One-Shot Multilingual Font Generation Via ViT

Jiarui Liu
University of Massachusetts, Amherst
300 Massachusetts Ave, Amherst, MA 01003
jiaruil@umass.edu

Zhiheng Wang
University of Massachusetts, Amherst
300 Massachusetts Ave, Amherst, MA 01003
zhihengwang@umass.edu

Abstract

Font design poses unique challenges for logographic languages like Chinese, Japanese, and Korean (CJK), where thousands of unique characters must be individually crafted. This paper introduces a novel Vision Transformer (ViT)-based model for multi-language font generation, effectively addressing the complexities of both logographic and alphabetic scripts. By leveraging ViT and pretraining with a strong visual pretext task (Masked Autoencoding, MAE) [10], our model eliminates the need for complex design components in prior frameworks while achieving comprehensive results with enhanced generalizability. Remarkably, it can generate high-quality fonts across multiple languages for unseen, unknown, and even user-crafted characters. Additionally, we integrate a Retrieval-Augmented Guidance (RAG) module to dynamically retrieve and adapt style references, improving scalability and real-world applicability. We evaluated our approach in various font generation tasks, demonstrating its effectiveness, adaptability, and scalability.

1. Introduction

Designing high-quality fonts is challenging, especially for logographic languages (e.g., Chinese, Japanese, Korean), where thousands of unique characters must be manually created. Existing font generation methods often focus on a single script or rely on large labeled datasets, limiting their ability to handle multiple languages and unseen, custom characters [3, 9, 12, 22].

To address these challenges, we propose a one-shot multilingual font generation model based on Vision Transformers (ViTs) [6, 10], capable of handling diverse scripts—including Chinese, Japanese, Chinese (CJK) and English—as well as user-invented glyphs. By leveraging ViT-based encoders and decoders, pretrained with a Masked Autoencoder (MAE) objective, our approach eliminates the need for intricate architectural tweaks and robustly captures both global structure and subtle stylistic elements. It supports handwriting and standard fonts without any reference character constraints, outperforming prior methods limited to narrow domains or requiring extensive character libraries [11, 16, 23, 26].

Additionally, we integrate a Retrieval-Augmented Guidance (RAG) module [5] to dynamically retrieve the most suitable style references from a known inventory, enabling the model to adapt to challenging or unusual characters. Unlike previous approaches that rely on base font references and fail when such references are unavailable, our model eliminates this dependency. It accepts any input shape, including hand-drawn designs, and demonstrates the capability to produce faithful, style-consistent outputs across multiple languages, including unseen fonts, unknown characters, and even user-crafted designs. Experiments show that our ViT-MAE-based system consistently generates high-quality, style-accurate fonts under one-shot conditions, establishing a robust foundation for versatile, cross-lingual font generation.

2. Related Work

Font design presents unique challenges, particularly for logographic languages like Chinese and Japanese, which require designing thousands of unique characters individually. While traditional font development remains labor-intensive, deep learning has revolutionized automatic font generation. Early approaches leveraged CNNs and GANs for image-to-image translation, with pioneering works by Azadi *et al*. [3] and Fogel *et al*. [7] demonstrating success in alphabetic languages. For logographic languages, models like "Rewrite" [18] and "zi2zi" [19] advanced the field through sophisticated style mapping between character pairs.

Recent research has shifted toward few-shot learning using GANs and diffusion models, enabling high-quality font generation from minimal reference characters. Notable advances include Zhang *et al*.'s [26] style-content separation networks and GAN-based frameworks by Li *et al*. [12], Wen *et al*. [21], and Yu *et al*. [25]. Hayashi *et al*. [8] proposed GlyphGAN for ensuring style consistency through independent control of character classes and style vectors.

For unsupervised scenarios, Xie *et al*.'s [22] DG-Font maintained structural integrity without extensive paired datasets, while Liu *et al*. [13] introduced FontTransformer for high-resolution synthesis in few-shot settings.

Parallel developments in handwritten text generation have emerged, with Pippi *et al*. [14] introducing Transformer-based models for handling unseen styles. MetaScript [23] and Tang *et al*. [17] achieved superior style fidelity in Chinese character generation. Recent diffusion models like FontDiffuser [24] and Diff-Font [9] have demonstrated more stable training and higher fidelity for glyph-rich languages.

Masked Autoencoding (MAE) [10] has emerged as a highly effective pretraining strategy for Vision Transformers (ViTs), enabling them to excel in spatial reasoning tasks. By masking a large portion of input image patches and reconstructing the missing information, MAE forces the ViT to learn robust representations of spatial structures and semantic content. This self-supervised approach not only enhances the model's ability to generalize across diverse visual tasks but also reduces the dependency on extensive labeled datasets. For font generation, where intricate spatial patterns and context relationships are crucial, MAE pretraining empowers the ViT to understand and synthesize complex glyph structures effectively.

Hiera, introduced by Ryali *et al*. [15], demonstrates that a simple hierarchical ViT pre-trained with MAE can outperform vision-specific architectures while reducing computational complexity. These approaches highlight the growing utility of ViT-based methods in capturing global context and long-range dependencies [6, 20]. By integrating MAE-style pretraining, recent models have effectively bridged the gap between handcrafted design and scalable, data-driven approaches [4, 10].

3. Dataset Description

Our dataset looks like something as follows:

Figure 1. From top to bottom are Chinese, Japanese, Korean and English. The five styles are randomly picked from the total of 308 styles.

We have Korean, Chinese, English and Japanese as the four languages that we are using in this model and 308 styles in total. we collected fonts from but not limite to 51 font[1] and google font[2]. In the pretraining phase, we skipped Korean. We used total of 154 styles from the other three language with 800k images for pretrain, with the rest of the images split to train validation and test in 8:1:1 ratio, meaning around 1M for training and 150k for validation. As discribed in the dataset section, all the font file have a relative reference character. In our training phase, we decide to let the input image has or don't have the access to the reference character. Thus, to test our model throughout, we decided to distributes it randomly to four sets: content font unseen (20k), content reference character unseen (50k), style reference character unseen (35k) and style font unseen (80k).

4. Method

4.1. Dataset Preparation and Pretraining

The default VitMAE model is able to take 80*80 images, however, our initial training shows that the model does not learn well from such big size. Therefore, we reduce the size of the model to 24*24, and used the patch size of 16.
Our goal is to pretrain the model with part of our dataset to let the model learn the encoder and decoder that we are going to use in the later training. This process also significantly reduced our learning time, and it is proven to be capable to reconstruct the font picture and learn its pattern which we will benefit when used for our encoder and decoder in the main model.
The original Vit-Mae model have a masked ratio of 0.75, after some experiment, we realized that this mask ratio might be too high for some character font pictures because of its complexity, thus we reduced the mask rate to 0.65. Below are the two pictures comparing the two mask ratio.

4.2. General Overview

Vision Transformers (ViTs) are well-suited for font image generation due to their ability to capture both global structures and long-range dependencies. Since each glyph can be considered a single-object input of fixed position and size, the self-attention mechanisms inherent in ViTs enable simultaneous modeling of content and style features.

Our approach builds upon a previous bi-encoder architecture by introducing a transformer-based generator guided by content and style embeddings. Specifically, we employ two ViT-based encoders: a content encoder that extracts the structural features of an input glyph, and a style encoder that summarizes visual style cues from multiple reference images. By combining these representations, our generator produces novel glyphs that retain the original content while reflecting the target style.

The training objective involves minimizing both recon-

(a) The reconstruction result using the VitMAE-base model with the 0.75 mask ratio, we can see that the a lot of strokes are not properly reconstructed

(b) This is the reconstruction result using the modified mask ratio of 0.65, after 17 epochs, we can see that the image is pretty well reconstructed.

Figure 2. The comparison showed that it is necessary to pretrain the VitMAE model in our dataset. This will help us to start with a confident encoder and decoder for the main model.

struction errors and perceptual differences between the generated output and the ground truth. In addition to standard pixel-wise losses (e.g., L2), we incorporate perceptual metrics derived from a pre-trained VGG19 network. This encourages alignment with high-level visual features. While adversarial methods could further enhance realism, our primary focus is on producing accurate, style-consistent reconstructions guided by explicit content and style features.

Building on prior work, our goals are:

1. **Enhanced Style and Content Feature Extraction:** We refine our architecture and training strategies to improve the efficiency and fidelity of feature extraction, ensuring that the model effectively captures both the structural details and stylistic nuances of glyphs.
2. **Improved Multi-Lingual Performance:** Since fonts often must support a variety of scripts, we adapt our model to handle multi-lingual datasets. This ensures that it generalizes well to diverse glyph shapes and stylistic variations.

4.3. Baseline

To benchmark our advancements, we implemented four baseline models that share a general structure with our main approach but differ in components:

1. **ViT from Scratch (Grayscale) + MSE:** A ViT encoder-decoder from scratch trained on grayscale inputs using only MSE loss.
2. **ViT from Scratch (RGB) + MSE:** Similar to the above, but trained on RGB inputs to utilize color information. We thought this one will be more computational expensive than the gray scale version, but it turns out they are nearly the same.
3. **Facebook-ViTMAE-base + MSE:** A ViTMAE-base model (from Facebook) serving as encoder and decoder, with MSE loss guiding reconstruction.
4. **Pretrained ViTMAE + MSE:** A ViTMAE model pretrained on our dataset, retaining MSE as the loss function.

With the baseline and the main model trained on the same dataset, the baselines provide reference points against which we can measure the improvements introduced by our combined-loss, cross-attention bi-encoder method.

4.4. Main Method – Cross-Attention Bi-Encoder With Combined Loss

Our core architecture integrates a cross-attention mechanism to fuse content and style representations effectively. This approach diverges from simpler additive or concatenation-based methods by allowing the content embedding (queries) to selectively attend to and incorporate the most relevant stylistic elements (keys and values) from the style embedding. Through this targeted interaction, we ensure that the structural fidelity of the glyph is preserved while layering on the nuanced aesthetic traits of the desired style.

The model is built upon a Vision Transformer MAE (Masked Autoencoder) backbone and comprises three principal components:

1. **Content Encoder:** A ViT-based encoder that processes the input glyph image, producing a content embedding representing the character's structure and global shape.
2. **Style Encoder:** Another ViT-based encoder, similarly initialized, processes reference style images to produce an embedding capturing stylistic attributes such as stroke thickness, curvature, and texture patterns.
3. **Cross-Attention Module and Decoder:** The cross-attention module uses the content embedding as queries and the style embedding as keys/values, ensuring that style information is integrated into the content representation in a controlled, attention-driven manner. A ViT-based decoder, adapted from the MAE framework, reconstructs the complete glyph from these fused embeddings, compensating for initially masked patches. This decoder leverages the learned representations to generate a final image that mirrors the style's character while maintaining correct glyph shape.

Choice of Perceptual Layers for Loss Computation

Our combined loss function uses perceptual losses derived from a pre-trained VGG19 network. We select specific layers to extract both content and style features, ensuring the model captures the structural integrity and stylistic nuances of the target glyph:

- **Content Layers (e.g., '$relu2_2$'):** Mid-level layers of VGG19 represent more abstract features than the very

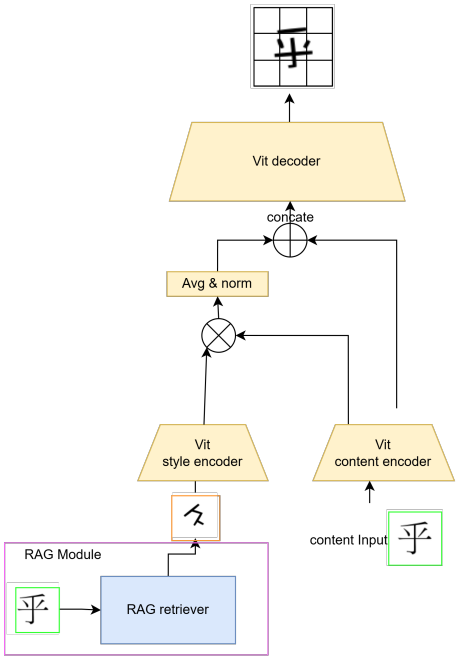

Figure 3. Our proposed model utilizes a cross-attention mechanism to guide the fusion of content and style embeddings, enhancing the flexibility and fidelity of glyph generation. Noted, the pink boxed RAG module is an add on to our main model. The green boxed font images are content input, and the orange boxed font image is style input.

early layers (which primarily capture edges and simple textures) but are not as abstract as the deepest layers. For fonts, these mid-level layers are well-suited to capture the shape and arrangement of strokes and overall glyph structure, ensuring the output maintains the identity of the character.

- **Style Layers (e.g., ‘$relu1_1$‘, ‘$relu2_1$‘, ‘$relu3_1$‘, ‘$relu4_1$‘, ‘$relu5_1$‘):** Lower-level layers focus on simple textures and patterns, while deeper layers capture increasingly complex structures. By sampling style features from multiple layers spanning early to deeper stages, we glean information about fine-grained textures, subtle stroke thickness variations, and higher-level aesthetic patterns. This multi-layer approach ensures that style loss guides the model to produce glyphs that faithfully reflect the chosen style across a wide range of visual attributes.

Combined Loss Formulation and Rationale

We define a combined loss that balances content fidelity, stylistic accuracy, and pixel-level precision:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{content} + \beta\mathcal{L}_{style} + \gamma\mathcal{L}_{MSE} \quad (1)$$

Here, $\mathcal{L}_{content}$ and $\mathcal{L}_{style}$ are the aforementioned perceptual losses derived from selected VGG19 layers, while $\mathcal{L}_{MSE}$ enforces pixel-level alignment.

From initial experiments, we observed that the raw scales of these losses differed substantially. If all losses were weighted equally, the MSE term risked becoming negligible, and the style cues might not be fully expressed. Since our primary objective is to perform style transfer, we emphasize $\mathcal{L}_{style}$ by setting a relatively higher weight $\beta = 0.4$ compared to $\alpha = 0.1$ for content. This ensures that the model focuses more on recreating the stylistic aspects accurately. At the same time, to maintain relevance for pixel-level details, we increase γ to 1.0, ensuring that the MSE term provides a stable, fine-grained anchor for sharpening edges and ensuring clear glyph boundaries.

Post-Training Refinement with L1 Loss

Following approximately 10 epochs of training with the combined loss, we introduce an additional short (0.5 epoch) refinement phase using an L1 loss in place of the perceptual losses:

$$\mathcal{L}_{refine} = \|I_{pred} - I_{gt}\|_1 \quad (2)$$

The motivation here is to smooth out residual artifacts, reduce subtle noise, and clean up the generated glyph images. L1 loss encourages sparsity in the error, which helps refine fine details and ensures the final output is both visually coherent and stylistically faithful.

Retrieval-Augmented Guidance (RAG) Module

To further enhance adaptability, we integrate a Retrieval-Augmented Guidance (RAG) module for scenarios where the user already has a set of style references available. In this scenario, the task input consists of a target content character and a desired style. The RAG module identifies the most suitable style reference from the known set and provides it to the model to generate the desired output.

The central hypothesis is that each character embodies a subset of the font style, and certain style features may not be fully represented in the input style reference characters. Characters with similar content or structure can provide additional style information during style transfer. **We hypothesize that the retrieval-enhanced method will extract the most relevant style information from the available reference set, enabling the transformer model to learn and replicate the style more precisely and efficiently.**

For each style, we use the content encoder to create embeddings for each character image and employ FAISS to build an exact search index.

Key Steps of the RAG Module

1. **Embedding Extraction:** For each available font style, we use our model's content encoder to generate content representations, which are concatenated to become the embedding. The embedding dimensions is `hidden_size × (patch_num + 1)`.
2. **FAISS Indexing:** We construct an exact search index for each style using FAISS. The IndexFlatL2 index is employed to perform similarity searches based on cosine similarity.
3. **Nearest Neighbor Retrieval:** Given a new content glyph and a target style, we compute the glyph's embedding and retrieve the most similar style references using nearest-neighbor search with our built embeddings.
4. **Font Character Generation:** The selected style reference exemplar is passed to the model, providing additional context to generate the target character.

The RAG module enhances scalability and flexibility, while also addressing some hard failure cases by dynamically retrieving style references that complement the content glyph. This approach, combined with our cross-attention bi-encoder ViT model and refined loss strategy, ensures a more robust and versatile glyph generation framework.

5. Benchmark Example – DiffusionFont

Among current state-of-the-art style transfer models, DiffuserFont is the only input-compatible method that we can feasibly test against. Although its focus is on Chinese characters, comparing our results with DiffuserFont provides a useful qualitative benchmark.

Figure 4. DiffuserFont captures content but deviates significantly in style.

While we illustrate with Chinese characters here, later evaluations consider all four languages in our dataset.

6. Experiment

As discussed in the Methods section, our experimental setup involves providing the model with a content image, a style image, and a corresponding ground truth image. The content and ground truth share the same underlying character, whereas the style and ground truth share the same stylistic attributes. Our approach utilizes content and style encoders derived from a pretrained ViTMAE model, enabling the network to efficiently capture content and style features. The different loss functions are then applied to balance the content fidelity, style accuracy, and pixel-level alignment, while the decoder component, also adapted from the pre-trained ViTMAE model, aims to reconstruct the target glyph.

6.1. Baseline models

As our baseline model uses the MSE difference between the ground truth and the generated image for the loss function, we believe that this is one of the naive ways to compare the difference between the two images. However, when we monitored the loss log, we realized that it takes around 3.5 hours to train on four 4090 per epoch and the loss goes down very slow. The sample results are shown as follows:

Figure 5. In each row, the first image is the content image, the second image is the style image, the third image is the ground truth image, and the last image is the generated image. The example is randomly picked from four testing set.

We used the same content input and style input for the three models to better compare it. All three models are trained with 10 epochs. At first glance, we can see that the stroke widths differ significantly from those in the ground truth. By examining the circled sections and comparing them to the corresponding ground truth images, it becomes clear that these models fail to capture the detailed stylistic elements we intended them to learn. However, we can see that the content is well preserved from the content input, thus MSEs are able to learn the content, but they do not perform well for transferring styles.

6.2. Bi-encoder Model

We run the same dataset for pretraining, training, validation, and testing across all models. The test results generated in the following are five randomly selected images from

the four testing set. Unfortuantely, as english only have 26 characters and 11 styles, we did not have it covered in the chart, but they are computed in Table 1 and Table 2 below.

6.2.1 Human Evaluation on Single Language Style Transfer

Figure 6. During the human evaluation, we covered the third column for the subject to determine the success of the style transfer. We have shown them 20 comparison in total, this is only an example. Because all of the subjects speaks english, so the enlish style transfer is not used for human evaluation

We recruited six subjects, divided into two groups. Group 1 (three subjects) spoke Chinese, Japanese, or Korean, and Group 2 (three subjects) spoke only English. Each subject rated style transfer on a scale of 0 (no transfer) to 2 (complete transfer).

All three participants in Group 1 rated the transfer as 2, indicating clear recognition of the intended style. In Group 2, two participants rated it as 2, while one gave a 1, noting difficulty in assessing small strokes. Group 1 members mentioned that viewing the characters from a distance helped confirm successful style transfer, while the Group 2 participant who scored it lower struggled with subtle details.

6.2.2 Cross-Language Style Transfer

Comparing Figures 7 and 8, we see that the bi-directional setup enables effective cross-language training. Furthermore, our model exhibits robustness beyond that of existing

Figure 7. The input content is Chinese, and the style input is Japanese hiragana. The generated result closely matches the ground truth, demonstrating our model's capability for cross-language style transfer.

(a)

(b)

Figure 8. Subfigures (a) and (b) show the same Japanese hiragana content paired with different Chinese style inputs. Both examples involve unseen content, unseen style, and no direct character reference.

font transfer methods: it does not require a character reference for either the style or content input. In contrast, other approaches cannot produce any output without a referenced character in their library. In the next section, we further demonstrate the effectiveness of our model.

6.2.3 Made-Up Content, Unseen Styles, and Handwriting

Figure 9. The input content is a made-up handwritten word, and the style input is also an unseen style. As a result, there is no ground truth reference for this example.

Conventional approaches typically focus solely on either

handwriting style transfer or standardized font transfer and often require a reference character in their library to produce any output. Our model, however, handles both scenarios seamlessly—even when no reference character is available. As shown in Figure 9, our method successfully interprets a purely handwritten, invented input and applies an unseen style, demonstrating its adaptability and robustness.

6.2.4 Numerical Comparison

Table 1 shows that our method attains relatively strong performance under challenging conditions. In contrast, as seen in Table 3, both DF-Font and CF-Font were trained for significantly longer (20k iterations vs. our 10), used a few-shot rather than one-shot approach, and benefited from more thoroughly tuned hyperparameters. It is also critical to note that our evaluation dataset is more difficult than those used in prior work. Moreover, our model, trained on a dataset of approximately 2.5 million samples across 308 styles, is substantially larger than the datasets employed by DF-Font and CF-Font—ours being more than 50 times their size. Their training setup required about 15 hours on four V100 GPUs for full convergence, while our setup, after an equivalent training duration on four RTX 4090 GPUs, reached only about 10 iterations due to the substantially larger dataset. Thus, if given more training time and computational resources, we expect our model's performance to surpass the current results. Furthermore, their models cannot generate characters not present in their reference libraries, whereas ours excels in generalization and scalability. Despite these disadvantages, our model's results are not far behind the current state of the art, and we are confident that with additional training iterations, we will achieve even better outcomes.

We have not included Diffusion-Font in our comparison. Although it is also a one-shot method and reports superior metrics compared to CF-Font and DG-Font in their paper, our empirical tests indicate that Diffusion-Font cannot consistently perform valid style transfer. Given these observed shortcomings, we remain skeptical about their reported results.

6.2.5 RAG Module

The evaluation results of our model with the Retrieval-Augmented Guidance (RAG) module (Table 2) indicate that the integration of RAG does not yield improvements across standard evaluation metrics, including L1 Loss, RMSE, SSIM, LPIPS, and FID, in unseen font scenarios. However, subsequent human analysis revealed that RAG effectively addresses specific failure cases and improves performance on hard examples, thereby enhancing the model's usability and adaptability. This demonstrates that while RAG does not directly impact traditional quantitative metrics, it plays a vital role in practical applications by ensuring robustness and reliability in challenging scenarios.

Figure 10. The target character includes the three-dot "water" radical, but the model initially generates the two-dot "ice" radical. The RAG module successfully retrieves the correct three-dot radical and enables the model to produce the desired glyph.

Figure 11. The RAG module resolves errors in generating the three-dot "water" radical by retrieving style information from a known set of characters.

Figure 12. For the challenging task of generating hiragana characters in a Chinese handwriting style, the RAG module retrieves structurally similar Chinese characters (historically related to the target hiragana) and produces highly accurate results.

7. Limitations and Future Work

- **Handwriting Input Potential:** Section 6.2.3 highlights the ability of our model to generate styles for made-up characters, showing our model's potential for handwriting input. However, due to time constraints, large-scale investigation was not conducted.
- **Expanding Writing Systems:** The model demonstrates strong performance across alphabetic-like scripts (e.g., Japanese Hiragana and English) and logographic characters (e.g., Kanji). Exploring its generalizability to other writing systems, such as Arabic or historical scripts like Linear A, would be an intriguing direction.
- **Few-Shot Generation:** Our model could easily extend to a few-shot setting by invoking the style encoder multiple times and averaging the style representation. While this approach has not been tested, investigating the model's performance in a few-shot scenario is an exciting avenue for future research.

Table 1. Performance metrics across different unseen font scenarios. **SS** refers to Style Font Unseen, **SC** refers to Style Reference Character Unseen, **CS** refers to Content Font Unseen, and **CC** refers to Content Reference Character Unseen. The dataset consists of all four languages. Our method demonstrates robust performance in font generation across multiple languages without requiring knowledge of input character content or style, highlighting its generalization.

Unseen Settings				**Metrics**				
SS	**SC**	**CS**	**CC**	**L1 Loss ↓**	**RMSE ↓**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**
✓	×	×	×	0.18697	0.54534	0.66566	0.19229	25.56092
✓	✓	×	×	0.18952	0.54987	0.66362	0.19494	25.23063
×	×	✓	×	0.18781	0.54672	0.66534	0.19200	25.52842
×	×	✓	✓	0.18879	0.54857	0.66485	0.19439	26.15635

Table 2. Performance metrics across different unseen font scenarios for our model enhanced with the RAG module. Surprisingly, RAG does not improve standard evaluation metrics, including L1 Loss, RMSE, SSIM, LPIPS, and FID.

Unseen Settings				**Metrics**				
SS	SC	CS	CC	**L1 Loss ↓**	**RMSE ↓**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**
✓	×	×	×	0.19858	0.56531	0.65361	0.20315	26.49332
✓	✓	×	×	0.19875	0.56551	0.65507	0.20319	26.37470
×	×	✓	×	0.19812	0.56434	0.65462	0.20121	26.61482
×	×	✓	✓	0.19930	0.56612	0.65337	0.20307	27.13604

Table 3. Performance metrics for unseen font style (SS). The evaluation results for DG-Font and CF-Font are sourced from the CF-Font paper.

Unseen Settings					**Metrics**				
SS	SC	CS	CC		**L1 Loss ↓**	**RMSE ↓**	**SSIM ↑**	**LPIPS ↓**	**FID ↓**
✓	×	×	×	DG-Font	0.07841	0.2442	0.6853	0.1198	27.98
✓	×	×	×	CF-Font	0.07394	0.2354	0.7007	0.1182	26.51

References

[1] 51Font, . 2

[2] Browse Fonts, . 2

[3] Samaneh Azadi, Matthew Fisher, Vladimir Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-Content GAN for Few-Shot Font Style Transfer, 2017. arXiv:1712.00516 [cs]. 1

[4] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEiT: BERT Pre-Training of Image Transformers, 2022. arXiv:2106.08254. 2

[5] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens, 2022. arXiv:2112.04426. 1

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, 2021. arXiv:2010.11929 [cs]. 1, 2

[7] Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, and Roee Litman. ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation, 2020. arXiv:2003.10557 [cs]. 1

[8] Hideaki Hayashi, Kohtaro Abe, and Seiichi Uchida. GlyphGAN: Style-Consistent Font Generation Based on Generative Adversarial Networks, 2019. arXiv:1905.12502 [cs]. 1

[9] Haibin He, Xinyuan Chen, Chaoyue Wang, Juhua Liu, Bo Du, Dacheng Tao, and Qiao Yu. Diff-Font: Diffusion Model for Robust One-Shot Font Generation. *International Journal of Computer Vision*, 2024. 1, 2

[10] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners, 2021. arXiv:2111.06377 version: 2. 1, 2

[11] Jeong-Sik Lee, Rock-Hyun Baek, and Hyun-Chul Choi. Arbitrary Font Generation by Encoder Learning of Disentangled Features. *Sensors*, 22(6):2374, 2022. Number: 6 Publisher: Multidisciplinary Digital Publishing Institute. 1

[12] Chenhao Li, Yuta Taniguchi, Min Lu, and Shin'ichi Konomi. Few-shot Font Style Transfer between Different Languages. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 433–442, 2021. ISSN: 2642-9381. 1

[13] Yitian Liu and Zhouhui Lian. FontTransformer: Few-shot High-resolution Chinese Glyph Image Synthesis via Stacked Transformers, 2022. arXiv:2210.06301 [cs]. 2

[14] Vittorio Pippi, Silvia Cascianelli, and Rita Cucchiara. Handwritten Text Generation from Visual Archetypes, 2023. arXiv:2303.15269 [cs]. 2

[15] Chaitanya Ryali, Yuan-Ting Hu, Daniel Bolya, Chen Wei, Haoqi Fan, Po-Yao Huang, Vaibhav Aggarwal, Arkabandhu Chowdhury, Omid Poursaeed, Judy Hoffman, Jitendra Malik, Yanghao Li, and Christoph Feichtenhofer. Hiera: A Hierarchical Vision Transformer without the Bells-and-Whistles, 2023. arXiv:2306.00989. 2

[16] Danyang Sun, Tongzheng Ren, Chongxun Li, Hang Su, and Jun Zhu. Learning to Write Stylized Chinese Characters by Reading a Handful of Examples, 2018. arXiv:1712.06424 [cs, stat]. 1

[17] Licheng Tang, Yiyang Cai, Jiaming Liu, Zhibin Hong, Mingming Gong, Minhu Fan, Junyu Han, Jingtuo Liu, Errui Ding, and Jingdong Wang. Few-Shot Font Generation by Learning Fine-Grained Local Styles, 2022. arXiv:2205.09965 [cs]. 2

[18] Yuchen Tian. kaonashi-tyc/Rewrite, 2024. original-date: 2016-10-26T03:11:46Z. 1

[19] Yuchen Tian. kaonashi-tyc/zi2zi, 2024. original-date: 2017-02-17T23:18:04Z. 1

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2023. arXiv:1706.03762 [cs]. 2

[21] Qi Wen, Shuang Li, Bingfeng Han, and Yi Yuan. ZiGAN: Fine-grained Chinese Calligraphy Font Generation via a Few-shot Style Transfer Approach. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 621–629, 2021. arXiv:2108.03596 [cs]. 1

[22] Yangchen Xie, Xinyuan Chen, Li Sun, and Yue Lu. DG-Font: Deformable Generative Networks for Unsupervised Font Generation, 2021. arXiv:2104.03064 [cs]. 1, 2

[23] Xiangyuan Xue, Kailing Wang, Jiazi Bu, Qirui Li, and Zhiyuan Zhang. MetaScript: Few-Shot Handwritten Chinese Content Generation via Generative Adversarial Networks, 2023. arXiv:2312.16251 [cs]. 1, 2

[24] Zhenhua Yang, Dezhi Peng, Yuxin Kong, Yuyi Zhang, Cong Yao, and Lianwen Jin. FontDiffuser: One-Shot Font Generation via Denoising Diffusion with Multi-Scale Content Aggregation and Style Contrastive Learning, 2023. arXiv:2312.12142 [cs]. 2

[25] Yong Yu. Few Shot POP Chinese Font Style Transfer using CycleGAN. *Journal of Physics: Conference Series*, 2171(1): 012031, 2022. Publisher: IOP Publishing. 1

[26] Yexun Zhang, Ya Zhang, Wenbin Cai, and Jie Chang. Separating Style and Content for Generalized Style Transfer, 2018. arXiv:1711.06454 [cs]. 1