

SAFont: Automatic Font Synthesis using Self-Attention Mechanisms

Chunying Ren, Shujing Lyu*, Hongjian Zhan, and Yue Lu

Shanghai Key Laboratory of Multidimensional Information Processing,
Department of Computer Science and Technology, East China Normal University,
Shanghai 200062, China

{cyren,hjzhan}@stu.ecnu.edu.cn, {ylu,sjlv}@cs.ecnu.edu.cn

Abstract. Automatic font synthesis, to save much time and labor, is an appealing solution for new font design. Although existing methods obtain success in font transformation to some extent, the quality of generated characters is not entirely satisfactory. Inspired by the superior performance of self-attention mechanisms in image generation task, this paper proposes a new model called SAFont for font transformation. The introduction of self-attention mechanisms is capable of modeling connections of strokes in a Chinese character. In addition, the definition of Edge Loss compels the model to generate characters with sharper edges. We show that SAFont is an effective model for font transformation from the experimental results.

Keywords: Automatic font synthesis · Self-attention mechanisms · Edge Loss.

1 Introduction

Chinese characters have been used continually over three millennia by more than a quarter of the world's population [1]. The use of Chinese characters is closely related to people's everyday life in East Asia. Recently, more and more netizens prefer to use Chinese characters with artistic or calligraphy style when they are online as this can show their distinctive style and taste. However, designing a new Chinese font with artistic or calligraphy style is a time-consuming and laborious task, requiring considerable efforts on manual design of benchmark characters [2]. Hence, automatic font synthesis, i.e. synthesizing characters of a certain font given limited manually designed samples is a meaningful task.

Most previous works [3, 4] leverage the hierarchical representation of simple strokes to learn font transformation. They first extract the representation of strokes from some existent characters and then combine strokes to generate nonexistent characters. The main problem of these methods is that the manual representations rely on preceding structural segmentation of characters which are severely affected by the prior knowledge. In addition, these methods focus on the local representation of the characters and ignore the overall style of characters.

Recently, several researchers model font synthesis as an image-to-image transformation problem. End-to-end methods based on Generative Adversarial Networks (GANs) [5] open up new horizons for the direction of font synthesis. On the one hand, directly transforming fonts end-to-end gets rid of the dependence on the hierarchical representation of simple strokes. On the other hand, owing to the ideology of adversarial learning which takes the overall style of Chinese characters into account, font synthesis has achieved remarkable progress. Zi2zi [6] and DCFont [7] learn to transform fonts based on pix2pix [8] using paired training data. Lyu et al. [9] employ two encoder-decoder networks to transform fonts where one is to learn font style representation and the other is to learn the style transfer. Chang et al. [2] design a hierarchical discriminator to provide more information for the generator and obtain remarkable success in Chinese font transformation.

However, these methods still have two obvious drawbacks:

- Fig.1 shows the uniqueness of the font with artistic or calligraphy style when combining strokes. The above mentioned methods [2, 6, 7, 9] only consider the migration of the overall style during font transformation. In fact, it can be noted that there are connections and changes among strokes rather than simply stacking strokes (cycled in red in Fig.1). However, these methods rarely take this relationship into account.
- For some complicated Chinese characters, their corresponding transformed results often loss part of local information. As we can see in Fig.2 (cycled in red), the generated results loss parts of strokes, or the edges of strokes are blurry.

To overcome the above problems, we propose a novel Chinese font transformation model, embedded with **self-attention mechanisms** [10] and **Edge Loss**. First, we introduce the self-attention mechanisms into font transformation models to capture geometric and structural patterns that exist consistently in some Chinese characters.

* Corresponding author.



Fig. 1. Some samples of Chinese characters with calligraphy style



Fig. 2. Some unsuccessful samples generated by previous models.

Zhang et al. [11] pointed out that convolutional GANs [14, 15] have difficulty in modeling some image classes when trained on multi-class datasets. Similar problems also exist at a multi-class database composed of Chinese characters. Different from the convolution operator, self-attention mechanisms are capable of helping the model build connections among strokes when transforming fonts [11]. With self-attention mechanisms, our model can generate results with more details. These details in a Chinese character (cycled in red in Fig.1) help to represent the style of Chinese characters. Second, to constrain the model to generate characters having sharper edges with less blur, we define a new loss called Edge Loss. Specially, the definition of Edge Loss introduces additional structural information into both generator and discriminator. The Edge Loss enables the discriminator to measure the discrepancy in distribution between the edges of generated and real characters, so that the generator is trained to produce results with less blur and shaper edges.

In summary, the main contribution of our work is to propose an end-to-end method for font synthesis, integrated with self-attention mechanisms and Edge Loss. We name our method SAFont. To our best knowledge, SAFont is the first attempt to integrate self-attention mechanisms into font transformation models. The experimental results show that we improve the quality of generated characters compared with state-of-the-art font synthesis algorithms. The generated Chinese characters obtained by SAFont maintain more details and shaper edges.

The remainder of the paper is organized as follows. We introduce the proposed SAFont for font transformation in Section 2. Then, the experimental results are illustrated in Section 3. Finally, we conclude the paper and discuss the future work in Section 4.

2 Methods

2.1 SAFont Network Architecture

Inspired by the successful experiences [6, 7] in designing a font transformation model, we design a novel font transformation model SAFont. SAFont consists of a style transfer network and a discriminator. The whole framework and specific structure details are shown in Fig.3.

Style Transfer Network: The style transfer network consists of five parts: the Content Encoder, the Style Transfer Module, the Generator and two Self-Attention Blocks. The Content Encoder and the Generator follow the Conv-InstanceNorm [12]-ReLU architecture and the Deconv-InstanceNorm-ReLU architecture, respectively. We replace ReLU with Tanh as the activation of the last layer of the Generator. The Style Transfer Module is stacked with six Res-Blocks which share the same architecture with ResNet [13]. Self-Attention Blocks are used for calculating the attention between feature maps from previous layers. We discuss them in detail in the next section. Simultaneously, to avoid losing location and structure information during down-sampling, we add several content skip-connections between the Content Encoder and the Generator to recapture the location and structure information.

As mentioned earlier, our model is end-to-end and the flow chart of our method is shown in Fig.3. First, the Content Encoder maps a specified source character to a content embedding. Then a self-attention block is used to calculate the attention between these content features. Features processed by self-attention mechanisms are fed into the Style Transfer Module. The outputs of the Style Transfer Module can be considered as the extracted features

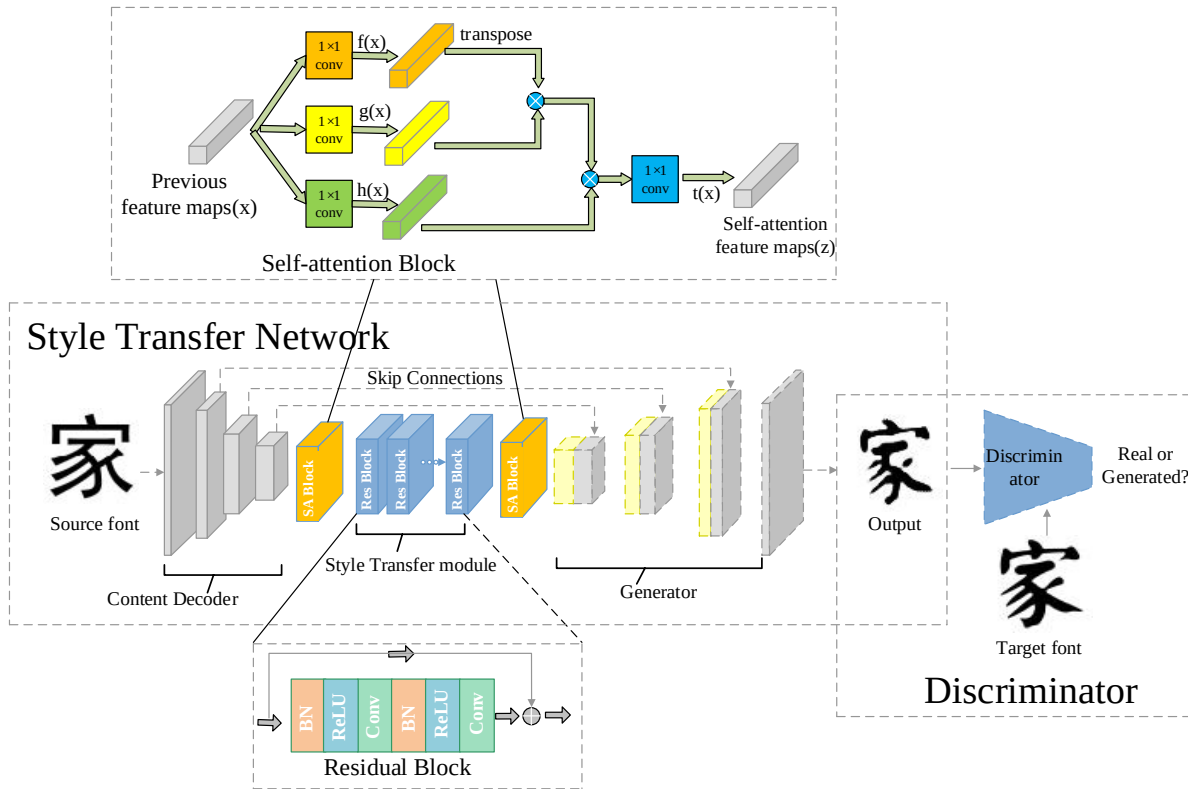


Fig. 3. The structure of SAFont. The middle part of the figure is the whole framework of SAFont, consisting of a **Style Transfer Network** and a **Discriminator**. In the top and bottle of the figure, we show the structure of **Self-Attention Block** and **Res-Block** in detail, respectively.

from the target domain. Similarly, we calculate the attention between the outputs of the Style Transfer Module again. Finally, all features are decoded via the Generator.

Discriminator: We employ a commonly used adversarial discriminator [8] which is stacked by several Conv-InstanceNorm-ReLU blocks to distinguish images from generated to real. The only difference is that we add an additional self-attention block in the middle of the discriminator to capture the relationships among strokes in a Chinese character.

2.2 Self-Attention Mechanisms

Since self-attention mechanisms are successfully applied in various tasks which must capture global dependencies, we keep and use the most of their advantages.

Most GAN-based models [14–16] for image generation are stacked with convolutional layers. However, convolution operator only focuses on local area of an whole image, thus it is inefficient to model long-range dependencies in images by using convolutional layers alone. Building a deeper convolutional network can improve this problem while it makes the network to be more difficult to train and slow to converge. In order to address this issue, self-attention [10, 11], also called intra-attention, is proposed to calculate the response at a position in a sequence by attending to all positions within the same sequence. SAGAN [11] is one of the first attempts to apply self-attention mechanisms in image generation. Owing to the adaptation of self-attention to the GAN framework, both generator and discriminator are capable of modeling relationships between widely separated spatial regions.

Self-attention mechanisms work as follows(described in the top of Fig.3): feature maps from previous hidden layer x are first transformed into two feature spaces f, g to calculate the attention, where f and g are two 1×1 convolution operators

$$\alpha_{j,i} = \frac{\exp(v_{ij})}{\sum_{i=1}^N \exp(v_{ij})}, \text{ where } v_{ij} = f(x_i)^T * g(x_j), \quad (1)$$

and $\alpha_{j,i}$ indicates the extent to which the model attends to the i^{th} location when synthesizing the j^{th} region. Then the output of attention layer is $z = (z_1, z_2, \dots, z_j, \dots, z_N)$ where

$$z_j = t\left(\sum_{i=1}^N \alpha_{j,i} h(x_i)\right) \quad (2)$$

and h and t are also two 1×1 convolution operators. Finally, the output of attention layer is multiplied by a scale parameter β and added back the input feature maps. The final output of SA-Block is represented as

$$y_i = \beta z_i + x_i \quad (3)$$

Note that four 1×1 convolution operators used for calculating the attention between feature maps add little extra parameters and computational complexity. Our method can model connections and changes among strokes during font transformation with the presence of self-attention mechanisms. With self-attention layers, we can obtain high quality characters with more details.

2.3 Loss Functions

Edge Loss: Pixel-level loss [2, 9] is widely used in existing font synthesis algorithms. All of them estimate the distribution consistency of two domains based on per-pixel discrepancy of generated and real characters. However, for font transformation task, the weights of pixels in a Chinese character image are different from each other. Contrary to the pixels which are used as background or filler, boundary pixels play a key role on determining the overall style of the Chinese characters. Hence, our model is demanded to pay more attention to the edge of each character. To avoid the failure cases shown in Fig.2 and reserve the edges of characters as much as possible, we define Edge Loss to constrain our model to generate results with shaper edges. We utilize Canny algorithm [20] to extract the edges of both generated and target images. L1-norm is used to measure the pixel distance between the two edges:

$$\mathcal{L}_{edge} = \mathbb{E}_{(x,y) \sim p_{data}} [\| C(G(x)) - C(y) \|_1] \quad (4)$$

where C denotes Canny algorithm, G denotes the generator, x and y are a pair of training sample from source and target domains, respectively.

Pixel-level Loss: Following by [2], we replace L1- or L2-norm with cross entropy function as per-pixel loss. Contrary to other image generation problems, each pixel in characters is normalized near 0 or 1 value in font transformation tasks. We regard our font transformation problem as a logistic regression problem. The pixel-level loss is defined as:

$$\mathcal{L}_{pix-wise}(G) = \mathbb{E}_{(x,y) \sim p_{data}} [-y(\log(\sigma(G(x)))) - (1-y)\log(1-\sigma(G(x)))] \quad (5)$$

where σ is sigmoid activation.

Adversarial loss: Since Goodfellow et al. [5] propose generative adversarial networks, adversarial losses have been successfully applied in various image generation tasks. The process of adversarial training helps the model to generate more authentic images. We select the hinge version [17–19] of adversarial loss to train our model:

$$\mathcal{L}_{GAN}(G) = -\mathbb{E}_{(x,y) \sim p_{data}} D(G(x)) \quad (6)$$

$$\begin{aligned} \mathcal{L}_{GAN}(D) = & -\mathbb{E}_{(x,y) \sim p_{data}} [\min(0, -1 + D(y))] - \\ & \mathbb{E}_{(x,y) \sim p_{data}} [\min(0, -1 - D(G(x)))] \end{aligned} \quad (7)$$

where D denotes the discriminator.

Finally, the total loss for the generation is:

$$\mathcal{L}_{total}(G) = \mathcal{L}_{GAN}(G) + \lambda_1 * \mathcal{L}_{edge} + \lambda_2 * \mathcal{L}_{pix-wise} \quad (8)$$

where λ_1 and λ_2 are used to balance the weights between the three losses.

3 Experiments

3.1 Datasets

All training data used in our experiments comes from the font website ¹. In our experiment, each font contains 3000 to 6000 grey-scale images in 64×64 format. These fonts are used as target fonts. We select the SimHei as source font. The training set consists of 2000 source characters and 2000 corresponding target characters from other styles, respectively. The remaining characters in source font are used for testing.

3.2 Performance Comparison

In this section, we illustrate our experimental results by transferring the font of SimHei to three fonts with other styles. In addition, we qualitatively and quantitatively evaluate the proposed method for Chinese font transformation problem.

Qualitative Results: To visually estimate the quality of these fonts generated by our method, we select several characters from each font to render a paragraph. For better demonstration, some generated characters are replaced by real characters and emphasized in red in Fig.4. The mixture of generated and real characters is beneficial to compare the style difference between each other.

In Fig.4, we can observe that the characters generated by our model are almost indistinguishable from real characters. Experimental results show that our model has the capability to transfer an existed printed font to other artistic or calligraphy fonts. More crucially, as mentioned earlier, our model has the capability to build connections and changes among strokes in a Chinese character. For example, in Fig.4 (c), there are full of connections and changes when transforming the font of SimHei to the font with a calligraphy style as a Chinese character in a calligraphy work is often completed by a continuous stroke. From the experimental results, SAFont performs well in modeling connections and changes which is like a calligrapher is creating calligraphy works.

Quantitative Results: We quantitatively compare the performance with three state-of-the-art methods for font transformation: AEGG [9], Zi2Zi [6] and HGAN [2]. Root Mean Square Error(RMSE) and Average Pixel Disagreement Ration(APDR) [21] are utilized as quantitative evaluation metrics. The results are shown in Table 1. It can be seen that SAFont achieves better results than AEGG and Zi2zi, and performs slightly better than HGAN.



Fig. 4. Parts of the Qualitative Results. (a) shows the text rendered using the source font SimHei. (b-d) show the corresponding texts rendered using the mixture of generated and real characters with three different styles.

¹ <https://www.izihun.com/>

Table 1. Quantitative Comparison.

Model	SimHei $\rightarrow$ font1		SimHei $\rightarrow$ font2		SimHei $\rightarrow$ font3	
	RMSE	APDR	RMSE	APDR	RMSE	APDR
AEGG [9]	22.128	0.178	23.156	0.221	25.112	0.235
Zi2Zi [6]	24.156	0.244	26.800	0.256	27.104	0.231
HGAN [2]	20.205	0.120	22.147	0.186	23.205	0.202
SAFont(our)	20.809	0.116	22.156	0.192	22.476	0.183

Table 2. Evaluation of Self-attention Mechanisms.

Position of SA-Block	SimHei $\rightarrow$ font1		SimHei $\rightarrow$ font2		SimHei $\rightarrow$ font3	
	RMSE	APDR	RMSE	APDR	RMSE	APDR
no attention	22.134	0.182	23.229	0.234	24.518	0.241
f_{64}	22.524	0.193	23.301	0.229	24.417	0.233
f_{128}	21.591	0.164	22.899	0.205	24.020	0.210
f_{256}	20.793	0.138	22.484	0.187	23.179	0.191
f_{512}	20.809	0.116	22.156	0.192	22.476	0.183

3.3 Analysis of Self-Attention Mechanisms

In this section, we conduct experiments to evaluate the effect of self-attention mechanisms. The results are shown in Table 2. The first row is the result without self-attention mechanisms and the last four rows are the results with self-attention blocks in different layers. In the first column of Table 2, f_{num} represents different levels of feature maps. Larger number denotes higher level feature maps.

Comparing the first row and other rows in Table 2, we can see that the values of RMSE and APDR are significantly reduced with the help of self-attention mechanisms. Experimental results show that self-attention mechanisms play a key role in improving the quality of generated characters. Simultaneously, from the last four rows in Table 2, similar to [11], we can conclude that SAGFont with self-attention mechanisms at the middle-to-high level feature maps performs better than that at the low level feature maps.

**Fig. 5.** Generated Results with/without Edge Loss

3.4 Effect of Edge Loss

From the results shown in the Fig.5, we can see that the use of **Edge Loss** helps to improve the quality of the generated images. It can be noted that the generated characters tend to be sharper with much less blur compared with those generated by models without Edge Loss. Experimental results show that Edge Loss is effective to constrain SAGFont to generate high quality Chinese characters.

4 Conclusion and Future Work

In this paper, we propose a novel Chinese font synthesis model called SAFont. We integrate self-attention mechanisms to our model and define an Edge Loss to get shaper edges in Chinese characters. Experimental results show that SAFont performs better than state-of-the-art font transformation algorithms. However, like most of the current methods, it is time-consuming as we need to retrain the model for each font transformation. In the future, we plan to adapt the ideology of transfer learning to our model to improve the generalization capability of our model.

Acknowledgement

The work is supported by Shanghai Natural Science Foundation (No. 19ZR1415900).

References

1. I. Taylor and M. M. Taylor: Writing and Literacy in Chinese, Korean and Japanese. In: Revised Edition, volume 14. John Benjamins Publishing Company (2014)
2. J. Chang, Y. Gu, Y. Zhang, Y. Wang: Chinese Handwriting Imitation with Hierarchical Generative Adversarial Network. In: 32nd British Machine Vision Conference, pp.290 (2018)
3. S. Xu, F. C. M. Lau, W. K. Cheung, Y. Pan: Automatic Generation of Artistic Chinese Calligraphy. In: IEEE Intelligent Systems 20(3), pp.32-39 (2005)
4. S. Xu, T. Jin, H. Jiang and others: Automatic Generation of Personal Chinese Handwriting by Capturing the Characteristics of Personal Handwriting. In: 21st Conference on Innovative Applications of Artificial Intelligence (2009)
5. I. J. Goodfellow, J. Pouget-Abadie, M. Mirza and others: Generative Adversarial Nets. In: 27th Advances in Neural Information Processing Systems, pp.2672-2680 (2014)
6. kaonashi-tyc Homepage, <https://github.com/kaonashi-tyc/zi2zi>. Last accessed 9 Jun 2017
7. Y. Jiang, Z. Lian, Y. Tang, J. Xiao: DCFont: an end-to-end deep chinese font generation system. In: SIGGRAPH Asia Technical Briefs, pp.22:1-22:4 (2017)
8. P. Isola, J. Zhu, T. Zhou, A. A. Efros: Image-to-Image Translation with Conditional Adversarial Networks. In: 30th IEEE Conference on Computer Vision and Pattern Recognition, pp.5967-5976 (2017)
9. P. Lyu, X. Bai, C. Yao and others: Auto-Encoder Guided GAN for Chinese Calligraphy Synthesis. In: 14th International Conference on Document Analysis and Recognition, pp.1095-1100 (2017)
10. A. Vaswani, N. Shazeer, N. Parmar and others: Attention is All you Need. In: 30th Advances in Neural Information Processing Systems, pp.6000-6010 (2017)
11. H. Zhang, I. J. Goodfellow, D. N. Metaxas, A. Odena: Self-Attention Generative Adversarial Networks. arXiv preprint arXiv:1805.08318 (2018)
12. D. Ulyanov, A. Vedaldi, V. S. Lempitsky: Instance Normalization: The Missing Ingredient for Fast Stylization. arXiv preprint arXiv:1607.08022 (2016)
13. K. He, X. Zhang, S. Ren, J. Sun: Deep Residual Learning for Image Recognition. In: 29th IEEE Conference on Computer Vision and Pattern Recognition, pp.770-778 (2016)
14. A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In: 4th International Conference on Learning Representations (2016)
15. Y. Chen, Y. Lai, Y. Liu: CartoonGAN: Generative Adversarial Networks for Photo Cartoonization. In: 31th IEEE Conference on Computer Vision and Pattern Recognition, pp.9465-9474 (2018)
16. C. Ledig, L. Theis, F. Huszar and others: Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In: 30th IEEE Conference on Computer Vision and Pattern Recognition, pp.105-114 (2017)
17. J. H. Lim, J. C. Ye: Geometric GAN. arXiv preprint arXiv:1705.02894 (2017)
18. T. Miyato, T. Kataoka, M. Koyama, Y. Yoshida: Spectral Normalization for Generative Adversarial Networks. In: 6th International Conference on Learning Representations (2018)
19. D. Tran, R. Ranganath, D. M. Blei: Deep and Hierarchical Implicit Models. arXiv preprint arXiv:1702.08896 (2017)
20. J. F. Canny: A Computational Approach to Edge Detection. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 8(6), pp.679-698 (1986)
21. M. Liu, O. Tuzel: Coupled Generative Adversarial Networks. In: 29th Advances in Neural Information Processing Systems, pp.469-477 (2016)