

# Scene text removal via cascaded text stroke detection and erasing

Xuwei BIAN<sup>1</sup>, Chaoqun WANG<sup>2</sup>, Weize QUAN<sup>1\*</sup>, Juntao YE<sup>1</sup>, Xiaopeng ZHANG<sup>1</sup>, and Dong-Ming YAN<sup>1</sup>

<sup>1</sup>NLPR, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

<sup>2</sup>School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China

## Abstract

Recent learning-based approaches show promising performance improvement for scene text removal task. However, these methods usually leave some remnants of text and obtain visually unpleasant results. In this work, we propose a novel “end-to-end” framework based on accurate text stroke detection. Specifically, we decouple the text removal problem into text stroke detection and stroke removal. We design a text stroke detection network and a text removal generation network to solve these two sub-problems separately. Then, we combine these two networks as a processing unit, and cascade this unit to obtain the final model for text removal. Experimental results demonstrate that the proposed method significantly outperforms the state-of-the-art approaches for locating and erasing scene text. Since current publicly available datasets are all synthetic and cannot properly measure the performance of different methods, we therefore construct a new real-world dataset, which will be released to facilitate the relevant research.

## Index Terms

scene text removal, text stroke detection, generative adversarial networks, cascaded network design, real-world dataset

## I. INTRODUCTION

Scene text is an important information carrier, and often appears in various scenarios. The problem of scene text removal can be stated as follows: given an image with appropriate amount of text [e.g., Fig. 1(a)], the goal is to remove the text in this image [e.g., Fig. 1(d)]. This task has many applications in our daily life, such as personal private information protection (hiding telephone numbers or home address from public photos), text translation (removing the original text and pasting new translated results), and so on.

Several approaches have been proposed to erase graphical text (e.g., subtitles) from color images [1]–[3]. For the challenging scenario of scene text removal, which usually has complex background and text with various fonts and sizes, etc, however, these methods often produce results with visual artifacts. Inspired by the notable success of deep learning in image transformation [4]–[6], recent works have introduced deep-learning-based approaches to solve this problem and have achieved promising results [7]–[9]. The learning-based methods can be roughly classified into two main categories, *i.e.*, text removal without/with using mask. The former simply takes the given image as input and removes all the texts from the whole input image. This kind of methods often left noticeable remnants of text or distort non-text area incorrectly, and cannot remove text locally. The latter usually uses a region mask, *i.e.*, a rectangle or polygon mask roughly indicating the text region [e.g., Fig. 1(b)], as additional input to facilitate the text removal.

Recent MTRNet [9] achieved noticeable improvement compared to prior works for scene text removal, by focusing on text regions via auxiliary/binary mask. In fact, their pipeline is similar to the general image inpainting tasks [10], [11]. However, there is an apparent difference between them: for the text removal, the pixel values of original input image in the regions indicated by auxiliary mask (*i.e.*, text regions) are known; whereas the corresponding values are unknown (corrupted) for the general image inpainting, *i.e.*, the so-called missing regions. Generally speaking, when the regions to be processed (indicated by mask) are larger, it becomes harder to fill or remove the corresponding regions not only for the image inpainting, but also for the text removal. In addition, for the scene text removal problem itself, there is no need to remove regions not covered by text strokes like MTRNet. In other words, the mask used by MTRNet covers some unnecessary/redundant regions (*i.e.*, non-stroke areas), especially when text strokes are scattered sparsely. It is obvious that if we can extract the exact text stroke, which means that we can preserve original contents of input image as much as possible, and then we could achieve better result. However, such precise areas are difficult to obtain, to best of our knowledge, there is no related research to focus on distinguishing text strokes from non-stroke area in the pixel-wise level.

In this paper, we propose a novel “end-to-end” framework based on *generative adversarial network* (GAN) to address this problem. The key idea of our approach is first to extract text strokes as accurately as possible, and then improve the text removal process. These two processes can be further enhanced via a simple cascade. In addition, current public datasets for scene text removal are all synthetic, which to some extent affect the generalization ability of trained models. To facilitate this

\*Corresponding author. E-mail: qweizework@gmail.com



Fig. 1. Sampled results of the proposed scene text removal method. From left to right: (a) input image, (b) the region mask, (c) text stroke mask obtained by our TSDNet, and (d) the final result. Each row shows a representative result of commonly appeared scenarios.

research and be close to real-world setting, we construct a new dataset with high quality. The main contributions of our work include:

- We design a text stroke detection network (TSDNet), which can effectively distinguish text strokes from non-text area.
- We propose a text removal generation network and combine it with TSDNet to construct a processing unit, which is cascaded to obtain our final network. Our method demonstrates the superior performance.
- We propose a weighted-patch-based discriminator to pay more attention to the text area of given images, making it easier for the generator to generate more realistic images.
- We construct a high-quality real-world dataset for the scene text removal task, and this dataset can be used to benchmark related text removal methods. It can also be used in other related tasks.

The remainder of this paper is organized as follows. Section II reviews relevant existing work. Section III introduces the motivation and network details of our method. Section IV presents the performance evaluations for our method and detailed comparisons with existing methods. Section V draws the conclusions and discusses the future working directions.

## II. RELATED WORK

### A. Scene text detection

Scene text detection is a fundamental step of scene understanding and is widely studied in the field of computer vision [12]. With the aid of deep learning, the performance of scene text detection framework has been significantly improved and surpassed traditional methods by large margins. Shi et al. [13] decompose text into two locally detectable elements of segments and links, which are simultaneously detected by a fully-convolutional network. Liu et al. [14] collect a curved text dataset called CTW1500 to facilitate the curved text detection task, and propose a method with the intergration of transverse and longitudinal sequence connection. Chen et al. [15] propose the concept of weighted text border and introduce attention module to boost the

detection performance. To obtain better detection performance, multi-scale pyramid input is widely used with the consumption of much more running time. He et al. [16] achieve remarkable speedup via a novel two-stage framework including a scale-based region proposal network and a fully convolutional network. CRAFT [17] effectively detect arbitrary text area by exploring each character and affinity between characters. In this work, we adopt this method as the tool to measure the performance of scene text removal (more details in Section IV-B).

### B. Text/non-text image classification

Another relevant research is text/non-text image classification, which identifies whether a image block contains text or not. Zhang et al. [18] first propose an effective method for text image discrimination, which is the suitable combination of maximally stable extremal region (MSER) [19], CNN, and bag of words (BoW) [20]. Bai et al. [21] propose a multi-scale spatial partition network to efficiently solve this task by predicting all image blocks simultaneously in a single forward propagation. Zhao et al. [22] investigate this task from two perspectives of the speed and the accuracy. They use a small and shallow CNN to accomplish high speed and then apply the knowledge distillation to improve its performance. Very recently, Gupta and Jalal [23] combine a text detector EAST [24] and classification subnetwork to achieve text/non-text image classification. Different from these previous works, our method mainly captures the exact position of text stroke, *i.e.*, in the pixel-wise level, instead of image block/patch-wise level, to effectively facilitate the text removal network.

### C. Scene text removal

Existing approaches of the scene text removal can be classified into two major categories: traditional non-learning methods and deep-learning-based methods.

Traditional approaches typically use color-histogram-based or threshold-based methods to extract text areas, and then propagate information from non-text regions to text regions depending on pixel/patch similarity [1]–[3]. These methods are suitable for simple cases, *e.g.*, clean and well focused text, whereas they have limited performance on complex scenarios, such as perspective distortion and complicated background, etc.

Recent learning-based approaches try to solve this problem with the powerful learning capacity of deep neural networks. Nakamura et al. [7] first propose a scene text erasing method (ST Eraser) based on convolutional neural network (CNN), and conducted text erasing patch by patch. This patch-based processing fails to localize text with complex shape and inevitably damaged the consistency and continuity of erased result. More recently, Zhang et al. [8] design an end-to-end trainable framework (EnsNet) with a conditional GAN to remove text from natural images. Different from [7] which erases text in an image patch by patch, EnsNet can erase the scene text on the whole image in an “end-to-end” manner. For these two works, they do not use mask, and thus need to localize and remove text simultaneously. Such kind of methods often suffer from inaccurate text localization and incomplete text removal. To solve this, Tursun et al. [9] develop a mask-based text removal network (MTRNet). Auxiliary mask is used to provide information on where the text is, and enables MTRNet to focus on text removal better. The additional information provided by mask is the main reason why MTRNet outperforms previous studies. MTRNet also supports partial/local text removal by providing mask purposefully. All of these existing approaches often leave some text strokes unchanged or generate unpleasant contents because they cannot appropriately and exactly pay attention to the text strokes. Another shortcoming of current methods is that their training datasets are all synthetic, because the collection of real-world datasets are difficult and time-consuming.

In addition, a closely related problem to scene text removal is the general image inpainting, which aims at synthesizing plausible contents to fill missing/hole regions of the corrupted input images. Image inpainting has been extensively studied with the aid of deep learning methods. More recently, several GAN-based approaches are proposed for this purpose [10], [25], [26] and show strong ability of generating reasonable contents for missing regions. We also use the GAN framework in our approach for text removal.

## III. PROPOSED METHOD

Region mask is commonly used in general image inpainting studies to indicate which regions should be filled. MTRNet [9] first introduces region mask into text removal task, and achieves good performance. Yet such direct introduction and application is inappropriate because it ignores the difference between scene text removal and general image inpainting. Region mask is suitable for image inpainting where it properly specifies the to-be-filled region. When directly applying in the text removal mask like MTRNet, unfortunately, it cannot reach pixel-level accuracy in distinguishing which regions are text strokes. Regarding the above inappropriateness, we believe that a more accurate text stroke mask can help to improve the performance of text removal methods. We therefore propose to decouple the text removal task into two sub-tasks: text stroke detection and stroke removal, and solve them separately. In the following, we explain the details of our network architectures and the training losses used in our network.

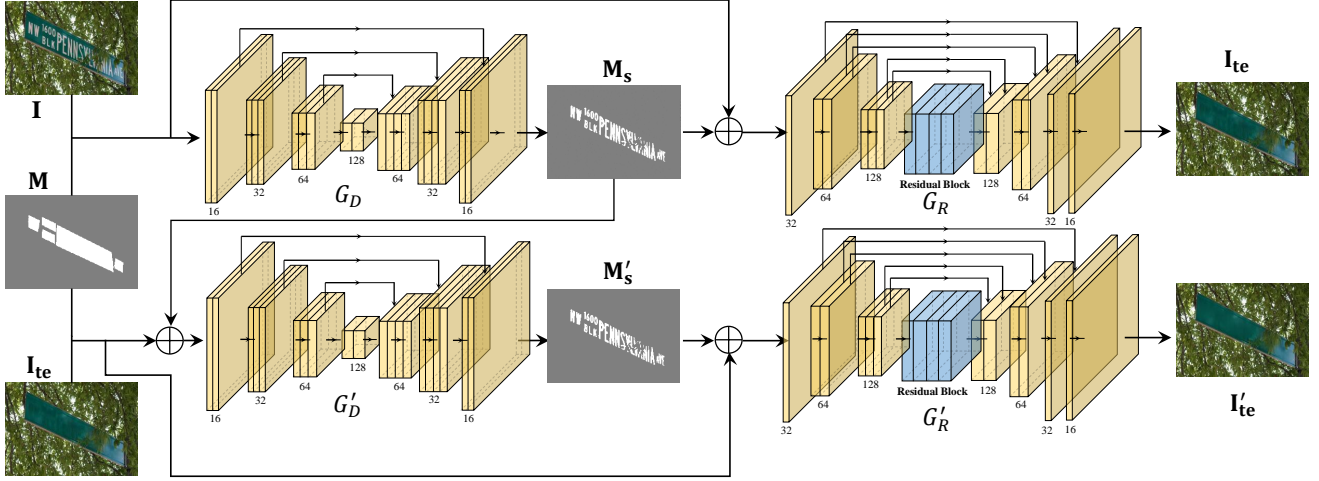


Fig. 2. The overall structure of the proposed generator, which consists of cascaded Text Stroke Detection and Text Removal Generation.  $\oplus$  indicates the concatenation of image, region mask and stroke mask. The convolutional kernel size of the first layer of  $G_R$  and  $G'_R$  is  $5 \times 5$ , and the remaining kernel size is  $3 \times 3$  in our proposed generator.

### A. Network architecture

We design and implement a text stroke detection network, and then combine it with our proposed text removal generation network to construct a processing unit. The final network is obtained by cascading this unit and combining with a weighted-patch-based discriminator.

1) *Cascaded generator*: The proposed generator is designed for the following two purposes, *i.e.*, 1) to detect text strokes in the input image accurately; 2) to inpaint the detected text strokes with proper content. To achieve the first goal, we construct a text stroke detection network (TSDNet). For the second goal, we propose a text removal generation network (TRGNet). The whole generator is obtained by cascading the group of TSDNet and TRGNet, as shown in Fig. 2. Note that, the parameters in these four networks are not shared. Technically, both TSDNet and TRGNet employ a U-Net-like architecture [27], since by comparing with simple encoder-decoder framework, the U-Net architecture with skip connection helps to recover the structure and the texture details of unmasked area from input images, as well as to avoid over-smoothing and undesired artifacts to some extent.

The inputs of the TSDNet (noted as  $G_D$ ), are a text image  $\mathbf{I}$  and a binary mask  $\mathbf{M}$  (indicating the text regions). The output is a float matrix  $\mathbf{M}_s$  with the same size as  $\mathbf{M}$ , ranging from 0 to 1, in which larger value indicates higher confidence that corresponding position of image  $\mathbf{I}$  is covered by text stroke.

$$\mathbf{M}_s = G_D(\mathbf{I}, \mathbf{M}). \quad (1)$$

The ground-truth of text stroke distribution is a binary mask  $\mathbf{M}_{gt}$ , in which 1 means corresponding position of  $\mathbf{I}$  is covered by text stroke. Different from  $\mathbf{M}$  which only specifies the rough region of certain text,  $\mathbf{M}_{gt}$  is a pixel-level annotation of text strokes. Practically,  $\mathbf{M}_{gt}$  can be obtained by binarizing the difference between paired text image  $\mathbf{I}$  and text-free image  $\mathbf{I}_{gt}$  (see more details in Section IV-A), and this stroke annotation is only used in the training stage as the supervised information to train TSDNet.

After obtaining the stroke mask  $\mathbf{M}_s$ , the TRGNet  $G_R$  is then applied to erase text from the input image  $\mathbf{I}$ .  $G_R$  takes three items as input, namely, text image  $\mathbf{I}$ , binary mask  $\mathbf{M}$ , and obtained stroke mask  $\mathbf{M}_s$  from  $G_D$ , and outputs text-erased image  $\mathbf{I}_{te}$ , *i.e.*,

$$\mathbf{I}_{te} = G_R(\mathbf{I}, \mathbf{M}, \mathbf{M}_s). \quad (2)$$

A TSDNet followed by a TRGNet (the first row of Fig. 2) can already detect and erase text effectively, but the result images ( $\mathbf{I}_{te}$ ) sometimes contain awkward artifacts and slight remnants of text. We observed that a simple cascade can eliminate such artifacts and bring visual improvement significantly. The designed architecture is as follows: the second TSDNet  $G'_D$  takes  $\mathbf{I}_{te}$ ,  $\mathbf{M}$ , and  $\mathbf{M}_s$  as input, and outputs  $\mathbf{M}'_s$ . Then, the second TRGNet  $G'_R$  takes  $\mathbf{I}_{te}$ ,  $\mathbf{M}$ , and  $\mathbf{M}'_s$  as input, and outputs the final text-erased result  $\mathbf{I}'_{te}$ . Through combining the previous outputs,  $G'_D$  obtains more accurate text stroke distribution in an incremental manner, and thus  $G'_R$  can reduce artifacts and inconsistency effectively.

Previous studies such as EnsNet and ST Eraser, which simply take an image/patch as input and try to erase text without any prior information, showed relatively limited performance. In this work, we use a binary mask (specifying the text region) as additional information to decrease the difficulty of detecting and erasing text at the same time, and design a TSDNet to

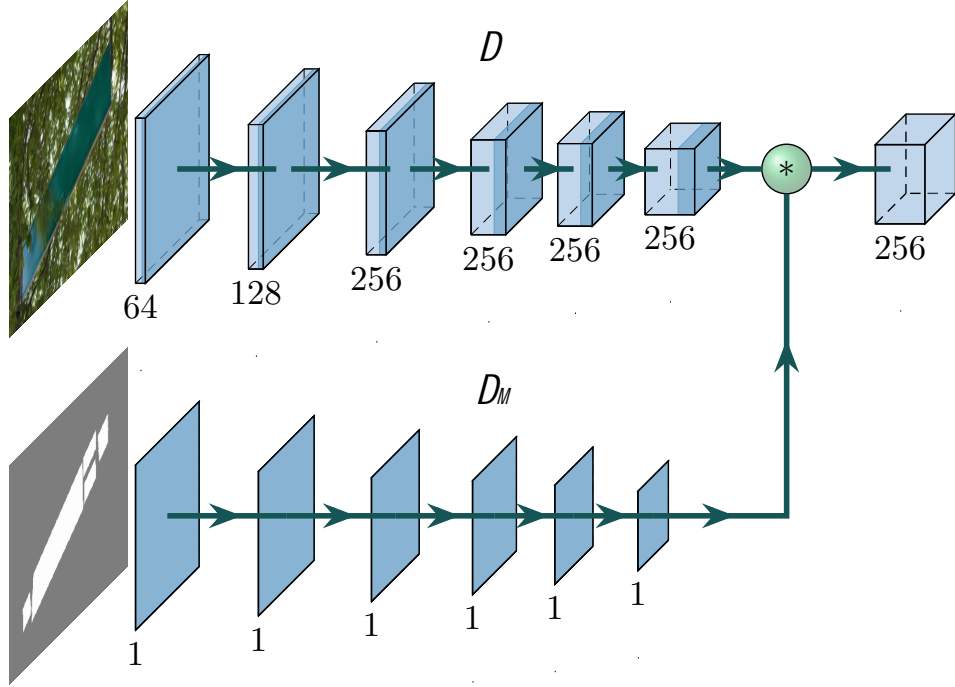


Fig. 3. Architecture of our proposed weighted-patch-based discriminator. \* means element-wise multiplication between two branches with broadcasting. The convolutional kernel size is  $5 \times 5$ .

provide more accurate instruction on which area should be removed. By doing so, we successfully decouple text removal into text stroke detection and stroke removal, and propose an effective solution and framework to solve these decoupled problems.

2) *Weighted-patch-based discriminator*: As text removal only needs to alter partial content of input image, thus a patch-based discriminator (see Fig. 3) is more suitable to effectively concentrate on altered areas. In this work, we use the discriminator proposed in SN-PatchGAN [26], [28] to discriminate the text-erased image patch by patch. We further improve the original discriminator by attaching an additional convolutional branch  $D_M$  to discriminator  $D$  for assigning different weights to different patches according to mask  $\mathbf{M}$ . The  $D_M$  has the same architecture as the  $D$ , but each layer only has one channel and weights in convolutional kernel are fixed to 1. By doing so, the patches covered by more text will be paid with more attention.

### B. Training loss

In this subsection, we present our loss functions for the generator and discriminator. To verify that our proposed method is valid, we use relatively simple loss function when training our network. For TSDNet  $G_D$  and  $G'_D$ , we use simple  $l_1$  loss,

$$\mathcal{L}_{TSD} = \mathbf{E} \left[ \|\mathbf{M}_s - \mathbf{M}_{\text{gt}}\|_1 + \lambda_t \cdot \|\mathbf{M}'_s - \mathbf{M}_{\text{gt}}\|_1 \right], \quad (3)$$

where  $\lambda_t$  balances the  $l_1$  loss of  $G_D$  and  $G'_D$ . We set  $\lambda_t = 10$  in all our experiments, as most text strokes have been detected by  $G_D$ .

For scene text removal task, our main goal is to remove the text and preserve the non-text regions, therefore, more attention have to be paid to masked area (indicated by  $\mathbf{M}$ ), especially detected stroke area (indicated by  $\mathbf{M}_s/\mathbf{M}'_s$ ). More precisely, we define the corresponding weight matrix  $\mathbf{M}_w$  and  $\mathbf{M}'_w$  for  $G_R$  and  $G'_R$  as following, respectively:

$$\begin{aligned} \mathbf{M}_w &= \mathbb{1} + \lambda_m \cdot \mathbf{M} + \lambda_s \cdot \mathbf{M}_s, \\ \mathbf{M}'_w &= \mathbb{1} + \lambda_m \cdot \mathbf{M} + \lambda_s \cdot \mathbf{M}'_s, \end{aligned} \quad (4)$$

where  $\mathbb{1}$  has same shape as  $\mathbf{M}$  and its all elements are 1. Then, the total loss of  $G_R$  and  $G'_R$  is defined as

$$\begin{aligned} \mathcal{L}_{TRG} = \mathbf{E} \left[ \|\mathbf{I}_{\text{te}} \odot \mathbf{M}_w - \mathbf{I}_{\text{gt}} \odot \mathbf{M}_w\|_1 \right. \\ \left. + \lambda_r \cdot \|\mathbf{I}'_{\text{te}} \odot \mathbf{M}'_w - \mathbf{I}_{\text{gt}} \odot \mathbf{M}'_w\|_1 \right], \end{aligned} \quad (5)$$

where  $\odot$  is the element-wise product operation, and  $\lambda_r$  is the balance parameter. In all our experiments, we set  $\lambda_m = 5$ ,  $\lambda_s = 5$ , and  $\lambda_r = 10$ .

For the objective function of patch-based GAN, we use the hinge version of adversarial loss [29], [30]. The corresponding loss function for generator and discriminator are respectively defined as

$$\mathcal{L}_G^{sn} = -\mathbf{E}\left[D_M(\mathbf{M}) \odot D(\mathbf{I}'_{te})\right], \quad (6)$$

$$\begin{aligned} \mathcal{L}_D^{sn} = & \mathbf{E}\left[ReLU(1 - D_M(\mathbf{M}) \odot D(\mathbf{I}_{gt}))\right] + \\ & \mathbf{E}\left[ReLU(1 + D_M(\mathbf{M}) \odot D(\mathbf{I}'_{te}))\right]. \end{aligned} \quad (7)$$

Note that,  $\odot$  means element-wise product with broadcasting in terms of depth.

To summarize, the total loss for our cascaded generator is the summation of Eqn. 3, 5 and 6:

$$\mathcal{L}_G = \mathcal{L}_{TSD} + \mathcal{L}_{TRG} + \mathcal{L}_G^{sn}. \quad (8)$$

Moreover, we observed that the perceptual loss [4] and the style loss [31] have no noticeable improvement for our task. One reason is that scene text is usually located in relatively flat area. The total variation loss [32] has no apparent effect on the erased result either, and thus is not used in our method.

#### IV. EXPERIMENTAL RESULTS

To evaluate our proposed method quantitatively and qualitatively, we compare our method with recent state-of-the-art text removal methods and general image inpainting methods, on synthetic dataset and our collected real-world dataset. Ablation study is also conducted to evaluate different components of our network.

##### A. Dataset

To train the deep model for text removal task, paired text image and text-free image are required. However, it is difficult to obtain such paired data for real-world scene images, this is why synthetic datasets are used for constructing text removal dataset in previous approaches. Currently, there are only two synthetic datasets, *i.e.*, the Oxford synthetic scene text detection dataset [33] and the SCUT synthetic text removal dataset [8]. These two datasets adopted the same synthetic technology proposed by [33], and shared the same drawback, *i.e.*, given a text-free image as background, a few or more images with synthesized text are obtained. For instance, the Oxford dataset synthesized 800,000 images using only 8,000 text-free images, which means that each 100 text images are synthesized using the same background image, leading to insufficiency of the diversity of background. Such kind of repetition would cause negative affect to the generalization ability of models.

In addition, synthetic data is only an approximation of real-world data. Current text synthesis technologies cannot generate realistic enough text images, which would restrain the text removal ability of models trained on synthetic data. When existing text removal methods are trained on synthetic dataset, and then tested on real-world data, we found that there often exists obvious text remnants and unsatisfactory artifacts, which is quantitatively analyzed in Section IV-E. To this end, we propose to construct a real-world dataset for the text removal scenario.

To construct such dataset, we first collect 5,070 images with text from ICDAR2017 MLT dataset [34], and 1,970 images captured from supermarkets and streets, etc. Then, post-processing are applied to obtain corresponding text-free images, region masks, and text stroke masks. We manually remove the text from these collected images and obtain text-free images as ground-truth using the inpainting tools in Photoshop<sup>®</sup>. The region masks are annotated by using VGG Image Annotator tool [35]. For the ground-truth stroke mask, we first compute the difference between paired text images and text-free images, and then turn it into a binary image. To enrich the diversity of our dataset, we also use synthesis method and then manually select 4,000 images with high realism. In total, we obtain 11,040 images as training set (Train<sub>rw</sub>). Several samples of our dataset are shown in Fig. 4. To construct testing set (Test<sub>rw</sub>), we additionally collect 1,080 real-world images and apply the same above post-processing to obtain the text-free images, region masks and text stroke masks.

In this work, we mainly conduct the experiments and comparisons on our real-world dataset. In the meanwhile, we also conduct experiments on a public synthetic dataset, *i.e.*, the Oxford dataset [33], which is much larger than the SCUT dataset [8]. For the Oxford dataset, we randomly select 75% of the whole dataset as training set (Train<sub>ox</sub>), and then randomly select 2,000 images from remaining set as testing set (Test<sub>ox</sub>). Note that, for these two datasets, *i.e.*, our real-world (RW) dataset and Oxford dataset, there is no overlapping between training set and testing set.

##### B. Evaluation metrics

We evaluate the performance from two different aspects: (1) can the method remove text from an image completely; (2) can text area be replaced with appropriate content. An accurate text detector is often used for the former evaluation metric. As for a text-erased image, the cleaner text is erased, the fewer text will be detected. In this work, we use the state-of-the-art text detector CRAFT [17] and use DetEval protocol [36] for evaluation (*i.e.*, recall, precision, and f-measure). For the second evaluation metric, we adopt general image inpainting metrics, and mainly use the following three indicators: 1) mean absolute error (MAE); 2) peak signal-to-noise ratio (PSNR); and 3) the structural similarity index (SSIM).



Fig. 4. Samples of our dataset. From left to right: images with text, text-free images, region masks, and text stroke masks.

### C. Implementation details

We implement our network using TensorFlow 1.13. The GPU version is TITAN RTX of NVIDIA® corporation. Input images are resized to  $256 \times 256$ . Adam optimizer [37] with a minibatch size of 16 is used to train our network, and its  $\beta_1$  and  $\beta_2$  are set to 0.5 and 0.9 separately. Initial learning rate is set to 0.0001. The model is trained for 10 epochs on our dataset and 6 epochs on Oxford dataset.

### D. Dataset comparison

Fig. 5 shows the comparison between Oxford synthetic dataset and our real-world dataset using two different networks (MTRNet and our proposed network). Each group of images contains the input (left), the result of model trained on Oxford dataset (middle), and the result of model trained on our dataset (right). It is obvious that the text in right image of each group is better removed, especially for our method (the second row in Fig. 5), which has no noticeable remnant and better preserves the original image details, such as lighting effect. This indicates that our dataset is more suitable for the scene text removal, even though the number of images in Oxford dataset is much larger than that of ours.

### E. Comparison with state-of-the-art methods

We quantitatively and qualitatively compare our method with state-of-the-art text removal methods: ST Eraser [7], EnsNet [8], and MTRNet [9], as well as recent image inpainting method: GatedConv [26]. We use the official implementation of EnsNet and GatedConv, and re-implemented ST Eraser and MTRNet.

Table I reports the quantitative comparison of the above five methods on the Oxford dataset and our dataset. We can find that our method is superior to other methods in MAE, PSNR, and SSIM by a large margin. When training on Train\_rw and testing on Test\_ox, our method achieves the best performance, and this phenomenon also exists when training on Train\_ox and testing on Test\_rw. This observation further indicates that our network has better generalization capability. For the cross-dataset validation, the results of training on Train\_ox and testing on Test\_ox are relatively similar with that of training on Train\_rw and testing on Test\_ox (see Table I column 9-14). However, the performance of training on Train\_rw and testing on Test\_rw is obviously better than that of training on Train\_ox and testing on Test\_rw (see Table I column 3-8), e.g., the PSNR and SSIM of EnsNet are improved by 7.37 (from 26.41 to 33.78) and 8.13% (from 87.30% to 95.43%), respectively. These two results imply that our dataset is more suitable for this scene text removal task, especially for the real-world applications.



Fig. 5. Oxford dataset vs. Our dataset. We use different datasets to train the same method and compare the generalization ability of obtained models. The first row corresponds to the results of MTRNet and the second row corresponds to that of our method. Every three consecutive images (in row) form a group. For each group, from left to right: (a) the input image, (b) text removed result by model trained on the Oxford dataset, and (c) the text removed result by model trained on our dataset.

TABLE I

QUANTITATIVE COMPARISON OF OUR METHOD AND STATE-OF-THE-ART METHODS. ALL METHODS ARE TRAINED AND TESTED ON THE OXFORD DATASET AND OUR DATASET SEPARATELY. FOR PSNR AND SSIM (IN %), HIGHER IS BETTER; FOR MAE, R (RECALL), P (PRECISION), AND F (F-MEASURE), LOWER IS BETTER.

Testing set		Test_rw						Test_ox					
Training set	Method	MAE	PSNR	SSIM	R(%)	P(%)	F(%)	MAE	PSNR	SSIM	R(%)	P(%)	F(%)
	Original image	-	-	-	43.25	40.68	41.93	-	-	-	48.24	67.93	56.42
Train_rw	ST Eraser(2017)	2.52	27.20	91.13	6.23	20.55	9.56	2.67	27.94	90.24	14.42	49.27	22.31
	EnsNet(2019)	1.22	33.78	95.43	1.94	20.18	3.53	1.89	31.37	93.03	7.25	49.84	12.65
	MTRNet(2019)	1.62	34.31	96.34	0.55	17.14	1.06	2.31	31.81	92.36	0.49	37.27	0.96
	GatedConv(2019)	1.42	34.82	96.10	<b>0.04</b>	<b>1.49</b>	<b>0.08</b>	2.50	30.42	89.52	<b>0.01</b>	<b>6.25</b>	<b>0.03</b>
	<b>Ours</b>	<b>0.75</b>	<b>39.44</b>	<b>97.56</b>	0.35	10.23	0.68	<b>1.63</b>	<b>34.40</b>	<b>93.97</b>	0.05	15.38	0.10
Train_ox	ST Eraser(2017)	4.26	21.52	82.20	28.34	36.17	31.78	5.77	20.92	77.05	21.56	48.60	29.87
	EnsNet(2019)	2.55	26.41	87.30	23.77	34.56	28.17	1.75	32.76	93.20	4.07	46.09	7.48
	MTRNet(2019)	1.89	32.53	92.68	11.07	34.95	16.82	2.24	31.79	92.06	1.79	42.75	3.43
	GatedConv(2019)	1.48	34.32	95.93	<b>0.02</b>	<b>1.27</b>	<b>0.04</b>	2.16	31.31	91.30	0.03	17.86	0.06
	<b>Ours</b>	<b>1.23</b>	<b>36.23</b>	<b>96.64</b>	0.33	10.89	0.64	<b>1.72</b>	<b>34.48</b>	<b>94.47</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>

Our method has the best performance among four scene text removal methods when evaluated by recall, precision, and f-measure. These three metrics are often lowest for GatedConv, because GatedConv first fully removes the text area and then fill the so-called missing region. Such processing can avoid incomplete text removal, bringing lowest recall, precision and f-measure, but can also bring obvious over-smoothing and boundary inconsistency to inpainted area, as shown in the following qualitative comparison.

Fig. 6 shows the text-erased images of all five methods. Compared with other text removal methods, our method is more effective in erasing text and inpainting text area with proper content. In the first row of Fig. 6, our result preserves more consistent texture details with original non-text areas, whereas that of other methods has obvious text remnants or visual inconsistency. Comparing the results in the fifth row, our result has no text remnant and well maintains the original structure, *i.e.*, the light transition. Furthermore, the text-erased images of our method show more reasonable texture details than that of GatedConv (comparing the fifth and sixth column in Fig. 6). The reason is that GatedConv first replaces the masked area with blank images, which will lose the useful detail information, thus resulting in the filling of texture itself becomes relatively more difficult. These results indicate that the reasonability of distinguishing text stroke area from non-text area in the masked region, which can guide the network to focus on text stroke area and preserve the useful information of original input image.

As analyzed earlier, using text region masks instead of text stroke masks is an important reason why existing scene text removal methods have limited performance. Inspired by this, our proposed novel and generic framework decouples the text removal problem into text stroke detection and stroke removal, and achieves superior performance. Following the pipeline of our “end-to-end” framework, a two-stage method, *i.e.*, first extracting text strokes with a semantic segmentation algorithm and then filling these holes with image inpainting approach (*e.g.*, GatedConv [26]), may be possible to handle this task. Fig. 7 compares the results of our method, GatedConv, and this two-stage method. In this experiment, we simply use the detected stroke mask via our TSDNet as the output of the first stage for the two-stage method. In the third and sixth row of Fig. 7, we find that the results of this two-stage method [(c) and (d)] both are visually unpleasant, whereas the results of our method



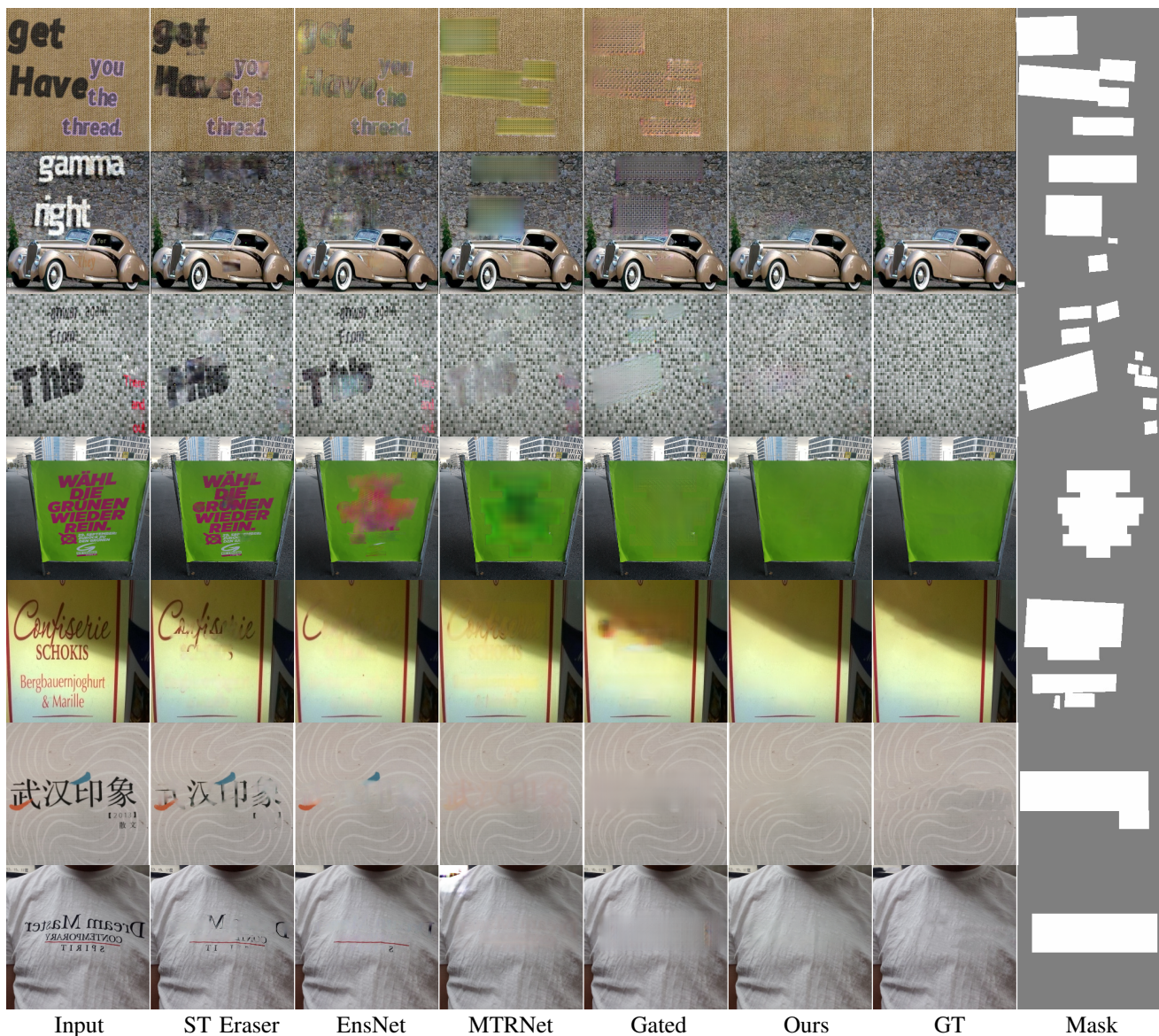


Fig. 6. Qualitative comparison of all methods. The top three rows are synthetic images, and the rest are real-world images. From left to right: input image, ST Eraser, EnsNet, MTRNet, GatedConv, Our method, ground-truth, and input mask.

[[a)] are consistently good. Noticeably, the final result of this two-stage method is sensitive to the output of the segmentation stage, *i.e.*, the detected text stroke mask, when directly using the stroke mask provided by our TSDNet, there is plenty of text remnants [see Fig. 7(c)], and this phenomenon is improved by dilating the corresponding stroke mask [see Fig. 7(d)]. The possible reason is that GatedConv relies so much on the surroundings of the masked area that a tiny noise would effect the final text removal results noticeably. On the contrary, our “end-to-end” framework only takes the detected stroke mask of our TSDNet as the middle information, and thus is robust to stroke mask.

#### F. Multi-lingual text removal and selective text removal

In this subsection, we also illustrate more results about multi-lingual text removal and selective text removal. We train MTRNet [9] and our method on our real-world dataset, and the corresponding results are reported in Fig. 8. Compared with MTRNet, our method can successfully remove the text in various languages. The reason is that our text stroke detection network focuses on the text, even learns the difference between various languages, and thus provides more useful information for the consecutive text removal generation network than the region mask.

In addition, our method can also accomplish the selective text removal. Given an auxiliary mask, where the desired removal text is indicating by a polygonal mask, our method can purposefully remove desired texts and does not affect the other text. Several sampled examples are shown in the last three rows of Fig. 8.

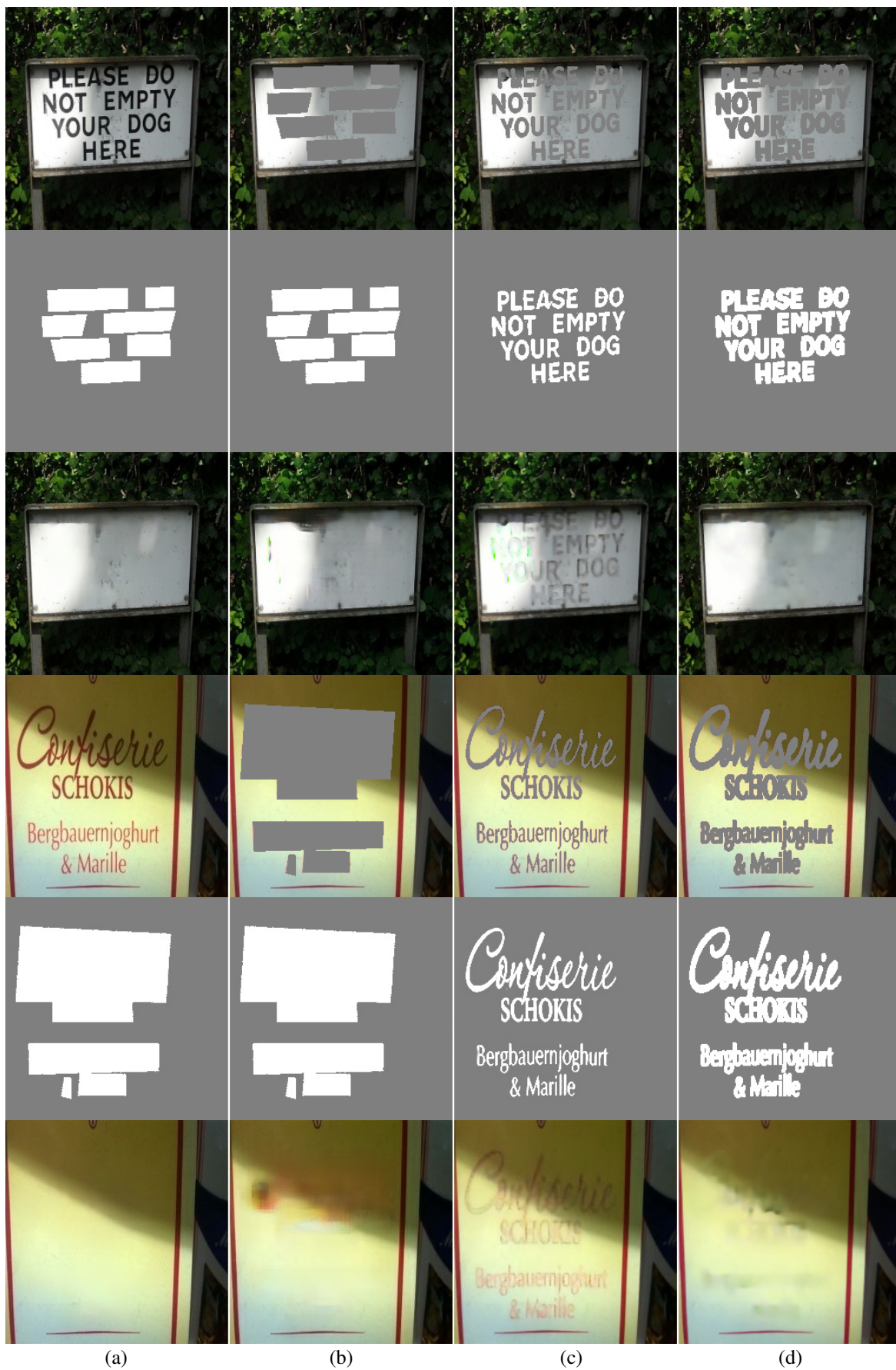


Fig. 7. Comparison of our method with a two-stage method. For each group (including three rows), the first and second rows are the inputs of network, and the third row is the final result. From left to right: (a) our method; (b) GatedConv with rectangle mask; (c) GatedConv with stroke mask, which is detected by our TSDNet; (d) GatedConv with dilated stroke mask.



Fig. 8. Results of multi-lingual text removal and selective text removal (the last three rows). From left to right: original image, input mask, MTRNet output, our output, and detected stroke mask of our method. Both models are trained on our dataset.

TABLE II

ABLATION STUDY. MODELS ARE TRAINED ON TRAIN\_RW AND TESTED ON TEST\_RW. tMAE IS THE MEAN ABSOLUTE ERROR BETWEEN DETECTED STROKE MASK AND GROUND-TRUTH STROKE MASK.

Method	MAE	PSNR	SSIM(%)	tMAE(%)
Baseline	1.59	35.00	95.42	-
WD	1.00	38.31	97.22	-
TSDNet	0.98	38.17	97.33	7.63
WD + TSDNet	0.92	38.47	97.48	4.85
Cascade	<b>0.75</b>	<b>39.44</b>	<b>97.56</b>	<b>4.73</b>

### G. Ablation study

Next, we study the effect of different components of our network. The corresponding results are reported in Table II and Fig. 9.

1) *Baseline*: For baseline model, we use a single TRGNet  $G_R$  (shown in Fig.2) as the generator, and use the discriminator proposed in SN-PatchGAN [26] as the discriminator (*i.e.*,  $D$  in Fig. 3). The inputs of TRGNet here are a text image  $I$  and a binary mask  $M$ . Comparing Table I and II, we find that such designed baseline model already shows similar performance with previous text removal methods, including EnsNet which does not use auxiliary mask and MTRNet which uses region mask. The visual results are also good as shown in the second column of Fig. 9.

2) *Weighted-patch-based discriminator (WD)*: Original discriminator proposed in SN-PatchGAN treats all patches equally. In our work, masked regions indeed are the focus of our attention. We propose a weighted-patch-based discriminator, which can pay more attention to masked area via assigning higher weight. Comparing the rows of “Baseline” and “WD” in Table II,



Fig. 9. Qualitative results of ablation study. The last column gives the best result.

it shows that our proposed discriminator can significantly improve the performance of the baseline model with the help of this weighted design. For example, the first row in Fig. 9 shows that our proposed weighted-patch-based discriminator can help to maintain the structure consistency of given image.

3) *Text stroke detection network (TSDNet)*: A TSDNet is added into baseline model to prove the effectiveness of accurate text stroke extraction. As shown in Table II, baseline model with TSDNet obtains much higher PSNR, SSIM, and much lower MAE (compare the rows of “Baseline” and “TSDNet”). Our proposed TSDNet can effectively distinguish whether the given area is text stroke or not. And this useful information can help TRGNet to remove masked areas more purposefully. When combining the WD and TSDNet (see “WD+TSDNet” in Table II), the performance can be further improved. Comparing the results in the second row of Fig. 9, it can be seen that our TSDNet can help to completely remove text from image (see the character “T”).

4) *Cascaded TSDNet and TRGNet (Cascade)*: Cascading of TSDNet and TRGNet can help to fix minor mistakes and slight text remnants of the first unit of TSDNet and TRGNet, such as, completing partial-detected text stroke, removing text residual, and fixing visual artifacts. We also experiment three cascade, and the text-erased results are a little bit blurry. A possible reason is that part of high frequency information is lost during cascading.

5) *The effect of stroke detection*: The text stroke detection is an important ingredient of our generic framework, here, we further discuss the effect of stroke detection performance on the final text removal. When inserting TSDNet into the Baseline, the performance is obviously improved, which validates our design of the TSDNet. Furthermore, the text stroke detection performance is enhanced via the cascaded design (tMAE of “Cascade” is significantly smaller than that of “TSDNet” in Table II), in the meanwhile, the text removal performance is better. This further illustrates that the improvement of stroke detection can enhance the final text removal result.

## V. CONCLUSION

In this work, we proposed a novel GAN-based framework to solve scene text removal task via decoupling text stroke detection and stroke removal. We designed and implemented a text stroke detection network and a text removal generation network, and constructed the final model by cascading the group of above two networks. Quantitative and qualitative results illustrate the superior performance of our proposed network. Our study implies that it is beneficial to know the position of text strokes for the scene text removal problem. To the best of our knowledge, our study is the first to reveal the importance of accurate text strokes to text removal task. In the meanwhile, we also constructed a versatile real-world dataset, including text images, ground-truth text-free images, and auxiliary masks, which can be used to benchmark text removal methods. Moreover, our approach can be used for quickly constructing the large scale text-free image dataset from images with text, and pixel-wise text stroke annotations can be obtained as well (*i.e.*, binarizing the difference between paired text image and text-free image). This kind of dataset will provide more and fine-grained supervised information to further improve the performance of scene text detection and recognition tasks.

Our method might generate implausible result if the text area is too large. We believe that a larger dataset with more diverse data can help to mitigate existing shortcomings. In the future, we plan to collect more real-world text images and construct a larger and richer dataset that can be used for both text removal task and other related research, *e.g.*, realistic text synthesis. In this work, we use the text region masks in an “off-line” manner, considering not very perfect performance of the current automatic text detectors and the requirement of partial text removal applications. We would like to design a more “complete” framework combining automatic text detection, which supports the refinement of possible detection errors and the selection of specific text region with simple user guidance. It would also be interesting to study text removal problem in the semi-supervised manner. We plan to share our source code and real-world dataset to the research community.

## REFERENCES

- [1] Khodadadi M, Behrad A. Text localization, extraction and inpainting in color images. In: Proceedings of the Iranian Conference on Electrical Engineering; 2012. p. 1035–1040.
- [2] Modha U, Dave P. Image Inpainting - Automatic Detection and Removal of Text From Images. International Journal of Engineering Research and Applications. 2012;2(2):930–932.
- [3] Wagh P D, Patil D R. Text detection and removal from image using inpainting with smoothing. In: Proceedings of the International Conference on Pervasive Computing; 2015. .
- [4] Johnson J, Alahi A, Fei-Fei L. Perceptual losses for real-time style transfer and super-resolution. In: European Conference on Computer Vision; 2016. p. 694–711.
- [5] Isola P, Zhu J-Y, Zhou T, et al. Image-to-Image Translation with Conditional Adversarial Networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 5967–5976.
- [6] Zhu J-Y, Park T, Isola P, et al. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In: Proceedings of the IEEE International Conference on Computer Vision; 2017. p. 2242–2251.
- [7] Nakamura T, Zhu A, Yanai K, et al. Scene Text Eraser. In: Proceedings of the International Conference on Document Analysis and Recognition. vol. 01; 2017. p. 832–837.
- [8] Zhang S, Liu Y, Jin L, et al. EnsNet: Ensconce Text in the Wild. In: Proceedings of the AAAI Conference on Artificial Intelligence; 2019. p. 801–808.
- [9] Tursun O, Zeng R, Denman S, et al. MTRNet: A Generic Scene Text Eraser. In: Proceedings of the International Conference on Document Analysis and Recognition; 2019. .
- [10] Iizuka S, Simo-Serra E, Ishikawa H. Globally and Locally Consistent Image Completion. ACM Transactions on Graphics. 2017;36(4):107:1–107:14.
- [11] Yu J, Lin Z, Yang J, et al. Generative Image Inpainting With Contextual Attention. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2018. .
- [12] Ye Q, Doermann D. Text Detection and Recognition in Imagery: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2015;37(7):1480–1500.
- [13] Shi B, Bai X, Belongie S. Detecting Oriented Text in Natural Images by Linking Segments. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 3482–3490.
- [14] Liu Y, Jin L, Zhang S, et al. Curved scene text detection via transverse and longitudinal sequence connection. Pattern Recognition. 2019;90:337 – 345.
- [15] Chen J, Lian Z, Wang Y, et al. Irregular scene text detection via attention guided border labeling. Science China Information Sciences. 2019;62(12):220103–.
- [16] He W, Zhang X-Y, Yin F, et al. Realtime multi-scale scene text detection with scale-based region proposal network. Pattern Recognition. 2020;98:107026.
- [17] Baek Y, Lee B, Han D, et al. Character Region Awareness for Text Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2019. p. 9365–9374.
- [18] Zhang C, Yao C, Shi B, et al. Automatic discrimination of text and non-text natural images. In: Proceedings of the International Conference on Document Analysis and Recognition; 2015. p. 886–890.
- [19] Matas J, Chum O, Urban M, et al. Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing. 2004;22(10):761 – 767.
- [20] Joachims T. Text categorization with Support Vector Machines: Learning with many relevant features. In: European Conference on Machine Learning; 1998. p. 137–142.
- [21] Bai X, Shi B, Zhang C, et al. Text/non-text image classification in the wild with convolutional neural networks. Pattern Recognition. 2017;66:437 – 446.
- [22] Zhao M, Wang R-Q, Yin F, et al. Fast Text/non-Text Image Classification with Knowledge Distillation. In: Proceedings of the International Conference on Document Analysis and Recognition; 2019. p. 1458–1463.
- [23] Gupta N, Jalal A S. Text or Non-text Image Classification using Fully Convolution Network (FCN). In: Proceedings of the International Conference on Contemporary Computing and Applications; 2020. p. 150–153.
- [24] Zhou X, Yao C, Wen H, et al. EAST: An Efficient and Accurate Scene Text Detector. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2017. p. 2642–2651.
- [25] Ren Y, Yu X, Zhang R, et al. StructureFlow: Image Inpainting via Structure-Aware Appearance Flow. In: Proceedings of the IEEE International Conference on Computer Vision; 2019. .

- [26] Yu J, Lin Z, Yang J, et al. Free-Form Image Inpainting With Gated Convolution. In: Proceedings of the IEEE International Conference on Computer Vision; 2019. .
- [27] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In: Medical Image Computing and Computer-Assisted Intervention; 2015. p. 234–241.
- [28] Miyato T, Kataoka T, Koyama M, et al. Spectral Normalization for Generative Adversarial Networks. In: International Conference on Learning Representations; 2018. .
- [29] Tran D, Ranganath R, Blei D. Hierarchical Implicit Models and Likelihood-Free Variational Inference. In: Proceedings of the Advances in Neural Information Processing Systems; 2017. p. 5523–5533.
- [30] Zhang H, Goodfellow I, Metaxas D, et al. Self-Attention Generative Adversarial Networks. In: Proceedings of the International Conference on Machine Learning; 2019. p. 7354–7363.
- [31] Gatys L A, Ecker A S, Bethge M. Image Style Transfer Using Convolutional Neural Networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 2414–2423.
- [32] Aly H A, Dubois E. Image up-sampling using total-variation regularization with a new observation model. *IEEE Transactions on Image Processing*. 2005;14(10):1647–1659.
- [33] Gupta A, Vedaldi A, Zisserman A. Synthetic Data for Text Localisation in Natural Images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 2315–2324.
- [34] Nayef N, Yin F, Bizid I, et al. ICDAR2017 Robust Reading Challenge on Multi-Lingual Scene Text Detection and Script Identification - RRC-MLT. In: Proceedings of the IAPR International Conference on Document Analysis and Recognition; 2017. p. 1454–1459.
- [35] Dutta A, Zisserman A. The VIA annotation software for images, audio and video. In: Proceedings of the 27th ACM International Conference on Multimedia; 2019. p. 2276–2279.
- [36] Wolf C, Jolion J-M. Object count/Area Graphs for the Evaluation of Object Detection and Segmentation Algorithms. *International Journal of Document Analysis and Recognition*. 2006;8(4):280–296.
- [37] Kingma D P, Ba J. Adam: A method for stochastic optimization. In: International Conference on Learning Representations; 2015. .