Journal of King Saud University - Computer and Information Sciences 36 (2024) 102028

HOSTED BY

Contents lists available at ScienceDirect

Journal of King Saud University - Computer and Information Sciences

journal homepage: www.sciencedirect.com

Full length article

One-shot font generation via local style self-supervision using Region-Aware Contrastive Loss

Jeong-Sik Lee [a], Hyun-Chul Choi [b,*]

[a] AI Lab, Kakao Entertainment, 221 Pangyoyeok-ro, Bundang-gu, Seongnam-si, Gyeonggi-do, 13493, Republic of Korea
[b] Department of Electronic Engineering, Yeungnam University, 280 Daehakro, Gyeongsan, Gyeongbuk, 38541, Republic of Korea

ARTICLE INFO

Keywords:
One-shot font generation
Region-aware contrastive loss
Local style
Style patch

ABSTRACT

Compositional scripts like Hangeul (Korean characters) and Chinese characters involve numerous characters, making manual font design labor-intensive and cost-ineffective work. Although many few-shot font generation methods have been introduced, they have at least one of the limitations, *i.e.*, lacking local styles of font, additional component labeling, and high complexity in network structure and training. To solve these limitations, given our observation that font style can be perceived at a patch-level rather than a component-level, we propose Region-Aware Contrastive loss (RAC-loss) so that the generator can capture the local style by self-supervision. The proposed loss maximizes the style information between patches of the generated image and the style reference image. And we introduce an attention mechanism to the patch-level contrastive loss to handle multiple patch correspondences. This attention learns style similarity between two glyph images, which serves as a patch-correspondence map. RAC-loss gives more fine-grained feedback to the generator than component-level loss, allowing it to incorporate local styles, even in a straightforward structure like a visual geometry group network (VGGNet). This results in a fast inference latency (3.02 ms), and the proposed method achieved 43.18 mean Fréchet Inception Distance (mFID) on the test dataset, a notable decrease of 5.42 compared to the previous method.

1. Introduction

Font design for languages with a considerable amount of characters, such as Hangeul (Korean characters) and Chinese characters, requires a huge amount of time and human resources. For example, combinational characters like Hangeul and Chinese character consist of 11,172 and 106,230 characters, respectively, making manual font design impractical. To address this issue, various automatic font generation methods have been studied. Early methods (Tian, 2024; Jiang et al., 2019; Gao and Wu, 2020) required retraining or fine-tuning their models to generate an arbitrary font that was not seen in a training phase.

In contrast, to avoid time-consuming retraining or fine-tuning, most of the recent Few-shot Font Generations (FFGs) (Zhang et al., 2018b; Gao et al., 2019; Li et al., 2020; Cha et al., 2020; Park et al., 2021; Xie et al., 2021; Liu et al., 2022; Tang et al., 2022; Kong et al., 2022) employ two encoders to extract text content and font style features, respectively. They merge these two features into a styled feature through transform layers and generate the glyph image by decoding this styled feature. Early FFGs (Zhang et al., 2018b; Gao et al., 2019; Li et al., 2020) struggled with generating an arbitrary font from a single style reference image, so they used multiple style reference images to extract common features. However, these methods often omit the local style (endpoint style, serif-ness, etc.) because they rely on the common feature of multiple style references, which is a global style representation that lacks local styles. This is particularly challenging for Chinese character, which has a complex structure and is highly sensitive to local detail. As a result, global style representation approaches produced unsatisfactory results in Chinese font generation.

To address the local style, recent studies (Cha et al., 2020; Park et al., 2021; Liu et al., 2022; Kong et al., 2022; Tang et al., 2022) have

* Corresponding author.
E-mail addresses: a2819z@ynu.ac.kr (J.-S. Lee), pogary@ynu.ac.kr (H.-C. Choi).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2024.102028
Received 21 December 2023; Received in revised form 2 April 2024; Accepted 2 April 2024
Available online 16 April 2024

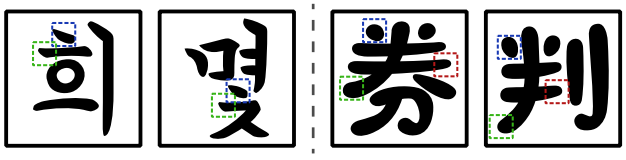

Fig. 1. Local style correspondence: Boxes of the same color in two images represent the patches in a similar style.

introduced the concept of glyph component (or stroke) into font generation. These component-based methods change the component style of the content glyph by leveraging the component of the style reference. However, dealing with components is complex as a component does not configure a single glyph, and its position, size, and rotation vary based on the glyph. Thus, disentangling the components is necessary, and it requires additional component labels, increasing the network complexity and time for both training and inference.

In this paper, we propose a Region-Aware Contrastive Font (RAC-Font), which releases the requirements of an additional component label and simplifies the structure of the generator. The proposed method is inspired by two characteristics of the font: (1) Font styles can be thought of per patch as shown in Fig. 1 instead of per component. (2) Patches with similar local styles in two glyph images are not matched one-to-one but rather one-to-many or many-to-many. We seamlessly incorporate these properties into the proposed Region-Aware Contrastive loss (RAC-loss), self-supervising the generator by maximizing the style similarity between patches of the generated image and style reference images. Matching the style between the patches is challenging without any supervision of locality, *e.g.*, components of a glyph image and the location of the component. Specifically, as shown in Fig. 1, patches with similar styles appear in different locations, making it impractical to simply match the local style between patches at identical locations.

Prior to our method, some studies in other fields (Park et al., 2020; Wang et al., 2021) tried to improve the performance of neural networks through the patch concept. CUT (Park et al., 2020) introduced the patch concept into image-to-image translation, but only patches at the same location were considered corresponding patches. Even though DenseCL (Wang et al., 2021) matched the patches across the different locations, it assumed one-to-one matching, which is inconsistent with the fact that a local style of font can appear in multiple patches. To address this issue, we introduced the attention mechanism into patch-level contrastive loss. This enhancement allows the neural networks to sample various positive patches by leveraging spatial attention derived from the style reference and generated images.

Specifically, RAC-loss employs an attention mechanism to get patch-relations with similar styles in two glyph images. This attention works like a soft mask to identify the corresponding patches. With this attention mask, we obtain a new patch by weighted summation of several patches with a similar style. This approach handles varying numbers of corresponding patches across glyph image combinations, whereas the previous hard thresholding method just selects patches based on a threshold or Top-K of attention values. Contrastive learning with the new patch encourages the corresponding patches to pull toward each other in the feature space. RAC-loss provides more refined self-supervision for capturing the local style within the generator. This encourages the generator to produce fonts that accurately reflect the local style, even without the complex structure that is necessary in the previous methods (Cha et al., 2020; Park et al., 2021; Liu et al., 2022; Tang et al., 2022).

The contribution of this paper is summarized as follows:

- We propose Region-Aware Contrastive loss (RAC-loss), which maximizes mutual information across patches at various locations, and this loss provides powerful self-supervision of local style to the generator.
- We propose a one-shot font generator with better performance than the few-shot font generator through RAC-loss and experimentally show the lower mFID values reduced by 3.09 and 5.42 in seen and unseen fonts, respectively, compared to the previous method.
- By simplifying the structure of the generator with the proposed RAC-loss, we achieved real-time one-shot font generation performance with four times faster inference time (3.02 ms) than the existing method that considers local styles (12.10 ms).
- Without any additional label, the proposed method considers patch-level font style, which is a more fundamental level than those of the previous FFGs, so our method can generate a font with a finer style.

In the remaining sections, we summarize the previous works related to font generation. Then, we explain our proposed method. Next, we compare our method with others and analyze the proposed method through experiments using two language datasets, *e.g.*, Hangeul and Chinese characters, and finally conclude this study.

2. Related work

Few-shot Font Generations (FFGs) (Zhang et al., 2018b; Gao et al., 2019; Li et al., 2020; Cha et al., 2020; Park et al., 2021; Xie et al., 2021; Liu et al., 2022; Tang et al., 2022; Kong et al., 2022) aim to generate arbitrary fonts given a few style reference images. Most methods (Zhang et al., 2018b; Gao et al., 2019; Li et al., 2020; Park et al., 2021; Xie et al., 2021) adopted the two encoders to disentangle the content and style features. Early FFGs (Zhang et al., 2018b; Gao et al., 2019; Li et al., 2020) extracted a global representation of font style and combined the global representation and content feature to generate a font. EMD (Zhang et al., 2018b) extracted both content and style features from multiple images, allowing for the disentanglement of the text content and font style. This disentanglement also made it possible to generate arbitrary fonts for the first time. However, since EMD utilized the global style feature, it failed to generate a font in a fine-grained local style. Although AGIS-Net (Gao et al., 2019) proposed a local discriminator and a local refinement loss to generate fine-grained font, they still lack local style considerations. The main problem with this kind of method is that its performance decreases rapidly as the number of given reference images decreases. In addition, these methods cannot capture the complex local style because both the generator and the discriminator only rely on global style features. To overcome the lack of local style, DG-Font (Xie et al., 2021) proposed the deformable skip connection. This allows DG-Font to consider the relationship between content and style reference images, but there are still some cases where font style transfer fails.

Since the above methods rely on global style in either the generator, discriminator, or both, it is difficult to create a handwriting font that reflects a distinct style depending on the component combination. To address this problem, the idea of extracting the style features of each component of a character (Cha et al., 2020; Park et al., 2021) has been explored. DM-Font (Cha et al., 2020) extracted each component style feature and saved that to dynamic memory, then utilized this memory for component-wise font generation. However, for languages with a huge number of components (Chinese characters), it requires a lot of labels and style reference images to cover all the components. LF-Font (Park et al., 2021) introduced a factorization module to reduce the number of reference images and labels, but additional component labels are still necessary. In addition, the performance decreased as the size of the reference was reduced.

Recent methods (Liu et al., 2022; Tang et al., 2022; Kong et al., 2022) have been studied to reduce the reliance on the vast number of component labels in existing component-based methods. XMP-Font (Liu et al., 2022) focused on strokes rather than components, so they reduced the number of additional labels to 28, and it used a cross-modal encoder, where each modal represents a stroke and a global,

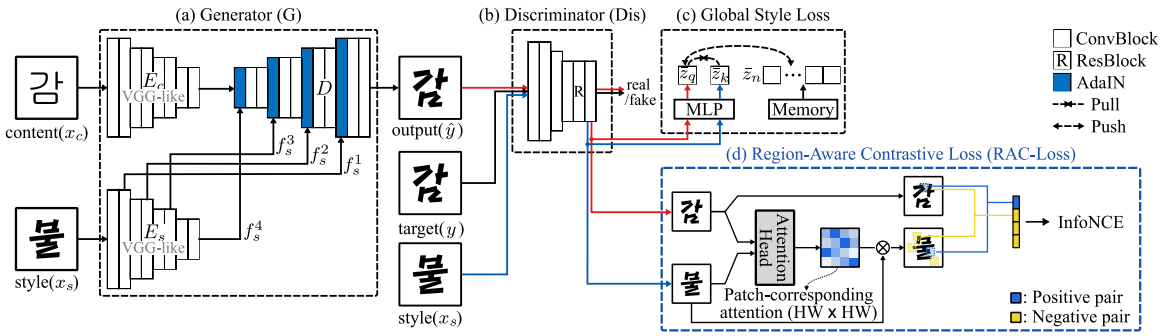

Fig. 2. Overview of proposed network structure: It consists of a generator (a), discriminator (b), global style loss (c), and Region-Aware Contrastive Loss (d). With Region-Aware Contrastive Loss ($\mathcal{L}_{\text{RAC}}$), the generator and discriminator can learn the style correlation between the patches of the two images—the generated image ($\hat{y}$) and the style reference image (x_s). With patch-corresponding attention, we can find corresponding patches to the query patch and obtain the aligned positive key patch by a weighted-sum of key patches according to their correlation with the query patch.

respectively, to consider both global and local styles. Tang et al. (2022) proposed a style aggregation module (SAM) that finds the corresponding style regions to content and aggregates them to generate an output font. However, to train the SAM module, a content-reference mapping, a mapping of multiple style references covering all components of each content, is required, which has the limitation of requiring a large number of reference images to cover all components, *e.g.*, 100 images to cover all Chinese components.

Unlike the previous studies that focused on the generator to improve performance, CG-GAN (Kong et al., 2022) focused on a discriminator. They proposed Component-Aware Module (CAM), which does component-wise classifying (or discrimination). Although its generator is simpler than the previous methods, the discriminator with CAM provides powerful feedback on local style to the generator, and it successfully generates the fine-grained font. Although our method also focuses on discriminators, as CG-GAN did, there are two notable differences in our method. First, our method accounts for local style without any additional labels, while CG-GAN requires weak supervision via component labels to account for local style. Second, our method can generate fonts with a more detailed style since we consider the local style at the patch-level, which is a finer level than the component-level.

Many methods (Cha et al., 2020; Zhang et al., 2019; Liu et al., 2022; Tang et al., 2022; Kong et al., 2022) have used various locality concepts such as component, stroke, and content-reference mapping, but they have the limitation of requiring additional labels for parts of font images.

3. Background theory

Contrastive learning is an unsupervised learning method that pulls positive samples from the query while pushing negative ones, resulting in a surprisingly positive effect on representation learning. SimCLR (Chen et al., 2020) analyzed the effects of various settings such as the augmentation method and batch size, on contrastive learning and established a basic framework for learning, but it required a lot of computational resources. MoCo (He et al., 2020) proposed a memory bank implemented as a queue to store negative samples, and it reduced the computational cost a lot. However, these methods (Chen et al., 2020; He et al., 2020) are less effective in improving the performance in dense prediction down-tasks, such as object detection and semantic segmentation, than in instance classification, because they learn the instance-level discriminative task with a spatially pooled feature that loses spatial information. Unlike the previous methods, DenseCL (Wang et al., 2021) performed contrastive learning using features that maintain a spatial dimension. This method maximized the mutual information of patch pairs that have top-1 cosine similarity between the two augmented images and improved the performance of dense prediction downstream-tasks. However, as mentioned in Section 1, for two font images, a patch in one image can match multiple patches in the other image; the top-1 matching of DenseCL is not appropriate for the FFG area.

Several works (Park et al., 2020; Kang and Park, 2020) extended this to generative models, following this progress in contrastive learning. CUT (Park et al., 2020) learned the image-to-image translation network by performing contrastive learning at the patch-level and reduced the learning complexity by removing the cyclic loss (Zhu et al., 2017). However, it is not possible to change the shape but only the texture, as in CycleGAN (Zhu et al., 2017), because only the same position can be a positive sample. Kang and Park (Kang and Park, 2020) defined the data-to-class and data-to-data relations in conditional GANs (Mirza and Osindero, 2014) and utilized contrastive loss for the data-to-data relation. They showed that utilizing a data-to-data relation performed better than the previous methods that used the data-to-class relation (Odena et al., 2017; Miyato and Koyama, 2018). However, Kang and Park considered data-to-data relationships only in instance-level but in patch-level, where local style in patch is also important for font generation.

4. Method

4.1. Overview

Contrastive learning has been studied in two branches: Instance-level (Chen et al., 2020; He et al., 2020) and patch-level (Wang et al., 2021). Unlike instance-level contrastive learning, which aims to capture high-level representations using a global pooled feature, patch-level contrastive learning maintains the spatial dimension of feature $z \in \mathbb{R}^{HW \times C}$, where HW and C denote spatial dimension and channel dimension, respectively. This approach aims to ensure that each corresponding pixel between the two spatial features exhibits a similar value, preserving local information and fine-grained details. Therefore, we adopt patch-level contrastive learning instead of instance-level learning, as our goal is to improve the quality of local styles in the font generation task. The previous patch-level contrastive methods (Park et al., 2020; Wang et al., 2021) either sampled only patches at the same location as a positive patch (Park et al., 2020) or treated other significant patches as negative patches (Wang et al., 2021). Conversely, in the Few-shot Font Generation (FFG) task, the style information, *e.g.*, style of endpoint, stroke thickness, etc., does not appear in the same region of the generated image and the style reference image. Additionally, similar style information may appear in several places. From these two perspectives, because the existing method of patch-level contrastive learning is not appropriate for FFG, we propose a new positive sampling method necessary for the FFG task.

As shown in Fig. 2, the proposed method consists of the generator (a), the discriminator (b), global style loss (c), and RAC-loss (d). The generator (Fig. 2a) includes a content encoder (E_c), style encoder (E_s), adaptive instance normalization (AdaIN) (Huang and Belongie, 2017),

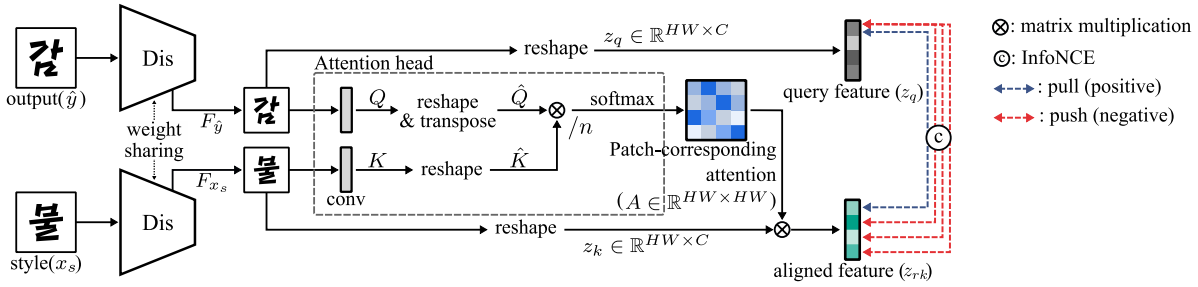

Fig. 3. The description of Region-Aware Contrastive Loss (RAC-loss). To find corresponding key patches to the query patch, the soft attention mechanism is adopted to match multiple key patches. Then, an aligned key patch is obtained by a weighted average of the matched key patches based on soft attention and is used for contrastive learning.

and decoder (D). Both encoders adopt the visual geometry group network (VGGNet) structure (Simonyan and Zisserman, 2015) (up to conv4_1), and the decoder has a mirror structure of one of the encoders. The encoders (E_c, E_s) extract features from one content image (x_c) and one style reference image (x_s), respectively. AdaIN (1) is used to blend the two features in the decoder for generating the glyph with the text of the content image and the style of the style image.

$$\mathrm{AdaIN}(f^i, f^i_s) = \sigma(f^i_s)\left(\frac{f^i - \mu(f^i)}{\sigma(f^i)}\right) + \mu(f^i_s), \tag{1}$$

where $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and standard deviation of the feature, respectively. AdaIN is commonly used in recent generative models (Karras et al., 2019; Park et al., 2019; Liu et al., 2019) for efficient global appearance changes, and we adopt it as well. We employ a multi-level style transfer structure that converts the decoded features ($\{f^i | i \in \{1,2,3,4\}\}$) into style features ($\{f^i_s | i \in \{1,2,3,4\}\}$) in the middle layers of the decoder for high-quality font generation. For multi-level style transfer, we extract the style features ($\{f^i_s | i \in \{1,2,3,4\}\}$) of each resolution in the style encoder. The discriminator (Dis) helps the generator produce a more realistic image by determining real and fake images using the generated image ($\hat{y}$) and target image (y). To handle the global style, we employ instance-level contrastive loss (He et al., 2020) (Fig. 2c) instead of using a conditional discriminator (Cha et al., 2020; Park et al., 2021; Tang et al., 2022; Kong et al., 2022). Further details will be discussed in Section 4.4.

Unlike the previous methods (Cha et al., 2020; Park et al., 2021; Liu et al., 2022), our generator relies on the global feature of the glyph image, excluding the use of the local feature (component or stroke), which may result in poor font generation. To introduce the local style with a simple structure of the generator or the discriminator, we propose Region-Aware Contrastive loss (RAC-loss) (Fig. 2d). By maximizing the mutual information between a patch of the generated image ($\hat{y}$) and a patch of the style reference image (x_s), this loss helps the discriminator capture the local style. The discriminator trained with RAC-loss gives the generator more accurate feedback about local style, enabling the generation of fonts with improved detail and quality.

4.2. Region-aware contrastive loss

We introduce spatial attention to patch-level contrastive learning (Fig. 2d), which enables the sampling of multiple positive samples between patches at different locations. The spatial attention is calculated from the features of a generated image (query feature) and a style reference image (key feature), yielding a newly aligned key feature correlated with a query feature. Patch-level contrastive learning is performed using the aligned key feature and the query feature. The details are as follows:

Patch-corresponding attention. Inspired by attention mechanisms that can learn semantically relevant between regions or patches in images (Dosovitskiy et al., 2021), we employ attention mechanisms to identify corresponding patches between the generated font images and style reference images. As shown in Fig. 3, the attention computation between the generated image ($\hat{y}$) and the style reference image (x_s) involves extracting features ($F_{\hat{y}}$, $F_{x_s} \in \mathbb{R}^{C\times H\times W}$) from the discriminator (Dis). These features ($F_{\hat{y}}$, F_{x_s}) are then projected to other embedding spaces ($Q, K \in \mathbb{R}^{C\times H\times W}$) as following (2), where (H, W) denotes the resolution of the feature and C is the number of channels.

$$\begin{aligned} Q &= \mathrm{conv}(F_{\hat{y}}), \quad Q \in \mathbb{R}^{C\times H\times W}, \\ K &= \mathrm{conv}(F_{x_s}), \quad K \in \mathbb{R}^{C\times H\times W}, \end{aligned} \tag{2}$$

where conv denotes 1×1 convolution. The two mapped features ($Q, K \in \mathbb{R}^{C\times H\times W}$) are reshaped into two sequences ($\hat{Q} \in \mathbb{R}^{HW\times C}, \hat{K} \in \mathbb{R}^{C\times HW}$). Matrix multiplication is performed on these sequences to obtain a tensor of the shape $HW\times HW$, followed by normalization with factor n. Depending on n, the operation of RAC-loss changes, which will be discussed in Section 4.3 for more detail. After normalization, softmax is applied to the normalized tensor to obtain patch-corresponding attention ($A \in \mathbb{R}^{HW\times HW}$), as expressed in (3):

$$A = \mathrm{s}\left(\cos(\hat{Q}, \hat{K})/n\right), \quad A \in \mathbb{R}^{HW\times HW}, \tag{3}$$

where $\mathrm{s}(x) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}$, $\cos(a,b) = \frac{a^T b}{\|a\|\cdot\|b\|}$, and $\|\cdot\|$ is L2-norm. The value of attention $A(i,j)$ represents the correlation between the ith patch of the generated image ($\hat{y}$) and the jth patch of the style reference image (x_s), utilized for positive or negative patch sampling.

Attention-aligned feature. The features ($F_{\hat{y}}, F_{x_s} \in \mathbb{R}^{C\times H\times W}$) are reshaped into sequences ($z_q, z_k \in \mathbb{R}^{HW\times C}$) and used as query and key features, respectively. Top-1 (or Top-K) sampling, as in DenseCL (Wang et al., 2021), is not suitable for soft attention because the number of patches in similar styles varies depending on the generated image and the style reference image. Instead, positive and negative samples are obtained using the attention-aligned feature ($z_{rk} \in \mathbb{R}^{HW\times C}$), which is a key feature corresponding to the query feature. The aligned feature is acquired by the matrix multiplication of the attention map (A) and key feature (z_k) as in (4).

$$z_{rk} = A\hat{z}_k, \quad z_{rk} \in \mathbb{R}^{HW\times C}. \tag{4}$$

This feature is a weighted-sum of the key feature values ($z_k(i), i \in \{1,2,\ldots,HW\}$) using attention ($A$) as a weight. Since the attention denotes the style correspondence between the query and key features, the ith aligned key feature ($z_{rk}(i) \in \mathbb{R}^C$) denotes the weighted-sum of key patches ($z_k(j), j \in \{1,2,\ldots,HW\}$) related to the ith query patch ($z_q(i)$).

Region-Aware Contrastive Loss. Since the ith aligned key feature value ($z_{rk}(i) \in \mathbb{R}^C$) represents the aggregation of the key feature values corresponding to the ith query feature patch ($z_q(i) \in \mathbb{R}^C$), it is used as a positive sample for the ith query feature patch. For the negative sample, the other attention-aligned feature values ($z_{rk}(j), \forall j \neq i$) are utilized. Since InfoNCE loss (Oord et al., 2018) is commonly employed to maximize the mutual information, we propose a new InfoNCE-like loss, Region-Aware Contrastive Loss (RAC-loss, $\mathcal{L}_{RAC}$), which is defined as (5).

$$\mathcal{L}_{RAC} = -\frac{1}{HW}\sum_{i=1}^{HW}\log\left[\frac{\exp(\cos(z_q(i), z_{rk}(i))/\tau)}{\sum_{j=1}^{HW}\exp(\cos(z_q(i), z_{rk}(j))/\tau)}\right], \tag{5}$$

where $\exp(x)$ denotes e^x. Leveraging the spatial attention (A) learned through self-supervision, this loss pulls multiple key patches corresponding to the query patch and pushes others. This allows the generator to produce a glyph image with an improved local style through self-supervision, eliminating the need for local labels such as component and stroke labels. In other words, the proposed method captures local styles without additional labels by maximizing the similarity between patches sharing similar styles rather than relying on component-wise classification (Park et al., 2021; Kong et al., 2022). Additionally, patches from other style images in the batch are not assigned as negative samples. Only negative patches in pairs of images are employed, following the approach in CUT (Park et al., 2020).

4.3. Temperature sampling

RAC-loss behaves in different ways depending on the normalizing factor (n). When n takes a very small value, the patch-corresponding attention (A) becomes a one-hot vector (hard attention), and each value of the aligned feature (z_{rk}) corresponds to one of the key features strongly related to the patch of each query feature. This is identical to the top-1 sampling of DenseCL (Wang et al., 2021), where only the patches with the top-1 similarity are sampled as positive samples.

For an appropriate value of n, the attention shifts toward soft attention, and the aligned feature (z_{rk}) is constructed with the weighted summation of the patches from multiple key features related to the ith query patch ($z_q(i)$). This approach aggregates the information of meaningful patches based on the learned patch-corresponding attention, resulting in a positive patch that is distinct from the top-k sampling method. Furthermore, positive patches are sampled flexibly without explicitly setting the parameter (K) for the number of positive patches because the aligned key feature is obtained using the patch-corresponding attention between query and key.

In this manner, our approach can be used more generally via sampling top-1 or top-K patches as positive patches, depending on the normalizing factor n. A detailed discussion on the comparison of sampling methods and the analysis of the normalizing factor (n) will be presented in Sections 5.4, 5.5, and 5.6, respectively. A value of 0.1 is used for both temperature value (τ in (5)) and normalizing factor (n in (3)) through grid search, and the result is presented in the experiment section.

4.4. Global style loss

Thanks to RAC-loss, the generator and discriminator can catch the local (or patch) style, but it still lacks supervision of the global style. For global style, the previous methods adopted the conditional discriminator (Cha et al., 2020; Park et al., 2021; Tang et al., 2022) or style classifier (Kong et al., 2022). However, these have over-fitting issues and rely solely on the relation between image embedding and class embedding (data-to-class relation), leading to suboptimal outcomes (Kang and Park, 2020). Similar to Kang and Park (Kang and Park, 2020), we leverage relations between multiple image embeddings (data-to-data relation) to capture the global style information of the font. For this purpose, the instance-level contrastive loss as shown in Fig. 2c is used following the settings of MoCo (He et al., 2020).

We set the generated image ($\hat{y}$) as a query and the style reference image (x_s) as a positive key. For the instance-level contrastive loss, the spatial dimension of both query and key features (z_q, z_k) is removed by average pooling. Then, we get the projected features ($\bar{z}_q, \bar{z}_k \in \mathbb{R}^C$) by MLP and use them as a positive pair. For negative samples, we use the samples ($\bar{z}_n \in \mathbb{R}^C$) stored in the memory queue. While some of the negative samples in the memory queue may potentially be positive pairs with $\bar{z}_q$, once the scales of both the memory queue and the dataset are adequately large, these stored positive samples can be treated as small noise. The memory queue, which stores the negative samples, has a size of 8192 and enqueues the key feature ($\bar{z}_k$) of the current mini-batch while dequeuing the oldest. We adopt InfoNCE (Oord et al., 2018) (6) with positive samples ($\bar{z}_q, \bar{z}_k$) and negative samples ($\bar{z}_n^i$) for the global style loss.

$$\mathcal{L}_C = -\log \frac{\exp(\cos(\bar{z}_q, \bar{z}_k)/\tau)}{\sum_{j=1}^{K} \exp(\cos(\bar{z}_q, \bar{z}_n^i)/\tau)}, \tag{6}$$

where K denotes the size of the memory queue. For the temperature value (τ in (6)), a commonly used value of 0.07 (Chen et al., 2020; He et al., 2020) is employed.

4.5. Training

In addition to the two contrastive losses, one for local style ($\mathcal{L}_{RAC}$) and the other for global style ($\mathcal{L}_C$), additional three losses are employed for training the font generation network. Firstly, to generate a realistic font image, hinge losses (Zhang et al., 2019) ((7) and (8)) are employed as the adversarial losses.

$$\begin{aligned}\mathcal{L}_{adv}^{G} = &- \mathbb{E}_{y\sim p_{data}} \max(0, -1 + Dis(y)) \\ &- \mathbb{E}_{\hat{y}\sim p_{gen}} \max(0, -1 - Dis(\hat{y})),\end{aligned} \tag{7}$$

$$\mathcal{L}_{adv}^{Dis} = - \mathbb{E}_{\hat{y}\sim p_{gen}} Dis(\hat{y}). \tag{8}$$

For the remaining losses aimed at generating the same font as the target font, pixel-level loss (9) normalized by the number of black pixels and feature-level loss (10) are utilized.

$$\mathcal{L}_{pixel} = \frac{1}{N_b(y)} ||y - \hat{y}||_1, \tag{9}$$

$$\mathcal{L}_{feat} = \sum_{i\in L} \left|\left|VGG_i(y) - VGG_i(\hat{y})\right|\right|_2, \tag{10}$$

where $N_b(\cdot)$ denotes the number of black pixels, which is the number of values less than 0.5 in the image normalized between 0 and 1. The feature matching loss, $\mathcal{L}_{feat}$, uses a pre-trained VGG-19 (Simonyan and Zisserman, 2015) network, and $VGG_i(\cdot)$ means an intermediate feature of VGG, *i.e.*, ReLU1_2, ReLU2_1, ReLU3_1, and ReLU4_1.

The aforementioned losses form a $\mathcal{L}_{\text{total}}$ (11) for training the generator and discriminator.

$$\begin{aligned}\mathcal{L}_{\text{total}} = &\min_{E_c,E_s,D} \max_{Dis} \lambda_{\mathcal{L}_{adv}} \mathcal{L}_{adv} + \lambda_{\mathcal{L}_{pixel}} \mathcal{L}_{pixel} \\ &+ \lambda_{\mathcal{L}_{feat}} \mathcal{L}_{feat} + \lambda_{\mathcal{L}_{RAC}} \mathcal{L}_{RAC} + \lambda_{\mathcal{L}_C} \mathcal{L}_C.\end{aligned} \tag{11}$$

$\lambda_{\mathcal{L}_{adv}}, \lambda_{\mathcal{L}_{pixel}}, \lambda_{\mathcal{L}_{feat}}, \lambda_{\mathcal{L}_{RAC}}, \lambda_{\mathcal{L}_C}$ are weights of each loss with values set to 0.1, 10, 1, 0.01, and 0.01, respectively, except for the ablation study.

The weights of the generator and the discriminator are initialized using He initialization (He et al., 2015), and spectral normalization (Miyato et al., 2018) is applied when updating the weights of both networks.

5. Experiments

5.1. Experimental setup

Since there is no available public dataset for benchmark and evaluation, we constructed two datasets, one for Hangeul and the other for Chinese characters, comprising 278 Hangeul fonts[1] and 373 Chinese fonts,[2] respectively. Both datasets were divided into test datasets of 40 fonts each and training datasets of the remaining fonts. Specifically, the test sets for Hangeul and Chinese fonts constituted 14% and 10% of the total dataset, respectively. For both Hangeul and Chinese datasets, we generated 2048 and 3300 character images of 64 × 64 size for each font, respectively. For the arbitrary font generation evaluation,

[1] https://noonnu.cc/
[2] https://chinesefontdesign.com/

Table 1
Comparison between the proposed method and the existing methods. N-shot means the number of images required to generate a font. Style representation indicates which level (global, component, stroke, patch) style feature is defined in. The additional label denotes additional required label data.

	N-shot	Style representation	Additional label
EMD (Zhang et al., 2018b)	10	global	
AGIS-Net (Gao et al., 2019)	10	global	
DG-Font (Xie et al., 2021)	1	global	
DM-Font (Cha et al., 2020)	N	component	✓(component)
LF-Font (Park et al., 2021)	10	component	✓(component)
XMP-Font (Liu et al., 2022)	28 (stroke)	stroke	✓(stroke)
FS-Font (Tang et al., 2022)	3 (100)	component	content-reference mapping
CG-GAN (Kong et al., 2022)	1	component	only training (component)
RAC-Font (ours)	1	patch	

we generated 2350 characters and 3750 characters for Hangeul and Chinese in the same size as for training, respectively.

For training parameters, we used Adam (Kingma and Ba, 2015) optimizer with $\beta_1 = 0$ and $\beta_2 = 0.9$, and the learning rate of 0.0003 for the generator and 0.0001 for the discriminator, respectively. The weight of the momentum encoder for global style loss ($\mathcal{L}_C$) was updated with a momentum of 0.999. The networks were trained for 10 epochs on the Hangeul dataset and 15 epochs on the Chinese dataset, respectively, with a batch size of 64. All experiments were conducted with Pytorch v1.9.0 framework, CUDA v11.1, and CuDNN v8.0.5 on a single NVIDIA RTX 3090 device.

5.2. Comparison to the previous FFGs

The proposed method was compared to other methods that can generate arbitrary fonts (EMD (Zhang et al., 2018b), AGIS-net (Gao et al., 2019), DG-Font (Xie et al., 2021), and CG-GAN (Kong et al., 2022)), both qualitatively and quantitatively. Since EMD and AGIS-Net have a structure that uses multiple images as input (N-shot font generation method), 10 images were used as input for EMD and AGIS-Net. The proposed method (RAC-Font), DG-Font, and CG-GAN used a single image as input. Unlike the methods that encode the entire image, DM-Font (Cha et al., 2020), LF-Font (Park et al., 2021), and XMP-Font (Liu et al., 2022) are excluded from the comparison group for a fair comparison because they require additional component-wise (or stroke-wise) labeling. FS-Font (Tang et al., 2022) is also excluded from the comparison group because there is no available code. All results of the comparison methods have been trained with our dataset using the official code. The summarized information about these methods is presented in Table 1.

5.2.1. Qualitative comparison

Fig. 4 shows the results of font generation in the Hangeul and Chinese test datasets. In the first row of Fig. 4a–b, the style reference images used for each font generation are included. For N-shot font generation methods (EMD and AGIS-Net), which require N reference style images, the N-1 reference style images are omitted for a more concise presentation. EMD (Zhang et al., 2018b) failed to generate thin fonts and struggled with generating Chinese fonts that have a more complicated structure than Hangeul. AGIS-Net (Gao et al., 2019) produced fonts with a good overall structure but lost local style at the endpoint of the stroke. DG-Font (Xie et al., 2021) had a problem that poorly reflected thickness and often reconstructed the content image as it is (green box in Fig. 4). The methods that use global styles (EMD, AGIS-Net, and DG-Font) were not able to represent local styles, such as serif-ness, varying thickness, connected or disconnected strokes, etc. CG-GAN (Kong et al., 2022) produced results with good local style, but it occasionally lost the global structure of the glyph (purple boxes in Fig. 4). In contrast, our method accurately generated the handwritten or cursive font styles, *e.g.*, serif-ness, thin stroke, varying thickness, and cursiveness. In the case of bold style, ours generated bold fonts without collapsing the glyph. For local styles such as the style at the endpoints (red boxes in Fig. 4) and the overlapped strokes (blue boxes in Fig. 4), our method shows better performance in detail. Furthermore, without explicitly considering content information, the proposed method proficiently generates unseen characters that were not used during training and no structural collapse of the content, such as the disappearance of components or strokes and an imbalance in the overall glyph.

5.2.2. Quantitative comparison

For quantitative comparison, we used several metrics that are widely used for generative model evaluation. First, we used the Structural Similarity Index Measure (SSIM) (Wang et al., 2004) which measures the structural similarity between two images in a pixel-level evaluation. We also employed Fréchet Inception Distance (FID) (Heusel et al., 2017) and Learned Perceptual Image Patch Similarity (LPIPS) (Zhang et al., 2018a) for perceptual level evaluation since SSIM may not perfectly align with human perception. FID measures the difference between the distributions of the ground truths and the generated images by using InceptionV3 (Szegedy et al., 2016). On the other hand, LPIPS compares the feature similarity between two images using a pre-trained VGG network (Simonyan and Zisserman, 2015). Both metrics are popularly used to evaluate perceptual similarity. In addition, we slightly modified FID to mFID (Liu et al., 2019) which is the average FID across font libraries. In last, to assess the spatial (local) characteristics of the font, we included spatial FID (sFID) (Nash et al., 2021). It is more sensitive to spatial variation than FID because it uses a feature with spatial dimensions.

Table 2 presents the quantitative evaluation results of Hangeul and Chinese characters. The RAC-Font (ours) is superior to other methods in terms of mFID evaluation. e.g., lower than CG-GAN by 2.09 and 5.42 for the Hangeul train and test datasets, respectively. Despite using a single image as input, RAC-Font outperforms the methods that use multiple images as input. e.g., 14.31/25.32 lower mFID than EMD and 10.13/14.37 lower mFID than AGIS-Net for the Hangeul train/test dataset, respectively. AGIS-Net shows high performance on the training dataset but poor performance on the test dataset. Additionally, our simple structure achieves better performance compared to DG-Font, which has a complex structure and performs well on the test dataset, *e.g.*, 9.11 and 10.88 lower mFID in Hangeul and Chinese test datasets, respectively. CG-GAN shows good performance in some metrics, especially SSIM, as it is supervised at a component-level. In certain perceptual-level metrics, such as a lower mFID value of 1.85 in the Chinese test dataset, its performance is slightly better than ours. However, it is difficult to recognize the difference in the quality of the generated results as shown in Fig. 4. These results illustrate that applying RAC-loss to a simple structure, *e.g.*, a convolution network without both attention and decomposition of components, can improve performance on the train and test datasets.

In addition, we measured the inference time of each method except EMD (Zhang et al., 2018b) because the official code of EMD is not compatible with our experimental environment (RTX 3090). To ensure accurate inference time measurement, we calculated the average of

Fig. 4. Qualitative comparison: All results were generated with unseen characters and test font dataset that were not used for training.

Table 2
Quantitative comparison: All values were calculated for 2350 Hangeul characters and 3750 Chinese characters. The ↑ and ↓ signs indicate that lower values are better and higher values are better, respectively.

		Hangeul				Chinese			
		SSIM ↑	LPIPS ↓	mFID ↓	sFID ↓	SSIM ↑	LPIPS ↓	mFID ↓	sFID ↓
Train	EMD (Zhang et al., 2018b)	0.7315	0.0819	35.16	18.07	0.7198	0.1390	78.15	24.60
	AGIS-Net (Gao et al., 2019)	0.7840	0.0752	30.98	14.25	0.7995	0.1047	64.94	20.99
	DG-Font (Xie et al., 2021)	0.7183	0.0742	38.95	12.43	0.7539	0.0934	35.92	17.08
	CG-GAN (Kong et al., 2022)	**0.8729**	0.0712	23.94	10.94	**0.8123**	0.0973	30.54	16.18
	RAC-Font (ours)	0.7213	**0.0709**	**20.85**	**8.46**	0.7633	**0.0928**	**26.81**	**16.10**
Test	EMD (Zhang et al., 2018b)	0.6720	0.1205	68.50	29.07	0.6332	0.1955	99.15	32.35
	AGIS-Net (Gao et al., 2019)	0.7149	0.1191	57.55	23.61	0.7630	0.1267	71.50	24.93
	DG-Font (Xie et al., 2021)	0.7031	0.1607	52.29	18.65	0.7185	0.1851	45.22	21.17
	CG-GAN (Kong et al., 2022)	**0.8576**	0.1183	48.60	**16.19**	**0.8091**	0.1168	**32.49**	19.92
	RAC-Font (ours)	0.7157	**0.1057**	**43.18**	17.61	0.7051	**0.1119**	34.34	**19.87**

Table 3
Results of inference time. The inference time is measured on a single NVIDIA RTX3090 GPU with a batch size of 1.

Method	Inference time (ms) ↓
EMD (Zhang et al., 2018b)	×
AGIS-Net (Gao et al., 2019)	2.65
DG-Font (Xie et al., 2021)	3.85
CG-GAN (Kong et al., 2022)	12.10
RAC-Font (ours)	3.02

1000 inference times after 100 warm-up runs, which is presented in Table 3. This result clearly shows that the proposed method outperforms recent methods, such as DG-Font, and even surpasses CG-GAN in terms of speed. Despite the increased complexity of the latter method, which is limited to the discriminator and shares similarities with our approach, the proposed method shows nearly four times faster speed with its simple structure, demonstrating its capability to effectively generate fonts in real-time.

5.3. Ablation study

In this section, we conducted the ablation study to compare the RAC-loss and the other patch-level contrastive losses, including DenseCL (Wang et al., 2021) and PatchNCE (Park et al., 2020). Table 4 presents the performance of each component. Baseline (config. A in Table 4) refers to the network trained without two contrastive losses ($\mathcal{L}_C$ and $\mathcal{L}_{RAC}$), and config. B/C/D/E are the networks trained by adding each independent component to the config. A.

Config. B denotes the result of the network trained with instance-level contrastive loss (6). Compared to the baseline, mFID increased by 4.01 and 5.65 in the Hangeul train and test datasets, respectively, and decreased by 2.67 and 3.65 in the Chinese datasets, respectively. Different performance patterns were observed between the two datasets. Due to the smaller scale of the Hangeul dataset in our setup, the key in the memory queue is more likely to have the same font classes as a query. Consequently, treating the font in the same class of the query as a negative sample seems to reduce the performance in the Hangeul dataset.

Config. C is a network trained with PatchNCE (Park et al., 2020) that employs positive patches in the same positions and negative patches in different positions. Compared to the baseline, PatchNCE shows worse performance, with an increase in mFID by 6.4 (or 7.65) for the Hangeul train (or test) datasets and by 0.93 (or 1.06) for the Chinese train (or test) datasets, respectively. This is even worse than instance-level contrastive learning (config. B in Table 4). This result suggests that applying a positive patch to the same location of two glyph images with different contents degrades the performance of tasks that require big shape changes, such as font generation.

Config. D is the result of a network trained with DenseCL (Wang et al., 2021), which samples only one patch with a top-1 similarity as positive.

Table 4
Results of ablation study: Config. A is a network trained with GAN loss (7)–(8), pixel loss (9), and feature loss (10). $\mathcal{L}_C$ denotes instance-level contrastive learning, and PatchNCE, DenseCL, and $\mathcal{L}_{RAC}$ denote patch-level contrastive loss, respectively. The ↑ and ↓ signs indicate that lower values are better and higher values are better, respectively.

	Configure	Hangeul				Chinese			
		SSIM ↑	LPIPS ↓	mFID ↓	sFID ↓	SSIM ↑	LPIPS ↓	mFID ↓	sFID ↓
Train	A. Baseline	0.7063	0.0738	32.70	28.81	0.6609	0.1213	41.30	25.55
	B. + $\mathcal{L}_C$	0.7073	0.0712	36.71	29.35	0.6521	0.1034	38.63	18.05
	C. + PatchNCE	0.7037	0.0759	39.10	34.36	0.6514	0.1227	42.23	28.51
	D. + DenseCL	0.7129	0.0730	26.90	17.11	0.7430	0.1052	33.57	16.16
	E. + $\mathcal{L}_{RAC}$	0.7124	**0.0702**	22.69	15.10	0.7606	0.1043	29.78	16.10
	F. config E. +$\mathcal{L}_C$	**0.7213**	0.0709	**20.85**	**8.46**	**0.7633**	**0.0928**	**26.81**	**15.13**
Test	A. Baseline	0.6899	0.1098	53.33	35.61	0.6624	0.1366	50.50	28.41
	B. + $\mathcal{L}_C$	0.6865	0.1075	58.98	38.04	0.6745	0.1181	46.85	20.13
	C. + PatchNCE	0.6816	0.1106	60.98	39.67	0.6610	0.1424	51.56	34.01
	D. + DenseCL	0.6912	0.1156	49.89	26.15	0.6959	0.1169	44.19	19.87
	E. + $\mathcal{L}_{RAC}$	0.6993	0.1061	43.87	23.69	0.6826	0.1155	37.24	19.14
	F. config E. +$\mathcal{L}_C$	**0.7157**	**0.1057**	**43.18**	**17.61**	**0.7051**	**0.1119**	**34.34**	**17.71**

Table 5
Comparison of mFID according to feature resolution: Experiments were conducted on the Hangeul datasets. 4 × 4 and 8 × 8 features were obtained by average pooling 16 × 16 feature.

	Feature resolution			
Dataset	4 × 4	8 × 8	16 × 16	32 × 32
Train	30.10	26.99	**22.69**	30.63
Test	53.55	49.73	**43.87**	52.89

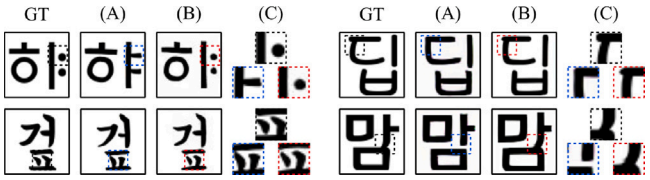

Fig. 5. Unseen font generation result according to the RAC-loss: (A) config. A in Table 4, (B) config. E in Table 4, (C) Enlarged view of the dashed box part of each row.

Unlike PatchNCE, DenseCL allows the sampling of positive patches from different locations, resulting in performance improvement over the baseline (config. A), *e.g.*, mFID for each Hangeul train and test: 32.70 → 26.90 and 53.33 → 49.89. Based on this result, it can be seen that patch-level contrastive learning using significant patch pairs at various regions as a positive patch is more effective for image-to-image translation tasks, such as font generation.

Config. E, the result of a network trained with RAC-loss, outperforms the previous two models trained with patch-level contrastive loss, namely PatchNCE (config. C in Table 4) and DenseCL (config. D in Table 4). A detailed comparison and analysis of DenseCL and RAC-loss will be dealt with in the later Section 5.4. In this configuration, a feature with a resolution of 16 × 16 was used to compute the RAC-loss. Choosing an appropriate resolution for the task and input image resolution is crucial; therefore, we conducted an ablation study to further investigate this. Table 5 presents the performance according to the feature resolution. Both small and large feature resolutions degrade the performance. For example, in the test dataset, the 4 × 4 feature raised mFID by 9.68, and the 32 × 32 feature raised mFID by 9.02 compared to the 16 × 16 resolution. This is because when the resolution of the feature is too low (high-level feature), the information in one pixel of the feature is compressed excessively. On the other hand, when the resolution is too high (low-level feature), the information in one pixel is insufficient, which negatively affects the learning process and results in performance degradation. This outcome demonstrates the importance of selecting a suitable resolution compared to the input image resolution.

Config. F is the result using both RAC-loss (config. E in Table 4) and instance-level contrastive learning (config. B in Table 4). When the instance-level contrastive loss was used alone (config. B), the tendency of performance varied depending on the data. However, when both patch-level and instance-level contrastive losses were used together (config. F), the performance improved compared to using patch-level loss alone (config. E), *e.g.*, the mFID of the Hangeul train and test data decreased from 22.69 to 20.85 and from 43.87 to 43.18, respectively. This result shows that instance-level and patch-level contrastive loss work complementary with each other, as mentioned in the previous study (Wang et al., 2021).

In addition to the quantitative comparison, we also performed a qualitative comparison according to RAC-loss, and the result is presented in Fig. 5. In the dashed box, the network trained with RAC-loss (B in Fig. 5) shows better results in some detailed aspects, such as breaking strokes, cursive strokes, shapes, etc., compared to the network trained without RAC-loss (A in Fig. 5).

The ablation study demonstrates that assigning positive patches with the proposed method outperformed simply assigning a positive sample to the same location (PatchNCE) and ignoring other meaningful patches (DenseCL).

5.4. Analysis of attention-based sampling

To show the effectiveness of the proposed attention-based positive sampling approach in one-to-many correspondence settings, such as the Few-shot Font Generations (FFGs), we compared and analyzed the proposed method with other one-to-many methods. Specifically, we modified the top-1 positive sampling of DenseCL to a top-K positive sampling for one-to-many correspondence, which we call DenseCL-top(K). Table 6 shows the comparison between DenseCL-top(K) (K = 1, 2, 3) and RAC-loss ($\mathcal{L}_{RAC}$).

As shown in Table 6, DenseCL-top(K) shows worse performance than the original DenseCL. To analyze this result, we present the histogram of the patch-corresponding attention (A) values in Fig. 6. The histogram illustrates that most cases have values between 0 and 0.1, with the remaining values distributed across other ranges. This suggests that only a few key patches are related to the query patch, while the rest have no association with the query patch at all. In such a scenario, simply sampling the top-K patches as positive samples has the potential to degrade performance because even patches with low correlation can be chosen as positive samples. On the other hand, the proposed

Table 6
Comparison of positive patch sampling methods. For both LPIPS and mFID, the lower is the better.

Method	Train		Test	
	LPIPS	mFID	LPIPS	mFID
DenseCL	0.0730	26.90	0.1106	49.89
DenseCL-top2	0.0755	27.73	0.1141	51.39
DenseCL-top3	0.0778	31.54	0.1134	53.32
$\mathcal{L}_{RAC}$	**0.0702**	**22.69**	**0.1061**	**43.87**

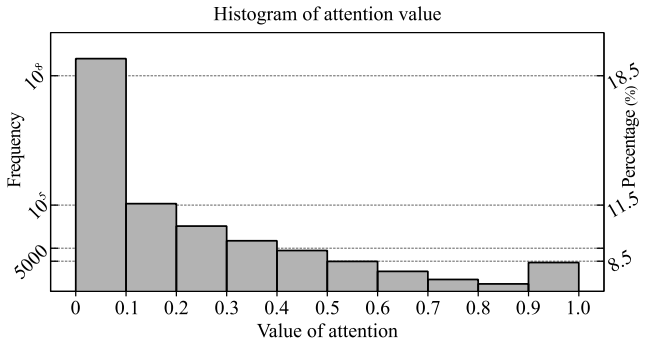

Fig. 6. Histogram of attention value with $n = 0.1$. The left and right y-axes denote frequency and percentage, respectively.

method ($\mathcal{L}_{RAC}$) aligns the key feature using patch-corresponding attention, which indicates the correlation between patches, rather than simply sampling the top-K patches. As a result, our method ($\mathcal{L}_{RAC}$) can sample an arbitrary number of positive patches at various spatial positions, resulting in a remarkable improvement in performance.

5.5. Patch correspondence visualization

To figure out how RAC-loss affects discriminator learning, we visualized the positive patch sampling result and analyzed it. Initially, we obtained the features ($F_{\hat{y}}, F_{x_s}$) of the generated image and style reference image using the discriminator. Subsequently, we compared the patch similarity of these two features and identified the patches with a similarity higher than 0.9. For concise visualization, we randomly selected only 5 patches among the matched patches and displayed them in Fig. 7.

When a network is not trained with RAC-loss ((C) and (D) in Fig. 7), matching occurs at the same location or between completely unrelated patches. In contrast, when RAC-loss is employed ((A) and (B) in Fig. 7), relatively similar styles are matched between patches. However, some matching results are slightly incorrect. This is because the weight of RAC-loss was smaller than the weight of the other losses during discriminator training. Nevertheless, even with this minor mismatching, the matching across patches in various areas was feasible to some extent, allowing the font generator to produce font images of high quality in detail with RAC-loss.

5.6. Analysis of RAC-loss

We analyzed the sensitivity of two parameters (n in (3) and τ in (5)) of the RAC-loss with the Hangeul dataset, and the result is presented in Fig. 8. As shown in Fig. 8, mFID is raised when both the temperature factor (τ in (5)) and attention normalizing factor (n in (3)) take very small values. Especially, a small value for n leads to performance degradation regardless of τ.

To demonstrate this performance degradation, we visualized the patch-corresponding attention according to the normalizing factor (n) as shown in Fig. 9. Given a specific spatial location in the query feature map ($F_{\hat{y}}$) as a query patch (red boxes in the 1st row of Fig. 9), we obtained the corresponding patches in the key feature map (F_{x_s}) (2nd and 3rd rows in Fig. 9). If the corresponding patches were incorrect (1st and 2nd columns in Fig. 9), when using the low value of n (Fig. 9b), the incorrectly matched patch was used as a positive patch, resulting in poor performance. On the other hand, when an appropriate value for n is used (Fig. 9b), other matched patches were employed as positive patches, and this led to relatively better performance. Additionally, choosing an appropriate value for n enables all different patches associated with a query patch to be treated as positive patches rather than negatives, as illustrated in the 3rd, 4th, and 5th columns of Fig. 9. Through the visualization and analysis, it is evident that the attention map clearly takes the font style into account at a lower level (patch-level) than the previous FFGs (Cha et al., 2020; Park et al., 2021; Liu et al., 2022; Tang et al., 2022; Kong et al., 2022) (component-level). This patch-level style can utilize various similar patches, which allows our method to generate a finer font than the other methods.

The training curve of RAC-loss is also presented in Fig. 10. In the early stage, the patch-corresponding attention (A) struggles to identify the style corresponding patch, leading to unstable training before 10,000 iterations. However, as the training of attention becomes stable, the proposed loss converges.

6. Conclusion

In this work, we proposed Region-Aware Contrastive Font (RACFont) that leverages the local style without any additional labels. We defined the local style on a per-patch level and adapted it into contrastive loss, which maximizes the mutual information of patches with similar styles. Through our observation, it was discovered that a single patch could match multiple patches located in different positions. To address this issue, spatial attention was utilized to identify the style correspondence between patches, and its values were used as weights to aggregate multiple patches, constructing a new aligned patch as a positive patch. This patch sampling strategy outperformed using only the patch with the top-1 similarity. The proposed network showed the lowest mFID values of 20.85 and 43.18 in the Hangeul train and test datasets, respectively, demonstrating lower mFID values of 3.09 and 5.42 compared to the previous work that required additional labels. Although our method had slightly worse performance than the previous work, for instance, a higher mFID value of 1.85 in the Chinese test dataset, this value does not show a significant visual difference, as discussed in the experiment section. Additionally, an ablation study revealed that sampling the positive sample in the same location and sampling only one patch as positive were not suitable for tasks requiring significant changes in the appearance of an image. Furthermore, we showed that the patch-corresponding attention map learned by the proposed method matches several patches in different positions. Then we analyzed this attention map and demonstrated that the proposed method can generate better fonts. Since the proposed method focuses on the discriminator, it can be directly applied to any Few-shot Font Generations (FFGs) and is expected to apply to various image-to-image translation domains. While CG-GAN (Kong et al., 2022) focused on the discriminator like the proposed method, it requires additional component labels, unlike the proposed method, which operates without the need for extra labels. Although we focused on maximizing the style information across patches, maximizing the content information between patches of words that share the same content but have different styles can be a possible future research to improve the quality of font generation for Chinese characters of complex structure. Additionally, while the proposed loss has successfully encouraged the generator to implicitly consider local style, introducing the patch concept directly into the generator will further enhance its performance in capturing local styles.

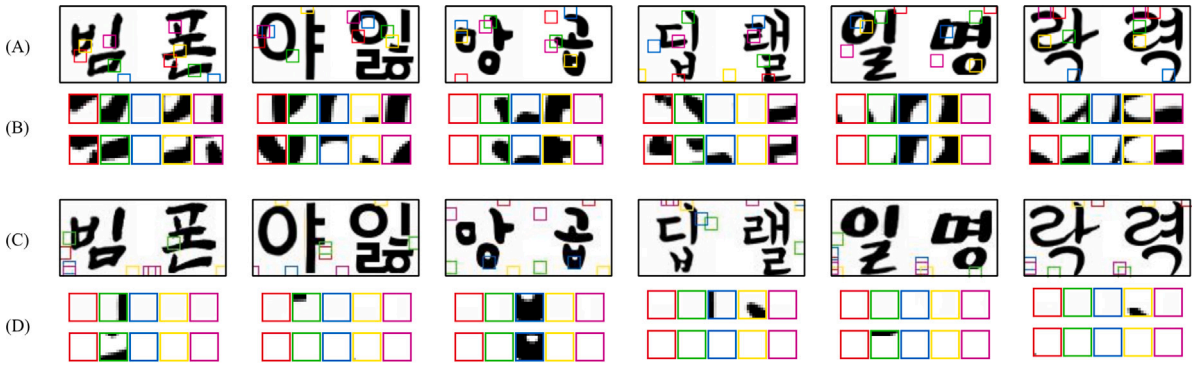

Fig. 7. Result of patch matching with more than a certain similarity: (A) Result of a network trained with $\mathcal{L}_{RAC}$, (B) The cropped and enlarged matched patch of (A), (C) Result of a network trained without $\mathcal{L}_{RAC}$, (D) The cropped and enlarged matched patch of (C).

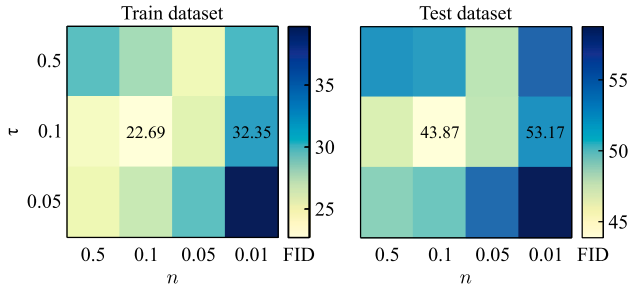

Fig. 8. mFID changes according to temperature value (τ) and normalizing factor (n): In each dataset, the lowest value of mFID is shown.

Fig. 9. Visualization of patch-corresponding attention when the red box in 1st row is a query. (a) and (b) show an attention map when the normalizing factor (n) is 0.1 and 0.01, respectively. With a low value of n (b), it behaves like DenseCL (Wang et al., 2021) which samples the one positive patch that has a top-1 similarity.

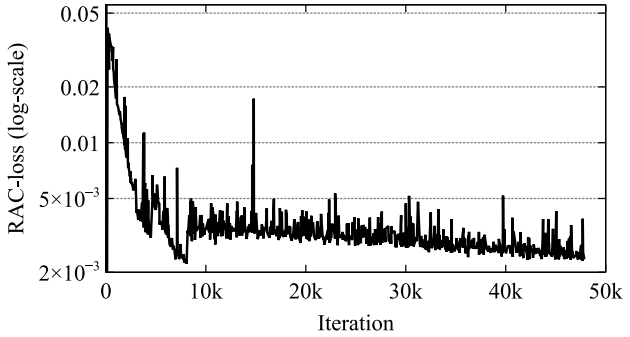

Fig. 10. Visualization of RAC-loss training curve.

CRediT authorship contribution statement

Jeong-Sik Lee: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Conceptualization. **Hyun-Chul Choi:** Writing – review & editing, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research was funded by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2022R1A2C2013541) in part and the 2023 Yeungnam University Research Grant in part.

References

Cha, J., Chun, S., Lee, G., Lee, B., Kim, S., Lee, H., 2020. Few-shot compositional font generation with dual memory. In: European Conference on Computer Vision. ECCV.
Chen, T., Kornblith, S., Norouzi, M., Hinton, G., 2020. A simple framework for contrastive learning of visual representations. In: International Conference on Machine Learning. PMLR, pp. 1597–1607.
Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations.
Gao, Y., Guo, Y., Lian, Z., Tang, Y., Xiao, J., 2019. Artistic glyph image synthesis via one-stage few-shot learning. ACM Trans. Graph. 38 (6), 1–12.
Gao, Y., Wu, J., 2020. GAN-based unpaired Chinese character image translation via skeleton transformation and stroke rendering. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34, (01), pp. 646–653.
He, K., Fan, H., Wu, Y., Xie, S., Girshick, R., 2020. Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9729–9738.
He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1026–1034.
Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S., 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. Adv. Neural Inf. Process. Syst. 30.
Huang, X., Belongie, S., 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1501–1510.
Jiang, Y., Lian, Z., Tang, Y., Xiao, J., 2019. Scfont: Structure-guided chinese font generation via deep stacked networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33, (01), pp. 4015–4022.
Kang, M., Park, J., 2020. ContraGAN: Contrastive learning for conditional image generation. In: Conference on Neural Information Processing Systems (NeurIPS).

Karras, T., Laine, S., Aila, T., 2019. A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4401–4410.

Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Kong, Y., Luo, C., Ma, W., Zhu, Q., Zhu, S., Yuan, N., Jin, L., 2022. Look closer to supervise better: One-shot font generation via component-based discriminator. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13482–13491.

Li, W., He, Y., Qi, Y., Li, Z., Tang, Y., 2020. FET-GAN: Font and effect transfer via K-shot adaptive instance normalization. In: Proceedings of the AAAI Conference on Artificial Intelligence. 34.

Liu, M.-Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz, J., 2019. Few-shot unsupervised image-to-image translation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10551–10560.

Liu, W., Liu, F., Ding, F., He, Q., Yi, Z., 2022. XMP-font: Self-supervised cross-modality pre-training for few-shot font generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7905–7914.

Mirza, M., Osindero, S., 2014. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y., 2018. Spectral normalization for generative adversarial networks. In: International Conference on Learning Representations.

Miyato, T., Koyama, M., 2018. CGANs with projection discriminator. In: International Conference on Learning Representations.

Nash, C., Menick, J., Dieleman, S., Battaglia, P., 2021. Generating images with sparse representations. In: Meila, M., Zhang, T. (Eds.), Proceedings of the 38th International Conference on Machine Learning. In: Proceedings of Machine Learning Research, Vol. 139, PMLR, pp. 7958–7968.

Odena, A., Olah, C., Shlens, J., 2017. Conditional image synthesis with auxiliary classifier gans. In: International Conference on Machine Learning. PMLR, pp. 2642–2651.

Oord, A.v.d., Li, Y., Vinyals, O., 2018. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748.

Park, S., Chun, S., Cha, J., Lee, B., Shim, H., 2021. Few-shot font generation with localized style representations and factorization. In: AAAI Conference on Artificial Intelligence.

Park, T., Efros, A.A., Zhang, R., Zhu, J.-Y., 2020. Contrastive learning for unpaired image-to-image translation. In: European Conference on Computer Vision.

Park, T., Liu, M.-Y., Wang, T.-C., Zhu, J.-Y., 2019. Semantic image synthesis with spatially-adaptive normalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2337–2346.

Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR.

Tang, L., Cai, Y., Liu, J., Hong, Z., Gong, M., Fan, M., Han, J., Liu, J., Ding, E., Wang, J., 2022. Few-shot font generation by learning fine-grained local styles. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7895–7904.

Tian, Y., 2024. Zi2zi. https://github.com/kaonashi-tyc/zi2zi, (Accessed 15 January 2024).

Wang, Z., Bovik, A., Sheikh, H., Simoncelli, E., 2004. Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13 (4), 600–612. http://dx.doi.org/10.1109/TIP.2003.819861.

Wang, X., Zhang, R., Shen, C., Kong, T., Li, L., 2021. Dense contrastive learning for self-supervised visual pre-training. In: Proc. IEEE Conf. Computer Vision and Pattern Recognition. CVPR.

Xie, Y., Chen, X., Sun, L., Lu, Y., 2021. DG-font: Deformable generative networks for unsupervised font generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5130–5140.

Zhang, H., Goodfellow, I., Metaxas, D., Odena, A., 2019. Self-attention generative adversarial networks. In: International Conference on Machine Learning. PMLR, pp. 7354–7363.

Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O., 2018a. The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR.

Zhang, Y., Zhang, Y., Cai, W., 2018b. Separating style and content for generalized style transfer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8447–8455.

Zhu, J.-Y., Park, T., Isola, P., Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Computer Vision (ICCV), 2017 IEEE International Conference on.