Few-Shot Content-Level Font Generation

Saima Majeed[1,2], Ammar Ul Hassan[1], and Jaeyoung Choi[1*]
[1] Soongsil University, Seoul, South Korea
[e-mail: choi@ssu.ac.kr]
[2] Air University, Islamabad, Pakistan
[e-mail: saimamajeed089@gmail.com]
[*]Corresponding author: Jaeyoung Choi

Received June 21, 2021; revised January 27, 2022; revised March 26, 2022; accepted April 14, 2022; published April 30, 2022

Abstract

Artistic font design has become an integral part of visual media. However, without prior knowledge of the font domain, it is difficult to create distinct font styles. When the number of characters is limited, this task becomes easier (e.g., only Latin characters). However, designing CJK (Chinese, Japanese, and Korean) characters presents a challenge due to the large number of character sets and complexity of the glyph components in these languages. Numerous studies have been conducted on automating the font design process using generative adversarial networks (GANs). Existing methods rely heavily on reference fonts and perform font style conversions between different fonts. Additionally, rather than capturing style information for a target font via multiple style images, most methods do so via a single font image. In this paper, we propose a network architecture for generating multilingual font sets that makes use of geometric structures as content. Additionally, to acquire sufficient style information, we employ multiple style images belonging to a single font style simultaneously to extract global font style-specific information. By utilizing the geometric structural information of content and a few stylized images, our model can generate an entire font set while maintaining the style. Extensive experiments were conducted to demonstrate the proposed model's superiority over several baseline methods. Additionally, we conducted ablation studies to validate our proposed network architecture.

Keywords: Generative adversarial networks, Hangul Fonts, Image-to-Image translation, Style transfer.

1. Introduction

With the continued advancement of technology in the digital era, images and videos have become major components of our daily activities. Therefore, visual media has become a major study target. Image processing, computer vision, and computer graphics can be utilized to study visual media. However, creating scalable generative models to capture visual media distributions remains as a significant challenge in machine learning. Extensive research has been conducted in the fields of style transfer [1], image generation [2], image classification [3], and image-to-image translation for capturing visual media distributions. These techniques can be utilized to generate fonts to simplify the font design process.

Fonts are necessary to represent text in specific styles and sizes. They play an important role in modern media because they are representations of text that not only add esthetic value to multimedia, but also add relevant property value. However, creating diverse font styles is difficult for novice users. Additionally, conventional calligraphic and artistic font design is a tedious and labor-intensive task based on the complexity of styles and content, particularly in the case of Chinese, Japanese, and Korean fonts. To create diverse font styles, a user must create different files for each distinct font style. For example, if a designer wants to create 100 different font styles, then they must create 100 different font files corresponding to each unique style by generating a set of characters for each language from scratch. This requires significant time and effort from designers. Therefore, an automated method is desirable to facilitate the font design process for large-scale-character languages. The problems associated with such a method can be resolved by combining techniques from the previously mentioned fields, where deep learning plays a vital role in font generation.

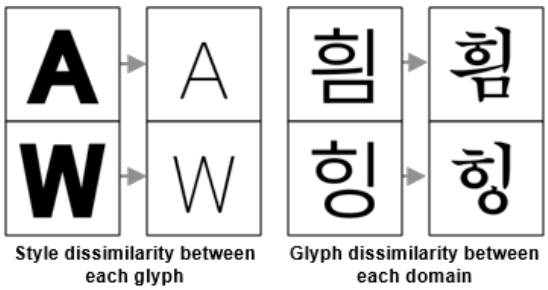

Fig. 1. Problems while transferring one font to another

A number of studies on image-to-image translation have been conducted and mapped onto the font design process to reduce the workload of font designers. However, such approaches work on conditional strategies and have significant limitations. (1) The method presented in [4] requires paired training data to learn style patterns in the target domain in a supervised manner. (2) The method from [5] does not require paired training examples, but it is only able to learn one-to-one mappings instead of many-to-many mappings for domains with complex relationships. (3) Additionally, based on fixed reference content, existing methods require fine-tuning on unseen domains. (4) The generalization ability of existing models is also limited by one-hot category embedding concepts. (5) Most font generation work focuses on font transfer, which entails transferring glyphs from one font to another font, but it is difficult for a model to capture the style patterns of a target domain when there are large differences between domains and glyph representations, as shown in **Fig. 1**. (6) Instead of using multiple images of a style to capture style features, existing methods only use a single image of the

target font domain. To address these issues, we propose a novel method for synthesizing fonts. First, to handle problem (5), we propose using the internal and external structures of content references instead of complete glyph images, which contain larger pixel values that are less relevant. Second, to preserve important style properties and achieve sufficient representation capabilities, we propose the use of multiple images of a target style for extracting style features. Unlike traditional approaches that focus on transferring styles by retaining content, the goal of our method is to maintain styles while altering content glyphs based on geometric representations instead of using one-hot content embedding, which limits generalization ability. To verify the capabilities of the proposed method, we conducted qualitative and quantitative experiments to confirm multi-style font generation with consistent characteristics.

2. Motivation

The primary goals of this work can be summarized as follows.

1. Style consistency: Perform content transfer instead of font transfer and verify the generated font styles using similarity and dissimilarity metrics.
2. Readability: Use k stylized images and character content features as inputs and generate a complete character set for the desired font style without compromising content readability or style.
3. Reduce fine-tuning: Instead of fine-tuning a model on a new font style, we find the closest style in a learned distribution to generate the remainder of the font set because our model does not require paired training examples or complete ground-truth labels for the corresponding domain.
4. Preserving structural properties: To preserve the structural properties of content, we make use of geometric representations of content instead of complete font images. This allows us to filter out less relevant information and only preserve structural properties as content. Using only the geometric structures of content can reduce the amount of data that must be processed.
5. Preserving style properties: To learn sufficient style representations, we make use of multiple font style images from each domain instead of using only a single image for style extraction.
6. Legibility: The legibility of generated characters is verified using a convolutional neural network.

3. Related Research

In recent years, many studies have utilized deep learning for font synthesis and generalization. Regarding architectures, these studies can be roughly subdivided into model-driven, deep neural network (DNN)-based, and Bayesian deep- learning-based approaches.

3.1 Prerequisite Knowledge

Large-scale data have empowered deep learning algorithms to achieve rapid advancement. Specifically, generative adversarial networks (GANs) have the ability to learn complex distributions and synthesize high-quality and semantically meaningful samples from standard data samples. Goodfellow et al. [6] introduced the first GAN in 2014. Their model outperformed other generative models at synthesizing images. Since then, GANs have become a popular research topic in computer vision and multimedia. A GAN consists of two

competitive neural networks called a generator and discriminator. The task of the generator is to generate realistic samples to deceive the discriminator by taking noise as inputs, whereas the task of the discriminator is to discriminate samples generated by the generator from real samples. Training continues until the generator is able to fool the discriminator in an adversarial game in a mix-max manner.

Two main research directions have been considered for GANs. The first research direction attempts to improve training, efficiency, and stability issues related to GANs using information theory [7, 8]. The second research direction focuses on GAN applications and architectures in the field of computer vision [9, 10]. GANs perform very well at learning complex distributions. However, there are three main challenges that must be considered: mode collapse, vanishing gradients, and a lack of output control. Therefore, the authors of [11] proposed conditional GANs (cGANs) as an improved GAN architecture. A cGAN has the same architecture as an original GAN, except that an extra condition is added to the generator and discriminator networks to address the aforementioned challenges related to GANs. The conditions applied in cGANs can be class labels, text, or images depending on the target problem. A cGAN empowers all image-synthesis applications. The following section discusses the strengths of GANs in terms of font generation.

3.2 Font generation

3.2.1 Model-driven based font generation studies

Manual font design requires professional skills, typography expertise, and artistic capabilities. Additionally, the design of fonts from scratch requires significant time and labor. Therefore, an automated process is required to synthesize fonts from a few handcrafted samples of an existing font. Recently, several model-driven font foundation studies have been presented. The authors of [12] proposed an outline-based interpolation method to synthesize new fonts by introducing heuristics and interpolation in a high-dimensional space. The concept of interpolating font parameters was initially introduced by Knuth [13]. Based on [13], this method was extended to Chinese characters [14] and some researchers have focused on parametric models [15]. In addition to parametric approaches, a number of other model-driven approaches have been proposed, including stroke-level font generation [16], statistics-based methods [17], contour-based methods [18], and transformation-based methods [19]. However, these methods suffer from problems such as a limited ability to extrapolate new fonts based on the point-wise correspondence between glyphs of the same characters in different fonts. Additionally, constructing handcrafted features and manually adjusting complicated parameters for diverse stylizations is a cumbersome task.

3.2.2 DNN-based font generation studies

Based on advancements in various fields of technology, the development of imaging and data has also accelerated. Data-driven approaches have become popular for image processing and analysis techniques. Recently, these techniques have been applied to automate the font synthesis process using DNNs in the fields of computer vision and computer graphics. Thus far, numerous deep-learning-based architectures have been proposed that are either specifically designed or appropriate for font synthesis. Such advancement allows researchers to model glyphs based on images. Image synthesis tasks using DNNs can be further subdivided into two categories: image-to-image translation and noise-to-image translation. A number of studies have been conducted on these subdivisions.

The concept of image synthesis dates back to image analogies [20], where a framework

utilizes two stages called the design phase and application phase to create analogous results by taking a pair of images as inputs. Upchurch [21] considered the image analogy concept, mapped it to an image synthesis task, and proposed a revised variational autoencoder to separate styles from content. Recently, several attempts have been made to utilize GANs for font synthesis applications. Zi2zi [22], which is an extended version of pix2pix [4], was the first attempt to use GANs to synthesize a target font using a combination of the methods discussed in [4, 23, 24]. Additionally, the authors of [25] proposed an example-guided font generation method using a U-net-based generator for GANs and claimed that their strategy makes it easier to balance loss functions compared to the method described in [22]. There are many variants of font generation that utilize GANs. For example, DCFont [26] is an endwise learning system that can generate a complete Chinese font library by considering a limited number of predesigned characters. Chinese font generation is framed as a mapping from an existing font to a personalized artistic style when utilizing the CycleGAN [5] concept. To learn and produce high-quality fonts, SCFont [27] recently introduced a stacked DNN to capture detailed content and synthesize high-quality fonts using a multi-stage architecture design and glyph-structure-guided information. MC-GAN [28] makes use of stacked cGANs to predict coarse glyph shapes using a glyph network and applies textures to output grayscale glyphs through an ornamentation network. This method yields impressive results, but it is limited to alphabetic characters and lacks generalization abilities based on the large number of parameters required for the ornamentation network. As a diverse one-stage framework, AGIS-Net [29] has been proposed. This method can generate shape and texture styles by considering a few predesigned samples.

In addition to image-based approaches, noise-based approaches for font synthesis have also emerged, providing diverse and high-quality results. GlyphGAN [30] is a method inspired by deep convolutional GANs [31] that considers the style consistency, legibility, and diversity characteristics of generated fonts in an unsupervised manner. However, this method is unable to control output styles because it learns distributions from noise. Therefore, mode collapse may occur. FontGAN [32] uses a combined GAN structure for styling and de-styling Chinese characters. However, none of the aforementioned architectures are specifically designed to synthesize large-scale fonts with high readability and consistency from a few samples of handcrafted stylized characters. Additionally, considering a fixed reference font as input content fails to capture the internal and outer structures of actual characters when the target domain has large dissimilarity from the content in terms of style. Furthermore, existing methods focus on font style transfer from one domain to another by maintaining content and altering style instead of performing content transfer. In this paper, we propose a method for performing content transfer by preserving styles from few-shot examples. Additional details and experiments for our model are described in the following sections, which highlight the superiority of our approach.

4. Proposed Network Description

4.1 Network Architecture Details

Font design mainly relies on underlying geometric structure and shape content. Internal (skeleton) and external (edges) representations convey information related to connected curves, strokes, boundaries, and other structural properties. This information can be utilized for modeling instead of considering a complete image as a source font. Such images contain irrelevant features that must be filtered out to obtain specific local details and structural

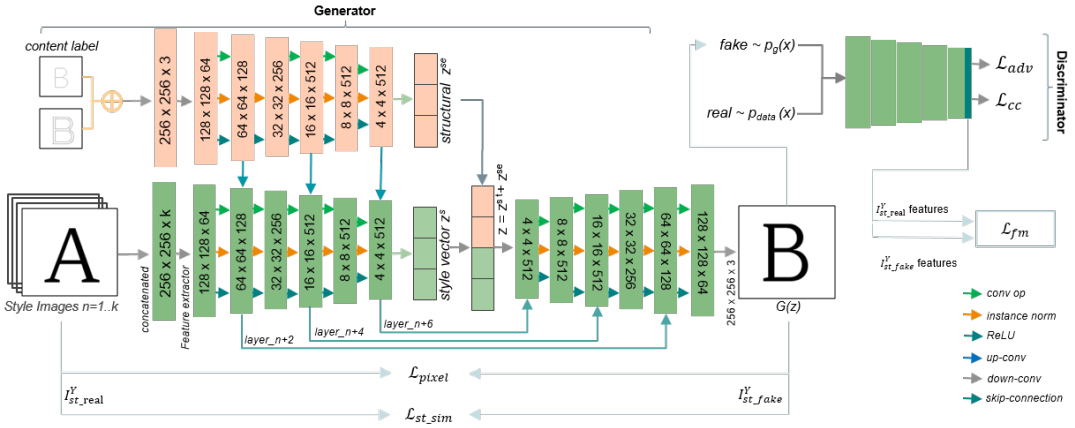

Fig. 1. The proposed network architecture. The Generator architecture consists of a content encoder, style encoder, and a decoder. The Discriminator takes both the real and generated images for the adversarial loss and the character classification loss. More details are provided in the main text.

information. Therefore, we utilize images containing edges and skeletons for a source font to preserve topological details for improved font synthesis. The content image inputs for our model contain edges and outlines that are concatenated channel-wise for enhanced structural and semantic guidance that is relevant to the content domain.

Additionally, to obtain style information regarding the target font, instead of using an individual target font image to extract style information, we consider a randomly selected set of k images from the target font domain. The reason for utilizing k stylized images for reference is to extract style-relevant and common features instead of style-irrelevant features, which should be ignored. This allows our model to devote additional attention to the target style. For example, if we input one bold font style image into our model to extract its style (here, style represents the boldness and strokes of the font), it will not be able to extract in-depth font-specific style information that could be mapped to source content images, leading to a loss of generalization ability. Therefore, at each iteration, $k < n$ images from the target domain are inputted into the style encoder by concatenating the k selected images in a channel-wise manner. To improve robustness, a combination of many different images is utilized for extracting styles.

Let I^X_{se} represent the concatenated edge and skeleton images of the content font. The target style reference is represented by I^Y_{st}, which is channel-wise concatenated with the k stylized images. The proposed model is based on an encoder-decoder architecture for font character synthesis, which is defined by a content edge-skeleton encoder (E_{se}), style encoder (E_{st}), style decoder (G_Y), and multi-task discriminator (D_Y). The complete network architecture is shown in **Fig. 2**.

The content font inputted into E_{se} is in the form of a geometric representation that contains the common features of the content characters after downsampling. Later, an encoded latent representation is passed into E_{st} to represent the content characters onto which the target style should be mapped. The content characters are style-irrelevant but are used for character embedding based on the target font characters. Additionally, the style encoder can learn various font representations by considering inter-style consistency based on k stylized target images instead of a single style image. During the training stage, the content font and reference style are encoded into latent variables as follows:

$$z^{se} = E_{se}(I^X_{se}), \tag{1}$$

$$z^{st} = E_{st}(I_{st}^{Y}), \tag{2}$$

where z^{se} and z^{st} represent the content character latent variables from content domain X, and z^{st} represents the style latent variable from the target style domain Y. Next z^{se} combined with z^{st} is fed into G^{Y} to synthesize images as follows:

$$I_{st\_fake}^{Y} = G_{Y}(z^{st}, z^{se}), \tag{3}$$

where $I_{st\_fake}^{Y}$ denotes a generated image with the content of z^{se} and style of z^{st}. We propose using skip connections between E_{st} and G_{Y} to facilitate enhanced style mapping. Additionally, the intermediate layer information of E_{se} is passed into E_{st} for content guidance. The goal of incorporating skip connections is to obtain features at all levels.

For adversarial training, a multi-task discriminator (D_{Y}) based on PatchGAN [4] was adopted in our model. The decoder outputs $I_{st\_fake}^{Y}$ are fed into the discriminator as fake samples along with real samples $I_{st\_\mathrm{real}}^{Y}$ for discrimination. The goal of D_{Y} is not only to represent realness and fakeness for given samples, but also to classify characters. The key change in the D_{Y} model is the introduction of a dense layer following the flattened layer, which is used for character classification. The size of this layer depends on the total number of content characters on which the model is trained, similar to the method discussed in [33]. This improves the training of GANs for image synthesis. Additionally, it incorporates character stylization based on a few samples of the target style.

During the testing stage, E_{st} encodes unseen input target-style images into z^{st} latent variables representing the target style. For content characters, E_{se} encodes the desired content. The concatenated latent variables in E_{st} and E_{se} are passed to G_{Y}, which then generates complete stylized font set characters by utilizing few-shot samples of the desired style. After training, the model has learned sufficient local details and structural information related to content characters. Therefore, it is able to map an unseen style to unseen content, because content is not fixed in the form of one-hot embedding.

4.2 Model Implementation Details

4.2.1 Encoder Details

To obtain font-style-specific and abstract details in an efficient manner, we employ two encoders, namely E_{se} and E_{st}. E_{se} uses six downsampling layers to obtain explicit features from content. Each layer uses a convolution operation and instance normalization [34] with a rectified linear unit (ReLU) as an activation function. Additionally, each second layer of information in E_{se} is transformed into the corresponding layer in E_{st} to obtain sufficient features at different scales from the given content's internal and external structures. E_{st} contains six downsampling layers and each layer concatenates the latent information from the previous layers to preserve features ranging from abstract features to specific style features. Because the content is style irrelevant, we focus on learning representations of various fonts while maintaining content. Therefore, for enhanced style extraction and transfer between font sets, we employ skip connections from all layers of E_{st} to G_{Y}.

4.2.2 Decoder Details

To map the concatenated final latent variables to the target content images, we employ one decoder, namely D_{Y}. This decoder contains the same number of upsampling convolutional

layers and uses the same operational settings as E_{st}. Additionally, to refine $I^{Y}_{st\_fake}$, skip connections and structural objectives are used to obtain refined and high-quality results.

4.2.3 Discriminator Details

In our approach, we employ a patch-level discriminative model. This model receives channel-wise concatenated positive and negative samples as inputs. It outputs score maps instead of single values based on a given positive and negative sample patch. Instead of the traditional ReLU, we employ the leaky ReLU function and perform no normalization. Leaky ReLU [35] has been proven to be effective at preventing the vanishing gradient problem and performing continuous optimization. To improve the discernibility of D_Y, a fully connected layer is implemented at the end of D_Y. This layer is responsible for classifying generated characters with their true character labels which indicate the true character content of the input image. Additionally, it helps quantify the recognition capabilities for the given content. The size of the fully connected layer is selected based on the total number of characters used for training the model for example, in case of English alphabets we used 26 characters upper-case letters hence 26 true labels are considered for the fully connected layer.

4.3 Learning Objective

An objective function plays a pivotal role in the performance and quality of a model for image synthesis. In this study, several loss functions were combined. The full objective for our model can be formulated as follows:

$$\mathcal{L}^{*} = \mathcal{L}_{adv} + \mathcal{L}_{cc} + \mathcal{L}_{L1} + \mathcal{L}_{st\_sim} + \mathcal{L}_{fm}. \tag{4}$$

4.3.1 Adversarial Loss

To achieve legibility, we employ conditional non-saturating GAN loss instead of min-max loss for training our model. This loss is an alteration of generator loss for handling the saturation problem with refined changes compared to min-max GAN loss. Similar to the original GAN loss function [6], the discriminator has the ability to classify generated images as fake images at the beginning of training because the quality of generated images is very low. Therefore, the term D(G(z)) in min-max moss has gradients near zero. To avoid this issue, non-saturating GAN loss is recommended for practical use [36]. Additionally, this function converges more quickly than the saturated GAN loss used in the original U-Net architecture. The adversarial loss in our model is formulated as follows:

$$\mathcal{L}_{adv}(D) = \mathbb{E}_{x,y}[\log D(\mathrm{x},\mathrm{y})] + \mathbb{E}_{x,z}[\log (1 - D(\mathrm{x}, G(\mathrm{x},\mathrm{z})], \tag{5}$$

$$\mathcal{L}_{adv}(G) = -\mathbb{E}_{x*}[\log D(G(\mathrm{x},\mathrm{c}))]. \tag{6}$$

Instead of minimizing loss, the generator maximizes the logarithmic probability of images to be predicted as real instead of fake. This function also uses a stable weight updating mechanism.

4.3.2 Content Classification Loss

To verify that content transfer works well, we classified the content of generated images. We employed content classification loss in the discriminator to classify generated content to make it recognizable by the network and learn content in a manner that is more efficient. This loss

can be formulated as follows:

$$\mathcal{L}_{cc} = -\mathrm{E}\left[\log D_{cls}\left(c^{*}|I^{Y}_{st\_real}\right)\right] - \mathrm{E}\left[\log D_{cls}\left(c|I^{Y}_{st\_fake}\right)\right], \quad (7)$$

where c* represents the label of the ground-truth image. By minimizing loss, D_y learns to classify generated images into the correct content classes and G_y learns to synthesize images that are more similar to the target content.

4.3.3 Pixel-wise loss

We adopted L1 regularized loss in our generator. Unlike other image synthesis tasks, font synthesis has more stringent requirements for the legibility and consistency of generated content. Therefore, to synthesize noisy images, we employed pixel-wise loss in the generator.

$$\mathcal{L}_{L1} = \mathbb{E}_{y,y'}\| I^{Y}_{st\_real} - I^{Y}_{st\_fake}||1 \text{ *lambda}, \quad (8)$$

where $I^{Y}_{st\_real}$ is a target style image preserving the same content given at the time of generation, $I^{Y}_{st\_fake}$ is a generated image preserving the style of I^{Y}_{st}, and lambda is a hyperparameter. The weight of $\mathcal{L}_{L1}$ is set to zero when the content contains no corresponding glyph images in the desired style. However, during training, we have ground-truth images of every piece of content in the target style. Therefore, the weights of $\mathcal{L}_{L1}$ will always be non-zero.

4.3.4 Structural Similarity Loss

To produce visually pleasing images, we adopted structural similarity loss in the generator. This type of loss can capture changes in structural information. It utilizes a Gaussian function to calculate the mean, variance, and covariance of given images. Therefore, regardless of contrastive conditions, it statistically calculates the differences between images as follows:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y+c_1)(\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)}. \quad (9)$$

The structural similarity index (SSIM) lies in the range of [0,1], where values close to one represent high similarity and values close to zero represent low similarity.

$$\mathcal{L}_{st\_sim} = \mathbb{E}_{y,y'}\,[1 - SSIM(y, G(z))], \quad (10)$$

where y represents $I^{Y}_{st\_real}$ and G(z) represents $I^{Y}_{st\_fake}$. The generator minimizes differences to learn structural information efficiently in the target domain.

4.3.5 Feature Matching Loss

In addition to basic pixel-level and structural-level loss, feature-level loss can also be utilized for high-quality image synthesis. Pixel-level loss yields visually acceptable results. However, it requires spatially aligned generated and target images to compute differences. Therefore, to compute differences at the feature level instead of the pixel level, feature matching loss is recommended for generating high-quality results. It computes the similarity between images using feature maps from different layers while ignoring the alignment of pixels.

$$\mathcal{L}_{fm} = \mathbb{E}_{y,y'}\,||f(I^{Y}_{st_{real}}) - f(G(z)||_2^2 \quad (11)$$

The generator minimizes the statistical differences between the features of $I^{Y}_{st_{real}}$ and $G(z)$. Features are extracted from an intermediate layer of the discriminator, where information is more distilled compared to VGG19 outputs. The L2-norm is computed between the means of the extracted feature spaces of I_X and I_Y.

5. Experiments and Results

In this section, we first introduce the parameter settings for the proposed model, target dataset, and baseline methods. Our model is then evaluated quantitatively and qualitatively to validate the superiority of the proposed approach. Finally, an ablation study is performed to demonstrate the importance of each component in the proposed method.

5.1 Experimental Parameter Settings

The proposed model was implemented using the TensorFlow software. We trained the model for 120 epochs on an NVIDIA 2080ti GPU using the Adam optimizer with a learning rate of 0.0002. The weights $\mathcal{L}_{ssim}$ and $\mathcal{L}_{L1}$ were set to 10.0 and 100.0, respectively. Additional architectural and implementation details were described in Section 5.2.

5.2 Dataset Details

Very few datasets are publicly available for font synthesis tasks. Only a few were provided in [28, 29]. Therefore, we used the open-source Google Fonts (https://fonts.google.com/) application to prepare a dataset for Latin characters (English alphabets). This application provides fonts for 29 different languages. For a Korean character dataset, we utilized font styles from the same source. To prepare images of the Latin and Korean datasets, we used the Unicode-based module [37]. After preparing the datasets, we detected and removed duplicate font images to generalize our model. We also excluded font styles that did not support specific characteristics of the Korean language. Additionally, to obtain geometric representations of content, such as the inner and outer structures of the content fonts, we utilized existing approaches for obtaining the outline data [38] and inner structures [39] of the content. We applied the same methods to the Latin and Korean content to obtain geometric representations. To achieve enhanced performance, we constructed our dataset using 3740 font styles for English alphabets and 100 font styles for Korean characters. Each image was 256 × 256 pixels in size. In each experiment, we utilized 80% of the dataset for training and the remaining 20% of the dataset for evaluation. To evaluate the proposed and baseline methods, we utilized few-shot stylized characters at inference time, even though we had ground-truth images. This was done (1) to see how well each model captured each style and (2) how well each model performed content-level transfer by preserving the style of each font using a few-shot character set.

5.3 State-of-the-art competitors

We considered two state-of-the-art methods for comparison: pix2pix and zi2zi. Pix2pix is based on GANs, which are commonly used for image synthesis. It uses paired image data to learn mappings between domains in a one-to-one manner. It cannot learn one-to-many mappings. Therefore, it can learn only one style at a time. Zi2zi extended the concept of pix2pix and added the style embedding concept to control and learn one-to-many styles simultaneously. Both methods aim to preserve content and change styles. The goal of our approach is to preserve styles and change content to reduce dissimilarity and fine-tuning

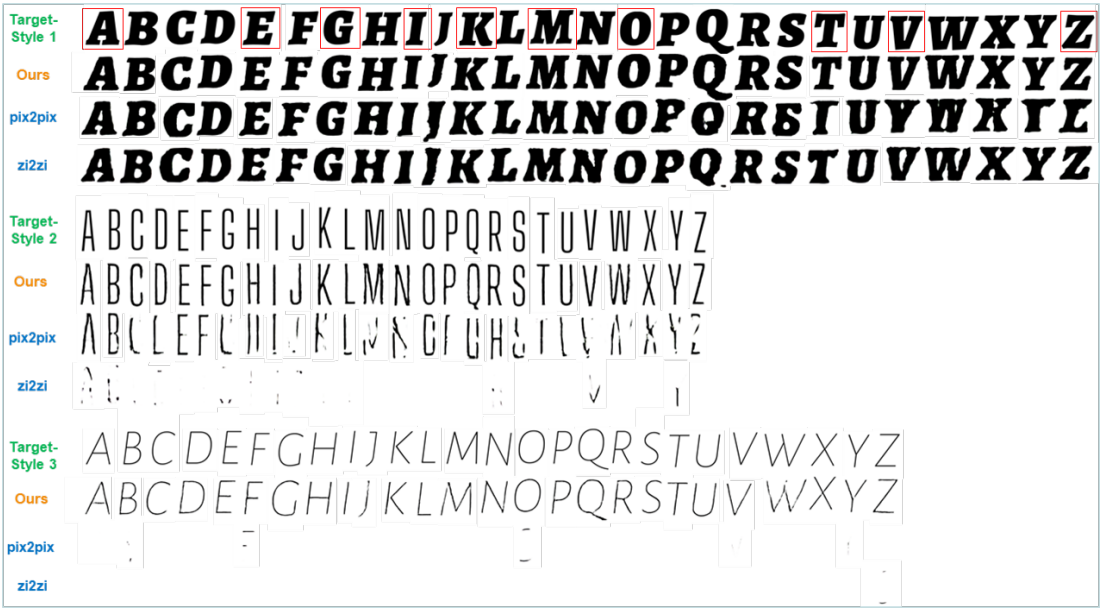

Fig. 2. Visual comparison of Latin glyph images synthesized by the proposed and baseline methods. The red box shows the few-shot stylized characters given to the model at inference time. The baseline methods (pix2pix and zi2zi) fail to generate the output images when the target font style is thin (as shown in the results of Target-Style 2, Target-Style 3).

constraints. Therefore, we utilized the same dataset for all three methods and learned content while preserving styles. Additionally, zi2zi requires fine-tuning on unseen font styles. We exclude fine-tuning because our model does not require fine-tuning on unseen fonts. To exclude fine-tuning, we remove the style classification layer from zi2zi and evaluate it in the same setting as the proposed method.

5.4 Experiment Results Evaluation

5.4.1 Qualitative Analysis:

In addition to comparing the effectiveness of our method to that of the baseline methods, we conducted a qualitative evaluation. For Latin and Korean characters, we trained our model using the aforementioned settings. At inference time, we use few-shot stylized Latin characters as inputs and performed content transfer by extracting a geometric representation of the content. As shown in **Fig. 3**, the synthesized font images from our model have higher quality than those from the baseline methods on very thin font styles. The baseline methods tend to generate completely blank images on very thin style fonts but yield acceptable results on bold style fonts. The red boxes in **Fig. 3** highlight the few-shot stylized characters given to the model at inference time to map content.

For Korean characters, we trained the model on 512 base characters. At inference time, we generated 2,350 characters using few-shot stylized characters for each distinct font style. As shown in **Fig. 4**, our model synthesizes realistic images and is able to capture small strokes and curve information. One possible reason for this good performance is the geometric representation of content in our model. By using geometric representations and multiple font style images, small stroke information can be captured. Additionally, the losses introduced in our model make it possible to learn structural and feature-level statistics of font styles in an

Fig. 4. Visual comparison of Korean glyph images synthesized by the proposed and baseline methods. The content and style were unseen for the model. Our model outperforms the baseline methods and has the ability to capture small stroke information.

efficient manner. Our model can capture sufficient style and structure representations to learn high and low-level font details higher and lower-level font details.

5.4.2 Quantitative Analysis:

Although visual appearance intuitively reflects the quality of synthesized results, to obtain a high-level performance indication for the entire dataset, quantitative evaluation is necessary. For quantitative evaluation, we performed reference- and non-reference-based image assessments. In many images generation tasks, reference-based evaluations can be performed when ground-truth data are available. However, non-reference-based evaluations can be performed when ground-truth data are not available. This method only requires a generated image whose quality is being assessed.

5.4.2.1 Reference-based image quality evaluation

For reference-based evaluation, we adopt three commonly used metrics for image assessment tasks: SSIM [40], spatial alignment measure (MSE), and peak signal-to-noise ratio (PSNR) [41]. All experiments on Latin and Korean characters were performed using the same metrics. The SSIM measures the similarity between the ground truth and generated images from a structural perspective. Values near zero indicate no similarity and values close to one indicate high similarity. MSE measures the difference between images at the pixel level. PSNR measures image distortion and noise between images. A high PSNR value indicates high image quality. MSE and PSNR estimate absolute error, whereas SSIM measures similarity by considering perceived changes in structural information.

As shown in **Table 1**, we statistically compared the proposed method to the baseline methods. We performed quantitative experiments by computing the average values of all characters based on the total number of characters in each distinct font style. The results shown in **Table 1** indicate that our method outperforms the baseline methods. Additionally, as shown in **Table 2**, we computed the image distortion and noise for the first five characters in Style-1 and Style-2 of the fonts shown in **Fig. 3**. To evaluate the image quality of the synthesized glyphs, we computed the PSNR on a few glyphs instead of all glyphs.

Table 1. Quantitative comparison on English glyph image dataset.

Font Styles	Spatial Alignment Measure			Structure Similarity Measure		
	pix2pix	zi2zi	Proposed	pix2pix	zi2zi	Proposed
Style-1	0.2712	0.2589	0.2232	0.9099	0.8984	0.9465
Style-2	0.1033	0.1775	0.2214	0.6913	0.3613	0.9412
Style-3	0.1933	0.1161	0.2217	0.1723	0.2123	0.9693

Table 2. Image distortion measure on English glyph images from character A to E. The higher PSNR value shows the high image quality.

Font Styles-1 Characters	Signal-Noise Ratio Measure			Font Styles-2 Characters	Signal-Noise Ratio Measure		
	pix2pix	zi2zi	Proposed		pix2pix	zi2zi	Proposed
A	19.18	17.69	22.34	A	13.83	13.17	19.13
B	19.54	16.94	21.61	B	21.24	11.12	25.02
C	18.41	18.78	21.10	C	16.14	11.18	21.94
D	24.79	18.89	23.07	D	15.83	11.06	21.88
E	19.96	18.27	19.99	E	23.62	10.03	25.90

Table 3. Quantitative comparison on Korean glyph image dataset.

Font Styles	Spatial Alignment Measure			Structure Similarity Measure		
	pix2pix	zi2zi	Proposed	pix2pix	zi2zi	Proposed
Style-1	0.3291	0.2972	0.2306	0.9255	0.9331	0.9541
Style-2	0.2912	0.2967	0.2496	0.9031	0.9284	0.9531
Style-3	0.3222	0.2781	0.2612	0.9412	0.9351	0.9321

We utilized the same quantitative evaluation measures for the Korean glyph images. As shown in **Tables 3** and **4**, our model outperforms the baseline methods from a structural and quantitative perspective. To compute image distortion measures, we only performed testing

on the first three Korean characters in Style-1 in **Fig. 4**.

Table 4. Image distortion measure on Korean glyph images of first three characters of Style-1.

Font Styles-1 Characters	Signal-Noise Ratio Measure		
	pix2pix	zi2zi	Proposed
각	23.91	21.72	26.61
강	22.15	22.05	27.84
갛	20.17	0.2147	23.51

5.4.2.2 No-reference image quality evaluation

To perform no-reference-based image quality assessment, we measured the accuracy of synthesized glyphs. For this evaluation, ground-truth images were not required. To measure the classification accuracy of the generated glyphs, we trained a classifier on the Korean and Latin datasets. The goal was to determine the legibility characteristics of the synthesized glyphs. We also analyzed how well the model captured and mapped the target style onto the content. In **Tables 5** and **6**, we present the legibility accuracy measures for the proposed method and baseline methods for the Latin and Korean synthesized glyphs, respectively. One can clearly see that our model yields higher classification accuracy than the baseline methods, which fulfills the readability and consistency characteristics of the glyphs between styles and contents.

Table 5. Legibility measure on English glyph images

Font Styles	Legibility Measure		
	pix2pix	zi2zi	Proposed
Style-1	0.8214	0.7963	0.9742
Style-2	0.5321	0.1018	0.9427
Style-3	0.2104	0.1103	0.9433

Table 6. Legibility measure on Korean glyph images

Font Styles	Legibility Measure		
	pix2pix	zi2zi	Proposed
Style-1	0.9227	0.9402	0.9518
Style-2	0.9185	0.9482	0.9612
Style-3	0.9582	0.9271	0.9708

5.5 Ablation Study

To investigate the effectiveness of the proposed approach, we performed an ablation study from the architecture, loss, and performance perspectives.

5.5.1 Fixed-reference font instead of geometric representation as content

We wished to answer the following questions. Why do we need to select inner and outer structures for content references instead of using complete font images? Why do we need to use *k* stylized images instead of a single style reference image? To answer these questions, we

conducted an experiment in which we trained our model using a fixed reference font image as content instead of a geometric representation and a single style reference image instead of a set of target stylized images. The model was trained using the same experimental settings discussed above. The only differences are the inputs for E_{se} and E_{st}.

As shown in **Fig. 5**, we determined that the generated content follows the structure and style of the reference font instead of the target font. Additionally, we found that using a single font style image to capture style information is insufficient, particularly when there is large dissimilarity between the content and style. This makes it difficult for the model to capture the underlying details of the target style. This issue is most prominent for very thin stylized target fonts because performing a collection of convolutional operations on a single thin-style image leads to a loss of spatial information in deeper layers, such as exact small stroke information. Selecting inner and outer structures for input content does not restrict the model to following a given content font image (non-geometric). Additionally, the model has sufficient structural information regarding the content in the form of underlying structures. Therefore, a thin style can also be mapped by taking advantage of content skeleton information, which is not possible if we use a complete font image.

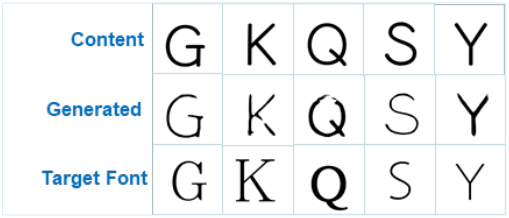

Fig. 5. Impact of conditioning on fixed font image

5.5.2 Impact of skip-layers between encoders

To verify the efficacy of skip layers, we conducted an experiment in which we removed the skip connections between E_{se} and E_{st}. We did not pass the content information to E_{st} and only the latent content information was added to the latent style. The connection between E_{st} and G_Y remained the same. As shown in **Fig. 6**, the model attempts to map the target style, but cannot capture content efficiently. Therefore, the model does not synthesize the correct content because it does not have sufficient content information. It only utilizes latent information, which is not sufficient for representing the content structure, particularly in the case of font synthesis tasks.

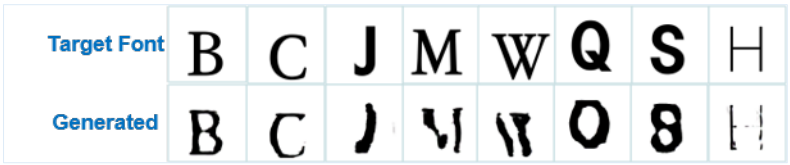

Fig. 6. Results without skip-layers between content and style extractor

5.5.3 Effects of loss functions

5.5.3.1 Effect of Structural Similarity Loss

We conducted an experiment by including and excluding structural similarity loss. We introduced this loss in the generator to learn structural and semantic information effectively

from the content and style perspectives. We trained the model without structural similarity loss and evaluated it using few-shot stylized characters. As shown in **Fig. 7**, the model can generate the target style, but it ignores small stroke information. Therefore, to learn small stroke information more effectively, $\mathcal{L}_{st\_sim}$ plays a vital role.

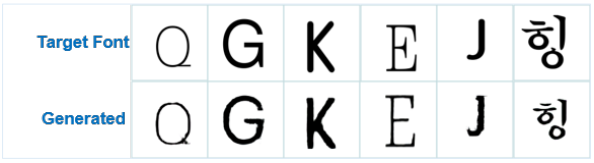

Fig. 7. Model results without structural similarity loss

5.5.3.2 Effect of Feature Matching Loss

We conducted an experiment by including and excluding feature matching loss. We introduced this loss in the generator to synthesize high-quality glyph images at the feature space level. Additionally, it is necessary to learn sufficient style representations from feature maps. As shown in **Fig. 8**, the model is able to learn the style and structure, but it has poor smoothness and overall image quality. Therefore, this loss plays a vital role in the synthesis of high-quality images. These effects are also visible in **Figs. 3** and **4**.

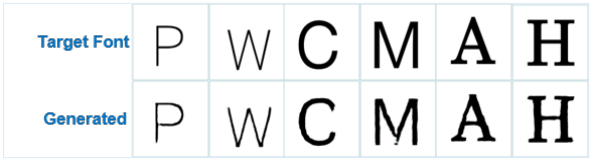

Fig. 8. Model results without feature matching loss

5.5.4 Cross-language assessment

To verify the generalization capabilities of the proposed method, we performed a cross-language assessment. We analyzed the model based on contents and styles that were not seen by the model at the time of training. We trained our model on Korean characters and *k* stylized images for each distinct font style. However, at the time of testing, we used unseen Chinese content characters and few-shot Chinese font style images as inputs for the model to verify its generalization capabilities. As shown in **Fig. 9**, the model yields impressive results because it has sufficient structural and style information. Therefore, it can efficiently capture contents and styles. The input forms of the content characters are the same as those for Korean alphabet characters in terms of their geometric representations.

5.5.5 Impact of distinct edge and skeleton encoders for content feature extraction

We wished to answer the following question. Why do we use a single encoder for extracting the inner and outer structures of content instead of using a separate encoder for each skeleton and edge? To answer this question, we constructed a model with three different encoders for the content skeletons, content edges, and style reference inputs. During training, we determined that the model size and computational cost increased when we downsampled each input image using deeper layers. At the time of testing, we found the model generated acceptable results overall, but it failed on fonts that had higher similarity to the content because of excessive information in the feature space regarding the target style and content.

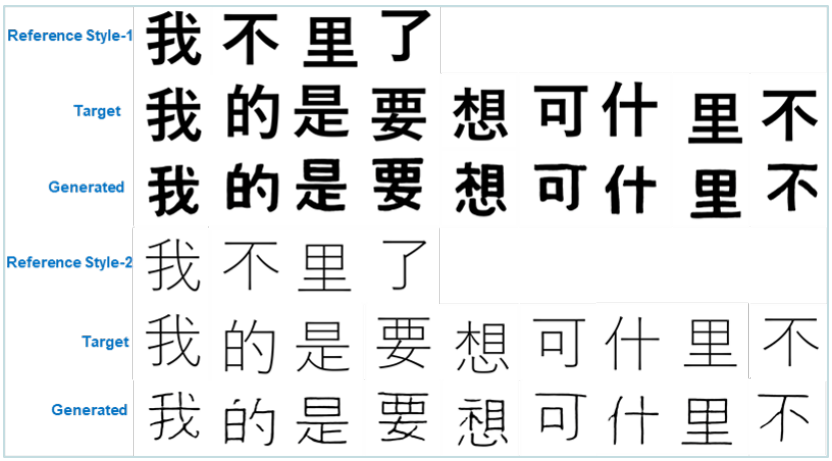

Fig. 9. Chinese characters synthesized by the proposed model. Note that the content and style were not seen by the model.

5.5.6 Failure cases

Can our model generate unseen content images during inference time? To get the answer to this question we conducted an experiment where we used our pre-trained model on English alphabets and during inference time, we gave it unseen character contents. By unseen character content we mean the characters which were not observed during training. For these contents we picked the Arabic language letters which are cursive and stylish and are very different from the English alphabets overall structure. We used 4 randomly selected style images representing the observed style of English alphabets and unseen content of Arabic letters as skeleton and edge inputs for our content encoder. We observed that our model somehow generated the contents by filling the shape of the contents but overall, the model failed to generate clean characters in both style and content wise scenario as shown in **Fig. 10**. The obvious possible reason to this failure is the unseen content of the letters which are very different to the seen content by the model.

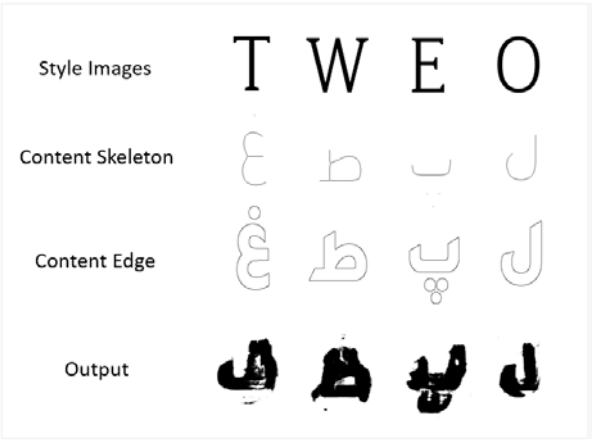

Fig. 10. Our proposed model fails at generating unseen content. In this case the model is unable to transfer seen style from English alphabets to unseen Arabic unseen content input.

6. Conclusion

In this paper, we proposed a geometric-content-guided cGAN that generates high-quality fonts. Our method is based on two separate encoders: one for content representation and another for style extraction from *k* input characters. By utilizing geometric information from the content and a few stylized images, our model can generate a complete font set with consistent styles and contents in terms of the target and source characters, respectively. Extensive experiments were conducted to demonstrate the superiority of the proposed method compared to state-of-the-art methods. An ablation study, as well as qualitative and quantitative experiments, demonstrated the effectiveness of the proposed method.

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-00218).

References

[1] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu and M. Song, "Neural Style Transfer: A Review," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 11, pp. 3365-3385, 1 Nov. 2020. Article (CrossRef Link).

[2] Y. Kataoka, T. Matsubara and K. Uehara, "Image generation using generative adversarial networks and attention mechanism," in *Proc. of 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, 2016. Article (CrossRef Link).

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. of Int. Conf. Learn. Representations*, 2015. Article (CrossRef Link).

[4] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Article (CrossRef Link).

[5] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proc. of ICCV*, 2017. Article (CrossRef Link).

[6] Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. of Advances in Neural Information Processing Systems (NIPS)*, pp. 2672–2680, 2014. Article (CrossRef Link).

[7] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *Proc. of International Conference on Learning Representations*, 2017. Article (CrossRef Link).

[8] K. J. Liang, C. Li, G. Wang, and L. Carin, "Generative adversarial network training is a continual learning problem," *arXiv preprint arXiv:1811.11083*, 2018. Article (CrossRef Link).

[9] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., "Photorealistic single image super-resolution using a generative adversarial network," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4681–4690, 2017. Article (CrossRef Link).

[10] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *Proc. of the IEEE International Conference on Computer Vision*, pp. 5907–5915, 2017. Article (CrossRef Link).

[11] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014. Article (CrossRef Link).

[12] N. D. Campbell and J. Kautz, "Learning a manifold of fonts," *ACM Transactions on Graphics (TOG)*, 33(4), 91:1-91:11, 2014. Article (CrossRef Link).

[13] KNUTH D. E., *TEX and METAFONT: New Directions in Typesetting*, Boston, MA, USA: American Mathematical Society, 1979. Article (CrossRef Link).
[14] XU S., LAU F., CHEUNG W. K., PAN Y, "Automatic generation of artistic Chinese calligraphy," *Intelligent Systems*, IEEE, 20(3), 32–39, 2005. Article (CrossRef Link).
[15] LAU, V. M. K, "Learning by example for parametric font design," in *Proc. of SIGGRAPH ASIA '09 Posters*, 5:1–5:1, 2009. Article (CrossRef Link).
[16] Z. Lian, B. Zhao, and J. Xiao, "Automatic generation of largescale handwriting fonts via style learning," in *Proc. of SIGGRAPH ASIA 2016 Technical Briefs*, ACM, pp. 1-4, 2016. Article (CrossRef Link).
[17] S. Yang, J. Liu, Z. Lian, and Z. Guo, "Awesome typography: Statistics-based text effects transfer," in *Proc. of IEEE Conf. Computer. Vis. Pattern Recognition*, pp. 2886–2895, 2016. Article (CrossRef Link).
[18] S. Uchida, Y. Egashira, K. Sato, "Exploring the world of fonts for discovering the most standard fonts and the missing fonts," in *Proc. of the 13th International Conference on Document Analysis and Recognition, ICDAR*, pp. 441–445, 2015. Article (CrossRef Link).
[19] H. Q. Phan H. Fu A. B. Chan, "Flexyfont: Learning transferring rules for flexible typeface synthesis," *Computer Graph Forum*, vol. 34(7), pp. 245-256, 2015. Article (CrossRef Link).
[20] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin, "Image analogies," in *Proc. of SIGGRAPH*, ACM, pp. 327–340, 2001. Article (CrossRef Link).
[21] P. Upchurch, N. Snavely, and K. Bala, "From A to z: Supervised transfer of style and content using deep neural network generators," *arXiv preprint arXiv:1603.02003*, 2016. Article (CrossRef Link).
[22] Y. Tian: zi2zi: "Master Chinese Calligraphy with Conditional Adversarial Networks," 2017. [Online]. Available: https://github.com/kaonashi-tyc/zi2zi
[23] A. Odena, C. Olah, J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. of the 34th International Conference on Machine Learning, ICML*, pp. 2642–2651, 2017. Article (CrossRef Link).
[24] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," *arXiv preprint arXiv:1611.02200*, 2016. Article (CrossRef Link).
[25] T. Miyazaki, T. Tsuchiya, Y. Sugaya, S. Omachi, M. Iwamura, S. Uchida, K. Kise, "Automatic generation of typographic font from a small font subset," *IEEE Computer Graphics and Applications*, vol. 40, no. 1, pp. 99-111, 2020. Article (CrossRef Link).
[26] Y. Jiang, Z. Lian, Y. Tang, and J. Xiao, "Dcfont: an end-to-end deep Chinese font generation system," in *Proc. of SIGGRAPH Asia 2017 Technical Briefs*, pp. 1–4. 2017. Article (CrossRef Link).
[27] Y. Jiang, Z. Lian, Y. Tang, and J. Xiao, "SCFont: Structure-guided Chinese font generation via deep stacked networks," in *Proc. of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 33(01), pp. 4015-4022, 2019. Article (CrossRef Link).
[28] S. Azadi, M. Fisher, V. Kim, Z. Wang, E. Shechtman, and T. Darrell, "Multi-content GAN for few-shot font style transfer," in *Proc. of the IEEE 555 Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7564–7573, 2018. Article (CrossRef Link).
[29] Gao Yue, Guo Yuan, Lian Zhouhui, Tang Yingmin, and Xiao Jianguo, "Artistic Glyph Image Synthesis via One-Stage Few-Shot Learning," *ACM Trans. Graph*, 38(6), 1-12, 2019, Article 185. Article (CrossRef Link).
[30] H. Hayashi, K. Abe, and S. Uchida, "GlyphGAN: Style-Consistent Font Generation Based on Generative Adversarial Networks," *Knowledge-Based Systems*, 186, 2019. Article (CrossRef Link).
[31] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2016. Article (CrossRef Link).
[32] X. Liu, G. Meng, S. Xiang, and C. Pan, "FontGAN: A Unified Generative Framework for Chinese Character Stylization and De-stylization," *arXiv preprint arXiv:1910.12604*, 2019. Article (CrossRef Link).

[33] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs", in *Proc. of the 34th International Conference on Machine Learning*, pp. 2642–2651, 2017. Article (CrossRef Link).
[34] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *arXiv preprint*, 2017. Article (CrossRef Link).
[35] A. L. Maas, A. Y. Hannun, and A. Y. Ng., "Rectifier nonlinearities improve neural network acoustic models," in *Proc. of ICML*, 2013. Article (CrossRef Link).
[36] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, "Are GANs created equal? a large-scale study," in *Proc. of NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*, December 2018. Article (CrossRef Link).
[37] D. H. Ko, A.U. Hassan, S. Majeed, and J. Choi, "Font2Fonts: A modified Image-to-Image translation framework for font generation," in *Proc. of SMA 2020*, September 17-19, 2020. Article (CrossRef Link).
[38] Canny, J., "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679-698, 1986. Article (CrossRef Link).
[39] T.-C. Lee, R.L. Kashyap and C.-N. Chu, "Building skeleton models via 3-D medial surface axis thinning algorithms," *CVGIP: Graphical Models and Image Processing*, 56(6), 462-478, 1994. Article (CrossRef Link).
[40] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, 13, 600-612, 2004. Article (CrossRef Link).
[41] I. Avcibas, B. Sankur and K. Sayood, "Statistical evaluation of image quality measures," *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 206-223, 2002. Article (CrossRef Link).

Saima Majeed https://orcid.org/0000-0003-0334-694X (ORCID ID)

Saima Majeed received her M.S. degree in Department of Computer Science and Engineering from Soongsil University, Seoul, South Korea in 2020. She is currently working as a Lecturer in Public Sector Air University, Islamabad, Pakistan. Her area of teaching is Artificial Intelligence, Machine learning, deep learning, and Programming related courses of undergraduate and graduate level students.

Ammar Ul Hassan https://orcid.org/0000-0001-6744-507X (ORCID ID)

Ammar Ul Hassan received B.S. degree in Department of Software engineering, from International Islamic University Islamabad, Pakistan in 2013. He then received his M.S. degree in Computer Science from Soongsil University, Seoul, South Korea in 2018. He is currently taking his Ph.D. degree in Department of computer science from Soongsil University Seoul, South Korea. He is working as a research associate in System Software laboratory under the supervision of Professor Jaeyoung Choi. His current research interests are Deep learning, Computer vision, Generative models, making font environment for new fonts in Linux Operating System.

Jaeyoung Choi http://orcid.org/0000-0002-4510-0385 (ORCID ID)

Jaeyoung Choi received the B.S. degree in Department of Control and Instrumentational Engineering, from Seoul National University, Seoul, Korea, in 1984, the M.S. degree in Department of Electrical Engineering, University of Southern California in 1986, and the Ph.D. degree in School of Electrical Engineering from Cornell University in 1991. He has previously worked at Oak Ridge National Laboratory (1992-1994) and University of Tennessee, Knoxville (1994-1995) as a postdoctoral research associate and a research assistant professor, respectively, where he had been involved with the ScaLAPACK project. He is currently a professor of School of Computer Science and Engineering at Soongsil University, Seoul, Korea. His current research interests include high performance computing and typography.