CLIPFont: Text Guided Vector WordArt Generation

Yiren Song
songyiren@sjtu.edu.cn
Yuxuan Zhang
zyx153@sjtu.edu.cn
Shanghai Jiaotong University
Shanghai, China

Abstract

Font design is a repetitive job that requires specialized skills. Unlike the existing few-shot font generation methods, this paper proposes a zero-shot font generation method called CLIPFont for any language based on the CLIP model. The style of the font is controlled by the text description, and the skeleton of the font remains the same as the input reference font. CLIPFont optimizes the parameters of vector fonts by gradient descent and achieves artistic font generation by minimizing the directional distance between text description and font in the CLIP embedding space. CLIP recognition loss is proposed to keep the category of each character unchanged. The gradients computed on the rasterized images are returned to the vector parameter space utilizing a differentiable vector renderer. Experimental results and Turing tests demonstrate our method's state-of-the-art performance. Project page: https://github.com/songyiren98/CLIPFont

1 Introduction

The artistic font has become an integral part of visual media. However, it is difficult to design new fonts without relevant knowledge and skills. This task is relatively easy when the number of characters is limited (e.g., Latin characters), but designing CJK (Chinese, Japanese, and Korean) characters presents challenges due to a large number of character sets and the complexity of glyph components. Existing methods implement few-shot font generation, inferring the styles of other characters by looking at some cases. But State-of-the-art methods [2, 6, 8, 9, 25] still require hundreds of characters to be written, are still labor intensive, and the generation quality is often poorer than the input examples. To solve this problem, this paper proposes CLIPFont, a zero-shot font generation framework.

CLIP [21] is a pre-trained model obtained by contrastive learning on 400 million image-text pairs, which has inspired a series of text-to-image generation works [3, 4, 12, 22, 23]. The purpose of this paper is to empower the field of font design with CLIP. Suppose there is a calligrapher whose text has a unique feature A, and the text description of feature A may be "Scribbled handwriting". The text he wrote constitutes the distribution A'. The goal of the previous few-shot font generation methods is to estimate the distribution A' by sampling a small number of samples, which is a difficult ill-posed problem. Therefore, previous studies have used a number of techniques to help estimate the distribution A', such as breaking down

the character into strokes and then recombining [16]. Whereas CLIPFont models feature A directly in the CLIP embedding space via text descriptions and use feature A to guide the stylization of the input font. Specifically, the optimization process minimizes the directional distance of font and text description in the CLIP embedding space. CLIPFont uses only the CLIP model as supervision and can handle any language text and symbols without fine-tuning on any font dataset. In addition to generating monochrome fonts, an effective color gradient enhancement strategy is proposed to disassemble the input character image into several polygonal pieces and achieves high-quality WordArt generation by optimizing the polygon shape and color.

Characters can be regarded as a special graphic with a clear category. Therefore, balancing the stylized effect with the recognizability of the font is the key to the research. We propose CLIP recognition loss to keep character categories unchanged by comparing the high-dimensional semantic feature of characters before and after optimization. Unlike the classic style transfer task which focuses on color and texture change, when designing a new WordArt, the shape of each stroke and decor needs to be designed. So in the CLIPFont framework, the shape and color optimization of polygons is decoupled, allowing us to independently control color and shape changes by setting the size of the learning rate. The skeleton of the font is another key of font design. There are usually deformation and distortion problems of the skeleton in the previous font generation methods. In CLIPFont, we want the skeleton of the generated character to be consistent with the input character. Specifically, CLIPFont optimizes the boundaries and colors of each polygon through gradient descent, while the overall position of the stroke does not undergo large change. Besides, similarity loss and CLIP recognition loss also have an effect on keeping the skeleton unchanged. For languages with a large number of characters, CLIPFont can be automatically generated based on existing fonts, which will greatly reduce the workload of font designers and enable them to create more diverse and unique fonts.

Overall, our contributions are as follows:

1. We propose the CLIPFont framework, which implements text-driven zero-shot vector font design for the first time, and models stylized font generation as a parameter optimization problem.

2. This paper proposes an effective vector font enhancement strategy to generate WordArt with color and texture. The CLIP recognition loss is proposed to alleviate the deformation of characters that is not conducive to recognition during the optimization process.

3. CLIPFont only utilizes the CLIP pre-trained model as supervision, without training or fine-tuning on any dataset, and can handle any language words and symbols.

2 Related Work

2.1 Style transfer and text effects transfer

Style transfer aims to transform a content image by transferring the semantic texture of a style image. Gatys et al. [7] use a pre-trained VGG network to transfer the style texture by calculating the style loss that matches the Gram matrices of the content and style features. The style loss defined by the Gram matrix has become the standard for later work[10, 14, 15, 17, 18]. As to artistic text effect transfer, Yang et al. [27] first proposed a texture-synthesis-based non-parametric method for transferring text effects. Azadi et al. [1] propose a data-driven MC-GAN that can generate stylized texts given a few examples. Yang et al. [28]

achieve text effects transfer by training a network to accomplish both the objective of style transfer and style removal, so that it can learn to disentangle and recombine the content and style features of text effects images. Wang et al. [24] present a novel framework to stylize the text with exquisite decor. Yang et al. [26] propose a font stylization method, which uses a style image to control the font style. Different from the above methods CLIPFont implement WordArt generation in the vector domain. Besides, the result of CLIPFont is controlled by the text prompt.

2.2 CLIP-guided image generation and manipulation

Recently, great progress has been made in text-driven image generation and editing based on CLIP pre-trained models. CLIP model has two encoders, one for image and one for text, that can convert images and text into the same embedding space by contrastive learning. One direction of work uses CLIP's gradient to guide a GAN's generator or diffusion model[3]. Other methods utilize CLIP to guide the optimization of a latent code and manipulate a specific image. Gal et al. [5] propose a model modification method using text conditions only and modulating the trained model into a novel domain without additional training images. The directional CLIP loss stylegan-NADA propose is widely used in follow-up work [11, 12]. Kwon and Ye [12] implements text-driven style transfer for the first time which is closest in effect to our work. While CLIPFont is specifically designed for the task of font generation and focuses on solving the problem of vector font optimization.

2.3 Few-shot font generation

Type design, especially for CJK, is a repetitive task that requires specialized knowledge. Some early font generation methods [6, 8, 9, 25] train the cross-domain translators between different font styles. More advanced architectures such as DM-Font [2], LF-Font [19], MX-Font [20] propose to use structure-aware style representations and learn the localized component-wise style representations. [16] propose a self-supervised cross-modality pre-training strategy. The above method reduces the workload of the designer to a certain extent, but still requires hundreds of characters. Furthermore, the results of few-shot font generation methods prediction tend to be of much lower quality than the input used for training, since inferring unseen characters from a few visible characters is an ill-posed problem. In addition, most of the above-mentioned methods can only realize the generation of monochromatic fonts in a specific language instead of word art. This paper proposed a text-driven zero-shot font generation method for the first time, which can generate both monochrome and artistic fonts. The definition of style is no longer limited to manually collected datasets but is controlled by textual descriptions, enabling an almost infinite variety of possibilities.

3 Method

The goal of CLIPFont is to start from the input vector character image and optimize vector characters' parameters to match the given description prompt. The overall schematic of our method is shown in Fig. 1. CLIPFont iteratively synthesizes WordArt through gradient descent. Starting from the input vector character, the color and shape of polygons are optimized to best match the given textual prompt. Before being passed into the CLIP encoder, vector

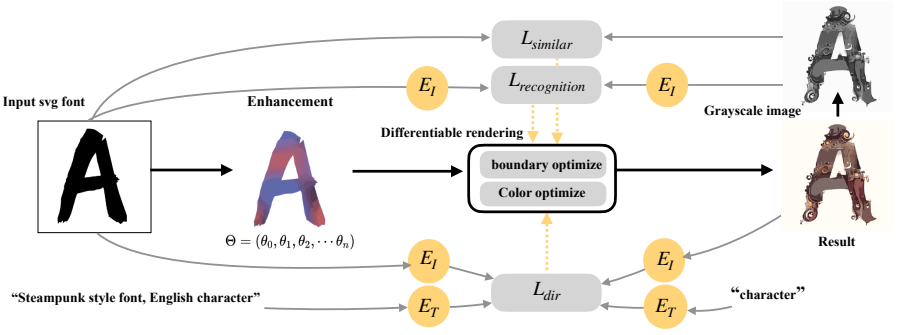

Figure 1: Overall schematics of CLIPFont. CLIPFont iteratively synthesizes stylized character through gradient descent. Starting from the input vector font, the color and boundary of polygons are optimized to best match the given textual prompt.

Figure 2: An example of the optimization process of CLIPFont. Input a vector font, after initialization and enhancement, CLIPFont optimizes the boundary and color of the vector font to gradually match the text prompt "English word, Steampunk."

character and prompt are enhanced. Differentiable vector rasterizer[13] is used to return the gradient computed on the rasterized image to the vector parameter space.

3.1 Vector font initialization and enhancement

Common fonts we use are stored in vector format for display at any scale without loss of quality. However, those fonts defined with very few control points are not suitable as the input of CLIPFont directly, because the number of control points to optimize is too small, resulting in lack of generation details. Therefore we adopt an enhancement strategy. Specifically, for each vector polygon, increase the number of control points on the boundary. In addition, vector polygon layers are stacked in the channel dimension to expand the parameter search space. Last but not the least, we apply a random color gradient to the input monochromatic font and divide it into multiple small polygons by color similarity.

3.2 Loss function

Compared to other types of designed graphics, font glyphs have distinct classes (i.e. each letter, number, symbol, etc. is a different class), and thus designing a generation procedure that is able to create clearly defined instances of different classes presents a challenge, as optimization process has to operate under strict constraints. The loss function CLIPFont use has three parts, which are CLIP recognition loss, directional CLIP loss, and similarity loss.

CLIP recognition loss. We hope that during the optimization process, the category of

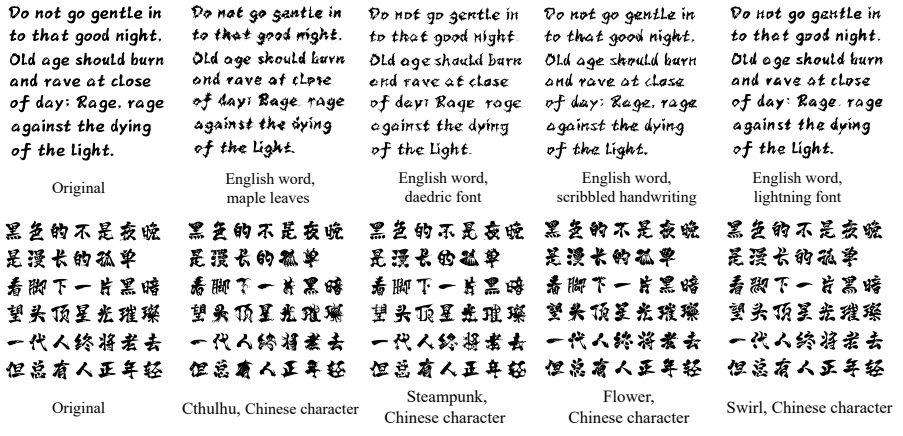

Figure 3: CLIPFont's monochrome font generation results. Input a font, CLIPVG optimizes the shape boundary of each character to match the text description.

the characters will not be changed. Therefore, we propose CLIP recognition loss, extract semantic features before and after optimization of individual characters, and compute high-dimensional semantic similarity. Before calculating the recognition loss, we grayscale the results, in order to avoid the effect of color change.

$$L_{recognition} = 1 - \frac{E_I\left(I_{\text{in}}\right) \cdot E_I\left(I_{\text{out}}\right)}{\left|E_I\left(I_{\text{in}}\right)\right| \left|E_I\left(I_{\text{out}}\right)\right|} \tag{1}$$

where I_{in} and I_{out} are the input vector image and result respectively.

Directional CLIP loss. StyleGAN-NADA proposed a directional CLIP loss that achieves robust semantic transfer, which aligns the CLIP-space direction between the text-image pairs of source and output. Directional CLIP loss has been used in lots of CLIP-based methods [11, 12]. We also apply the L_{dir}, which is defined as:

$$\begin{aligned} \Delta T &= E_T\left(t_{target}\right) - E_T\left(t_{src}\right) \\ \Delta I &= E_I\left(I_{out}\right) - E_I\left(I_{in}\right) \\ L_{dir} &= 1 - \tfrac{\Delta I \cdot \Delta T}{|\Delta I||\Delta T|} \end{aligned} \tag{2}$$

where E_T and E_I are the text and image encoders of CLIP, respectively. t_{target}, t_{src} are the description text of the effect we want and the text description of input content, respectively.

Similarity loss The $L_{similar}$ is used to measure the similarity between the generated result and the input font. $L_{similar}$ is defined as:

$$L_{similar} = \lambda_2 L_2 + \lambda_{lpips} L_{lpips} \tag{3}$$

where L_{lpips}[29] uses deep features to measure image similarity, and pays attention to differences in high-level features while L_2 monitors pixel-level differences. Similar to computing the recognition loss, we first grayscale the results, otherwise, the similarity loss will hinder positive color changes.

Total loss. CLIPFont uses three different losses as overall loss function. Firstly, we use $L_{recognition}$ to keep high-dimensional semantics and categories of characters unchanged.

Secondly, we apply $\lambda_d L_{dir}$ for texture generation and positive shape change. Finally, $L_{similar}$ can avoid results from becoming too cluttered. Therefore, our total loss function L_{total} is formulated as:

$$L_{total} = \lambda_r L_{recognition} + \lambda_d L_{dir} + L_{similar} \tag{4}$$

4 Experiments

4.1 Experimental setting

We describe the basic parameter settings of our model in this section.

Figure 4: CLIPFont's WordArt generation results. Input a font, CLIPFont optimizes the shape boundary of each character to match the text description and can handle any language characters.

Data augmentation. To improve the robustness of the feature embedding of the CLIP model, text enhancement and image enhancement are proposed. Before calculating the L_{dir} and $L_{recognition}$, the image of the font undergoes perspective transformation and random cropping. To reduce noise in text embedding, we also use a prompt engineering technique proposed by Radford et al. [21]. Specifically, several texts with the same meaning are made, such as connecting "a photo of" with the prompt and feeding them to the text encoder. Finally, we use the average embedding instead of the original single text condition.

Optimizer setting. Considering the memory capacity and time cost, we set the canvas to 1024 pixels and run for 150 iterations. CLIPFont optimizes the borders and colors of

Figure 5: The interpolation results between two different prompts.

Figure 6: Stylized results of different input fonts. CLIPFont can well maintain the skeleton characteristics.

the polygons that make up the character. Learning rates are important parameters that determines how drastically the generated results change. We set 0.2 for control points and 0.01 for color. CLIPFont can generate stylized characters one by one, or generate images containing multiple characters at one time through random sampling. Regarding the superposition enhancement strategy, CLIPFont superimposes the vector font enhanced by the color gradient three times along the channel dimension as input. Training time per image is less than one minute on an RTX2080Ti GPU.

4.2 WordArt and monochrome font generation results

The generation result of CLIPFont is shown in Fig. 2 and Fig.3. When the enhancement and color optimization are canceled, a monochrome font can be obtained. The contours of the characters produce positive changes to fit the text description, making it semantically close to the prompt. From the results, our font generation result is more like a decor style based on the input font. Different from the previous method, CLIPFont can handle arbitrary languages and symbols. Fig.4 shows the results of the interpolation between two different control texts, meaning we can have finer control over the results with different prompts and weights.Due to the flexibility of natural language, we can obtain an infinite variety of stylized fonts through a variety of text differences. Fig.5 shows the results of different languages and different input fonts. CLIPFont can well maintain the skeleton characteristics of the input font. Fig.6 shows the results of different languages and different input fonts, CLIPFont can well maintain the skeleton characteristics of the input font. Besides, we can choose whether to optimize the background color or not.

Figure 7: Ablation study results. Used prompts are "English word, Cthulhu" and "Cthulth, Chinese character" for top and bottom, respectively. (a) Original font. (b) When we apply all loss fuction and enhancement, we can get the best result. Lines annotated with alphabets are ablation study results of (c) CLIP recognition loss, (d) CLIP dir loss, (e) similar loss, (f) vector font enhancement.

4.3 Ablation study

To verify the necessity of each component in our method, we performed an ablation study as Fig.7 shows. We can obtain the best results when we use all the proposed loss functions and enhancement strategies. When L_{dir} is removed, the model loses stylization ability. When removing $L_{regognition}$, characters in the result are hard to identify, such as the letter G, N, Q, and S. When $L_{similar}$ is removed, the results become cluttered and illegible. The last column shows the effect of removing the overlay enhancement strategy, resulting in a lack of detail and texture.

4.4 Comparison with baseline

Because CLIPFont is the first zero-shot font generation method, it is quite different from existing font generation in principle and effect, so a fair comparison cannot be made. Therefore, in this section, we compare CLIPFont with the state-of-art method CLIPstyler[12], which is a CLIP-guided style transfer method and can be applied to the font generation task as well. As shown in Fig.8, CLIPstyler only changes the background's color and generates limited texture. While CLIPFont has the following advantages: (1) the results are in vector format and can be arbitrarily scaled without loss of detail, (2) the character produces a shape change that matches the text description, (3) the result has more texture and detail, (4) the method can control whether the background changes or not.

4.5 Turing experiment and user study

To demonstrate the effectiveness of our method, we conduct a Turing experiment and user study. We mixed 20 artworks created by CLIPFont with 20 artworks by humans and invited 50 volunteers to identify whether each one was created by a human or not. All volunteers have not seen the generation results of CLIPFont before. The average accuracy rate is only 56 percent, which means that the WordArt created by CLIPFont has a quality comparable to that of a human artist. Similarly, we also conducted the Turing experiment on monochrome fonts,

Figure 8: Compare experimental results. CLIPstyler is the state-of-the-art CLIP-guided image stylization method.

but the participants' accuracy rate was 68 percent, which means that the monochrome font results generated by CLIPFont are still have some different with that of human designers. In user study, we asked 50 volunteers to choose the preferred generation results, and the rate of liking CLIPFont was as high as 94%.

5 Limitations

CLIPFont is implemented based on diffvg [13], and can only optimize continuously changing control point coordinates, color transparency, etc. It cannot generate discrete decisions such as adding and removing graphs, and key points. What's more, textual guidance is also inherently limited by the ambiguity of natural language prompts. In addition, CLIPFont relies on the pre-trained models obtained from 400 million image-text pairs, so it may learn the bias and discrimination contained in the dataset.

6 Conclusion

This paper proposes a vector WordArt generation framework CLIPFont, which optimizes the directional CLIP loss between vector fonts and text descriptions through gradient descent to achieve WordArt and monochrome font generation. It greatly reduces the workload of font designers and achieves state-of-art results. CLIPFont does not need to be trained or fine-tuned on any font dataset, relies only on CLIP supervision, and can handle arbitrary languages or symbols. Ablation experiments, Turing study and user study demonstrate the effectiveness of the method proposed in this paper.

References

[1] Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. Multi-content gan for few-shot font style transfer. In *Proceedings*

of the IEEE conference on computer vision and pattern recognition, pages 7564–7573, 2018.

[2] Junbum Cha, Sanghyuk Chun, Gayoung Lee, Bado Lee, Seonghyeon Kim, and Hwalsuk Lee. Few-shot compositional font generation with dual memory. In *European Conference on Computer Vision*, pages 735–751. Springer, 2020.

[3] Katherine Crowson. Vqgan + clip. https://colab.research.google.com/drive/1L8oL-vLJXVcRzCFbPwOoMkPKJ8-aYdPN. 2021.

[4] Kevin Frans, LB Soros, and Olaf Witkowski. Clipdraw: Exploring text-to-drawing synthesis through language-image encoders. *arXiv preprint arXiv:2106.14843*, 2021.

[5] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021.

[6] Yiming Gao and Jiangqin Wu. Gan-based unpaired chinese character image translation via skeleton transformation and stroke rendering. In *National Conference on Artificial Intelligence*, 2020.

[7] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[8] Yaoxiong Huang, Mengchao He, Lianwen Jin, and Yongpan Wang. Rd-gan: Few/zero-shot chinese character style transfer via radical decomposition and rendering. In *European Conference on Computer Vision*, 2020.

[9] Yue Jiang, Zhouhui Lian, Yingmin Tang, and Jianguo Xiao. Scfont: Structure-guided chinese font generation via deep stacked networks. pages 4015–4022, 2019.

[10] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[11] G. Kim, T. Kwon, and J. C. Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. 2021.

[12] Gihyun Kwon and Jong Chul Ye. Clipstyler: Image style transfer with a single text condition. *arXiv preprint arXiv:2112.00374*, 2021.

[13] Tzu-Mao Li, Michal Lukáč, Michaël Gharbi, and Jonathan Ragan-Kelley. Differentiable vector graphics rasterization for editing and learning. *ACM Transactions on Graphics (TOG)*, 39(6):1–15, 2020.

[14] Xueting Li, Sifei Liu, Jan Kautz, and Ming-Hsuan Yang. Learning linear transformations for fast image and video style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3809–3817, 2019.

[15] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6649–6658, 2021.

[16] Wei Liu, Fangyue Liu, Fei Ding, Qian He, and Zili Yi. 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*, pages 7905–7914, 2022.

[17] Zhi-Song Liu, Li-Wen Wang, Wan-Chi Siu, and Vicky Kalogeiton. Name your style: An arbitrary artist-aware image style transfer. *arXiv preprint arXiv:2202.13562*, 2022.

[18] Dae Young Park and Kwang Hee Lee. Arbitrary style transfer with style-attentional networks. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5880–5888, 2019.

[19] Song Park, Sanghyuk Chun, Junbum Cha, Bado Lee, and Hyunjung Shim. Few-shot font generation with localized style representations and factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 2393–2402, 2021.

[20] Song Park, Sanghyuk Chun, Junbum Cha, Bado Lee, and Hyunjung Shim. Multiple heads are better than one: Few-shot font generation with multiple localized experts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13900–13909, 2021.

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[22] Yingtao Tian and David Ha. Modern evolution strategies for creativity: Fitting concrete images and abstract concepts. *arXiv preprint arXiv:2109.08857*, 2021.

[23] Yael Vinker, Ehsan Pajouheshgar, Jessica Y Bo, Roman Christian Bachmann, Amit Haim Bermano, Daniel Cohen-Or, Amir Zamir, and Ariel Shamir. Clipasso: Semantically-aware object sketching. *arXiv preprint arXiv:2202.05822*, 2022.

[24] Wenjing Wang, Jiaying Liu, Shuai Yang, and Zongming Guo. Typography with decor: Intelligent text style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5889–5897, 2019.

[25] S. J. Wu, C. Y. Yang, and Y. J. Hsu. Calligan: Style and structure-aware chinese calligraphy character generator. 2020.

[26] S. Yang, J. Liu, W. Yang, and Z. Guo. Context-aware text-based binary image stylization and synthesis. *IEEE Transactions on Image Processing*, pages 1–1, 2018.

[27] Shuai Yang, Jiaying Liu, Zhouhui Lian, and Zongming Guo. Awesome typography: Statistics-based text effects transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7464–7473, 2017.

[28] Shuai Yang, Jiaying Liu, Wenjing Wang, and Zongming Guo. Tet-gan: Text effects transfer via stylization and destylization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1238–1245, 2019.

[29] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.