Construction of Scene Tibetan Dataset Based on GAN

Guowei Zhang¹, Weilan Wang¹*, Penghai Zhao¹ and Jincheng Li¹

¹Key Laboratory of China's Ethnic Languages and Information Technology of Ministry of Education, Lanzhou, Northwest Minzu University, Gansu, 730030, China

*Corresponding author’s e-mail: wangweilan@xbmu.edu.cn

Abstract. For the research of Tibetan scene text detection and recognition, it is time-consuming and laborious to collect and annotate natural scene data labels manually. Therefore, artificial synthetic data is of great significance for promoting relevant work. This paper focuses on the study of replacing other languages in the scene with Tibetan, while maintaining the style of the original text. We decompose the problem into three sub-networks: text style transfer network, background inpainting network and fusion network. Firstly, the text in the foreground image is replaced by the text style transfer network to generate the foreground image. Then the background inpainting network erases the original text in the style image, and uses the surrounding information around the text to fill the text area to generate the background image. Finally, the generated foreground image and background image are used to generate the target image by the fusion network. We experimented with conversions from English to Tibetan and English to English to verify the generalization and robustness of the network. Experimental results show that its accuracy (SSIM, PSNR) on some datasets (SVT, ICDAR 2013) has been improved to some extent.

1. Introduction

As the most direct representation of human high-level semantic information, text plays an indispensable and important role in image understanding. In the past decades, the scene text detection algorithm based on deep learning has been greatly developed, and the detection accuracy has been greatly improved compared with traditional methods. The success of deep learning in computer vision can be attributed to the following three aspects: (a) high-capacity models; (b) rapidly increasing computing power; (c) the availability of large-scale labeled data[1]. With the rapid development and popularization of high-performance mobile and wearable devices, scene text detection and recognition technology has become a research hotspot in the field of computer vision. Compared with document OCR, scene text detection and recognition still have big problems, such as the diversity of text, complex and changeable image background, and interference from external factors (uneven illumination, low resolution, etc.). At present, the scene text datasets published by academic circles (such as ICDAR2013, 2015, CTW, etc.) are mostly limited to mainstream characters (English, Chinese, etc.), and there are few datasets for the detection and recognition of ethnic minority languages and multilinguals. Therefore, the sourced scene text datasets of ethnic minority languages and multiliguals has high research value and significance.

Scene text detection and recognition is a challenging but very useful task. In the era of data-driven deep learning, the size of the datasets will directly determine the performance of the model. In recent years, more and more text image synthesis methods have been used for the task of scene text detection and recognition. These methods combine different rendering algorithms for modeling to create images
with text from images without text. However, if the synthesized image cannot be perfectly integrated with the image in the real scene, this will affect the training accuracy of the subsequent model.

This paper focuses on the replacement of Tibetan in realistic scenes. The task of replacing Tibetan text in natural scenes can be expressed as: replace text present in an image with arbitrary user-supplied Tibetan text in a way that will not allow distinguishing the resulting image from the original one. Therefore our goal is to perform text replacement while retaining the original image text font, color, size and image background, so that the resulting image has a more realistic visual effect. The main challenge is to transfer the original image text style and background restoration. This paper replace the scene text through GAN on the basis of SRNet[2]. By adopting a divide-and-conquer strategy, the problem is decomposed into three sub-networks: text style transfer network, background inpainting network, and fusion network. Each sub-network can be addressed by using an encoder-decoder structure. In the text style transfer network, two images (text image and style image) will be prepared, and the features of these two images will be extracted forward through the encoder mechanism, and the context information will be learned through SE-ReNet[3].The introduction of a skeleton-based learning mechanism makes the text stroke structure in the resulting image more complete. But unfortunately, when the generator produces a series of bad skeleton generations, however instead of bouncing back, it grows worse and finally leads to mode collapse. We use two methods to solve this problem: 1) use a binary style image corresponding to the text image to guide the generation of the resulting image; 2) keep the discriminator ahead of the generator. Then, in the background inpainting network, we follow the idea of the U-net[4]network to erase the text area on the style image and fill it with appropriate texture. The fusion network is responsible for organically fusing the generated foreground and background to generate the final result image. Since there are no paired datasets in real scenes, this article uses synthetic data for training and tests on real scene images.

2. Related work
In recent years, text detection and recognition tasks made great leaps in the recovery depth neural network technology, but these models also bring many challenges. On the one hand, massive amounts of data are required to effectively train; on the other hand, the cost of data acquisition is very high and sometimes quite difficult. Therefore, in scene text tasks, data synthesis and data synthesis algorithms play an important role. At the same time, data synthesis can also provide detailed label information that a large amount of real data does not have due to cost issues.

2.1. GAN
GAN[5] was proposed by Ian Goodfellow in 2014. It solves the problem of unsupervised learning by training two neural networks. These two competing networks are called a generator (G) network and a discriminator (D) network. The purpose is to map random noise to samples and distinguish between real samples and generated samples. Formally speaking, the formal form of the GAN goal involves finding the Nash equilibrium of the min-max problem of G and D:

\[
\min_G \max_D \mathbb{E}_{x \sim q_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p(z)}[\log (1 - D(G(z)))]
\]  

(1)

GAN has a large number of practical use cases, such as image generation[6][7], artwork generation[8] and video generation[9]. In addition, it can also through adversarial training[10] to improve image quality, style transfer[11] image restoration[12] and other interesting tasks. CGAN[13] improved the original GAN model, turning the unsupervised GAN into a conditionally generative adversarial model. DCGAN[14] proposed an important architecture change to solve the problems of training instability, mode collapse and internal covariate conversion. It is the first time that Convolutional Neural Network (CNN) has been introduced into GAN and achieved very good results. StackGAN[15] is used to explore the conversion of text to image and create the internal representation of text description. StyleGAN[16] draws on style transfer and proposes a style-based generator to achieve better decoupling of latent space to generate high-definition (1024*1024) vivid face images.
2.2. Image Synthesis & Text Style Transfer
Image synthesis has been extensively studied in computer vision research, in which text image synthesis has been studied as a method of image data enhancement to train accurate and robust DNN models. Jaderberg[17]proposed a method of synthesizing natural scene text, and used the large-scale realistic data synthesized by this method to train DNN models and perform scene text detection tasks. Long[18] render images with the help of the UE4 which renders text and scene as a whole and allows for better text area proposal and access to accurate scene information. Style transfer[19] generates an output image with original content but new style by capturing the visual style of the input image and the visual content of the style image. MC-GAN[20]proposed a two-stage architecture by extending the traditional CGAN network. This is the first time that an end-to-end network has been used to solve the problem of font style transfer. Wang[21]proposed a method based on deep learning for the task of glyph synthesis, using font attributes and the values to create corresponding texts.

3. Methodology
During the training, we take the content image $I_c$ and the style image $I_s$ as the input of the network, and expect the resulting image $O_t$ — the replaced text image should have the same background and foreground styles as the text existing in the style image. As shown in Figure 1, the network consists of three parts: text style transfer network, background inpainting network, and fusion network. Each subnetwork adopts an encoder-decoder structure: the encoder layer used to extract the feature, and the decoder layer is used to restore to the original image size. The text style transfer network extracts the text features of $I_c$ and the style features of $I_s$ and integrates them together to get the foreground image. The background inpainting network erases the text information in $I_s$ by means of UNet-like method and fills in the appropriate texture to get the restored background image $O_b$. The fusion network gradually and seamlessly integrates the foreground image and the background image together to get the final output $O_t$.

3.1. Text Style Transfer Network
This part is responsible for the style transfer of the foreground text, such as text color, font, shape and other information. Our goal is to generate an image with specified text content and also containing text style features in the style image. Firstly, $I_c$ and $I_s$ are encoded through several subsampling convolution layers. In order to fully combine style and content feature, three SE-ResNet blocks are added after the three-layer convolution, whose architecture is illustrated in Table 1. In the decoder
stage, three up-sampled transpose convolution layers are used to generate the foreground image $O_f$. In this process, incomplete text structure and broken strokes often occur, leading to ambiguous semantic information of text. We adopt the method of [2] to introduce $I_c$ corresponding skeleton diagram $I_{sk}$ into network to train the generation of foreground image. Dice Loss is used to optimize the skeleton branch, and its loss function can be expressed as:

$$L_{sk} = 1 - \frac{2\sum_{i=1}^{N}(O_{sk})_i(O_{sk})_i}{\sum_{i=1}^{N}(I_{sk})_i + \sum_{i=1}^{N}(O_{sk})_i}$$  \hspace{1cm} (2)$$

where $N$ is the number of pixel; $I_{sk}$ is the skeleton ground truth; $O_{sk}$ is the predicted output of the skeleton branch.

SE-ResNet Module: In the real scene, there are not only texts arranged horizontally but also a lot of curving texts. Since the output of the convolutional layer does not consider the dependence on each channel, in order to better learn the position information of the text in the style image, the SE block (the module as shown in figure 2) is added on the basis of the residual block to allow the network to selectively enhance the features with large amounts of information, so that subsequent processing can make full use of these features, and useless features are suppressed. According to our experiments, this module significantly improves the image generation effect, especially for images with curved text.

We found that when the network generated a series of bad skeleton images, the network did not optimize them, but made the resulting images worse. In order to prevent the network from generating a series of bad skeletons and causing the model to collapse, we use the binary style image $I_{bi}$ corresponding to $I_c$ to guide the training of the skeleton image. We define $L_{bi}$ on the basis of $L1$ loss to optimize this branch, which can be expressed as:

$$L_{bi} = \frac{1}{\|I_f - O_f\|_1} + \frac{1}{\|I_f - O_f\|_1}$$  \hspace{1cm} (3)$$

$$L_{bi} = \frac{1}{\sum_{i=1}^{N}(I_{bi}) + \sum_{i=1}^{N}(O_{bi})(L_{bi} - 1)}$$  \hspace{1cm} (4)$$

where $I_f$ is the ground truth of the foreground text image after style transfer; $O_f$ is the foreground image predicted by the text style transfer network; $I_{bi}$ the ground truth of the binary style image, $(I_{bi})_{m \times n}$ is the number of pixels in the binary image.

The loss of the text style transfer network can be defined as:

$$L_t = \|I_{sc} - G_t(I_c, I_f)\|_1 + (L_{sk} + L_{bi})$$  \hspace{1cm} (5)$$

where $I_{sc}$ is the ground truth of the text style transfer network; $G_t$ represents the text style transfer network.

Table 1. The encoder architecture of text transfer network

| Layer   | Type               | Parameter                                      |
|---------|--------------------|------------------------------------------------|
| Layer_1 | Conv_BN_ReLu       | Kernel_size=3, Stride=1, Channels=32, BN, ReLu |
|         | Strided_Conv       | Kernel_size=3, Stride=2, Channels=64           |
| Layer_2 | Conv               | Kernel_size=3, Stride=1, Channels=64           |
|         | Conv_BN_ReLu       | Kernel_size=3, Stride=1, Channels=64, BN, ReLu |
|         | Strided_Conv       | Kernel_size=3, Stride=2, Channels=128          |
| Layer_3 | Conv               | Kernel_size=3, Stride=1, Channels=128          |
|         | Conv_BN_ReLu       | Kernel_size=3, Stride=1, Channels=128, BN, ReLu |
|         | SE-ResNet          | SE-ResNet block                                |
| Layer_4 | SE-ResNet          | SE-ResNet block                                |
|         | Strided_Conv       | Kernel_size=3, Stride=2, Channels=256          |
| Layer_5 | Conv               | Kernel_size=3, Stride=1, Channels=256          |
|         | Conv_BN_ReLu       | Kernel_size=3, Stride=1, Channels=256, BN, ReLu |
3.2. Background Inpainting Network

In addition to the text style of learning and transfer, the background inpainting network is also crucial for the final image fusion. The UNet-like structure is used to erase the text in the style image to generate a more realistic natural scene image. We take $I_g$ as the input of the background inpainting module, and the text is erased through the encoder-decoder framework. The three-layer dilate convolution is added after the encoder to capture multi-scale context information and we set the dilation rates to 2, 4, and 8 respectively. In addition, in the process of up-sampling at each level of the network, the feature map of the corresponding position of the coder layer is connected on the channel by using the skip-connection. At the same time, the features of the decoding layer are input to the subsequent decoding stage of the fusion module to assist the fusion process and effectively improve the background blur and virtual conditions. A successive convolution layer can then learn to assemble a more precise output based on this information. Through the fusion of low-level features and high-level features, the network can retain more high-resolution detailed information contained in high-level feature maps and improve image restoration quality.

Use $L_1$ loss and GAN loss to optimize the network, and use $G_b$ and $D_b$ to represent the generator and discriminator of the background inpainting network respectively:

$$L_B = \mathbb{E}[\log D_b(I_b, I_s)] + \log(1 - D_b(O_b, I_s)) + \gamma_b ||I_b - O_b||_1$$

(6)

where $I_b$ and $O_b$ is the ground truth of the background image and the predicted one respectively; $\gamma_b$ is the balance factor set to 10.

3.3. Fusion Network

The fusion network aims to fuse the output obtained from the first two modules to generate a complete text image. As shown in Figure 1, this part of the network also follows the encoder-decoder structure, which combines the encoding features of the foreground image with the features of the decoding stage of the background inpainting module, so that the foreground and background can be seamlessly combined in an appropriate and progressive way. We use $G_f$ and $D_f$ to represent the generator and discriminator of the fusion network respectively, and the loss is defined as:

$$L_F = \mathbb{E}[\log D_f(I_f, I_c)] + \log(1 - D_f(O_f, I_c)) + \gamma_f ||I_f - O_f||_1$$

(7)

where $I_f$ ground truth of the resulting image; $O_f$ is the network prediction image; the balance factor $\gamma_f$ is set to 10

To generate more realistic scene images, we also introduce style loss[19] and perceptual loss[22] into the fusion module, and use the pre-trained VGG-19 model to extract image features, so we collectively refer to them as VGG-loss. The specific loss function is expressed as follows:

$$L_{VGG} = \eta_1 L_{per} + \eta_2 L_{style}$$

$$L_{per} = \mathbb{E}\left[\sum_i^i \frac{1}{M_i} \left\| \mathcal{D}_i(I_i) - \mathcal{D}_i(O_i) \right\|_1 \right]$$

(9)

$$L_{style} = \mathbb{E}\left[\left\| G_j^\varphi(I_i) - G_j^\varphi(O_i) \right\|_1 \right]$$

(10)

Where $\mathcal{D}_i$ is the activation map from relu1_1 to relu5_1 layer of VGG-19; $M_i$ is the element size of feature map obtained by the $i$-th layer. $G$ is the Gram matrix. $\eta_1$ and $\eta_2$ are the balance factors and set to 1 and 500 respectively. The whole framework of fusion model can present as:

$$L_F = L_F' + L_{VGG}$$

(11)
4. Experiments

4.1. Details
Unlike document OCR, the natural scene background is usually more complicated, and the text style is different, so the synthetic data in the training phase should be as close to the real scene as possible. We used 15 English fonts, 1 standard Tibetan font, and 1,000 background images to generate 600,000 training images and 3,000 test images. The Tibetan font was in himalaya. We adjust the size of the input image to 64×256 and the batch size is 16 and use the Adam optimizer to optimize the network. In reference[14] the method, the encoder stage we use strided convolutions for down-sampling, and the decoder stage uses transposed convolution for up-sampling.

4.2. Metrics
We use MSE, PSNR and SSIM three metrics to evaluate the model:

MSE (Mean Square Error): Also known as $l_2$ error, it can be used to evaluate the degree of data change. The smaller the value of MSE, the better the accuracy of the prediction model to describe the experimental data.

PSNR (Peak Signal-to-Noise Ratio): It is the most common and widely used objective evaluation index for images. It is based on the error between corresponding pixels, that is, based on error-sensitive image quality evaluation. However, many experimental results have shown that the PSNR score cannot be exactly the same as the visual quality seen by the human eye. It is possible that the higher PSNR looks worse than the lower PSNR.

SSIM (Structural Similarity): It evaluates the similarity of two images through three aspects: luminance, contrast, and structure. The larger the SSIM value, the smaller the image distortion.
4.3. Datasets

Evaluate our method on the following public datasets.

The Street View Text (SVT): It is harvested from Google Street View. Image text in this data exhibits high variability and often has low resolution. It contains 350 images, 249 test images (647 word images can be cropped), 101 training images (257 word images can be cropped).

SynthText: The dataset by the natural scene image including the word, whose main used for text detection in natural scenes, the dataset consists of 800,000 images composed of approximately 8,000,000 words synthesis examples.

ICDAR 2013: A total of 350 street scene images. Among them, 249 is a test image (you can cut out 647 word images). 101 images are training images (257 word images can be cut out).

4.4. Effectiveness of Model

In this section, in order to evaluate our method, the following text replacement method is selected to compare with our method: SRNet[2]. Figure 3 shows the replacement effect of adding SE-ResNet module and binary style image based on SRNet.

![Figure 3. Some specific results of ablation study](image)

We present the evaluation model in the text of the scene dataset on some of the sample results in figure 4 and 5. As can be seen from the figure, our model can replace the text in the style image while retaining the original font, color, size, and background.
4.5. Quantitative result
Table 2 shows the quantitative results of our method compared with SRNet. It can be seen from the table that our proposed method has greatly improved the indicators across different languages. For Tibetan, $l_2$ error decreased by 0.01, PSNR increased by 0.76, and SSIM increased by 0.04.

Table 2. Quantitative results on synthetic dataset

| Method       | Tibetan   | English   |
|--------------|-----------|-----------|
|              | $l_2$ | PSNR | SSIM | $l_2$ | PSNR | SSIM |
| SRNet        | 0.039 | 15.04 | 0.569 | 0.032 | 15.26 | 0.584 |
| w/o SE-ResNet| 0.035 | 15.16 | 0.581 | 0.028 | 15.44 | 0.607 |
| w/o binary image | 0.030 | 15.54 | 0.587 | 0.023 | 15.79 | 0.625 |
| ours         | 0.029 | 15.80 | 0.609 | 0.021 | 16.05 | 0.641 |

4.6. Limitation
Due to the limited amount of training data, the geometric attribute space and font space cannot be fully utilized. When the text in the style image is an artistic font or a more complex background, our method does not show good effects. Figure 6 shows Some cases where text replacement failed in real scenes.
5. Conclusion
This paper improves an end-to-end network based on SRNet to solve the task of text replacement in natural scenes. Using a divide-and-conquer strategy, the problem is decomposed into three sub-networks: the text style transfer network, the background inpainting network and the fusion network. Each sub-network uses a feedforward model. The text style transfer network and the fusion network are respectively composed of an encoder-decoder structure based on SE-ResNet block and Res block, and the background inpainting network is composed of an encoder-decoder structure based on dilated convolution. In the style transfer network, we introduce the SE-ResNet Module to better learn the spatial position information of the text. In order to make the text stroke structure in the generated image more complete, use the binary style image corresponding to $I_c$ to guide the training of the skeleton branch; in the background inpainting network, a network structure similar to U-Net is adopted, with the help of skip-connection and dilate convolution to better erase text information, and the features of the decoder stage are input to the decoding stage of the subsequent fusion network to assist the fusion process and effectively improve the background blur and virtual situation. Quantitative results on some public scene text datasets demonstrate the effectiveness of our method.

In future work, on the one hand, we will improve the Tibetan scene dataset, and on the other hand, we will explore more generalized and lighter text image synthesis methods to solve the task of text replacement in more complex scenes.

Acknowledgments
The authors acknowledge the National Natural Science Foundation of China (No.61772430, No.61375029), program for Leading Talent of State Ethnic Affairs Commission, supported by Program for Innovative Research Team of SEAC ([2018]98), and also supported by the Gansu Provincial first-class discipline program of Northwest Minzu University (No.11080305).

References
[1] Sun, C., Shrivastava, A., Singh, S., Gupta, A. (2017). Revisiting unreasonable effectiveness of data in deep learning era. In Proceedings of the IEEE international conference on computer vision. pp. 843-852.
[2] Wu, L., Zhang, C., Liu, J., Han, J., Liu, J., Ding, E., Bai, X. (2019, October). Editing text in the wild. In Proceedings of the 27th ACM international conference on multimedia. pp. 1500-1508.
[3] Hu, J., Shen, L., Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7132-7141.

[4] Ronneberger, O., Fischer, P., Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention. pp. 234-241. Springer, Cham.

[5] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets[C]//Advances in neural information processing systems. 2014: 2672-2680.

[6] Brock, A., Donahue, J., Simonyan, K. (2018). Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096.

[7] Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1125-1134.

[8] Gatys, L. A., Ecker, A. S., Bethge, M. (2015). A neural algorithm of artistic style. arXiv preprint arXiv:1508.06576.

[9] Chan, C., Ginosar, S., Zhou, T., Efros, A. A. (2019). Everybody dance now. In Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5933-5942.

[10] Xie, C., Tan, M., Gong, B., Wang, J., Yuille, A. L., Le, Q. V. (2020). Adversarial examples improve image recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 819-828.

[11] Zhu, J. Y., Park, T., Isola, P., Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision. pp. 2223-2232.

[12] Liu, G., Reda, F. A., Shih, K. J., Wang, T. C., Tao, A., Catanzaro, B. (2018). Image inpainting for irregular holes using partial convolutions. In Proceedings of the European Conference on Computer Vision (ECCV). pp. 85-100.

[13] Mirza, M., Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[14] Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.

[15] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D. N. (2017). Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In Proceedings of the IEEE international conference on computer vision. pp. 5907-5915.

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

[17] Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A. (2014). Synthetic data and artificial neural networks for natural scene text recognition. arXiv preprint arXiv:1406.2227.

[18] Long, S., Yao, C. (2020). Unrealtext: Synthesizing realistic scene text images from the unreal world. arXiv preprint 2003.10608.

[19] Gatys, L. A., Ecker, A. S., Bethge, M. (2016). Image style transfer using convolutional neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2414-2423.

[20] Azadi, S., Fisher, M., Kim, V. G., Wang, Z., Shechtman, E., Darrell, T. (2018). Multi-content gan for few-shot font style transfer. In Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7564-7573.

[21] Wang, Y., Gao, Y., Lian, Z. (2020). Attribute2Font: creating fonts you want from attributes. ACM Transactions on Graphics (TOG), 39(4), 69-1.

[22] Johnson, J., Alahi, A., Fei-Fei, L. (2016, October). Perceptual losses for real-time style transfer and super-resolution. In European conference on computer vision. pp. 694-711. Springer, Cham.
[23] Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.