Cross-Lingual Font Style Transfer with Full-Domain Convolutional Attention

Hui-huang Zhao[a,b,*], Tian-le Ji[b], Paul L. Rosin[c], Yu-Kun Lai[c], Wei-liang Meng[d,e], Yao-nan Wang[a]

[a]National Engineering Laboratory for Robot Visual Perception and Control Technology, Hunan University, China
[b]School of Computer Science and Technology, Hengyang Normal University, 421002, China
[c]School of Computer Science & Informatics, Cardiff University, Cardiff, UK
[d]State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, 100098, China
[e]School of Artificial Intelligence, Beijing University of Posts and Telecommunications, Beijing, 100876, China

Abstract

In this paper, we propose a new cross-lingual font style transfer model, FCAGAN, which enables font style transfer between different languages by observing a small number of samples. Most previous work has been on style transfer of different fonts for single language content, but in our task we can learn the font style of one language and migrate it to another. We investigated the drawbacks of related studies and found that existing cross-lingual approaches cannot perfectly learn styles from other languages and maintain the integrity of their own content. Therefore, we designed a new full-domain convolutional attention (FCA) module in combination with other modules to better learn font styles, and a multi-layer perceptual discriminator to ensure character integrity. Experiments show that using this model provides more satisfying results than the current cross-lingual font style transfer methods. Code can be found at https://github.com/jtlxlf/FCAGAN.

Keywords:
Cross-lingual; Full-domain Convolutional Attention; Multi-layer Perceptual Discriminator; Font Style Transfer.

1. Introduction

Fonts are an essential visual design, a carrier of high-level semantic information, and the primary way to convey our messages. In today's globalized world, language barriers no longer constrain information exchange, and fonts from diverse language systems are ubiquitous in our daily lives. Designers of posters, advertisements, and logos often have to consider supporting multiple languages in their work. However, designing a large number of complex glyphs in a consistent style (which is required for good graphic design) for different languages is both time-consuming and expensive. Recently, with the rapid development in the field of deep learning, several font generation methods have emerged to help designers reduce the workload of font design. Some of these methods can accomplish the task of

[*]Corresponding author
Email addresses: happyday.huihuang@gmail.com (Hui-huang Zhao), jtl2653656389@gmail.com (Tian-le Ji), RosinPL@cardiff.ac.uk (Paul L. Rosin), LaiY4@cardiff.ac.uk (Yu-Kun Lai), weiliang.meng@ia.ac.cn (Wei-liang Meng), yaonan@hnu.edu.cn (Yao-nan Wang)

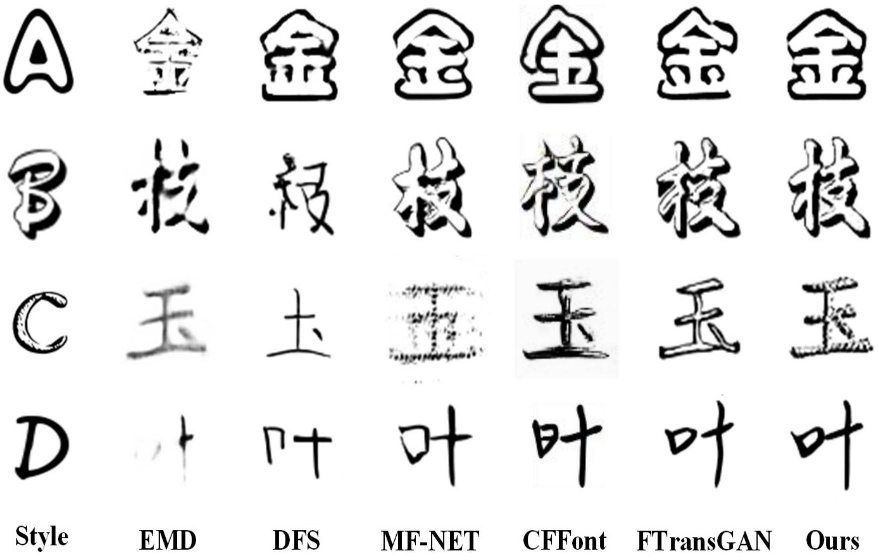


Figure 1: Column 1 is the style font image, columns 2, 3, 4, 5 and 6 are the generated results of font style transfer methods such as EMD [1], DFS [2], MF-NET [3], CFFont [4] and FTransGAN [5]. In that order, and column 6 is the result generated by our model. It can be seen that the current method suffers from missing textual content and insufficient style learning ability, and our model can generate better results.

cross-lingual generation, allowing users to extend their language's font style to other languages' texts and create a new multilingual font library. Although these methods have shown promising results, challenges still exist, such as missing character strokes or difficulties in learning the font styles of different languages.

To address these issues, we propose a new model, FCAGAN, which generates a font library of the target language's font styles by observing multiple samples from the target language. We only need to enter characters from one language as the content map and characters from another language as the style map, and the model will make the source language characters have the style of the other language, thus enabling font style migration. Because it uses an end-to-end model, it has a high level of real-time performance. Similar to a traditional Generative Adversarial Network (GAN), our network contains a generator and two discriminators. The generator consists of two encoders (a content encoder and a style encoder) and a decoder. The content encoder extracts the content features of the source language characters, while the style encoder extracts the style features of the other language characters. The outputs of the two encoders are then fed to the decoder and undergo a simple kind of fusion to generate the target image. In the style encoder, we design a new full-domain convolutional attention-based style extractor, which is described in 3.1.1. It mainly includes three modules: a full-domain convolutional attention (FCA) module, an adaptive fusion module, and a channel-dependent multi-level feature extraction module. In full-domain convolutional attention (FCA), we use full-domain convolution to construct attention maps for feature maps of different layers, which helps this attention module obtain a more detailed representation of the style. Full domain convolution is different from the traditional convolution since it introduces a convolution kernel that can span the entire feature map, thus obtaining the full domain information of the feature map, effectively taking into account both global and local information. The adaptive fusion module is used to assist FCA for better results, and the channel-dependent multi-level feature extraction module is responsible for extracting the final style code from multiple style feature maps. The specific details are shown in figure 2. We believe that using attention networks and combining multi-layer features can better extract style information from character images and accomplish font style migration for arbitrary languages. We then design a multi-layer perceptual discriminator to check similarity from both content and style perspectives. To validate the effectiveness of our proposed approach, our experiments compare FCAGAN with other approaches that allow for cross-language font style migration. In conclusion, our contribution is as follows:

- A new model FCAGAN is proposed, applying an end-to-end solution to cross-lingual font style transfer.
- We synthesize the advantages of self-attentive mechanism and convolution, and propose the full-domain convolutional attention (FCA) module to extract information from feature maps of different sizes. Furthermore, we use a channel-dependent multi-level feature extraction module to extract style encodings from style feature maps at different scales.
- We designed a multi-layer perceptual discriminator that takes into account both global and local aspects and more accurately identifies whether the samples are real or not.
- The experimental results demonstrate that our model performs well in cross-lingual font style transfer, effectively extracting font styles and ensuring the integrity of content.

2. Related Work

2.1. Style Transfer

The style transfer method works by combining the content of one image with the style of another image, aiming to simultaneously preserve the content as much as possible, whilst also swapping style as best as possible, ultimately generating a new image. The style transfer methods are divided into fixed style transfer and arbitrary style transfer. A representative work on fixed style transfer is Gatys et al.'s neural style migration [6], which generates target images by using convolutional neural networks (CNNs) to extract content and style features and optimise content loss and style loss. And in the area of arbitrary style migration, Huang et al. proposed adaptive instance normalization [7], which uses an encoder-decoder network to achieve efficient style transfer of arbitrary images. As mentioned earlier, font style transfer requires considering structural variations among fonts, their distinctive stylistic features, and ensuring the semantic content of characters remains intact. Therefore, font style migration can be borrowed from image style transfer, but it also needs to be improved to some extent for font characteristics.

2.2. Image-to-Image Translation

Image-to-image translation approaches convert the appearance of images from a source domain to a target domain, e.g., horses to zebras. [8] With the continual development of Generative Adversarial Networks, image-to-image translation models like PixPix [9], CycleGAN [10], etc. have gained increasing popularity. However, these methods are only applicable between two domains and are less practical. StarGANv2 [11] enhances the flexibility of the mapping network by incorporating domain information and designing a new mapping network for generative style coding, ultimately enabling multi-source domain to multi-target domain image transformation. However, none of these methods can solve the task of translating images of unknown target style. Thus, FUNIT [12] proposes a few-shot unsupervised image generation method. They combine adversarial training with a novel network design (encoder-decoder), testing with just a few sample images specified to generate images with that style and other content. InjectionGAN [13] unifies generative adversarial networks(GAN) and VAE [14] to explore the latent space and combines the input image with latent variables to address the inefficiency of redundant translation. Similarly, some few-shot image translation methods [15] [16] have been proposed, and they can also generate the original image into an arbitrary style.

2.3. Attention Mechanism

The attention mechanism can be seen as a signal processing mechanism, enabling the network to prioritize crucial aspects of the input data, thereby enhancing overall network performance. It is used for many vision tasks, such as image classification, object detection, semantic segmentation, etc. Attention can also be divided into four basic categories: channel attention, spatial attention, temporal attention, and branching attention [17]. Each of these methods can focus on localised areas to propagate features, and they can be combined to obtain a more diverse set of support relationships from a number of different levels [18]. Later, Vaswani et al. proposed the self-attention mechanism [19] and used it in the Transformer module, which greatly improved the accuracy and parallelism of the translation model. Self-Attention is a unique attention mechanism that can correlate information from different locations in a sequence, effectively capture distant dependencies, is highly adaptable, and is gradually playing an increasingly important role in computer vision. Previously, SA-GAN [20] successfully applied a self-attentive layer to the image generation task. Guo et al. [21] proposed a new linear attention mechanism, KLA, and first optimally decomposed large kernel convolutions in spatial and channel directions.

Recently, networks often use a combination of convolutional and attention layers to improve performance when processing font tasks. In this work, we employ a combination of full-domain convolution and attention mechanisms to enhance feature extraction. The full-domain convolution can generate an attention map by acquiring the full-domain information of the feature map, which can effectively process local contextual information while considering global and local features, improve generalization ability, and better acquire font style information.

2.4. Font Generation

Font generation is an image generation technique that generates other characters in the same style as the sample, based on a model and a sample of characters. In the field of computer vision, it can also be seen as a particular case of image style transfer. Image style transfer refers to transferring the styles (e.g. color, texture, brushstrokes, etc.) of one image onto another image, while font generation specifically converts the style of characters to a designated font style.

The traditional approach to font generation is based primarily on contour shape modeling. Zhou et al. [22] proposed a Chinese character radical composition model to generate handwritten fonts. This method ensures that the structure of the characters is correct, but it is difficult to capture the detail and personality of handwritten fonts. With the popularity of generative adversarial networks(GAN), font generation methods such as zi2zi[23], PEGAN [24], and CalliGAN [25] have emerged. These methods typically require a pair of matched images as training data and then use the network to learn the mapping between different image domains. However, in real life, the acquired sample images are generally uneven, and none of these methods can solve the unpaired font migration task. Zhang et al. used the idea of extracting and combining character style and content features to design a font migration network EMD (encoder mixer decoder) [1] by exploiting the conditional dependency between the style and content of text images. The method extracts font style and content features from the style image set and content image set, respectively, and then synthesizes the input to the decoder through a mixer (Mixer) without matching the source-target character image pairs, which is more flexible and versatile. Jiang et al. have designed an end-to-end font generation network, DCFont [26], which means that no human intervention is required during offline training and online generation. MC-GAN [27] provides an end-to-end solution for style migration of few-sample fonts by designing a style encoder to check for commonalities and differences between multiple fonts. It only needs a few samples to generate an entire library of fonts.

Recently, several methods have been proposed for migrating few-shot font styles. Zhu et al. [28] proposed a deep feature similarity framework for style migration of a few samples, which generates target characters by exploiting the deep feature similarity between the input content character images and style character images. Subsequently, several methods were proposed for various font generation tasks. Deepimitator [29] and SDT [30] can imitate and generate personal handwriting from a small number of handwriting images using sequence processing models (e.g., RNN). FontTransformer [31] and DeepVecFont-v2 [32] can generate high-quality fonts from a small number of samples by using a multilayered attention-based Transformer. VQFont [33] utilizes cross-attention for feature fusion and achieves better results in few-shot font generation within a single language. Junbum, Cha, et al. propose a few-shot font generation method, DM-Font [34], which achieves good results in implementing the transfer of Chinese font styles to foreign language fonts such as Korean and Thai. However, they require external auxiliary information such as character component information to be added at training time, and for multilingual datasets, character component information is exceptionally tedious to obtain. Li et al. proposed a cross-lingual font style transfer network (FTransGAN) [5], which is theoretically applicable to various languages and does not require the assistance of other components, but also suffers from incomplete character generation and insufficient style learning ability. Our method introduces full-domain convolutional attention to enhance the ability of style extraction, and improves the discriminator architecture to strengthen adversarial training for ensuring character integrity.

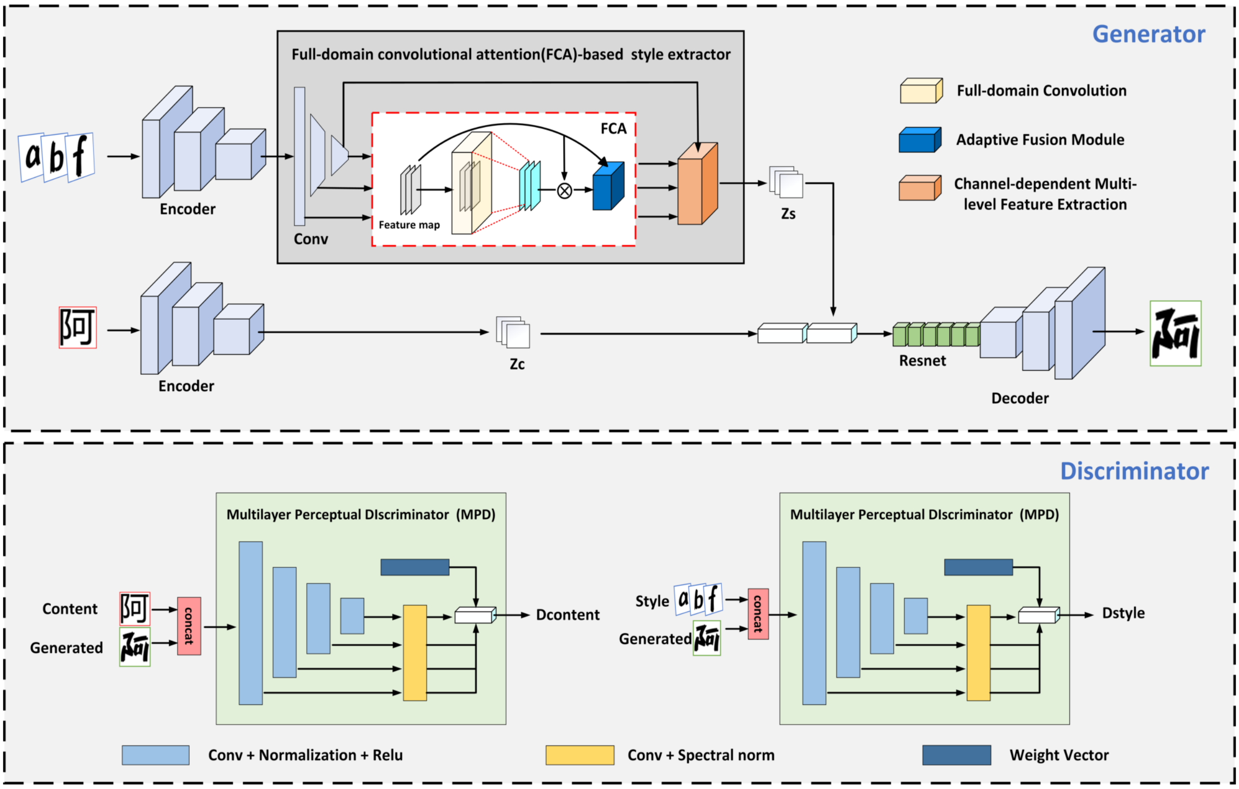

Generator
Full-domain convolutional attention(FCA)-based style extractor
FCA
Feature map
Conv
Encoder
Zs
Zc
Encoder
Resnet
Decoder
Full-domain Convolution
Adaptive Fusion Module
Channel-dependent Multi-level Feature Extraction
Discriminator
Multilayer Perceptual DIscriminator (MPD)
Content
Generated
concat
Dcontent
Style
Generated
concat
Dstyle
Multilayer Perceptual DIscriminator (MPD)
Conv + Normalization + Relu
Conv + Spectral norm
Weight Vector

Figure 2: Network Architecture. The generator consists of three parts: style encoder, content encoder and decoder. The content encoder consists of three convolutional layers (Conv layers). The style encoder is based on the structure of the content encoder with the addition of a full-domain convolutional attention-based style extractor, as described in Section 3.1. Zs and Zc denote the extracted style coding and content coding, respectively. The decoder consists of six ResNet blocks and transposed convolution layers (Deconv layers). Multi-Layer Perceptual Discriminator is used to distinguish between true and false for content and style. where the weight vector is the optimal weights obtained after many experiments.

3. Method Description

Our method aims to achieve cross-lingual font style transfer. In the network, we extract content features using a single source language character image, while deep style features are accurately captured by utilizing multiple character images of the same font in the target language. Therefore, we input a content image of a source language but multiple target style images of another language. For example, if the content image is a Chinese character, then the style image could be six English font character images.

We will then use these images to generate a new image that will contain the content of the selected Chinese character image and the image of the selected English font style. Furthermore, we set all the input and output images to be grayscale images of size 64×64. To train the model, we randomly selected images from large-scale Chinese and English datasets as content images and style images, respectively, and input them into the network.

As shown in Figure 2, we use an end-to-end training approach where the network has one generator and two discriminators (content and style discriminator). The two discriminators have different inputs but the same structure and are used to discriminate the truth of a patch locally. See 5 for details of implementation. The generator consists of a style encoder, a content encoder, and a decoder. The two encoders extract style features and content features, respectively, and then the outputs of the two encoders are concatenated in the channel dimension as the input to the decoder. The content encoder consists of three convolutional layers, each followed by a BatchNorm layer and a ReLU layer. The style encoder is based on the structure of the content encoder with the addition of a full-domain convolutional attention-based style extractor, which consists of three modules: a full-domain convolutional attention (FCA) module, an adaptive fusion module, and a channel-dependent multi-level feature extraction module. The details are described in 3.1. The decoder consists of six ResNet blocks and transposed convolution layers.

3.1. Full-domain Convolutional Attention-based Style Extractor

As shown in Figure 3, its input comes from the feature map s_1 obtained after three convolutional layers, and the feature maps s_2 and s_3 obtained by subjecting s_1 to two downsampling processes. During the downsampling process, since the receptive field in the feature map output by the shallower layer is small, it only contains local features. As the convolutional layer deepens, the receptive field will gradually expand until it even contains almost global features. In this way, the network can acquire both local and global features at all levels.

3.1.1. Full-domain Convolutional Attention Module

We use the feature map $s_i(i = 1, 2, 3)$ as the input to the attention module. The key to the attention mechanism is the construction of the attention graph, and in this module, we use a full-domain convolutional layer to construct the attention graph. This approach combines the excellent ability of the traditional attention mechanism to obtain long-range dependencies and the ability of convolution to extract contextual information. The full domain convolution kernel can cover the entire graph information so that each region of the output feature graph contains global information. Of course, for feature maps of different sizes, different convolution is used to obtain the full domain information of the feature map. Since the feature maps s_1, s_2 and s_3 are of different sizes after downsampling, we notice that using a single convolution to obtain full-domain information for larger feature maps requires massive computational consumption. To solve this problem, we use multi-layer convolution to achieve the same effect.

As shown in Figure 4, the large convolution kernel used for larger feature maps can be divided into a small convolution kernel and a dilated convolution kernel. To increase its channel adaptation, we connect a convolution kernel size of 1×1 to the later convolution [35]. In order to keep the input and output sizes consistent, we need to compute the full-domain convolution parameters for the feature maps of different layers. The calculation is detailed in Algorithm 1.

After the above steps to obtain the attentional map M_i, we multiply s_i with Mi to obtain the feature map $Att(s_i)$.

$$Att(s_i) = M_i \times s_i \tag{1}$$

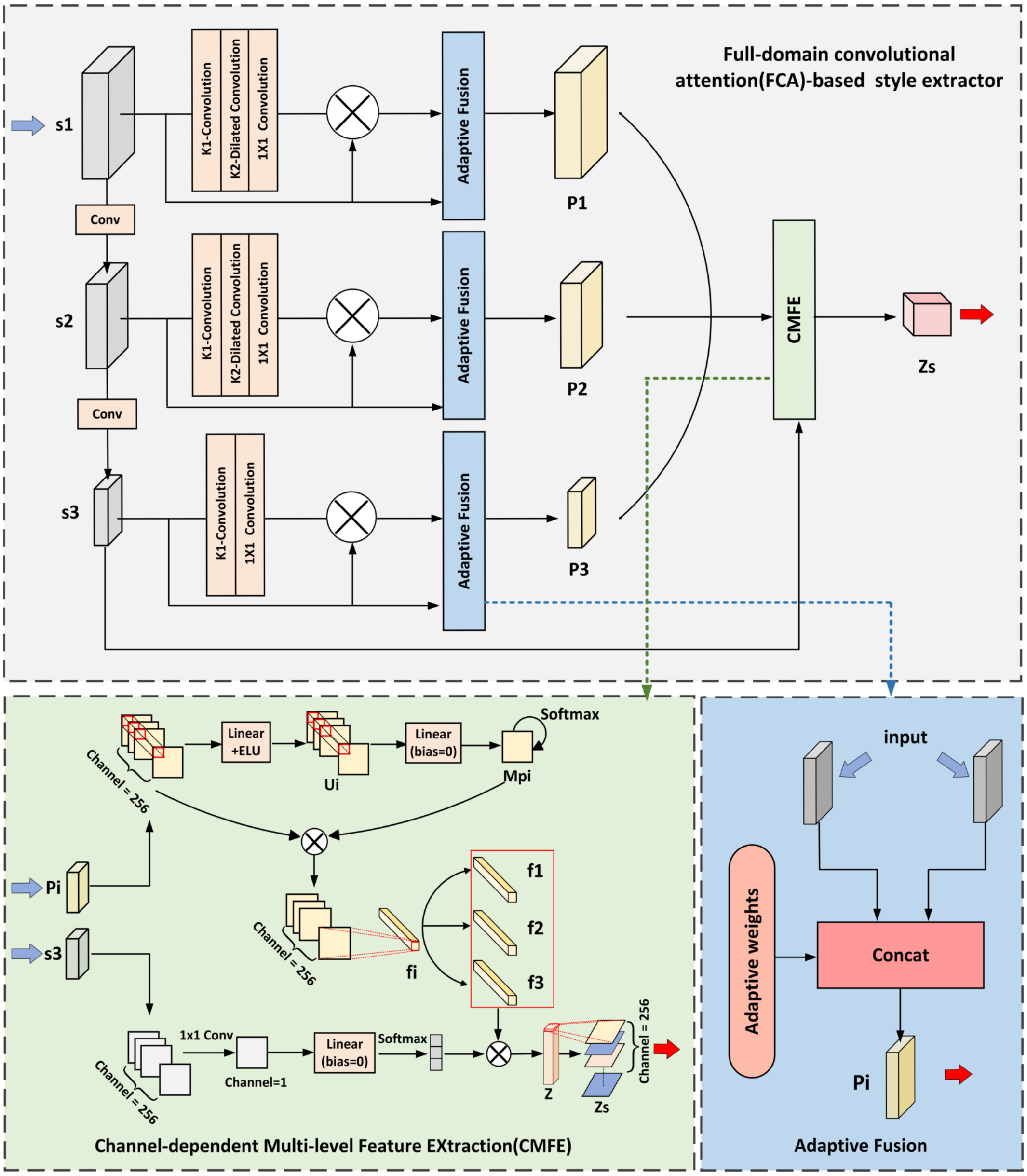


Figure 3: A specific design diagram of a style extractor based on full-domain convolutional attention. It consists of several modules, where blue arrows indicate the input and red arrows indicate the output. Among them, s1 is the input style feature map. It undergoes convolutional downsampling to obtain s2 and s3, which have different receptive fields. p1, p2, and p3 are the style encodings obtained by applying global convolutional attention and adaptive fusion modules (marked in blue) to each of them. Then, they are combined with s3 and input into the CMFE module (marked in green) for further refinement to obtain the final style encoding Zs.

3.1.2. Adaptive Fusion Module

After that, we use an adaptive fusion module to enhance the network's performance. Using the attention Weighted fusion mitigates the loss of font structure and information caused by attention mechanisms, while adaptive methods dynamically adjust the weights to assist the network in focusing on more crucial features.

Specifically, we provide two adaptive parameters w_1 and w_2 as the weights of the input images s_i and $Att(s_i)$,

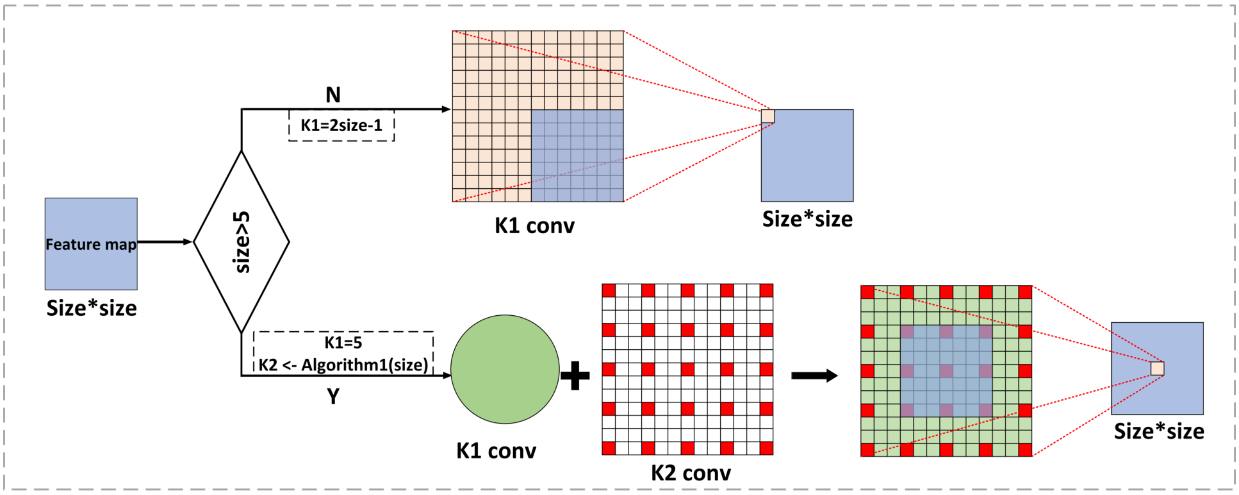


Figure 4: The full-domain convolution process in a two-dimensional view. Our approach can algorithmically design the convolution in such a way that each field of the output feature map contains the full domain information of the feature map and keeps the feature map scale constant. For smaller feature maps, we use a large convolution kernel (only K1-convolution) to achieve this. For larger feature maps, we use a weighted sum of the convolution of each local region instead of a single large convolution kernel. This helps reduce computational resource consumption. The K1-convolution represents the convolution in each local region, while the K2-convolution refers to the process of obtaining a weighted sum of the K1-convolution results from each region. The details regarding the parameter can be found in Algorithm 1.

respectively. In this case, we can consider them as learning parameters to train the model by back-propagation and adjust the size according to the gradient of the loss function. Finally, the output image P_i is obtained by weighted summation of s_i and $Att(s_i)$, where w_1 and w_2 are initialized to 0.5, and a constraint is added to require that their sum is always 1.

$$\begin{cases} w_1 + w_2 = 1 \\ P_i = w_1 \times s_i + w_2 \times Att(s_i) \end{cases} \tag{2}$$

The details of this module and the calculation process are detailed in Figure 3 and Algorithm 1.

3.1.3. Channel-dependent Multi-level Feature Extraction Module

As shown in Figure 3, the output feature maps P_1, P_2 and P_3 are of different sizes and contain style features in different perceptual domains due to downsampling. In order to better learn the style features and match the output size of the content encoder, we use this module to extract the style features we need from $P_i(i = 1, 2, 3)$. Here, we will use $V_r(r = 1, 2, 3, ..., W \times H)$ to denote each region of the feature map P_i.

We follow a self-attentive mechanism approach to make the network focus more on regions that contain important stylistic information. The network needs to construct an attention graph Mp_i to determine the value of the contribution of each region V_r of P_i to the overall style. We consider the correlation between channels, so the attention graph Mp_i is required to be the same for each channel of P_i. Therefore, the network first needs to effectively fuse the values of each region V_r on different channels before constructing the attention graph. Specifically, we first transform Pi into a feature map U_i with 100 channels using the linear layer and the ELU function as the neural network layer, and then use the adaptive vector v_i of length 100 as the weights to weight the fusion of regions V_r on different channels, and

Algorithm 1 Full-domain Convolutional Attention

```
Input: Feature maps $s_i$
Data: Parameters of undecomposed Full-domain Convolution: $K_s, Pad_s$; Parameters of Full-domain Convolution: $K_1, Pad_1, K_2, Pad_2, dila_2$
$size = Shape(s_i)[-1]$ //Find the size of the input feature map
$Pad_s = (K_s - 1)/2 \leq K_s - size \rightarrow K_s \geq 2size - 1$
//Ensure that the convolution kernel covers the full domain of the feature map
if $size \leq 5$ then
    $K_1 = K_s = 2size - 1$
    $Pad_1 = Pad_s = size - 1$
    $AttConv = Conv(K_1, Pad_1)$
else if $size > 5$ then
    $K_1 = 5, Pad_1 = 2, dila_2 = 3$
    $K_2 = 2n + 1, n \in N^*$ //Convolution kernel size is odd.
    $K_s = 3(K_2 - 1) + 1$ //When dilation rate is 3, the relationship between $K_s$ and $K_2$
    $K_s = 6n + 1 \geq 2size - 1 \rightarrow n \geq (size - 1)/3, n \in N^*$
    $K_2 = 2n_{min} + 1, Pad_2 = Pad_s = 3n_{min}$
    $AttConv = Conv(K_1, Pad_1)+Conv(K_2, Pad_2, dila_2)$
end if
$M_i = AttConv(s_i)$
$M_i = Pointwise\_Conv(M_i)$ //Calculate the attention map $M_i$
$Att(s_i) = M_i \times s_i$
$Pi = w_1 \times s_i + w_2 \times Att(s_i), w_1 + w_2 = 1$ //Adaptive Fusion
Output: Feature maps $P_i$
```

finally obtain the attention map Mpi by softmax. Where v_i is randomly initialized and jointly trained for the whole model.

$$Ui = ELU(W_c P_i + b_c) \tag{3}$$

$$Mp_i = softmax(U_i^T v_i) \tag{4}$$

Then we use the attention map Mp_i to extract the important style information from the feature map. Since there are multiple channels, we should end up with a one-dimensional feature vector $f_i (i = 1, 2, 3)$.

$$f_i = \sum_{r=1}^{H \times W} Mp_i(r) \times P_i(r) \tag{5}$$

where $Mp_i(r)$ denotes the value of each pixel point of Mp_i and $P_i(r)$ denotes the value of each pixel point of P_i.

The vectors f_1, f_2, and f_3 contain different degrees of style information, and we use the style image itself to determine the importance of the respective information and finally extract more accurate style features. Therefore, we use a pointwise convolution to reduce the dimensionality of s_3 and then assign weights g_1, g_2, g_3 to the feature vectors f_1, f_2, f_3 using a fully connected layer and a softmax function.

$$g_1, g_2, g_3 = softmax(tanh(w_k s_3 + b_k)) \tag{6}$$

$$Z = \sum_{i=1}^{3} g_i \times f_i \tag{7}$$

where Z is the weighted sum of the three feature vectors.

The size of the content feature map output by the content encoder is $C \times H \times W$, and Z is a C-dimensional vector, so we need to expand Z to Zs to match the size of the content feature map. The detailed calculation process is described in Algorithm 2.

Algorithm 2 Channel-dependent Multi-level Feature Extraction

```
Input: $P_i (i = 1, 2, 3)$, $s_3$
$U_i = ELU(Linear(P_i)), channel : 256 => 100$
$L_i = Linear(P_i), channel : 100 => 1$
$Mp_i = Softmax(L_i)$
$F_i = Hadamard\ Product(U_i, Mp_i)$
for $field\ in\ F_i$ do
    $f_i = f_i + field$
end for
$Vector = Tanh(Linear(s_3))$
$Vs = Softmax(Vector)$
$Z = f_i Vs$
$Zs = Expand(Z)$
Output: $Z_s$
```

3.2. Discriminator

However, cross-lingual font generation is a complex task, as ensuring that the generated font images are satisfactory in both content and style is challenging due to the inconsistent languages between content images and style images.

Therefore, we propose a multilayer perceptual discriminator that allows the network to pay better attention to global and local information, and the details are shown in Figure 5. We believe the discriminator can be seen as a feature extractor [36]. Inspired by UNet [37], we think that the performance of the network can be effectively improved by processing the discriminatory results of the outputs of the different layers. In a common patchGAN, the image is mapped to an $N \times N$ matrix after various convolutional layers, and the evaluation value of each point in the $N \times N$ matrix is used to evaluate the veracity of a small region (i.e., the perceptual field) in the original image. However, in character images, the style of the text should be determined by a combination of global and local information. So we extract feature maps with different receptive fields from four different layers and then output four various discriminations through the discriminator. We set a weight vector to guide the discriminator's global and local attention degree. In this way, the discriminator has a more accurate discrimination ability, which helps the network generate better images.

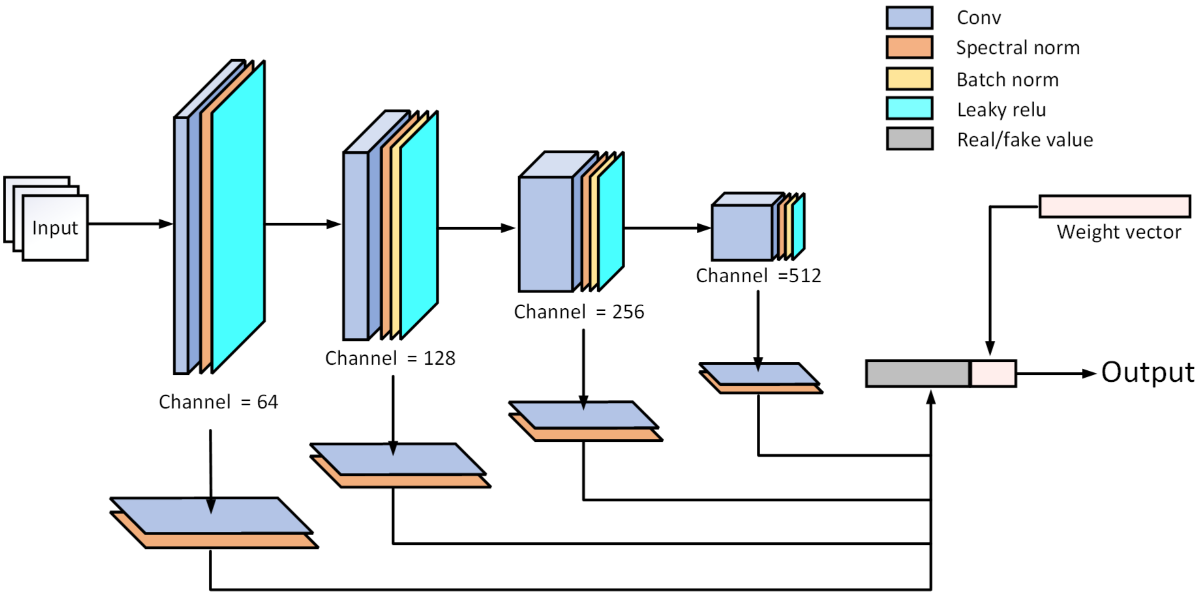


Figure 5: Detailed Structure of Multi-Layer Perceptual Discriminator.

3.3. Loss Functions

As mentioned earlier, our model consists of two discriminators, Dcontent and Dstyle. Dcontent receives the generated image and content image, checking whether they are the same characters, and Dstyle receives the generated image and style image, checking whether they are the same style. We concatenate the respective input images in the channel dimension and input them into the discriminator. The real images are also inputted to the discriminator in the same way as the generated images, and the adversarial loss is constructed based on the results obtained from both. The whole model is trained jointly in an end-to-end manner. We define three loss functions in the network: L_1 loss, style loss L_{style}, and content loss $L_{content}$.

We aim to stabilize GAN training and make the results convergent and closer to real images. Among them, $L_{content}D$ and $L_{style}D$ use the quadratic smoothing hinge loss function. $L_{content}G$ and $L_{style}G$ continue to use the traditional hinge loss. Hinge loss measures the distance between the model's predicted output and the actual output and is used in networks to minimize this distance. It encourages the model to find a decision boundary that maximizes the difference between the two classes. The quadratic smoothing hinge loss helps the model to converge better and produce higher quality results while ensuring its excellent generalization ability. We construct the loss function based on the output of the discriminator, as shown in Eqs. 8 and 9. where L_{style} and $L_{content}$ are obtained by summing the respective corresponding losses.

$$\begin{cases} L_{content} \quad = L_{content}G + L_{content}D, \\ L_{content}G = -W_c E_{\hat{x},c\sim P(\hat{x},c)}[D_c(\hat{x}, c)], \\ L_{content}D = \frac{1}{2} W_c E_{\hat{x},c\sim P(\hat{x},c)}[max(0, 1 - D_c(\hat{x}, c))^2] + \\ \qquad\qquad \frac{1}{2} W_c E_{I_g,c\sim P(I_g,c)}[max(0, 1 - D_c(I_g, c))^2] \end{cases} \tag{8}$$

$$\begin{cases} L_{style} \quad = L_{style}G + L_{style}D, \\ L_{style}G = -W_s E_{\hat{x},s\sim P(\hat{x},s)}[D_s(\hat{x}, s)], \\ L_{style}D = \frac{1}{2} W_s E_{\hat{x},s\sim P(\hat{x},s)}[max(0, 1 - D_s(\hat{x}, s))^2] + \\ \qquad\qquad \frac{1}{2} W_s E_{I_g,s\sim P(I_g,s)}[max(0, 1 - D_s(I_g, s))^2] \end{cases} \tag{9}$$

where I_g is the ground truth image, $\hat{x}$ is the generated image, c and s denote the content and style images, respectively, and E denotes the mean value. W_c/W_s is the weight value of the discriminator output, and D_c/D_s is the real/fake value of the discriminator output.

To stabilize our training, we also use L_1 loss in our objective function to calculate the pixel-by-pixel error between the generated image and the ground truth image as follows:

$$L_1 = E_{\hat{x},I_g\sim P(\hat{x},I_g)}[\left\|I_g - \hat{x}\right\|_1] \tag{10}$$

where I_g is the ground truth image, $\hat{x}$ is the generated image, and E denotes the mean value. Above all these loss terms, the loss function of our cross-lingual font style transfer model is as follows:

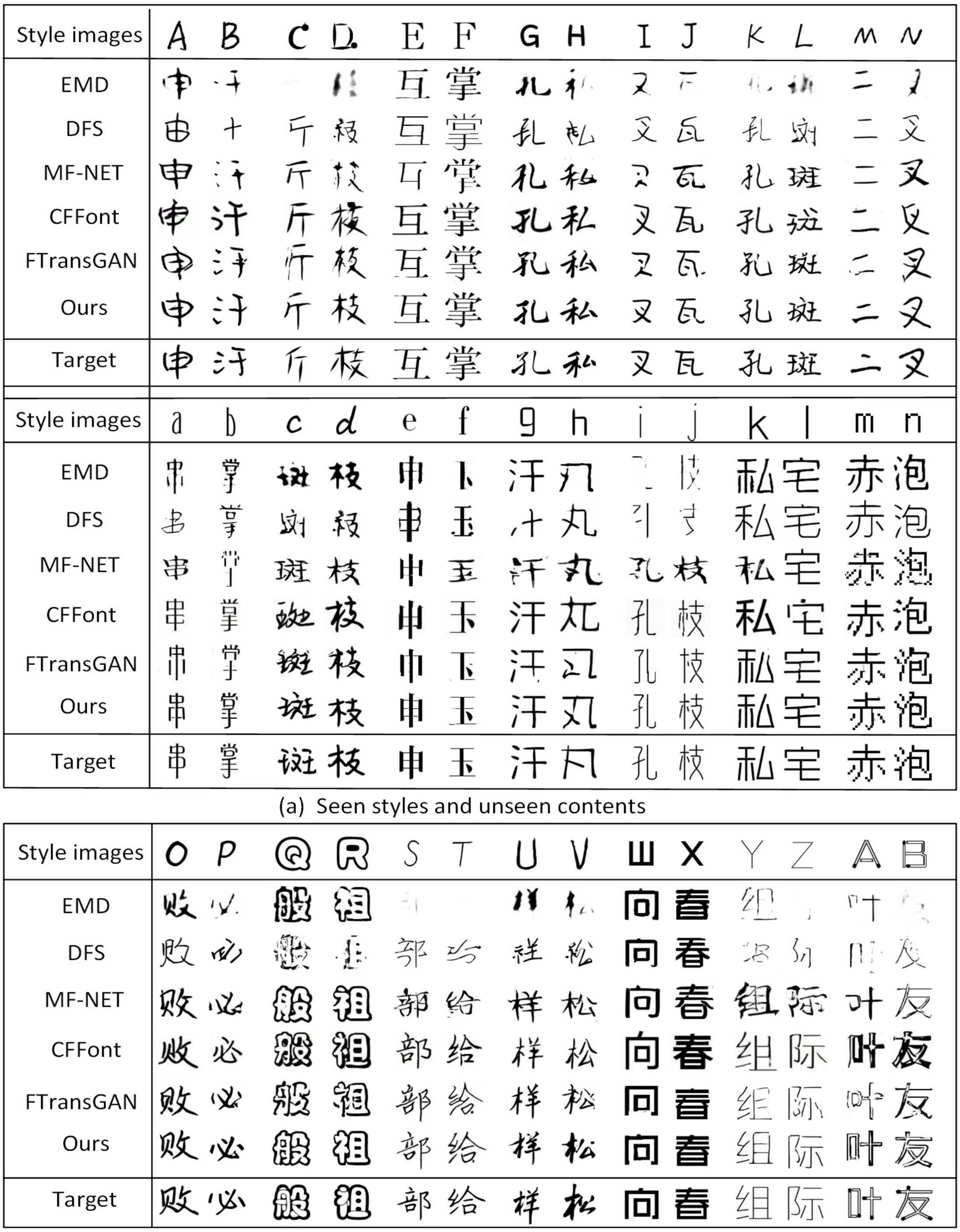
$$L = \lambda_1 L_1 + \lambda_s L_{style} + \lambda_c L_{content} \tag{11}$$

where λ_1, λ_s, and λ_c are the three weights used to balance these terms.

4. Experiments

4.1. Font Dataset and Compared Methods

To conduct our experiments, we use a multilingual dataset for training and testing. In our experiments, English style images are used as style inputs to the network, and Chinese content images are used as content inputs to the network. Chinese characters are so numerous and have such complex geometrical structures, so simple glyph generation methods often cannot guarantee the structural correctness of the resulting Chinese glyphs [31]. On the other hand, English has fewer and simpler structures. Therefore, this design scheme can better demonstrate the ability of our model to migrate font styles between different languages.



(a) Seen styles and unseen contents

(b) Unseen styles and seen contents

Figure 6: Visual comparison of our FCAGAN (row 7) with EMD [1](row 2), DFS [2] (row 3), MF-net [3] (row 4), CFFont [4] (row 5) and FTransGAN [5] (row 6) on different test datasets. Row 1 shows the stylized font image and row 8 shows the target image.

During testing, the model was able to transfer the style of English letters to Chinese characters by inputting only a small number of English style images and the desired Chinese content images. To test our method, we selected 30

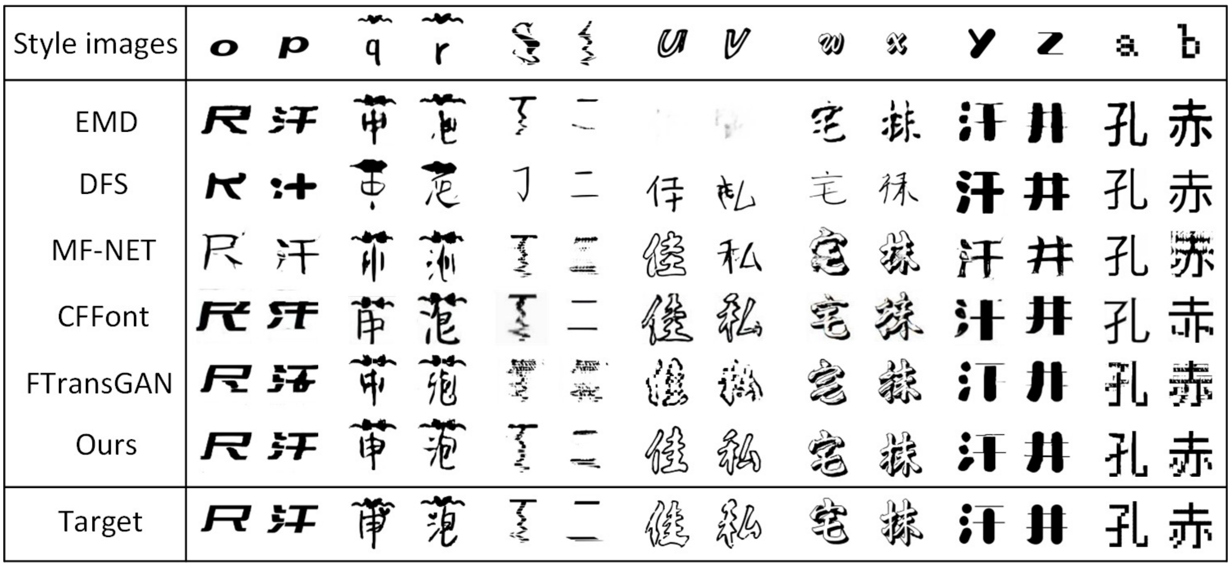


Figure 7: Visual comparison of complex style fonts in terms of content completeness. Comparative Methods: EMD [1](row 2), DFS [2] (row 3), MF-net [3] (row 4), CFFont [4] (row 5), FTransGAN [5] (row 6).

unseen Chinese characters and 30 varieties of unseen English fonts as the test set. We used a standard font (such as Microsoft YaHei) as the input of Chinese content, so as to ensure that the synthesized character type conforms to our expectations. Subsequently, we selected 800 fonts to build the training set for comparison experiments and ablation studies. Each font contains approximately 1000 Chinese characters and 52 Latin characters of the same style, with the size of each character image being 64 × 64. In addition, we also separately selected 400 fonts and 1200 fonts to build the training set to evaluate the impact of different dataset sizes on the generation results.

At present, among the known models for font style conversion, most of them cannot perform style conversion between any two languages well. We have selected several networks that can be used on arbitrary language datasets for comparison, including EMD [1], DFS [2], FTransGAN [5], MF-NET [3] and CFFont [4]. English fonts were chosen uniformly as the style map and Chinese images as the content map. English has a simple structure compared to other languages, so extracting styles from English fonts for conversion to more complex Chinese fonts is a challenging task, and we are also able to visually better compare the ability of each model to migrate font styles between different languages.

4.2. Implementation Details

In the following experiments, we fed six images of the target font into the style encoder and set λ_1=100 and λ_c=λ_s=1. We trained 20 epochs with an initial learning rate of lr=0.0002, which gradually reduced in each epoch starting from the 11th epoch. Our learning rate dynamic adjustment scheme is shown in Figure 8. We can observe that the L1 loss undergoes significant changes from epoch 0 to 10, while it remains relatively stable from epoch 11 to 20. Starting from the 10th epoch (indicated by the vertical line), the learning rate lr decreases by 0.0002 per epoch. This helps the network to search for local optimal solutions.

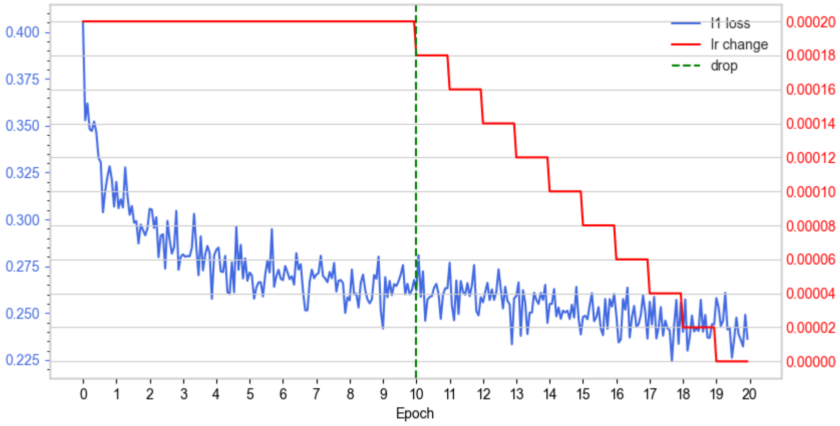


Figure 8: The blue line in the figure represents the change in L1 loss, the red line represents the change in learning rate (lr), and the green line represents the point in time at which the learning rate begins to change.

4.3. Evaluation Metrics

Evaluating font style conversion can be a complex task, as the effectiveness of converting different font styles may vary subjectively. Therefore, our evaluation criteria need to consider multiple aspects. To address this, we have established three distinct types of metrics for model evaluation: pixel-level evaluation, perceptual-level evaluation, and artistic-level evaluation.

1) Pixel level evaluation

The pixel-level evaluation compares pixels at the exact location between the model-generated image and the sample real image. We use multi-scale structural similarity (MS-SSIM) to evaluate the performance of the model. Structural similarity (SSIM) is a metric used to measure the similarity of two images. It takes into account not only the pixel-level similarity of luminance and color but also the similarity of image structural information, so it better reflects the human eye's perception of image similarity. Multi-scale structural similarity (MS-SSIM) is calculated by dividing the image into multiple scales, calculating an SSIM value for each scale, and then weighting and averaging these SSIM values to obtain the final similarity index. It can detect the structural information of images at different scales, capture the image features, and evaluate the structural similarity of images more comprehensively

2) Perceptual level evaluation

FID (Fréchet Inception Distance) is a metric proposed by Salimans et al. [38] for evaluating the differences between images generated by models and real images. It is mainly based on two aspects: first, the extraction of image features using the Inception network, and second, the use of the Fréchet distance to measure the differences in the distribution of these features. Liu et al. [12] proposed a modified FID metric called the conditional version of mFID (conditional mFID). It calculates the FID metric for each target class and then takes the average value. This modified conditional version of the mFID metric can be better suited for scenarios such as few-sample learning because it considers the similarity differences between different target classes and better reflects the performance of the generative

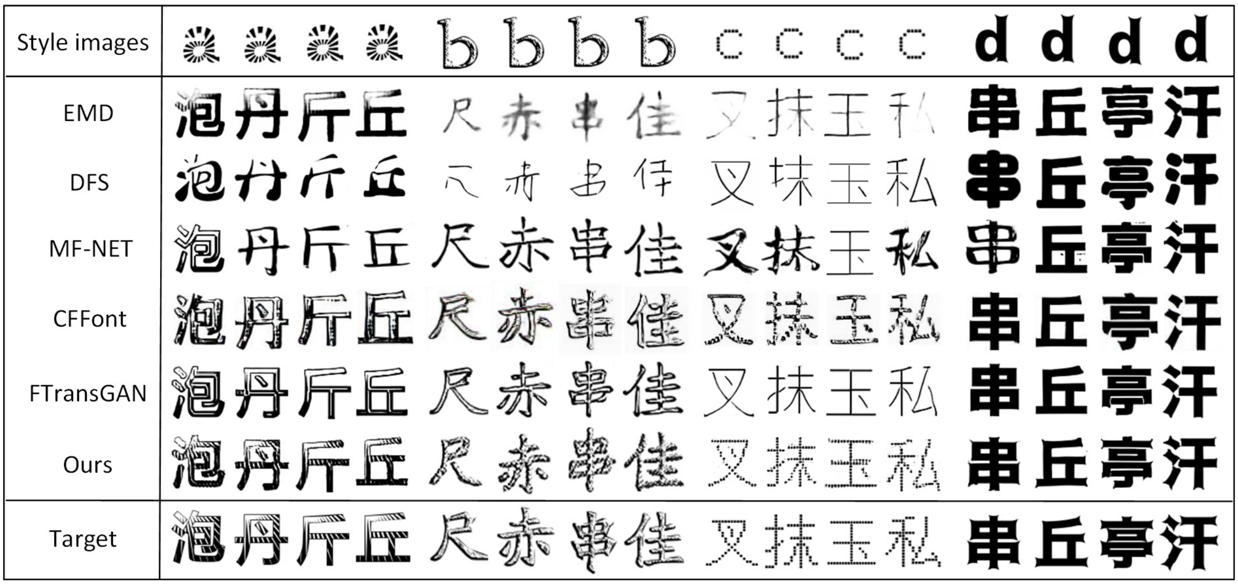


Figure 9: Visual comparison of the style transfer capabilities of complex stylised fonts. Comparative Methods: EMD [1](row 2), DFS [2] (row 3), MF-net [3] (row 4), CFFont [4] (row 5), FTransGAN [5] (row 6).

model on different target classes.

In this paper, we evaluate the generated images from both style and content perspectives. We trained two multi-layered ResNet networks to classify and extract features for both character content and font style. During evaluation, we used the pre-trained ResNet network models to compare the generated images with the real images in terms of content and style. We calculated the percentage of generated images that correctly matched the content category and style category, and selected the highest values as the metrics for content perception accuracy and style perception accuracy, respectively. Additionally, we used the content and style features extracted by this network as activations to calculate content FID and style FID, and took the average as the metrics for content perception mFID and style perception mFID.

3) Art level evaluation

To explore the ability of our method on font style transfer, we used ArtFID [39], a method for quantitative evaluation of neural style transfer proposed by Wright and Ommer, to evaluate the ability of the model. Previously, the quantitative assessment of style migration still lagged compared to other visual domains. To address this issue, Matthias adopted a different approach from other evaluation schemes, taking into account both the degree of content preservation and style matching of the generated stylized imag/es and combining the two to form a more convincing metric for assessing style migration-ArtFID. In our experiments, we evaluated ArtFID for each font in the test set, and the final metric presented is the average of the results obtained by evaluating all fonts.

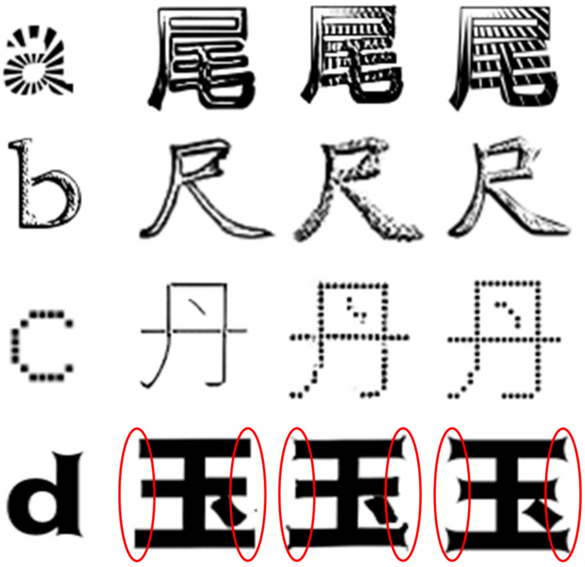


Figure 10: A detailed demonstration of the comparative learning ability of complex stylised fonts. Column 1 shows the stylised font image, column 2 shows the results of the baseline method, column 3 shows the generated results of our model, and column 4 shows the target image.

4.4. Quantitative and Qualitative Results

To better describe the quality of images generated by our method, we divide font styles into regular and complex styles. In Figure 6(a) and Figure 7, we respectively demonstrate the visual effects of simple and complex styles on an unknown content dataset. In Figure 6(a), EMD [1], DFS [2] and MF-net [3] all have obvious problems of missing strokes, while CFFont [4] and FtransGAN [5] also have missing issues for some finer strokes. In contrast, our method has better content completeness. In Figure 7, complex font styles have a greater impact on the generation effects of other models. Relative to Chinese characters, the cross-language font generation capability of the CFFont [4] method is slightly insufficient during testing, and the content fusion module sometimes cannot find a suitable solution when facing complex font styles, so it does not have advantages in generating effects for characters from the test set with unknown content. FTransGAN [5] accurately learns the overall style, but its detail style extraction and content completeness are not as good as ours. As shown in Table 1, our method generally outperforms other methods on this dataset in terms of indicators.

For datasets with unknown style and known content, Figure 6(b) shows the visual results of all methods. Among them, EMD [1] and DFS [2] fail to recognize character content, and MF-net [3] does not satisfy the generation of character images in terms of shape and stroke details. CFFont [4] uses a content fusion module to extract content features, which can enhance the generation effect of content. However, the multiple base font images used by the network for content reconstruction can also mislead the network, resulting in generated font images that are similar to the target font but not identical. This also explains why this method has a better content-aware mFID but slightly inadequate style performance in the unknown style test set in Table 1. FTransGAN [5] still has the problem of missing content details in its generated results. In Table 1, this model achieves a higher style-aware accuracy, but this only

Methods	Pixel-level	Content-aware		Style-aware		↓ArtFID
	↑MS-SSIM	↑Accuracy(%)	↓mFID	↑Accuracy(%)	↓mFID	
Seen styles and **Unseen** contents						
EMD [1]	0.467	81.2	116.9	24.4	597.1	14.19
DFS [2]	0.231	89.2	150.0	2.7	820.9	12.49
MFNet [3]	0.477	96.0	76.2	45.4	428.9	13.18
CFFont [4]	0.499	96.7	51.3	57.8	311.1	12.99
FtransGAN [5].	0.492	97.0	49.6	58.3	308.1	12.10
Ours	**0.502**	**97.7**	**43.5**	**63.0**	**286.6**	**11.98**
Unseen styles and **seen** contents						
EMD [1]	0.388	85.5	184.4	4.4	623.2	13.87
DFS [2]	0.201	91.7	230.7	0.7	662.4	12.38
MFNet [3]	0.354	97.7	135.7	8.9	552.7	12.65
CFFont [4]	0.389	99.9	**80.4**	9.5	441.4	12.45
FtransGAN [5].	0.383	99.8	97.6	**11.7**	419.4	12.33
Ours	**0.393**	**99.9**	96.6	10.8	**416.2**	**11.73**

Table 1: Quantitative Evaluation on the adopted dataset. ↑ means larger numbers are better, ↓ means smaller numbers are better.

represents a decent effect on a certain style. On the contrary, certain fonts may yield better outcomes, but the style-aware mFID does not stand out, indicating a relatively weaker generalization capability.

To demonstrate the remarkable ability of our method to capture font style details, in Figure 9,we selected some distinctive fonts as references. These fonts have the same basic outlines as regular fonts but possess unique designs that are more challenging to handle in specific aspects. In Figure 10(a), there are white fine lines interspersed within the characters, and compared to the baseline, our method clearly exhibits this alternating black-and-white style. In Figure 10(b), we can observe that due to the scarcity of black pixels within the strokes, these details are easily overlooked in style capture. However, our method is able to capture them, resulting in results closer to the reference. In Figure 10(c), the characters are composed of individual dots connected together, and although the spacing between each dot is minimal, our method still pays attention to this detail. In Figure 10(d), apart from the overall bold style, there is also a slight local indentation at the end of the strokes. Capturing such subtle styles in English characters with fewer strokes and simpler structures is a challenging task, and our method demonstrates superior performance in this aspect. In summary, our method is highly sensitive to style details, demonstrating good performance when dealing with complex styles that involve multiple details.

4.5. Ablation Studies

We will conduct experiments to verify the effectiveness of each key component in the model. We use the same settings for training and evaluation for each part. In Table 2, All denotes our proposed model.

1) Version A represents the model after replacing the full-domain convolutional attention (FCA) module in the proposed network with the traditional self-attention. Compared to the traditional SA-GAN [20], FCA captures long-

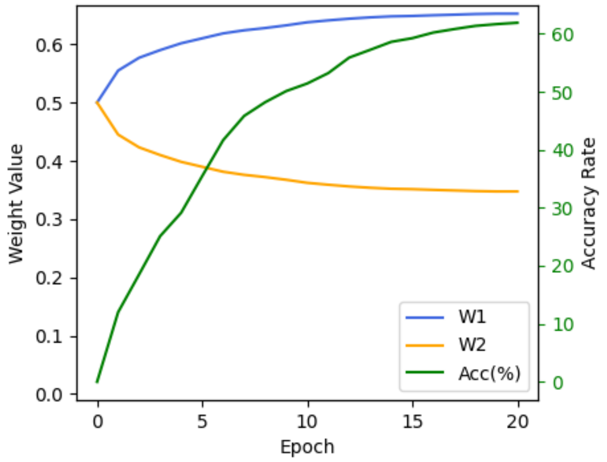


Figure 11: Variation curves of the adaptive weights w1,w2 and the corresponding style-aware accuracy(Acc).

term dependencies better and acquires more subtle features. Therefore, when the model does not use FCA, the learning ability of the network decreases, especially for styles already present in the training set.

2) Version B represents the model after removing the adaptive fusion module from the proposed network. As shown in Figure 11, the module adaptively adjusts the fusion parameters to follow the network to obtain the best values. From the metrics in Table 2, it can be seen that the removal of this module not only negatively affects the overall performance, but also reduces the performance of the model in the unknown style domain. It can be seen that this module can effectively solve some overfitting phenomena and improve the generalization ability of the model.

3) Version C represents our proposed network without using the multi-layer perceptual discriminator and the loss function designed for it. As can be seen in Table 2, this scheme has a great impact on the overall learning ability of the network, especially on the completeness of the content. In the face of different styles and complex contents, this superior discriminator is better able to distinguish between generated and real samples. It uses perceptual features of varying scales in different domains to provide finer guidance for the generator in terms of content and style learning, resulting in overall improved performance.

4) Version D represents our proposed network without the improved channel-dependent multi-level feature extraction module. As can be seen in Table 2, this scheme has a beneficial effect on the overall performance of the network, although it makes a slight impact on the unknown content accuracy and unknown style-aware FID. The module can extract more critical information from multiple features at different levels and integrate them, while allowing the importance of high and low frequency information to be determined by the image itself, further enhancing the feature extraction capability of the network.

In summary, we consider each module essential in the model. These modules play their respective roles to help the model to have stronger learning ability and generalisation, so that the results can achieve better performance in pixel-level, perceptual-level, and art-level metrics.

Methods	Pixel-level	Content-aware		Style-aware		↓ArtFID
	↑MS-SSIM	↑Accuracy(%)	↓mFID	↑Accuracy(%)	↓mFID	
Seen styles and Unseen contents						
All	0.502	97.7	**43.5**	**63.0**	**286.6**	**11.98**
Version A	0.498	97.0	50.8	58.9	306.5	12.73
Version B	0.501	97.3	48.4	62.2	300.9	12.51
Version C	**0.503**	97.0	46.2	60.0	305.1	12.57
Version D	0.501	**98.0**	43.9	62.5	290.9	12.08
Unseen styles and seen contents						
All	**0.393**	**99.9**	**96.6**	10.8	416.2	**11.73**
Version A	0.385	99.8	98.3	**11.7**	418.6	12.02
Version B	0.390	99.9	97.4	10.0	428.4	12.38
Version C	0.393	99.8	96.9	10.9	**414.9**	11.86
Version D	0.387	99.9	96.8	10.5	415.6	11.98

Table 2: Ablation study on the adopted dataset.↑ means larger numbers are better, ↓ means smaller numbers are better. For ease of observation, we mark the optimal values in bold and the worst values in underline.

The data proves that we produce the best results when we blend the various components, achieving excellent results on pixel-level, perceptual-level, and art-level metrics.

4.6. Effect of Dataset Size

To investigate the impact of sample size on model performance, we conducted additional experiments using training sets with 400 and 1200 fonts respectively. We compared our method with the approach (FTransGAN [5]) that yielded similar generated results in Table 1. Table 3 reflects the results obtained by these methods with training sets containing 400, 800, and 1200 fonts. It can be observed that as the sample size increases, there is a significant improvement in various aspects of performance. However, our method still produces relatively excellent results, demonstrating that our approach outperforms the previous method in overall performance, rather than being limited to gains for individual fonts.

In addition, when there is a wide variety of font styles in the samples, there may be a potential issue in correctly distinguishing between different styles in the input. To evaluate our model, we conducted two separate tests on 60 font styles, with different content in the style images for the two tests. As shown in Figure 12, the results for two different English characters are most similar only when they are in the same font style. Therefore, our model is not influenced by the content of style images and will not be misled into thinking that similar font styles are the same.

4.7. Chinese Font Style to English Character

Figure 13 demonstrates the model's ability to migrate Chinese font styles to english characters. Compared with other models, FCAGAN can better ensure the integrity of English character content while learning Chinese font styles.

		Seen styles and **Unseen** contents			
Dataset size	Methods	↑MS-SSIM	↓Content-aware mFID	↓Style-aware mFID	↓ArtFID
400	FTransGAN	0.529	44.2	239.2	**12.43**
	Ours	**0.536**	**42.6**	**229.2**	12.72
800	FTransGAN	0.492	49.6	308.1	12.10
	Ours	**0.502**	**43.5**	**286.6**	**11.98**
1200	FTransGAN	0.518	34.7	261.1	12.11
	Ours	**0.528**	**32.5**	**243.5**	**12.01**
		Unseen styles and **seen** contents			
Dataset size	Methods	↑MS-SSIM	↓Content-aware mFID	↓Style-aware mFID	↓ArtFID
400	FTransGAN	0.335	83.8	423.1	12.66
	Ours	**0.351**	**83.4**	**415.9**	**12.48**
800	FTransGAN	0.383	97.6	419.4	12.33
	Ours	**0.393**	**96.6**	**416.2**	**11.73**
1200	FTransGAN	0.390	80.7	335.2	12.09
	Ours	**0.406**	**79.6**	**336.2**	**11.76**

Table 3: Experimental results on the effect of dataset size. ↑ means larger numbers are better, ↓ means smaller numbers are better.

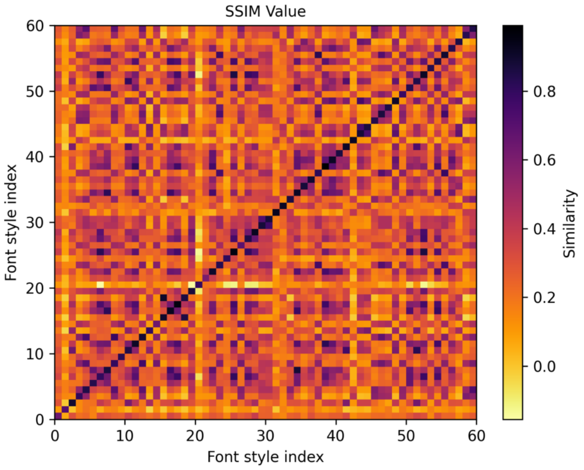


Figure 12: The heat map of the SSIM matrix. The horizontal and vertical axes of the chart represent the indices of 60 different fonts for two sets of distinct English letters. The color of the coordinates represents the SSIM value between the generated results obtained from different English character images as style inputs. The darker the color, the more similar the results are.

4.8. Statistical Significance Test

In order to demonstrate the superiority of the method, we perform a test for statistical significance of the difference between the proposed network model and baseline (FTransGAN[5]). Our sample source contains two sets of image sets, which come from the generation results of our model FCAGAN and the generation results of baseline method. Both sets of images contain the same 50 random fonts, and each font contains the same Chinese content. We calculate its style FID value for each font, so our sample contains two sets of data, each with 50 style FID values. The specific

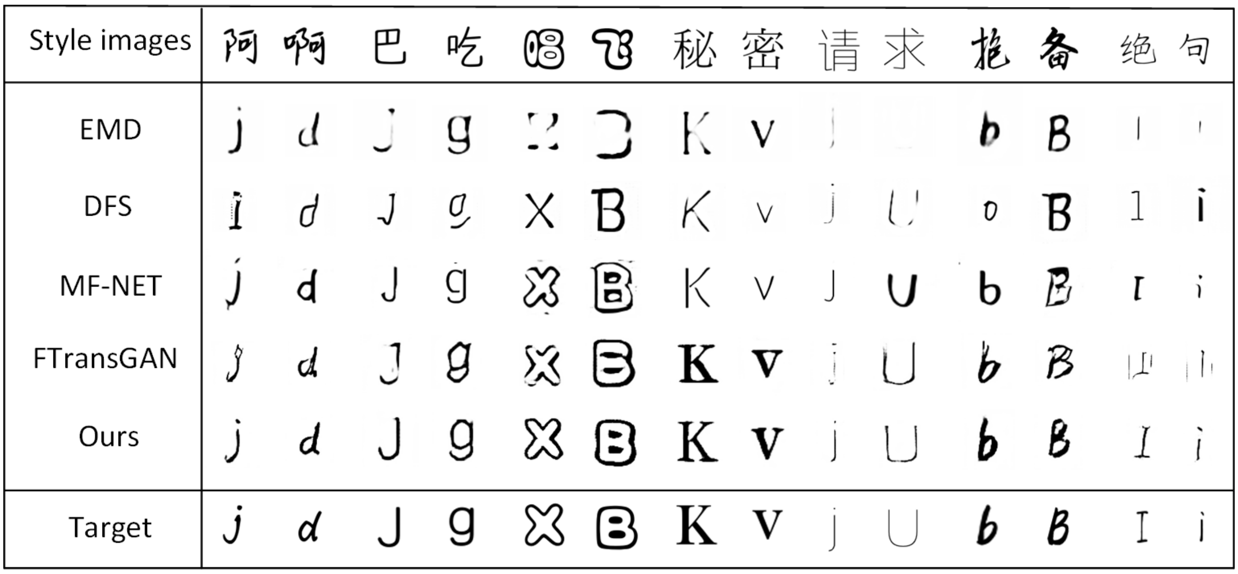


Figure 13: Visual comparison of Chinese font styles migrating to English characters. Comparative Methods: EMD [1](row 2), DFS [2] (row 3), MF-net [3] (row 4), FTransGAN [5] (row 5).

	Average value ↓	H-statistic	P-value
FCAGAN	**280.54**	4.71	0.029
Baseline	330.26		

Table 4: Experimental results of the Statistical Significance Test. This includes the mean of the sample data corresponding to the two models, and the H-statistic and P-value obtained from the Kruskal-Wallis test

data are shown in Figure 14.

We formulate the null hypothesis H_0: the performance of our proposed method is similar to the baseline, i.e., the two data sets in the sample are in the same overall distribution. In addition, we set the significance level $\alpha = 0.05$. We used a rank-based non-parametric test and Table 4 shows the results of the Kruskal-Wallis test on both sets of data.

The p-value is less than the α-value, and so the original hypothesis H_0 is rejected, indicating that the FID metrics obtained by our method, and thus its performance, are significanctly better than those of the baseline method.

4.9. Limitations

In order to investigate the model's ability to generalise to other languages, we use it to test the transfer of known English styles to (a) Japanese, (b) Korean (b) and (c) Thai, respectively. The experimental results are shown in Figure 15. Since the structure of Japanese is simpler than that of Chinese, the results generated in experiment (a) are generally better. Korean is similar to Chinese in that the characters are made up of components. In experiment (b), the model basically meets the requirements in terms of content completeness, with only minor stroke misalignments in individual words. In experiment (c), the generation result of style letter *r* is content-incorrect. Secondly, since there are many similar words in Thai, the requirements for content completeness are higher. In the marked red box, some

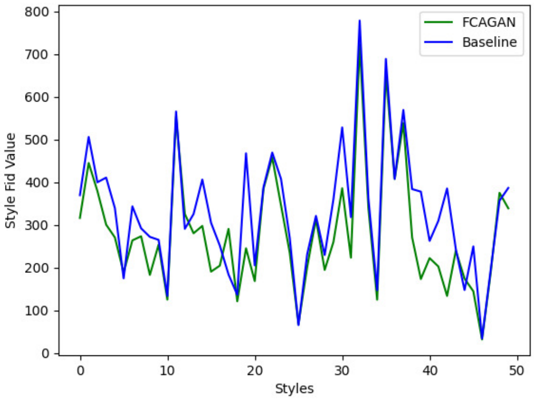


Figure 14: Visual presentation of two sets of experimental data from the Statistical Significance Test. Where the smaller the FID value, the better the effect.

strokes are different from the content image, which may cause people to regard it as another similar character. Our model should further improve the precise expression of content and avoid content recognition errors.

In FCAGAN, we lack the processing of content features, which may lead to some content errors in the generated results when facing other languages. Therefore, in future work, we can consider adding some auxiliary factors (such as stroke information in various languages) to help generate better results.

5. Conclusion

We propose an end-to-end approach to transfer font styles between languages using only a few samples. The key idea is to construct a style encoder based on the full-domain convolutional attention module and utilize a multi-layer perceptual discriminator to guide it in accomplishing more precise font style extraction. Additionally, the adaptive fusion module is employed to enhance the model's generalization capability, and the channel-dependent multi-level feature extraction module assists in feature refinement. To obtain higher quality results and ensure stable network training, we introduce an improved quadratic smoothing hinge loss. Extensive experiments demonstrate that our proposed FCAGAN exhibits superior performance for cross-lingual font style transfer tasks, surpassing existing methods. It has the ability to capture subtle styles that are difficult to capture, thereby generating results that are more satisfactory. We hope the proposed method can provide valuable aid for issues related to style extraction in style transfer tasks and challenges regarding character integrity in font generation tasks. In the future, we will strive to further strengthen the model's capability of extracting styles and contents, aiming for more satisfactory results when faced with different languages and unusual fonts.

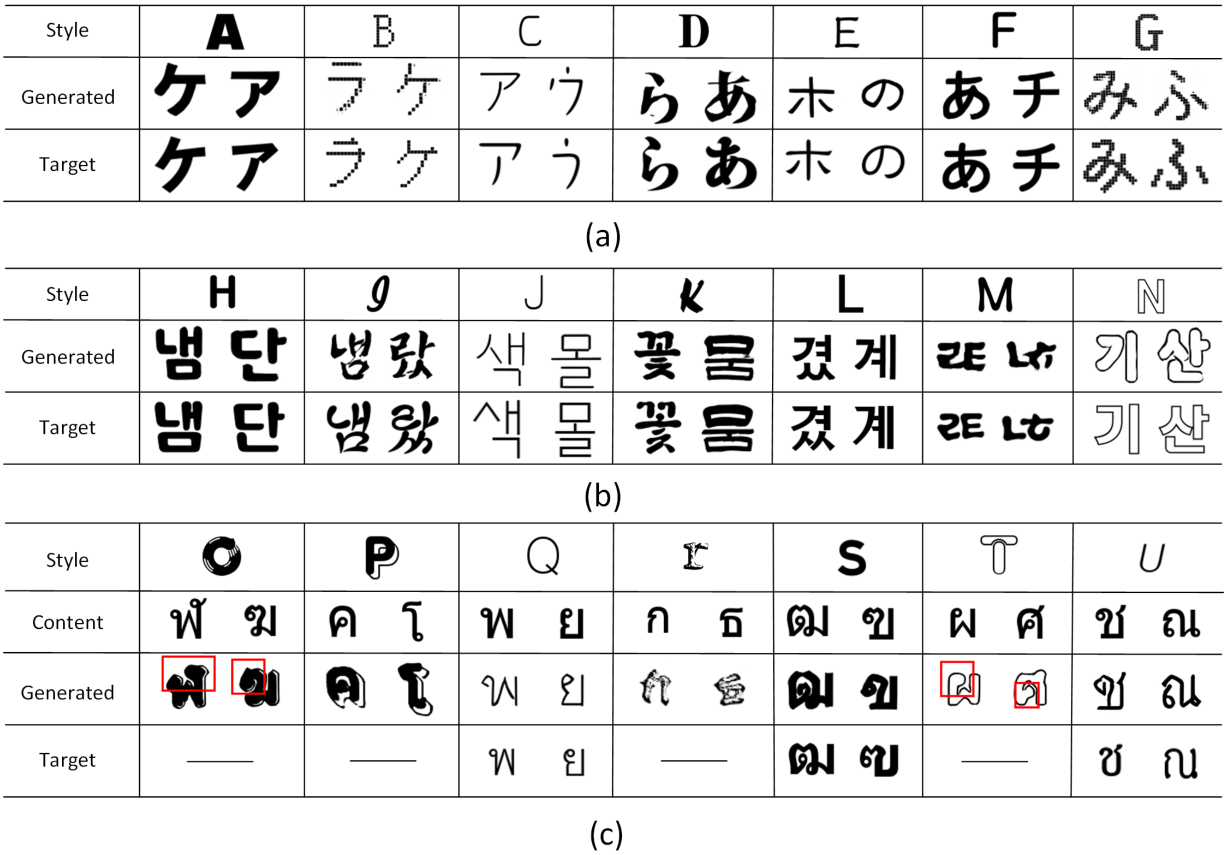

Style
Generated
Target
A
B
C
D
E
F
G
(a)
Style
Generated
Target
H
J
L
M
N
(b)
Style
Content
Generated
Target
Q
S
U
(c)

Figure 15: Cross-lingual inference on (a) Japanese glyphs, (b) Korean glyphs, and (c) Thai glyphs, where the horizontal line indicates that Thai has no relevant groundtruth for the style. Since the generated Thai glyphs are not easily observable, content references are added to the second line in c.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61772179), Hunan Provincial Natural Science Foundation of China (2023JJ50095, 2020JJ4152, 2022JJ50016) and the Science and Technology Plan Project of Hunan Province (2016TP1020).

References

[1] Y. Zhang, Y. Zhang, W. Cai, Separating style and content for generalized style transfer, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 8447–8455.

[2] A. Zhu, X. Lu, X. Bai, S. Uchida, B. K. Iwana, S. Xiong, Few-shot text style transfer via deep feature similarity, IEEE Transactions on Image Processing 29 (2020) 6932–6946.

[3] Y. Zhang, J. Man, P. Sun, Mf-net: A novel few-shot stylized multilingual font generation method, in: Proceedings of the 30th ACM International Conference on Multimedia, 2022, pp. 2088–2096.

[4] C. Wang, M. Zhou, T. Ge, Y. Jiang, H. Bao, W. Xu, CF-Font: Content fusion for few-shot font generation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 1858–1867.

[5] C. Li, Y. Taniguchi, M. Lu, S. Konomi, Few-shot font style transfer between different languages, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 433–442.

[6] L. A. Gatys, A. S. Ecker, M. Bethge, Image style transfer using convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2414–2423.

[7] X. Huang, S. Belongie, Arbitrary style transfer in real-time with adaptive instance normalization, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 1501–1510.

[8] Z. Cao, W. Wang, L. Huo, S. Niu, Unsupervised class-to-class translation for domain variations, Pattern Recognition 138 (2023) 109346.

[9] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.

[10] J.-Y. Zhu, T. Park, P. Isola, A. A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2223–2232.

[11] Y. Choi, Y. Uh, J. Yoo, J.-W. Ha, StarGAN v2: Diverse image synthesis for multiple domains, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 8188–8197.

[12] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, J. Kautz, Few-shot unsupervised image-to-image translation, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 10551–10560.

[13] W. Xu, K. Shawn, G. Wang, Toward learning a unified many-to-many mapping for diverse image translation, Pattern Recognition 93 (2019) 570–580.

[14] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).

[15] K. Baek, Y. Choi, Y. Uh, J. Yoo, H. Shim, Rethinking the truly unsupervised image-to-image translation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 14154–14163.

[16] X. Chen, C. Xu, X. Yang, L. Song, D. Tao, Gated-GAN: Adversarial gated networks for multi-collection style transfer, IEEE Transactions on Image Processing 28 (2) (2018) 546–560.

[17] M.-H. Guo, T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, S.-M. Hu, Attention mechanisms in computer vision: A survey, Computational Visual Media 8 (3) (2022) 331–368.

[18] M. Feng, H. Hou, L. Zhang, Y. Guo, H. Yu, Y. Wang, A. Mian, Exploring hierarchical spatial layout cues for 3d point cloud based scene graph prediction, IEEE Transactions on Multimedia (2023).

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[20] H. Zhang, I. Goodfellow, D. Metaxas, A. Odena, Self-attention generative adversarial networks, in: International conference on machine learning, PMLR, 2019, pp. 7354–7363.

[21] M.-H. Guo, C.-Z. Lu, Z.-N. Liu, M.-M. Cheng, S.-M. Hu, Visual attention network, arXiv preprint arXiv:2202.09741 (2022).

[22] B. Zhou, W. Wang, Z. Chen, Easy generation of personal Chinese handwritten fonts, in: 2011 IEEE international conference on multimedia and expo, IEEE, 2011, pp. 1–6.

[23] Y. Tian, Master Chinese calligraphy with conditional adversarial networks (2017).

[24] D. Sun, Q. Zhang, J. Yang, Pyramid embedded generative adversarial network for automated font generation, in: 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, 2018, pp. 976–981.

[25] S.-J. Wu, C.-Y. Yang, J. Y.-j. Hsu, CalliGAN: Style and structure-aware Chinese calligraphy character generator, arXiv preprint arXiv:2005.12500 (2020).

[26] Y. Jiang, Z. Lian, Y. Tang, J. Xiao, DCFont: an end-to-end deep Chinese font generation system, in: SIGGRAPH Asia 2017 Technical Briefs, 2017, pp. 1–4.

[27] S. Azadi, M. Fisher, V. G. Kim, Z. Wang, E. Shechtman, T. Darrell, Multi-content GAN for few-shot font style transfer, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7564–7573.

[28] X. Zhu, M. Lin, K. Wen, H. Zhao, X. Sun, Deep deformable artistic font style transfer, Electronics 12 (7) (2023) 1561.

[29] B. Zhao, J. Tao, M. Yang, Z. Tian, C. Fan, Y. Bai, Deep imitator: Handwriting calligraphy imitation via deep attention networks, Pattern Recognition 104 (2020) 107080.

[30] G. Dai, Y. Zhang, Q. Wang, Q. Du, Z. Yu, Z. Liu, S. Huang, Disentangling writer and character styles for handwriting generation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 5977–5986.

[31] Y. Liu, Z. Lian, FontTransformer: Few-shot high-resolution Chinese glyph image synthesis via stacked transformers, Pattern Recognition 141 (2023) 109593.

[32] Y. Wang, Y. Wang, L. Yu, Y. Zhu, Z. Lian, DeepVecFont-v2: Exploiting transformers to synthesize vector fonts with higher quality, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 18320–18328.

[33] W. Pan, A. Zhu, X. Zhou, B. K. Iwana, S. Li, Few shot font generation via transferring similarity guided global style and quantization local style, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 19506–19516.

[34] J. Cha, S. Chun, G. Lee, B. Lee, S. Kim, H. Lee, Few-shot compositional font generation with dual memory, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX 16, Springer, 2020, pp. 735–751.

[35] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).

[36] K. Kim, S. Park, E. Jeon, T. Kim, D. Kim, A style-aware discriminator for controllable image translation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18239–18248.

[37] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: Medical Image Computing and Computer-Assisted Intervention, Proceedings, Part III 18, Springer, 2015, pp. 234–241.

[38] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training GANs, Advances in neural information processing systems 29 (2016).

[39] M. Wright, B. Ommer, ArtFid: Quantitative evaluation of neural style transfer, in: DAGM German Conference on Pattern Recognition, Springer, 2022, pp. 560–576.

Hui-huang Zhao received his PhD degree in 2010 from XiDian University. He was a Sponsored Researcher in the School of Computer Science and Informatics, Cardiff University. Now he is a Visiting Professor in National Engineering Laboratory for Robot Visual Perception and Control Technology, Hunan University. His main research interests include machine learning and image processing.

Tian-le Ji received a B.S. degree from Huaihua University, Huaihua, China, in 2022 and he is currently working toward a Master's degree at Hengyang Normal University, Hengyang, China. Since 2022, his current research interests include computer vision and image style transfer.

Paul L. Rosin is currently a professor with the School of Computer Science and Informatics, Cardiff University, U.K. Previous posts include lecturer with the Department of Information Systems and Computing, Brunel University London, U.K., research scientist with the Institute for Remote Sensing Applications, Joint Research Centre, Ispra, Italy, and lecturer with the Curtin University of Technology, Perth, Australia. His research interests include low level image processing, performance evaluation, shape analysis, facial analysis, cellular automata, non-photorealistic rendering, and cultural heritage. For more information, please visit `https://users.cs.cf.ac.uk/Paul.Rosin/`.

Yu-kun Lai received his bachelor's degree and PhD degree in computer science from Tsinghua University in 2003 and 2008, respectively. He is currently a Professor in the School of Computer Science & Informatics, Cardiff University. His research interests include computer graphics, geometry processing, image processing and computer vision. He is on the editorial boards of *IEEE Transactions on Visualization and Computer Graphics* and *The Visual Computer*. For more information, please visit `https://users.cs.cf.ac.uk/Yukun.Lai/`.

Wei-liang Meng received his PhD degree in Computer Application from State Key Laboratory of Computer Science at Institute of Software, Chinese Academy of Sciences in 2010. He is currently an Associate Professor in State Key Laboratory of Multimodal Artificial Intelligence Systems at Institute of Automation, Chinese Academy of Sciences. His main research field includes artificial intelligence, computer vision, 3D scene analysis, 3D geometry processing and computer graphics.

Yao-nan Wang received the Ph.D. degree in electrical engineering from Hunan University, Changsha, China, in 1994. He was a PostDoctoral Research Fellow with the Normal University of Defence Technology, Changsha, from 1994 to 1995. From 1998 to 2000, he was a Senior Humboldt Fellow in Germany. From 2001 to 2004, he was a Visiting Professor with the University of Bremen, Bremen, Germany. Since 1995, he has been a Professor with the College of Electrical and Information Engineering, Hunan University. His current research interests include robotics and image processing.