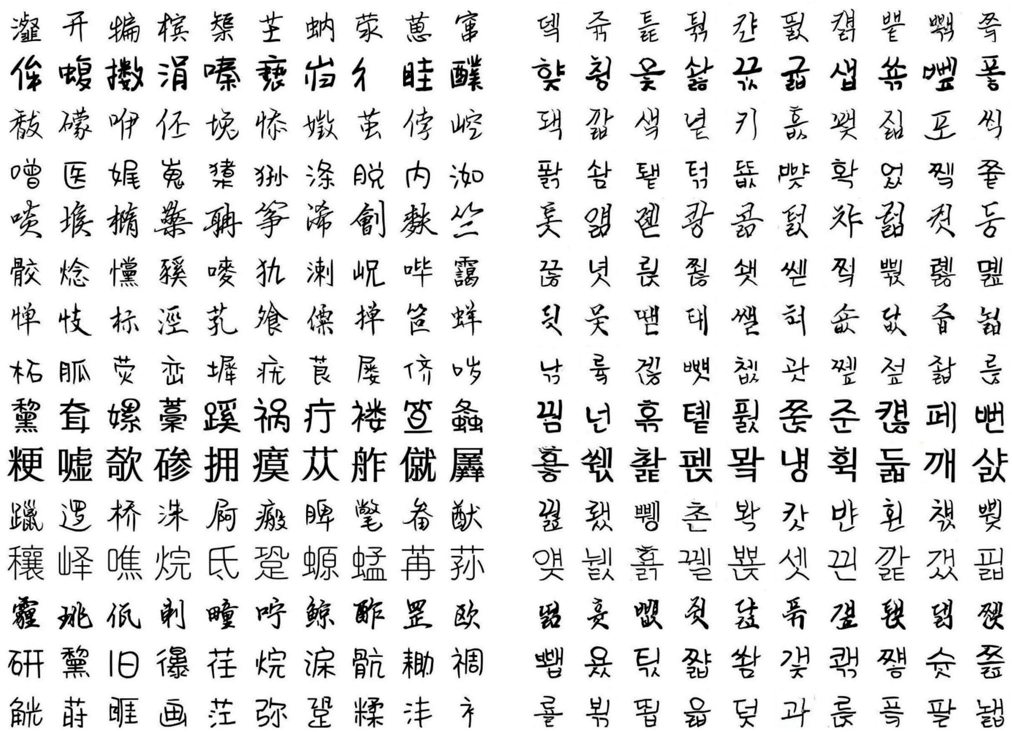

Figure 3. Cross lingual font generation.

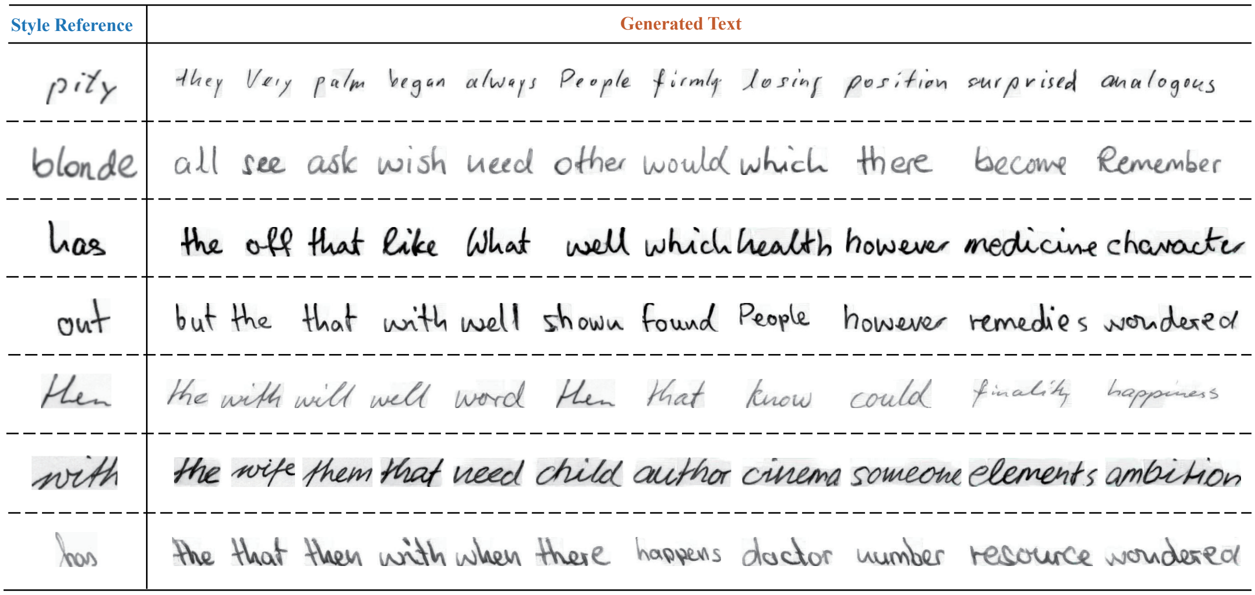

Figure 4. Visual comparison for synthesizing handwritten words.

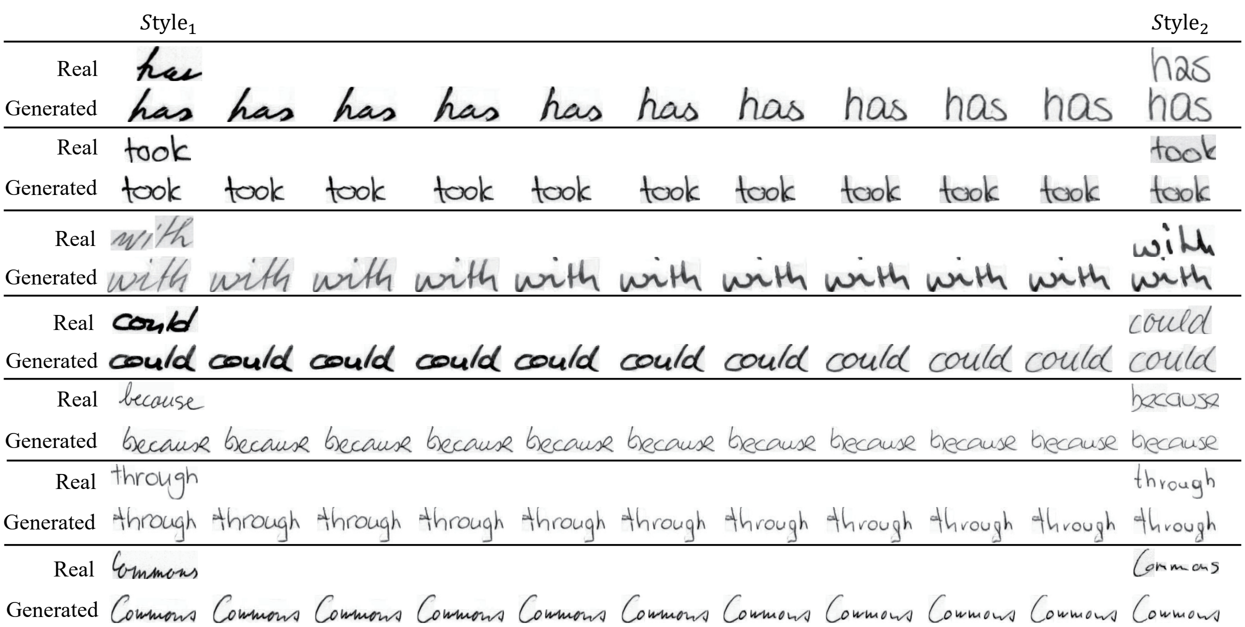

Figure 5. Style interpolation between two different styles.

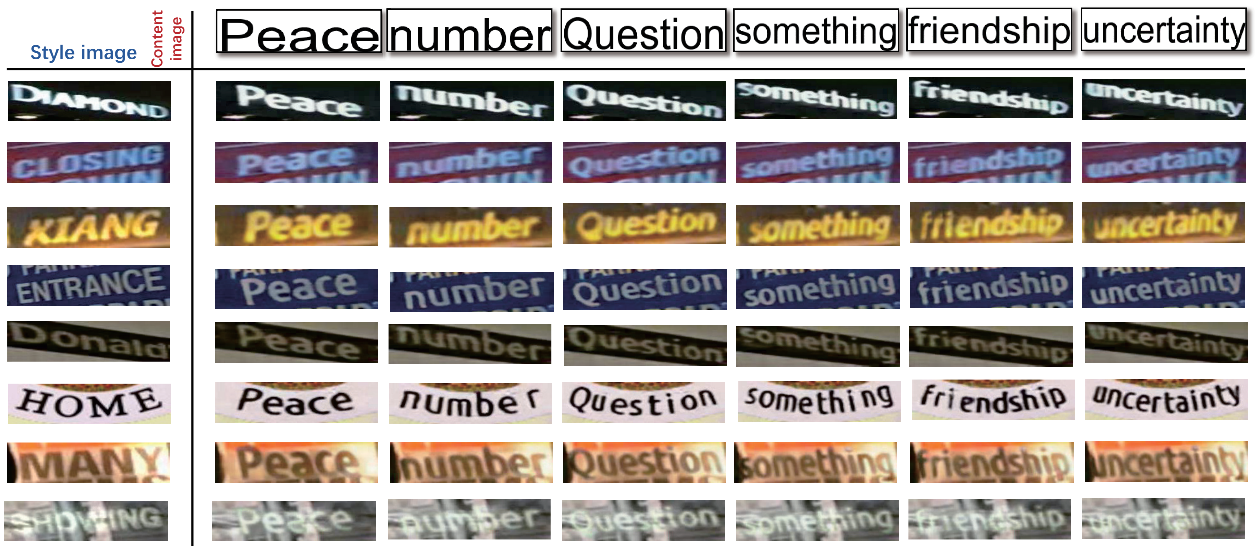

Figure 6. Additional scene text editing results.

References

[1] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, pages 694–711, 2016. 1

[2] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. ICDAR 2015 competition on robust reading. In *ICDAR*, pages 1156–1160, 2015. 1

[3] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. Icdar 2013 robust reading competition. In *ICDAR*, pages 1484–1493, 2013. 1

[4] Simon M Lucas, Alex Panaretos, Luis Sosa, Anthony Tang, Shirley Wong, Robert Young, Kazuki Ashida, Hiroki Nagai, Masayuki Okamoto, Hiroaki Yamamoto, et al. ICDAR 2003 robust reading competitions: entries, results, and future directions. In *ICDAR*, pages 105–122, 2003. 1

[5] Anand Mishra, Karteek Alahari, and CV Jawahar. Scene text recognition using higher order language priors. In *BMVC*, 2012. 1

[6] Trung Quy Phan, Palaiahnakote Shivakumara, Shangxuan Tian, and Chew Lim Tan. Recognizing text with perspective distortion in natural scenes. In *ICCV*, pages 569–576, 2013. 1

[7] Anhar Risnumawan, Palaiahankote Shivakumara, Chee Seng Chan, and Chew Lim Tan. A robust arbitrary text detection system for natural scene images. *Expert Systems with Applications*, pages 8027–8048, 2014. 1

[8] Edgar Schonfeld, Bernt Schiele, and Anna Khoreva. A U-Net Based Discriminator for Generative Adversarial Networks. In *CVPR*, pages 8207–8216, 2020. 1

[9] Kai Wang, Boris Babenko, and Serge Belongie. End-to-end scene text recognition. In *ICCV*, pages 1457–1464, 2011. 1

[10] Moonbin Yim, Yoonsik Kim, Han-Cheol Cho, and Sungrae Park. SynthTIGER: Synthetic Text Image GEneratoR Towards Better Text Recognition Models. In *ICDAR*, pages 109–124, 2021. 1

[11] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-Attention Generative Adversarial Networks. In *ICML*, pages 7354–7363, 2019. 1

[12] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. The spatially-correlative loss for various image translation tasks. In *CVPR*, pages 16407–16417, 2021. 1