Font Style that Fits an Image –
Font Generation Based on Image Context

Taiga Miyazono, Brian Kenji Iwana, Daichi Haraguchi, and Seiichi Uchida
{iwana,uchida}@ait.kyushu-u.ac.jp
Kyushu University, Fukuoka, Japan

Abstract. When fonts are used on documents, they are intentionally selected by designers. For example, when designing a book cover, the typography of the text is an important factor in the overall feel of the book. In addition, it needs to be an appropriate font for the rest of the book cover. Thus, we propose a method of generating a book title image based on its context within a book cover. We propose an end-to-end neural network that inputs the book cover, a target location mask, and a desired book title and outputs stylized text suitable for the cover. The proposed network uses a combination of a multi-input encoder-decoder, a text skeleton prediction network, a perception network, and an adversarial discriminator. We demonstrate that the proposed method can effectively produce desirable and appropriate book cover text through quantitative and qualitative results. The code can be found at https://github.com/Taylister/FontFits.

Keywords: Text generation · Neural font style transfer · Book covers

1 Introduction

Fonts come in a wide variety of styles, and they can come with different weights, serifs, decorations, and more. Thus, choosing a font to use in a medium, such as book covers, advertisements, documents, and web pages, is a deliberate process by a designer. For example, when designing a book cover, the title design (i.e., the font style and color for printing the book title) plays an important role [14,25]. An appropriate title design will depend on visual features (i.e., the appearance of the background design), as well as semantic features of the book (such as the book content, genre, and title texts). As shown in Fig. 1 (a), given a background image for a book cover, typographic experts determine an appropriate title design that fits the background image. In this example, yellow will be avoided for title design to keep the visual contrast from the background; a font style that gives a hard and solid impression may be avoided by considering the impression from the cute rabbit appearance. These decisions about fonts are determined by the image context.

This paper aims to generate an appropriate text image for a given context image to understand their correlation. Specifically, as shown in Fig. 1 (b), we
Fig. 1. (a) Our question: Is there any title design that fits a book cover? (b) Our approach: If there is a specific design strategy, we can realize a neural network that learns the correlation between the cover image and the title design and then generate an appropriate title design.

Fig. 2. Overview of the proposed framework.

attempt to generate a title image that fits the book cover image by using a neural network model trained by 104,295 of actual book cover samples. If the model can generate a similar title design to the original, the model catches the correlation inside it and shows the existence of the correlation.

Achieving this purpose is meaningful for two reasons. First, its achievement shows the existence of the correlation between design elements through objective analysis with a large amount of evidence. It is often difficult to catch the correlation because the visual and typographic design is performed subjectively with a huge diversity. Thus, the correlation is not only very nonlinear but also very weak. If we prove the correlation between the title design and the book cover image through our analysis, it will be an important step to understand the “theory” behind the typographic designs. Secondly, we can realize a design support system that can suggest an appropriate title design from a given back-
This system will be a great help to non-specialists in typographic design.

In order to learn how to generate a title from a book cover image and text information, we propose an end-to-end neural network that inputs the book cover image, a target location mask, and a desired book title and outputs stylized text. As shown in Fig. 2, the proposed network uses a combination of a Text Encoder, Context Encoder, Skeleton Generator, Text Generator, Perception Network, and Discriminator to generate the text. The Text Encoder and Context Encoders encode the desired text and given book cover context. The Text Generator use skeleton-guided text generation [29] to generate the text, and the Perception Network and adversarial Discriminator refine the results.

The main contributions of this paper are as follows:

– We propose an end-to-end system of generating book title text based on the cover of a book. As far as we know, this paper presents the first attempt to generate text based on the context information of book covers.
– A novel neural network is proposed, which includes a skeleton-based multi-task and multi-input encoder-decoder, a perception network, and an adversarial network.
– Through qualitative and quantitative results, we demonstrate that the proposed method can effectively generate text suitable given the context information.

2 Related Work

Recently, font style transfer using neural networks [1] has become a growing field. In general, there are three approaches toward neural font style transfer, GAN-based methods, encoder-decoder methods, and NST-based methods. A GAN-based method for font style transfer uses a conditional GAN (cGAN). For example, Azadi et al. [2] used a stacked cGAN to generate isolated characters with a consistent style learned from a few examples. A similar approach was taken for Chinese characters in a radical extraction-based GAN with a multi-level discriminator [10] and with a multitask GAN [28]. Few-shot font style transfer with encoder-decoder networks have also been performed [33]. Wu et al. [29] used a multitask encoder-decoder to generate stylized text using text skeletons. Wang et al. [26] use an encoder-decoder network to identify text decorations for style transfer. In Lyu et al. [18] an autoencoder guided GAN was used to generate isolated Chinese characters with a given style. There also have been a few attempts at using NST [6] to perform font style transfer between text [1,7,21,24].

An alternative to text-to-text neural font style transfer, there have been attempts to transfer styles from arbitrary images and patterns. For example, Atarsaikhan et al. [1] proposed using a distance-based loss function to transfer patterns to regions localized to text regions. There are also a few works that use context information to generate text. Zhao et al. [32] predicted fonts for web pages using a combination of attributes, HTML tags, and images of the
web pages. This is similar to the proposed method in that the context of the text is used to generate the text. Yang et al. [30] stylized synthetic text to become realistic scene text. The difference between the proposed method and these methods is that we propose inferring the font style based on the contents of the desired medium and use it in an end-to-end model to generate text.

3 Automatic Title Image Generation

The purpose of this study is to generate an image of the title text with a suitable font and color for a book cover image. Fig. 2 shows the overall architecture of the proposed method. The network consists of 6 modules: Text Encoder, Context Encoder, Text Generator, Skeleton Generator, Perception Network, and Discriminator. The Text Encoder and Context Encoder extracts text and styles from the input text and style cover. The Generator generates the stylized text suitable for the book cover input, and the Skeleton Generator creates a text skeleton to guide the Generator. The Perception Network and Discriminator help refine the output of the Text Generator.

3.1 Text Encoder

The Text Encoder module extracts character shape features from an image $I_t$ of the input text. These features are used by the Text Generator and the Skeleton Generator to generate their respective tasks. As shown in Fig. 3, the Text Encoder input is an image of the target text rendered on a background with a fixed pixel value.

The encoder consists of 4 convolutional blocks with residual connections [9]. The first block consists of two layers of $3 \times 3$ convolutions with stride 1. The subsequent blocks contain a $3 \times 3$ convolutional layers with stride 2 and two convolutional layers with stride 1. Leaky Rectified Linear Units (Leaky ReLU) [19] are used as the activation function for each layer. The negative slope is set to 0.2. There are skip-connections between second and third convolutional blocks and the convolutional blocks of the Skeleton Generator, which is described later.

3.2 Context Encoder

The Context Encoder module extracts the style features from the cover image $S_c$ and the location mask image $S_m$. Examples of the covers $S_c$ and the location mask $S_m$ are shown in Fig. 3. The cover image $S_c$ provides the information about the book cover, such as color, objects, layout, etc.) for the Context Encoder to predict the style of the font and the location mask $S_m$ provides target location information. It should be noted that the cover image $S_c$ has the text inpainted, i.e., removed. Thus, the style is inferred solely based on the cover and not on textual cues.

The Context Encoder input is constructed of $S_c$ and $S_m$ concatenated in the channel dimension. Also, the Context Encoder structure is the same as the Text Encoder.
Encoder, except that the input is only 2 channels (as opposed to 3 channels, RGB, for the Text Encoder). As the input of the generators, the output of the Context Encoder is concatenated in the channel dimension with the output of the Text Encoder.

3.3 Skeleton Generator

In order to improve the legibility of the generated text, a skeleton of the input text is generated and used to guide the generated text. This idea is called the Skeleton-guided Learning Mechanism [29]. This module generates a skeleton map, which is the character skeleton information of the generated title image. As described later, the generated skeleton map is merged into the Text Generator. By doing so, the output of the Text Generator is guided by the skeleton to generate a more robust shape.

This module is an upsampling CNN that uses four blocks of convolutional layers. The first block contains two standard convolutional layers, and the three subsequent blocks have one transposed convolutional layer followed by two standard convolutional layers. The transposed convolutional layers have $3 \times 3$ transposed convolutions at stride 2, and the standard convolutional layers have $3 \times 3$ convolutions at stride 1. All the layers use Batch Norm and LeakyReLU. In addition, there are skip-connections between the Text Encoder and after the first layer of the second and third convolutional blocks.

To train the Skeleton Generator, the following skeleton loss $L_{\text{skeleton}}$ is used to train the combined network:

$$L_{\text{skeleton}} = 1 - \frac{2 \sum_{i=1}^{N} (T_{\text{skeleton}})_i (O_{\text{skeleton}})_i}{\sum_{i=1}^{N} (T_{\text{skeleton}})_i + \sum_{i=1}^{N} (O_{\text{skeleton}})_i},$$  

(1)
where \( N \) is the number of pixels, \( T_{\text{skeleton}} \) is true skeleton map, and \( O_{\text{skeleton}} \) is the output of Skeleton Generator. This skeleton loss is designed based on DiceLoss [20].

### 3.4 Text Generator

The Text Generator module takes the features extracted by Text Encoder and Style Encoder and outputs an image of the stylized title. This output is the desired result of the proposed method. It is an image of the desired text with the style inferred from the book cover image.

The Text Generator has the same structure as the Skeleton Generator, except with no skip-connections and with an additional convolutional block. The additional convolutional block combines the features from the Text Generator and the Skeleton Generator. As described previously, the Skeleton Generator generates a skeleton of the desired stylized text. To incorporate the generated skeleton into the generated text, the output of the Skeleton Generator is concatenated with the output of the fourth block of the Text Generator. The merged output is further processed through a \( 3 \times 3 \) convolutional layer with a stride of 1, Batch Normalization, and Leaky ReLU activation. The output \( O_t \) of the Text Generator has a tanh activation function.

The Text Generator is trained using a reconstruction loss \( L_{\text{reconst}} \). The reconstruction loss is Mean Absolute Error between the generated output \( O_t \) and the ground truth title text \( T_t \), or:

\[
L_{\text{reconst}} = |T_t - O_t|.
\]

(2)

While loss guides the generated text to be similar to the original text, it is only one part of the total loss. Thus, the output of the Text Generator is not strictly the same as the original text.

### 3.5 Perception Network

To refine the results of the Text Generator, we use a Perception Network. Specifically, the Perception Network is used to increase the perception of the generated images [13,29]. To do this, the output of the Text Generator is provided to a Perception Network, and two loss functions are added to the training. These loss functions are the Content Perceptual loss \( L_{\text{content}} \) and the Style Perceptual loss \( L_{\text{style}} \). The Perception Network is a VGG19 [22] that was pre-trained on ImageNet [5].

Each loss function compares differences in the content and style of the features extracted from the Perception Network when provided the generated title and the ground truth title images. The Content Perceptual loss minimizes the distance between the extracted features of the generated title images and the original images, or:

\[
L_{\text{content}} = \sum_{i \in \mathcal{F}} |P_i(T_t) - P_i(O_t)|,
\]

(3)
where $P$ is the Perception Network, $P_l(\cdot)$ is the feature map of $P$’s $l$-th layer given the generated input image $O_t$ or ground truth image $T_t$, and $F$ is the set of layers used in these loss. In this case, $F$ is set to the relu1_1, relu2_1, relu3_1, relu4_1, and relu5_1 layers of VGG19. The Style Perceptual loss compares the texture and local features extracted by the Perception Network, or:

$$
L_{style} = \sum_{l \in F} |\Psi^P_l(T_t) - \Psi^P_l(O_t)|,
$$

where $\Psi^P_l(\cdot)$ is a Gram Matrix, which has $C_l \times C_l$ elements. Given input $I \in \{O_t, T_t\}$, $P_l(I)$ has a feature map of shape $C_l \times H_l \times W_l$. The elements of Gram Matrix $\Psi^P_l(I)$ are given by:

$$
\Psi^P_l(I)_{c,c'} = \frac{1}{C_l H_l W_l} \sum_{h=1}^{H_l} \sum_{w=1}^{W_l} P_l(I)_{h,w,c} P_l(I)_{h,w,c'},
$$

where $c$ and $c'$ are each element of $C_l$. By minimizing the distance between the Gram Matrices, a style consistency is enforced. In other words, the local features of both images should be similar.

### 3.6 Discriminator

In addition to the Perception Network, we also propose the use of an adversarial loss $L_{\text{adversarial}}$ to ensure that the generated results are realistic. To use the adversarial loss, we introduce a Discriminator to the network architecture. The Discriminator distinguishes between whether the input is a real title image or a fake image. In this case, we use the true tile image $T_t$ as the real image and the Text Generator’s output $O_t$ as the fake image.

The Discriminator in the proposed method follows the structure of the DCGAN [17]. The Discriminator input goes through 5 down-sampling $5 \times 5$ convolutional layers with stride 2 and finally a fully-connected layer. The output is the probability that the input is a true title image. Except for the output layer, the LeakyReLU function is used. At the output layer, a sigmoid function is adopted. The following adversarial loss is used to optimize the entire generation model and the Discriminator:

$$
L_{\text{adversarial}} = \mathbb{E}_{I_t,T_t} [\log D(I_t, T_t)] + \mathbb{E}_{I_t,S} [\log \{1 - D(I_t, G(I_t, S))\}],
$$

where $G$ is the whole generation module, $D$ is the discriminator, $S$ is the style condition($S_c, S_m$), $T_t$ is the true title image, and $I_t$ is the input title text.

### 3.7 Title image generation model

As we have explained, the proposed network can generate title text suitable for a style. As a whole, the Text Encoder receives an input text image $I_t$ and the Context Encoder receives a style image $(S_c, S_m)$, and the Skeleton Generator
outputs a skeleton map \( O_{\text{skeleton}} \) and the Text Generator outputs a title image \( O_t \). This process is shown in Fig. 2. The training is done with alternating adversarial training with the Discriminator and the rest of the modules in an end-to-end manner. The Text Generator, Skeleton Generator, Text Encoder, and Context Encoder are trained using a total loss \( L_{\text{total}} \) through the Text Generator, where

\[
L_{\text{total}} = w_1 L_{\text{reconst}} + w_2 L_{\text{skeleton}} + w_3 L_{\text{adversarial}} + w_4 L_{\text{content}} + w_5 L_{\text{style}}.
\]

Variables \( w_1 \) to \( w_5 \) are weights for each of the losses.

4 Experimental setup

4.1 Dataset

In the experiment, as shown in Fig. 3, to train the network, we need a combination of the full book cover image without text, a target location mask, a target plain input text, the original font, and the skeleton of the original words. Thus, we use a combination of the following datasets and pre-processing to construct the ground truth.

We used the Book Cover Dataset [12]. This dataset consists of 207,572 book cover images\(^1\). The size of the book cover image varies depending on the book cover, but for this experiment, we resize the images to 256×256 pixels in RGB.

To ensure that the generated style is only inferred by the book cover and not any existing text, we remove all of the text from the book covers before using them. To remove the text, we first use Character Region Awareness for Text Detection (CRAFT) [4] to detect the text, then cut out regions defined by the detected bounding-boxes with dilated with a 3×3 kernel. CRAFT is an effective text detection method that uses a U-Net-like structure to predict character proposals and uses the affinity between the detected characters to generate word-level bounding-box proposals. Then, Telea’s inpainting [23] is used to fill the removed text regions with the plausible background area. The result is images of book covers without the text.

For the title text, CRAFT is also used. The text regions found by CRAFT are recognized using the scene text recognition method that was proposed by

\(^1\) https://github.com/uchidalab/book-dataset
Using the text recognition method, we can extract and compare the text regions to the ground truth. Once the title text is found, we need to extract the text without the background, as shown in Fig. 4. We generate a text mask (Fig. 4b) using a character stroke separation network to perform this extraction. The character stroke separation network is based on pix2pix [11] and it is trained to generate the text mask based on the cropped detected text region of the title. By applying the text mask to the detected text, we can extract the title text without a background. The background is replaced with a background with pixel values (127.5, 127.5, 127.5). Since the inputs are normalized between [-1, 1], the background represents a “0” value. Moreover, a target location mask (Fig. 3b) is generated by masking the bounding-box of the title text. The plain text input is generated using the same text but in the font “Arial.” Finally, the skeleton of the text is obtained using the skeletonization method of Zhang et al. [31].

| Original Book Cover | Style Cover + Mask | Input Text | True Title | Generated Title |
|---------------------|--------------------|------------|------------|-----------------|
| ![Original Book Cover](image1.png) | ![Style Cover + Mask](image2.png) | LYNAGH | LYNAGH | Windows |
| Migraine | Migraine | Migraine | Migraine | Migraine |
| FINLAND | FINLAND | FINLAND | FINLAND | Windows |
| HOME | HOME | HOME | HOME | Windows |
| Windows | Windows | Windows | Windows | Windows |

| Font MSE | 5.4 \times 10^{-4} | 4.7 \times 10^{-6} | 2.8 \times 10^{-4} | 1.2 \times 10^{-3} | 5.4 \times 10^{-7} |
| Color MSE | 6.0 \times 10^{-3} | 1.1 \times 10^{-3} | 4.0 \times 10^{-3} | 4.2 \times 10^{-2} | 3.7 \times 10^{-2} |

Fig. 5. Successful generation results.
4.2 Implementation details

In this experiment, for training the proposed model on high-quality data pairs, only images where the character recognition results [3] of the region detected by CRAFT (Fig. 4a) and the image created by the character stroke separation network (Fig. 4c) match were used. As a result, our proposed model has been trained on 195,922 title text images and a corresponding 104,925 book cover images. Also, 3,702 title text images and 1,000 accompanying book cover images were used for the evaluation. The training was done end-to-end using batch size 8 for 300,000 iterations. We used Adam [15] as optimizer, and set the learning coefficient $\eta = 2 \times 10^{-4}$, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. We also set $w_1 = 1$, $w_2 = 1$, $w_3 = 1.0 \times 10^{-2}$, $w_4 = 1$, and $w_5 = 1.0 \times 10^3$ for the weighting factors of the losses. The weights are set so that the scale of each loss value is similar. For $w_3$, we set a smaller value so that Discriminator does not have too much influence on the generation module’s training and can generate natural and clear images [16].

4.3 Evaluation metrics

For quantitative evaluation of the generate title images, we use standard metrics and non-standard metrics. For the standard metrics, Mean Absolute Error (MAE), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity (SSIM) [27] are used. We introduce two non-standard metrics for our title generation task, Font-vector Mean Square Error (Font MSE) and Color Mean Square Error (Color MSE). Font MSE evaluates the MSE between font style vectors of the original (i.e., ground-truth) and the generated title images. The font style vector is a six-dimensional vector of the likelihood of six font styles: serif, sans, hybrid, script, historical, and fancy. The font style vector is estimated by a CNN trained with text images generated by SynthText [8] with 1811 different fonts. Color MSE evaluates the MSE between three-dimensional RGB color vectors of the original and the generated title images. The color vector is given as the average color of the stroke detected by the character stroke separation network. These evaluations are only used in the experiment where the target text is the same as the original text. It should be noted that the three standard evaluations MAE, PSNR, and SSIM, can only be used when the target text is the same as the original text. However, we can use Font MSE and Color MSE, even when the generated text is different because they measure qualities that are common to the generated text and the ground truth text.

5 Experimental results

5.1 Qualitative evaluation

This section discusses the relationship between the quality of the generated images and the book cover images by showing various samples generated by the network and the corresponding original book cover images. Fig. 5 shows the samples where the network generates a title image close to the ground truth.
Fig. 6. Failure results.

| Original Book Cover | Style Cover + Mask | Input Text | True Title | Generated Title |
|---------------------|--------------------|------------|------------|-----------------|
| ![Image](image1.jpg) | ![Image](image2.jpg) | ![Image](image3.jpg) | ![Image](image4.jpg) | ![Image](image5.jpg) |

| Font MSE | 1.3 x 10^{-3} | 2.9 x 10^{-4} | 2.0 x 10^{-3} | 3.3 x 10^{-1} | 3.3 x 10^{-1} |
| Color MSE | 6.9 x 10^{-2} | 3.4 x 10^{-1} | 1.9 x 10^{-1} | 4.6 x 10^{-2} | 8.1 x 10^{-3} |

Successfully, that is, with smaller Font MSE and Color MSE. From the figure, we can see that the strokes, ornaments, and the size of the text are reproduced. Especially, the first example shows the serif font is also reproducible even if the input text is always given as a sans-serif image.

Among the results in Fig. 5, the “Windows” example clearly demonstrates that the proposed method can predict the font and style of the text given the book cover and the target location mask. This is due to the book cover being a recognizable template from the “For Dummies” series in which other books with similar templates exist in the training data. The figure demonstrates that the proposed method effectively infers the font style from the book cover image based on context clues alone.

Fig. 6 shows examples where the generator could not successfully generate a title image close to the ground truth. The first, second, and third images in Fig. 6 show examples of poor coloring in particular. The fourth and fifth images
show examples where the proposed method could not predict the font shape. For the “SUGAR” and “BEN” images, there are no clues that the color should be red. In the “DEADLY” book cover image, one would expect light text on the dark background. However, the original book cover used dark text on a dark background. For the “Beetles” and “Furrow” examples, the fonts are highly stylized and difficult to predict.

Fig. 7 shows several additional results including successful and failure cases. Even in this difficult estimation task from a weak context, the proposed method gives a reasonable style for the title image. The serif for “Make” and the thick style for “GUIDEBOOK” are promising. We also observe that peculiar styles, such as very decorated fonts and vivid colors, are often difficult to recover from the weak context.

Finally, in Fig. 8, we show results of using the proposed method, but with text that is different from the ground truth. This figure demonstrates that we can produce any text in the predicted style.
5.2 Ablation study

To measure the importance of the components of the proposed method, quantitative and qualitative ablation studies are performed. The network of the proposed method consists of six modules and their associated loss functions. Therefore, we measure the effects of the components. All experiments are performed with the same settings and the same training data.

The following evaluations are performed:

- **Proposed**: The evaluation with the entire proposed model.
- **Baseline**: Training is performed only using the Text Encoder and Text Generator with the reconstruction loss $L_{\text{reconst}}$.
- **w/o Context Encoder**: The proposed method but without the Context Encoder. The results are expected to be poor because there is no information about the style to learn from.
- **w/o Skeleton Generator**: The proposed method but with no guidance from the Skeleton Generator and without the use of the skeleton loss $L_{\text{skeleton}}$.
- **w/o Discriminator**: The proposed method but without the Discriminator and the adversarial loss $L_{\text{adversarial}}$.
- **w/o Perception Network**: The proposed method but without the Perception Network and the associated losses $L_{\text{content}}$ and $L_{\text{style}}$.

The quantitative results of the ablation study are shown in Table 1. The results show that Proposed has the best results in all evaluation methods except one, Color MSE. For Color MSE, w/o Perception Network performed slightly better. This indicates that the color of text produced by the proposed method without the Perception Network was more similar to the ground truth. However,
Table 1. Quantitative evaluation results.

| Method                         | MAE   | PSNR | SSIM | Font MSE | Color MSE |
|-------------------------------|-------|------|------|----------|-----------|
| Baseline                      | 0.041 | 20.49| 0.870| 0.174    | 0.102     |
| Proposed                      | 0.035 | 21.58| 0.876| 0.062    | 0.064     |
| w/o Context Encoder           | 0.036 | 20.79| 0.872| 0.126    | 0.093     |
| w/o Discriminator             | 0.035 | 21.46| 0.814| 0.105    | 0.081     |
| w/o Perception Network        | 0.061 | 19.39| 0.874| 0.085    | 0.058     |
| w/o Skeleton Generator        | 0.035 | 21.09| 0.874| 0.112    | 0.080     |

Fig. 9. Results of the ablation study.

as shown in Fig. 9, the Perception Network is required to produce reasonable results. In the figure, the colors are brighter without the Perception Network, but there is also a significant amount of additional noise. This is reflected in the results for the other evaluation measures in Table 1.

Also from Fig. 9, it can be observed how important each module is to the proposed method. As seen in Table 1, the Font MSE and Color MSE are much larger for w/o Context Encoder than the proposed method. This is natural due to knowing the style information being provided to the network. There are no hints such as color, object, texture, etc. Thus, as shown in Fig. 9, w/o Context Encoder only generates a basic font with no color information. This also shows that the book cover image information is important in generating the title text. A similar trend can be seen with w/o Discriminator and w/o Skeleton Network. The results show that the Discriminator does improve the quality of the font and the Skeleton Network ensures the structure of the text is robust.
6 Conclusion

In this study, we proposed a method of generating the design of text based on context information, such as the location and surrounding image. Specifically, we generated automatic book titles for given book covers using a neural network. The generation of the title was achieved by extracting the features of the book cover and the input text with two encoders, respectively, and using a generator with skeletal information. In addition, an adversarial loss and perception network is trained simultaneously to refine the results. As a result, we succeeded in incorporating the implicit universality of the design of the book cover into the generation of the title text. We obtained excellent results quantitatively and qualitatively and the ablation study confirmed the effectiveness of the proposed method. The code can be found at https://github.com/Taylister/FontFits. In the future, we will pursue the incorporation of the text back onto the book cover.

Acknowledgement

This work was in part supported by MEXT-Japan (Grant No. J17H06100 and Grant No. J21K17808).

References

1. Atarsaikhan, G., Iwana, B.K., Uchida, S.: Guided neural style transfer for shape stylization. PLOS ONE 15(6), e0233489 (Jun 2020)
2. Azadi, S., Fisher, M., Kim, V., Wang, Z., Shechtman, E., Darrell, T.: Multi-content GAN for few-shot font style transfer. In: CVPR (2018)
3. Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S.J., Lee, H.: What is wrong with scene text recognition model comparisons? dataset and model analysis. In: ICCV (2019)
4. Baek, Y., Lee, B., Han, D., Yun, S., Lee, H.: Character region awareness for text detection. In: CVPR (2019)
5. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: CVPR (2009)
6. Gatys, L., Ecker, A., Bethge, M.: A neural algorithm of artistic style. J. Vis. 16(12), 326 (2016)
7. Gomez, R., Biten, A.F., Gomez, L., Gibert, J., Karatzas, D., Rusinol, M.: Selective style transfer for text. In: ICDAR (2019)
8. Gupta, A., Vedaldi, A., Zisserman, A.: Synthetic data for text localisation in natural images. In: CVPR (2016)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385 (2015)
10. Huang, Y., He, M., Jin, L., Wang, Y.: RD-GAN: Few/zero-shot chinese character style transfer via radical decomposition and rendering. In: ECCV (2020)
11. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2017)
12. Iwana, B.K., Raza Rizvi, S.T., Ahmed, S., Dengel, A., Uchida, S.: Judging a book by its cover. arXiv preprint arXiv:1610.09204 (2016)
13. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: ECCV (2016)
14. Jubert, R.: Typography and graphic design: from antiquity to the present. Flammarion (2007)
15. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
16. Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W.: Photo-realistic single image super-resolution using a generative adversarial network. In: CVPR (2017)
17. Li, J., Jia, J., Xu, D.: Unsupervised representation learning of image-based plant disease with deep convolutional generative adversarial networks. In: CCC (2018)
18. Lyu, P., Bai, X., Yao, C., Zhu, Z., Huang, T., Liu, W.: Auto-encoder guided GAN for chinese calligraphy synthesis. In: ICDAR (2017)
19. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: ICML (2013)
20. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 3DV (2016)
21. Narusawa, A., Shimoda, W., Yanai, K.: Font style transfer using neural style transfer and unsupervised cross-domain transfer. In: ACCV Workshops (2019)
22. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
23. Telea, A.: An image inpainting technique based on the fast marching method. J. Graphics Tools 9(1), 23–34 (2004)
24. Ter-Sarkisov, A.: Network of steel: Neural font style transfer from heavy metal to corporate logos. In: ICPRAM (2020)
25. Tschichold, J.: The new typography: A handbook for modern designers, vol. 8. University of California Press (1998)
26. Wang, W., Liu, J., Yang, S., Guo, Z.: Typography with decor: Intelligent text style transfer. In: CVPR (2019)
27. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. 13(4), 600–612 (2004)
28. Wu, L., Chen, X., Meng, L., Meng, X.: Multitask adversarial learning for chinese font style transfer. In: IJCNN (2020)
29. Wu, L., Zhang, C., Liu, J., Han, J., Liu, J., Ding, E., Bai, X.: Editing text in the wild. In: ACM ICM (2019)
30. Yang, X., He, D., Kifer, D., Giles, C.L.: A learning-based text synthesis engine for scene text detection. In: BMVC (2020)
31. Zhang, T., Suen, C.Y.: A fast parallel algorithm for thinning digital patterns. Commun. the ACM 27(3), 236–239 (1984)
32. Zhao, N., Cao, Y., Lau, R.W.: Modeling fonts in context: Font prediction on web designs. Computer Graphics Forum 37(7), 385–395 (2018)
33. Zhu, A., Lu, X., Bai, X., Uchida, S., Iwana, B.K., Xiong, S.: Few-shot text style transfer via deep feature similarity. IEEE Trans. Image Process. 29, 6932–6946 (2020)