Editing Text in the Wild

Liang Wu*
Huazhong University of Science and Technology
liangwu1995@gmail.com

Chengquan Zhang*
Department of Computer Vision Technology (VIS), Baidu Inc.
zhangchengquan@baidu.com

Jiaming Liu*
Department of Computer Vision Technology (VIS), Baidu Inc.
liujiaming03@baidu.com

Junyu Han
Department of Computer Vision Technology (VIS), Baidu Inc.
hanjunyu@baidu.com

Jingtuo Liu
Department of Computer Vision Technology (VIS), Baidu Inc.
liujingtuo@baidu.com

Errui Ding
Department of Computer Vision Technology (VIS), Baidu Inc.
dingerrui@baidu.com

Xiang Bai†
Huazhong University of Science and Technology
xbai@hust.edu.cn

ABSTRACT

In this paper, we are interested in editing text in natural images, which aims to replace or modify a word in the source image with another one while maintaining its realistic look. This task is challenging, as the styles of both background and text need to be preserved so that the edited image is visually indistinguishable from the source image. Specifically, we propose an end-to-end trainable style retention network (SRNet) that consists of three modules: text conversion module, background inpainting module and fusion module. The text conversion module changes the text content of the source image into the target text while keeping the original text style. The background inpainting module erases the original text, and fills the text region with appropriate texture. The fusion module combines the information from the two former modules, and generates the edited text images. To our knowledge, this work is the first attempt to edit text in natural images at the word level. Both visual effects and quantitative results on synthetic and real-world dataset (ICDAR 2013) fully confirm the importance and necessity of modular decomposition. We also conduct extensive experiments to validate the usefulness of our method in various real-world applications such as text image synthesis, augmented reality (AR) translation, information hiding, etc.

KEYWORDS

Text Editing; Text Synthesis; Text Erasure; GAN

*Equal contribution. This work was mainly done when Liang Wu was an intern at Baidu Inc.
†Corresponding author.

MM '19, October 21–25, 2019, Nice, France

ACM ISBN 978-1-4503-6889-6/19/10…$15.00
https://doi.org/10.1145/3343031.3350929

ACM Reference Format:
Liang Wu, Chengquan Zhang, Jiaming Liu, Junyu Han, Jingtuo Liu, Errui Ding, and Xiang Bai. 2019. Editing Text in the Wild. In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3343031.3350929

1 INTRODUCTION

Text in images/videos, or known as scene text, contains rich semantic information that is very useful in many multi-media applications. In the past decade, scene text reading and its application have witnessed significant progresses [4, 16, 26, 34, 38]. In this paper, we focus on a new task related to scene text: scene text editing. Given a text image, our goal is to replace the text instance in it without damaging its realistic look. As illustrated in Fig. 1 (*a*), the proposed scene text editor produces realistic text images by editing each word in the source image, retaining the styles of both the text and background. Editing scene text has drawn increasing attention from both academia and industry, driven by practical applications such as text image synthesis [33], advertising photo editing, text image correction, augmented reality translation [5].

As shown in Fig. 1 (*b*), there are two major challenges for scene text editing: text style transfer and background texture retention. Specially, the text style consists of diverse factors such as language, font, color, orientation, stroke size and spatial perspective, which makes it hard to precisely capture the complete text style in source image and transfer them to the target text. Meanwhile, it is also difficult to maintain the consistency of the edited background, especially when text appears on some complex scenes, such as menu and street store sign. Moreover, if the target text is shorter than the original text, the exceeding region of characters should be erased and filled with appropriate texture.

Considering these challenges, we propose a style retention network (SRNet) for scene text editing which learns from pairs of images. The core idea of SRNet is to decompose the complex task into several simpler, modular and joint-trainable sub networks: text conversion module, background inpainting module and fusion module, as illustrated in Fig. 2. Firstly, the text conversion module (TCM)

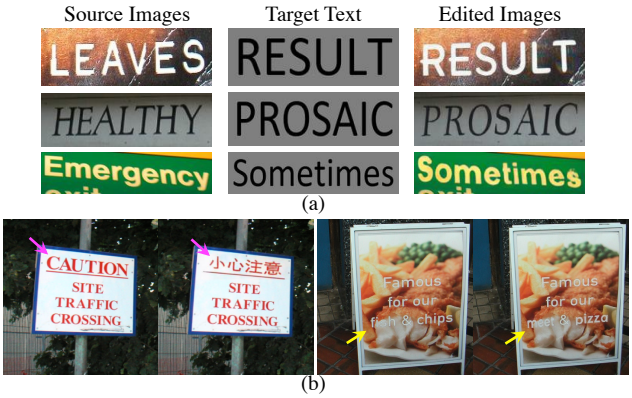

Figure 1: (a) The process of scene text editing. (b) Two challenges of text editing: rich text style and complex background.

transfers the text style of the source image to the target text, including font, color, position, and scale. In order to keep the semantics of the target text, we introduce a skeleton-guided learning mechanism to the TCM, whose effectiveness has been verified in Exp. 4.4. At the same time, the background inpainting module (BIM) erases the original text stroke pixels and fills them with appropriate texture in a bottom-up feature fusion manner, following the general architecture of a "U-Net" [23]. Finally, the fusion module automatically learns how to fuse foreground information and background texture information effectively, so as to synthesize edited text image.

Generative Adversarial Networks (GAN) models [7, 11, 40] have achieved great progress in some tasks, such as image-to-image translation, style transfer, these methods typically apply the encoder-decoder architecture that embeds the input into a subspace then decodes it to generate desired images. Instead of choosing such a single branch structure, the proposed SRNet decomposes the network into modular sub networks, while decomposes the complex task into several easy-to-learn tasks. This strategy of network decomposing has been proven useful in recent works [1, 3]. Besides, the experiment results of SRNet are better than pix2pix [11], a successful method used in image-to-image translation, which further confirms the effectiveness and robustness of SRNet. Compared with the work of character replacement [24], our methods works in a more efficient word-level editing way. In addition to the ability to edit scene text image in the same language (such as the English words on ICDAR 2013), SRNet also shows very encouraging results in cross-language text editing and information hiding tasks, as exhibited in Fig. 7, 8.

The major contribution of this paper is the style retention network (SRNet) proposed to edit scene text image. SRNet possesses obvious advantages over existing methods in several folds:

- To our knowledge, this work is the first to address the problem of word or text-line level scene text editing by an end-to-end trainable network;
- We decompose SRNet into several simple, modular and learnable modules, including a text conversion module, a background inpainting module and the final fusion module, which enables SRNet to generate more realistic results than most image-to-image translation GAN models;
- Under the guidance of stroke skeleton, the proposed network can keep the semantic information as much as possible;
- The proposed method exhibits superior performance on several scene text editing tasks like intra-language text image editing, AR translation (cross-language), information hiding (e.g. word-level text erasure), etc.

2 RELATED WORK

2.1 GAN

Recently, GANs [7] have attracted increasing attention and made great progress in many fields ,including generating images from noise [19], image-to-image translation [11], style transfer [40], pose transfer [41], etc. The framework of GANs consists of two modules: generator and discriminator, where the former aims to generate data close to the realistic distribution while the latter strives to learn how to distinguish between real and fake data. DCGAN [22] firstly used convolutional neural networks (CNN) as the structures of generator and discriminator, improved training stability of GAN. Conditional-GAN [19] generated the required images under the constraints of given conditions, and achieved significant results in pixel-level alignment image generation task. Pix2pix [11] implemented the mapping task from image to image, which was able to learn the mapping relationship between input domain and output domain. Cycle-GAN [40] accomplished the cross-domain conversion task under the unpaired style images while achieving excellent performance. However, existing GANs are difficult applied in text editing task directly, because the text content changes while the shape of text needs change greatly, and the complex background texture information also need to be preserved well when editing a scene text image.

2.2 Text Style Transfer

Maintaining the scene text style consistency before and after editing is extremely challenging. There are some efforts attempting to migrate or copy text style information from a given image or stylized text sample. Some methods focus on character-level style transfer, for example, Lyu *et al*. [17] proposed an auto-encoder guided GAN to synthesize calligraphy images with specified style from standard Chinese font images. Sun *et al*. [29] used a VAE structure to implement a stylized Chinese character generator. Zhang *et al*. [37] tried to learn the style transfer ability between Chinese characters at the stroke level. Other methods focus on text effects transfer, which can learn visual effects from any given scene image and bring huge commercial value in some specific applications like generating special-effects typography library. Yang *et al*. [31, 33]proposed a patch-based texture synthesis algorithm that can map the sub-effect patterns to the corresponding positions of the text skeleton to generate image blocks. It is worth noting that this method is based on the analysis of statistical information, which may be sensitive to glyph difference and thus induce a heavy computational burden. Recently, TET-GAN [32] used the GAN to design a light-weight framework that can simultaneously support stylization and destylization on a variety of text effects. Meanwhile, MC-GAN [2] used two sub-networks to solve English alphabet glyph transfer and effect transfer respectively, which accomplished the few-shot font style transfer task.

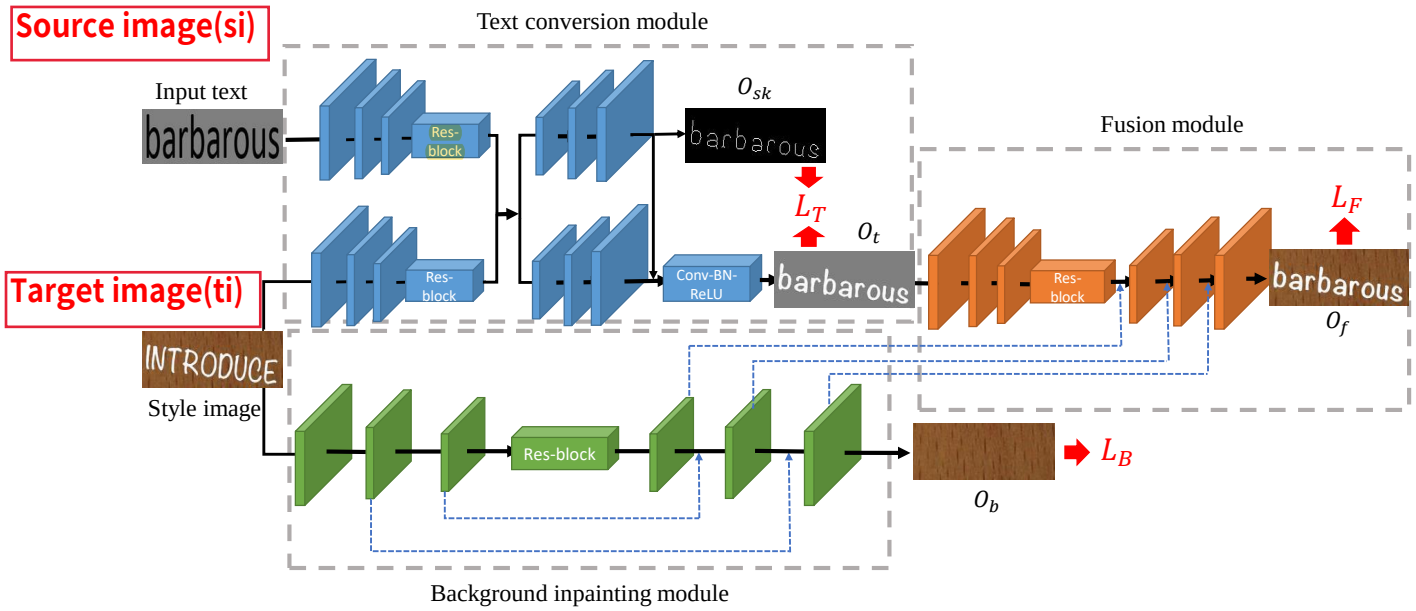

Figure 2: The overall structure of SRNet. The network consists of a skeleton-guided text conversion module, a background inpainting module and a fusion module.

Different from these existing methods, the proposed framework in this paper is trying to solve the migration problem of arbitrary text styles and special effects at a word or text-line level, rather than at the character level. In practice, word-level annotations are much easier to obtain than character-level annotations, and editing word is more efficient than editing characters. Besides, word-level editors favor word-level layout consistency. When dealing with words of different lengths, our word-level editor can adjust the placement of foreground characters adaptively, while character-level methods ignore.

2.3 Text Erasure and Editing

Background texture needs to be consistent with that before editing for scene text editing. There are some related works of text erasure, trying to erase the scene text stroke pixels while completing image inpainting on corresponding positions. Nakamura *et al*. [21] proposed an image-patch based framework for text erasure, but large computational cost is induced due to the sliding window based processing mechanism. EnsNet [35] firstly introduced the generative adversarial network to text erasing, which can erase the scene text on the whole image in an end-to-end manner. With the help of refined loss, the visualization results looks better than those of pix2pix [11]. Our background inpainting module is also inspired by generative adversarial networks. In the process of text editing, we only pay attention to background erasure at word-level, therefore, the background inpainting module in SRNet can be designed more light and still have good erasure performance which is illustrated in Fig. 8.

We noticed that a recent paper [24] try to study the issue of scene text editing, but it can only transfer the color and font of a single character in one process while ignoring the consistency of background texture. Our method integrates the advantages of the approaches of text style transfer and text erasing. We propose a style retention network which can not only transfer text style by an efficient manner (word or text-line level processing mechanism) but also retain or inpaint the complete background regions to make the result of scene text editing more realistic.

3 METHODOLOGY

We present a style retention network (SRNet) for scene text editing. During training, the SRNet takes as input a pair of images (I_s, I_t) where I_s is the source style image and I_t is the target text image. The outputs $((T_{sk}, T_t), T_b, T_f)$ where T_{sk} is the target text skeleton, T_t is the foreground image which has the same text style as I_s. T_b is the background of I_s and T_f is the final target text image. In order to effectively tackle the two major challenges mentioned in Sec. 1, we decompose the SRNet into three simpler and learnable sub networks: 1) text conversion module, 2) background inpainting module and 3) fusion module, as illustrated in Fig. 2. Specifically, the text style from source image I_s is transferred to the target text with the help of a skeleton-guided learning mechanism aiming to retain text semantics(Sec. 3.1). Meanwhile the background information is filled by learning an erasure or inpainting task (Sec. 3.2). Lastly, the transferred target image and completed background are fused by the text fusion network, generating the edited image (Sec. 3.3).

3.1 Text Conversion Module

We render the target text into a standard image with a fixed font and background pixel value setting to 127, and the rendered image is denoted as target text image I_t. The text conversion module (blue part in Fig. 2) takes the source image I_s and the target text image I_t as inputs, and aims to extract the foreground style from

Text Conversion Modules提取source image(si)的前景和target image(ti)的背景，输出带有si中文字的文本和ti中文字的风格

用的是FCN的encoder+decoder结构, si和ti分别输入后接3层下采样、4个residual blocks，两边concatenate，后接三层上采样（？），再接一层conv+bn+lRelu得到结果

the source image I_s and transfers it to the target text image I_t. In particular, the foreground style contains text style, including font, color, geometric deformation, and so on. Thus, the text conversion module outputs an image O_t which has the semantics of the target text and the text style of the source image. An encoder-decoder FCN is adopted in this work. For encoding, the source image I_s is encoded by 3 down-sampling convolutional layers and 4 residual blocks [9], the input text image I_t is also encoded by the same architecture, then two features are concatenated along their depth axis. For decoding, there are 3 up-sampling transposed convolutional layers and 1 Convolution-BatchNorm-LeakyReLU blocks to generate the output O_t. Moreover, we introduce a skeleton-guided learning mechanism to generate more robust text. We use G_T to denote the text conversion module and the output can be represented as:

$$O_t = G_T(I_t, I_s). \tag{1}$$

Skeleton-guided Learning Mechanism. Different from other natural objects, humans distinguish different texts mostly according to the skeleton or glyph of text. It is necessary to maintain the text skeleton in I_t after transferring the text style from source style image I_s. To achieve this, we introduce a skeleton-guided learning mechanism. Specifically, we add a skeleton response block which is composed of 3 up-sampling layers and 1 convolutional layer followed by a sigmoid activation function to predict a single channel skeleton map, and then concatenate the skeleton heatmap and decoder output along depth axis. We use the dice loss [18] instead of the cross-entropy loss to measure the reconstruction quality of the skeleton response map since it is found to yield more accurate results. Mathematically, the skeleton loss is defined as:

$$\mathcal{L}_{sk} = 1 - \frac{2\sum_i^N (T_{sk})_i (O_{sk})_i}{\sum_i^N (T_{sk})_i + \sum_i^N (O_{sk})_i}, \tag{2}$$

where N is the number of pixell; T_{sk} is the skeleton ground truth map; O_{sk} is output map of the skeleton module.

We further adopt the $L1$ loss to supervise the output of text conversion module. Combing with the skeleton loss, the text conversion loss is:

$$\mathcal{L}_T = \|T_t - O_t\|_1 + \alpha \mathcal{L}_{sk}, \tag{3}$$

where T_t is the ground truth of text conversion module, and α is regularization parameter, which is set to 1.0 in this paper.

3.2 Background Inpainting Module

In this module, our main goal is to obtain the background via a word-level erasure task. As depicted in the green part in Fig. 2, this module takes only the source image I_s as its input, and outputs a background image O_b, in which all text stroke pixels are erased and filled with proper texture. The input image is encoded by 3 down-sampling convolutional layers with stride 2 and follows with 4 residual blocks, then the decoder generates the output image with original size via 3 up-sampling convolutional layers. We use the leaky ReLU activation function after each layer while tanh function for the output layer. We denote the background generator as G_B. In order to make the visual effects more realistic, we need to restore the texture of background as much as possible. U-Net [23], which proposes to add skip connections between mirrored layers, proven remarkably effective and robust at solving object segmentation and image-to-image translation tasks. Here, we adopt this mechanism in the up-sampling process, where previous encoding feature maps with the same size are concatenated to reserve richer texture. This helps to restore the lost background information during the down-sampling process.

Different from other full text image erasure methods [21, 35], our method aims at word-level image inpainting task. Text appearing in word-level image tends to be relatively standard in scale, so our network structure has possesses simple and neat design. Inspired by the work of Zhang *et al*. [35], the adversarial learning is added to learn more realistic appearance. The detailed architecture of the background image discriminator D_B is described in Sec. 3.4. The whole loss function of background inpainting module is formulated as:

$$\mathcal{L}_B = \mathbb{E}_{(T_b, I_s)}[\log D_B(T_b, I_s)] + \mathbb{E}_{I_s} \log[1 - D_B(O_b, I_s)] + \beta \|T_b - O_b\|_1, \tag{4}$$

where T_b is the ground truth of background. The formula is combined by adversarial loss and $L1$ loss, and β is set to 10 in our experiments.

3.3 Fusion Module

The fusion module is designed to fuse the target text image and background texture information harmoniously, so as to synthesize edited scene text image. As the orange part illustrates in Fig. 2, the fusion model also follows the encoder-decoder FCN framework. We feed the foreground image, generated by text conversion module, to the encoder, which consists of three down-sampling convolutional layers and residual blocks. Next, a decoder with three up-sampling transposed convolutional layers and Convolution-BatchNorm-LeakyReLU blocks to generates the final edited image. It is noteworthy that we connect the decoding feature maps of the background inpainting module to the corresponding feature maps with the same resolution in the up-sampling phase of the fusion decoder. In this way, the fusion network outputs the images whose background details are substantially restored; text object and background are fused well while achieving synthesis realism in the appearance. We use G_F and O_f to denote the fusion generator and its outputs respectively. Besides, the adversarial loss is added here, and the detailed structure of the corresponding discriminator D_F will be introduced in Sec. 3.4. In summary, we can formulate the optimization objectives of the fusion module as the following:

$$\mathcal{L}'_F = \mathbb{E}_{(T_f, I_t)}[\log D_F(T_f, I_t)] + \mathbb{E}_{I_t} \log[1 - D_F(O_f, I_t)] + \theta_1 \|T_f - O_f\|_1, \tag{5}$$

where T_f is the ground truth of edited scene images. We choose $\theta_1 = 10$ to keep balance between adversarial loss and $L1$ loss.

VGG-Loss. In order to reduce distortions and make more realistic images, we introduce the VGG-loss to the fusion module that includes perceptual loss [13] and style loss [6]. As the name suggests, the perceptual loss L_{per} penalizes results that are not perceptually similar to labels by defining a distance measure between activation maps of a pre-trained network (we adopt the VGG-19 model [28] pretrained on ImageNet [25]). Meanwhile, the style loss L_{style} computes the differences in style. The VGG-loss L_{vgg} can

Figure 3: Some results on ICDAR2013 dataset. Images from left to right: input images and edited results. It should be noted that on the third row we replaced the words whose lengths is different from the original text; the last row shows some cases with long text.

be represented by:

$$\mathcal{L}_{vgg} = \theta_2 \mathcal{L}_{per} + \theta_3 \mathcal{L}_{style}, \tag{6}$$

$$\mathcal{L}_{per} = \mathbb{E}[\sum_i \frac{1}{M_i} \|\phi_i(T_f) - \phi_i(O_f)\|_1], \tag{7}$$

$$\mathcal{L}_{style} = \mathbb{E}_j[\|G_j^{\phi}(T_f) - G_j^{\phi}(O_f)\|_1], \tag{8}$$

where ϕ_i is the activation map from relu1_1, relu2_1, relu3_1, relu4_1 and relu5_1 layer of VGG-19 model; M_i is the element size of the feature map obtained by the $i-th$ layer; G is Gram matrix $G(F) = FF^T \in \mathbb{R}^{n \times n}$; the weights θ_2 and θ_3 set to 1 and 500 respectively. The whole training objectives of the fusion model is:

$$\mathcal{L}_F = \mathcal{L}'_F + \mathcal{L}_{vgg}. \tag{9}$$

3.4 Discriminators

Two discriminators sharing the same structure as PatchGAN[11] are applied in our network. They are composed of five convolution layers to reduce the scale to 1/16 of the original size. The discriminator D_B in background inpainting module concatenate I_s with O_b or T_b as input to judge whether the erased result O_b and the target background T_b is similar, while the discriminator D_F in fusion module concatenate I_t and O_f or T_f to measure the consistence between the final output O_f and the ground truth image T_f.

3.5 Training and Inference

In the training stage, the whole network is trained in an end-to-end manner, and the overall loss of the model is:

$$\mathcal{L}_G = \arg\min_G \max_{D_B, D_F} (\mathcal{L}_T + \mathcal{L}_B + \mathcal{L}_F), \tag{10}$$

Figure 4: Examples of synthetic data. From top to bottom: style image, target image, foreground text, text skeleton, background.

Following the training procedures of GAN, we alternately train the generator and discriminators. We synthesize the image pairs with similar style except text as our training data. Besides, the foreground, text skeleton and background images can be obtained with the help of text stroke segmentation masks. The generator takes I_t, I_s as input with the supervision of T_{sk}, T_t, T_b, T_f and outputs the text replaced image O_t. For the adversarial training, (I_s,O_b) and (I_s,T_b) are fed into D_B to chase for background consistency; (I_t,O_f) and (I_t,T_f) are fed into D_F to ensure accurate results.

In the inference phase, given the standard text image and the style image, the generator can output the erased result of style image and edited image. For the whole image, we crop out the target patches according to the bounding box annotations and feed them to our network, then we paste the results to original locations to get the visualization of whole image.

4 EXPERIMENTS

In this section, we present some results in Fig. 3 to verify that our model has a strong ability of scene text editing, and we compare our method with other neural network based methods to prove the effectiveness of our approach. An ablation study is also conducted to evaluate our method.

4.1 Datasets

The datasets used for the experiments in this paper are introduced as following:

Synthetic Data We improve the text synthesis technology [8] to synthesize data in a pair of style but with different text, the main idea is to select fonts, color, parameters of deformation randomly to generate styled text, then render it on the background image, and, at the same time, we can get the corresponding background, foreground text and text skeleton after image skeletonization [36] as ground truth (Fig. 4). In our experiments, we resize the text image height to 64 and keep the same aspect ratio. The training set consists of a total of 50000 images and the test set contains 500 images.

Real-world Dataset The ICDAR 2013 [14] is a natural scene text data set organized by the 2013 International Conference on Document Analysis and Recognition for competition. This dataset focuses on the detection and recognition of horizontal English text in natural scenes, containing 229 training pictures and 233 test pictures. The text in each image has a detailed label and all text is annotated by horizontal rectangles. Every image has one or more text boxes. We crop the text regions according to the bounding box and input the cropped images to our network, then paste the results back to their original location. Noted that we only train our model on synthetic data, and all real-world data is used for testing only.

4.2 Implementation Details

We implemented our network architecture based on pix2pix [11]. Adam [15] optimizer is adopted to train the model with $\beta_1 = 0.5$, $\beta_2 = 0.999$ until the output tends to be stable in training phase. Learning rate is initially set to 2×10^{-4} and gradually decayed to 2×10^{-6} after 30 epochs. We chose $\alpha = \theta_2 = 1, \beta = \theta_1 = 10, \theta_3 = 500$ to make the loss gradient norms of each part close in back propagation. We apply spectral normalization [20] to both generator and discriminator and use batch normalization [10] in generator only. The batch size is set to 8 and the input images is resized to $w \times 64$ with the aspect ration unchanged. In training, we get the batch data randomly and the image width is resized to the average width, when testing we can input images with variable width to get desired results. The model takes about 8 hours to train with a single NVIDIA TITAN Xp graphics card.

4.3 Evaluation Metrics

We adopt the commonly used metrics in image generation to evaluate our method, which includes the following: 1) MSE, also known as ℓ_2 error; 2) PSNR, which computes the the ratio of peak signal to noise; 3) SSIM [30], which computes the mean structural similarity index between two images. A lower ℓ_2 error or higher SSIM and PSNR mean the results are similar to ground truth. We only calculate the above mentioned metrics on the synthetic test data, because the real dataset does not have paired data. On the real data, we calculate the recognition accuracy to evaluate the quality of the generated result. Since the input of our network is cropped image, we only compute those metrics on the cropped regions. Additionally, visual assessment is also used in real dataset to qualitatively compare the performance of various methods.

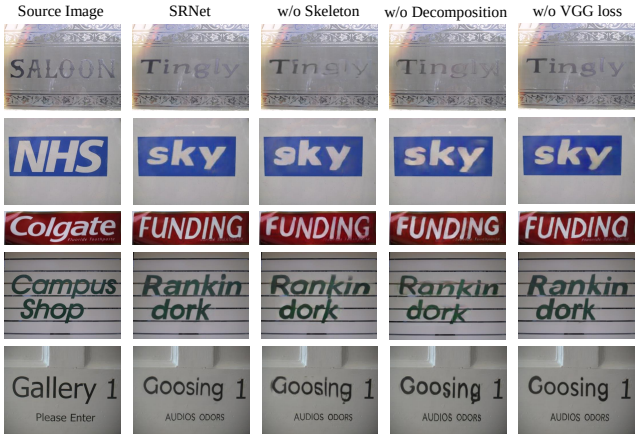

Figure 5: Sample results of ablation study.

The adopted text recognition model is an attention-based text recognizer [27] whose backbone is replaced with a VGG-like model. It is trained on Jaderberg-8M synth data [12] and ICDAR 2013 training data, and them are augmented by random rotation and random resize in x-$axis$. Each text editing model renders 1000 word images based on ICDAR 2013 testing data as their respective test sets. Recognition accuracy is defined as Equ. 11, where y refers to the ground truth of $n-th$ sample, and y' refers to its corresponding predicted result; N refers to the number of samples in the whole test set.

$$seq\_acc = \frac{\sum_{n \in N_{test}} (\mathbb{I}(y == y'))}{N_{test}}. \tag{11}$$

4.4 Ablation Study

In this section, we study the effects of various components of the proposed network with qualitative and quantitative results. Fig. 5 shows the results of different settings such as: removing the skeleton guided module, without decomposition strategy, and removing the vgg loss L_{vgg}(perceptual loss and style loss).

Skeleton-guided Module. After the removal of skeleton module, due to the lack of supervision information of the text skeleton during training, the text structure after transfer is prone to yield local bending even breakage, which is easy to affect the quality of the generated images. In contrast, the full-module method maintains the transfer text structure well and learns the deformation of the original text correctly. From Tab. 1, we can see that the results are worse than full model on all metrics, especially a significant decline appeared in SSIM. This shows skeleton-guided module has a positive effect on the overall structure.

Benefits from Decomposition. A major contribution of our work is to decompose the foreground text and background to different modules. We also conduct experiments on models that did

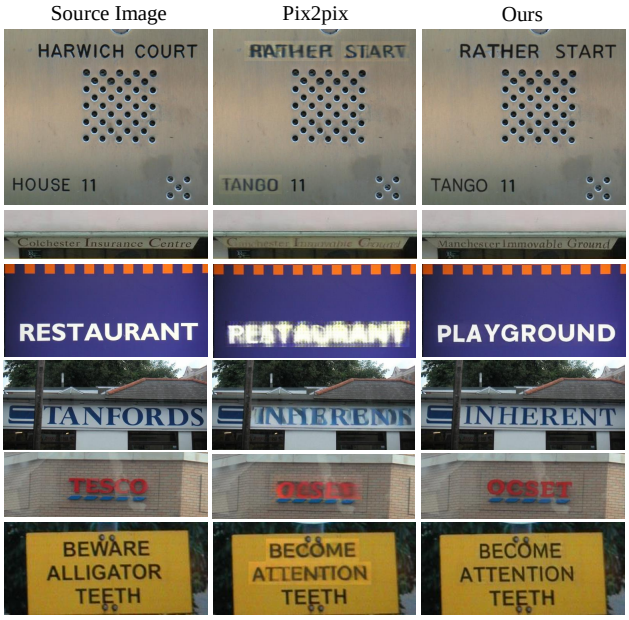

Figure 6: A comparison of our model with pix2pix.

Table 1: Quantitative evaluation results.

method	ℓ_2 error	PSNR	SSIM	seq_acc
pix2pix [11]	0.092	16.54	0.63	0.717
without skeleton	0.025	20.08	0.64	0.798
without decomposition	0.064	18.56	0.66	0.786
without vgg loss	0.022	20.39	0.74	0.778
SRNet	**0.014**	**21.12**	**0.79**	**0.827**

not decompose the foreground text from background. In short, we removed the background inpainting branch, so the foreground text feature and background feature are processed by the foreground module simultaneously. From the Fig. 5, we can find the results are not satisfactory. The original text still remains in the synthetic image, and the text and the background are very vague. From Tab. 1, we can find the metrics of no-decomposition are generally the worst,which verifies that the mechanism of decomposition is helpful to learn clear strokes and reduce learning complexity.

Discussion of VGG Loss. As can be seen from these examples in Fig. 5, the results look unrealistic in appearance without the VGG loss. In this setting, we can find some details like characters in same word has different scales, the structure of text is not maintained well, etc. The results on all metrics are worse than full model, which also illustrates the importance of this component.

4.5 Comparison with Previous Work

Note that there was no work focusing on word-level text editing task before, so we choose pix2pix [11] network, which can complete the image translation task to compare with our method. In order to make pix2pix network implement multiple style translation, we concatenate the style image and the target text in depth as input of the network. Both methods maintain the same configurations during training. As can be seen from the Fig. 6, our method completes the foreground text transfer and retention of the background texture correctly; the structure of the edited text is regular; the font is consistent as before and the texture of background is more reasonable, while the results are similar to the real picture in the overall sense. Quantitative comparison with pix2pix can be found in Tab. 1. It indicates that our method is superior to the pix2pix method in all of the metrics.

Table 2: Comparison SRNet with previous methods on ICDAR2013, lower value means better effect. Note that our method erased text according to the word-level annotations.

Detection	Erasure Methods	F-measure(%)
EAST [39]	Original image	75.37
	Pix2Pix [11]	17.78
	Scene text eraser [21]	16.03
	Ensnet [35]	10.51
	SRNet	**4.64**

4.6 Cross-Language Editing

In this section, we conduct an experiment on cross-language text editing task to check the generalization ability of our model. The application can be used in visual translation and AR translation to improve visual experience. Considering that the relation of Latin fonts and non-Latin fonts are not mapped well, for convenience, we only complete translation tasks from English to Chinese. In the training phase, we adopt the same text image synthesis method mentioned in Sec. 4.1 to generate large amounts of training data. It is worth noting that we map all English fonts to several common Chinese fonts manally by analyzing the stroke similarity from the size, thickness, inclination, etc. We evaluate it on the ICDAR2013 test set and use the translation results as input text to check the generalization of our model. The results are shown in Fig. 7, from which we can see that even if the output is Chinese characters, the color, geometric deformation and background texture can be kept very well, and the structure of characters is the same as the input text. These realistic results show the superior synthesis performance of our proposed method.

4.7 Text Information Hiding

The subtask that extracts the background information can also output the erased image. Different from the two text erasing methods [21, 35], in many cases, the entire image is not required to remove all text, it is more practical to erase part of the text in an image. We are aiming at the word-level text erasure which can select text area freely in the picture needed to be erased. As the erasure examples shown in Fig. 8, we can see that the locations of original text are filled with appropriate textures. Tab. 2 shows the detection results on erased images. Due to the particularity of our method, we erased the cropped images and pasted them back to compare with other methods.

4.8 Failure Cases

Although our method is capable of most scene images, there are still some limitations. Our methods may fail when the text have

Figure 7: The translation examples. Left: input images, right: translation results.

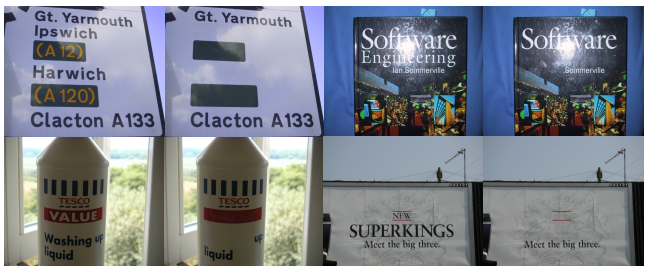

Figure 8: The erasure examples. Left: input images, right: erasure results. We erase the text randomly in every image.

very complex structures or rare font shapes. Fig. 9 shows some failed cases of our method. In the top row, although the foreground text has been transferred successfully, it can be found that the shadow of the original text still remains in the output image. In the middle row of images, our model fails to extract the style of text with such a complicated spatial structure, and the result of the background erasure is also sub-optimal. In the bottom row of images, the boundaries surrounding the text are not transfered with text. We attribute these failure cases to the inadequacy of these samples in training data, so we assume they could be alleviated by augmenting the training set with more font effects.

Figure 9: The failure cases. Left: source images; right: edited results.

5 CONCLUSION AND FUTURE WORK

This paper proposes an end-to-end network for text editing task, which can replace the text from scene text image while maintaining the original style. We mainly divide it into three steps to achieve this function: (1) extract foreground text style and transfer to input text with the help of skeleton; (2) erase the style image with appropriate texture to get background image; (3) merge the transferred text with the erased background. To our best knowledge, this paper is the first work to edit text image in the word-level. Our method has achieved outstanding results in both subjective visual realness and objective quantitative scores on ICDAR13 dataset. At the same time, the network also have the ability to erase text and edit on cross-language situation, and the effectiveness of our network has been verified through the comprehensive ablation studies.

In the future, we hope to solve text editing in more complex scenarios while making the model easier to use. We will edit text between more language pairs to fully exploit the ability of the proposed model. We will try to propose new evaluation metrics to evaluate the quality of text editing properly.

ACKNOWLEDGMENTS

This work is supported by NSFC 61733007, to Dr. Xiang Bai by the National Program for Support of Top-notch Young Professionals and the Program for HUST Academic Frontier Youth Team 2017QYTD08. We sincerely thank Zhen Zhu and Tengteng Huang for their valuable discussions and continuous help to this paper.

REFERENCES

[1] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Learning to Compose Neural Networks for Question Answering. In *NAACL-HLT*. 1545–1554.
[2] Samaneh Azadi, Matthew Fisher, Vladimir G Kim, Zhaowen Wang, Eli Shechtman, and Trevor Darrell. 2018. Multi-content gan for few-shot font style transfer. In *CVPR*. 7564–7573.
[3] Guha Balakrishnan, Amy Zhao, Adrian V Dalca, Fredo Durand, and John Guttag. 2018. Synthesizing images of humans in unseen poses. In *CVPR*. 8340–8348.
[4] Shancheng Fang, Hongtao Xie, Zheng-Jun Zha, Nannan Sun, Jianlong Tan, and Yongdong Zhang. 2018. Attention and Language Ensemble for Scene Text Recognition with Convolutional Sequence Modeling. In *ACM Multimedia*. ACM, 248–256.

[5] Victor Fragoso, Steffen Gauglitz, Shane Zamora, Jim Kleban, and Matthew Turk. 2011. TranslatAR: A mobile augmented reality translator. In *WACV*. IEEE, 497–502.

[6] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *CVPR*. 2414–2423.

[7] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NeurIPS*. 2672–2680.

[8] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. 2016. Synthetic data for text localisation in natural images. In *CVPR*. 2315–2324.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[10] Sergey Ioffe and Christian Szegedy. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *ICML*. 448–456.

[11] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*. 1125–1134.

[12] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. 2014. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. *arXiv preprint arXiv:1406.2227* (2014).

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*. Springer, 694–711.

[14] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. 2013. ICDAR 2013 robust reading competition. In *ICDAR*. IEEE, 1484–1493.

[15] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*. 13.

[16] Shangbang Long, Xin He, and Cong Yao. 2018. Scene Text Detection and Recognition: The Deep Learning Era. *arXiv preprint arXiv:1811.04256* (2018).

[17] Pengyuan Lyu, Xiang Bai, Cong Yao, Zhen Zhu, Tengteng Huang, and Wenyu Liu. 2017. Auto-encoder guided gan for chinese calligraphy synthesis. In *ICDAR*, Vol. 1. IEEE, 1095–1100.

[18] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. 2016. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *IC3DV*. IEEE, 565–571.

[19] Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[20] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. 2018. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957* (2018).

[21] Toshiki Nakamura, Anna Zhu, Keiji Yanai, and Seiichi Uchida. 2017. Scene text eraser. In *ICDAR*, Vol. 1. IEEE, 832–837.

[22] Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*.

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*. Springer, 234–241.

[24] Prasun Roy, Saumik Bhattacharya, Subhankar Ghosh, and Umapada Pal. 2019. STEFANN: Scene Text Editor using Font Adaptive Neural Network. *arXiv preprint arXiv:1903.01192* (2019).

[25] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. ImageNet Large Scale Visual Recognition Challenge. *IJCV* 3, 115 (2015), 211–252.

[26] Baoguang Shi, Xiang Bai, and Cong Yao. 2017. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE TPAMI* 39, 11 (2017), 2298–2304.

[27] Baoguang Shi, Mingkun Yang, Xinggang Wang, Pengyuan Lyu, Cong Yao, and Xiang Bai. 2018. Aster: An attentional scene text recognizer with flexible rectification. *IEEE TPAMI* (2018).

[28] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

[29] Danyang Sun, Tongzheng Ren, Chongxun Li, Hang Su, and Jun Zhu. 2017. Learning to write stylized chinese characters by reading a handful of examples. *IJCAI* (2017).

[30] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.

[31] Shuai Yang, Jiaying Liu, Zhouhui Lian, and Zongming Guo. 2017. Awesome typography: Statistics-based text effects transfer. In *NeurIPS*. 7464–7473.

[32] Shuai Yang, Jiaying Liu, Wenjing Wang, and Zongming Guo. 2019. Tet-gan: Text effects transfer via stylization and destylization. In *AAAI*, Vol. 33. 1238–1245.

[33] Shuai Yang, Jiaying Liu, Wenhan Yang, and Zongming Guo. 2018. Context-Aware Unsupervised Text Stylization. In *ACM Multimedia*. ACM, 1688–1696.

[34] Chengquan Zhang, Borong Liang, Zuming Huang, Mengyi En, Junyu Han, Errui Ding, and Xinghao Ding. 2019. Look More Than Once: An Accurate Detector for Text of Arbitrary Shapes. In *CVPR*. 10552–10561.

[35] Shuaitao Zhang, Yuliang Liu, Lianwen Jin, Yaoxiong Huang, and Songxuan Lai. 2019. Ensnet: Ensconce text in the wild. In *AAAI*, Vol. 33. 801–808.

[36] TY Zhang and Ching Y Suen. 1984. A fast parallel algorithm for thinning digital patterns. *Commun. ACM* 27, 3 (1984), 236–239.

[37] Yexun Zhang, Ya Zhang, and Wenbin Cai. 2018. Separating style and content for generalized style transfer. In *CVPR*. 8447–8455.

[38] Zheng Zhang, Chengquan Zhang, Wei Shen, Cong Yao, Wenyu Liu, and Xiang Bai. 2016. Multi-oriented text detection with fully convolutional networks. In *CVPR*. 4159–4167.

[39] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. 2017. EAST: an efficient and accurate scene text detector. In *CVPR*. 5551–5560.

[40] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*. 2223–2232.

[41] Zhen Zhu, Tengteng Huang, Baoguang Shi, Miao Yu, Bofei Wang, and Xiang Bai. 2019. Progressive Pose Attention Transfer for Person Image Generation. In *CVPR*. 2347–2356.