Modeling Stroke Mask for End-to-End Text Erasing

Xiangcheng Du\textsuperscript{1*}, Zhao Zhou\textsuperscript{1,2*}, Yingbin Zheng\textsuperscript{2}, Tianlong Ma\textsuperscript{1}, Xingjiao Wu\textsuperscript{1}, Cheng Jin\textsuperscript{1,3†}

\textsuperscript{1}School of Computer Science, Fudan University, Shanghai, China \textsuperscript{2}Videt Technology, Shanghai, China \textsuperscript{3}Key Laboratory of Culture & Tourism Intelligent Computing of Ministry of Culture & Tourism, Shanghai, China

Abstract

Scene text erasing aims to wipe text regions in scene images with reasonable background. Most previous approaches employ scene text detectors to assist localization of the text regions. However, detected text boxes contain both text strokes and background clutters, and directly inpainting on the whole boxes may remain text artifacts and make regions unnatural. In this paper, we present an end-to-end network that focuses on modeling text stroke masks that provide more accurate locations to compute erased images. The network consists of two stages, i.e., a basic network with stroke generation and a refinement network with stroke awareness. The basic network predicts the text stroke masks and initial erasing results simultaneously. The refinement network receives the masks as supervision to generate natural erased results. Experiments on both synthetic and real-world scene images demonstrate the effectiveness of our framework in producing high quality erasing results.

1. Introduction

Scene text contains numerous privacy information, and exposed information can easily be used for illegal purposes. Recently, scene text erasing technology has obtained much attention and is widely used in securing sensitive private information [13, 20]. The text erasing task aims to erase text and maintain original background content as possible, which is extremely challenging.

A popular practice is to decompose the text erasing into two subtasks, i.e., text detection and image inpainting. The networks for these subtasks can be trained separately, and the text contents are erased within the text bounding boxes. With the development of recent text detectors and image inpainting, the approaches under this pipeline achieve good performance on the scene text erasing benchmarks.

However, due to the text boxes from text detectors containing both text strokes and background content, directly inpainting the whole boxes will also replace the background content, which may make the regions unnatural. Some erasing samples are illustrated in Figure 1. We believe accurate modeling of the real text segments would improve the visual effects of the erasing and reduce the artifacts.

In this paper, we propose a new erasing method called Stroke-Aware Erasing Network (SAEN), which is built upon the modeling of the text stroke mask to compute the erased images. Starting from a basic encoder-decoder network to generate an initial erased image, an additional decoder is connected after the basic encoder and designed to predict the text stroke masks which provide the text location information. Since the erased regions usually contain remnants of texts, we further introduce another encoder-decoder structure that considers the concatenation of predicted stroke and imprecise initial erased image as input, leading to the stroke-aware module to generate a more realistic and natural erased image.

\*These authors contributed to the work equally.
†Corresponding author. Email: jc@fudan.edu.cn

Figure 1. From left to right: original scene images, text erased images by text boxes-aware approach, and text erased images by the proposed method. Text boxes-aware results are unnatural due to redundant background content, and our method leverages text stroke masks to generate more natural results.
Our contributions are summarized as follows:

- We propose an end-to-end stroke-aware erasing network, which can naturally and effectively erase scene texts. It could distinguish text strokes from the non-text areas and provide important location information for text erasure.

- We design a stroke-aware erasing module, which could enhance erasing performance. The module aims to erase residual text by combining the initial-erased results and stroke masks. Meanwhile, adversarial techniques are applied to generate natural images.

- Extensive evaluations demonstrate the superior performance of the proposed method. Notably, we achieve state-of-the-art results on SCUT-Syn and SCUT-EnsText datasets compared with the existing methods.

The rest of our paper is organized as follows. Section 2 briefly reviews the scene text erasing approach. In Section 3, we give the details of the proposed Stroke-Aware Erasing Network. We show the experimental results conducted on the datasets to demonstrate the effectiveness in Section 4. Section 5 concludes the paper.

2. Related Work

Early text erasing approaches mainly focus on the text properties, such as color, fonts, and layout. The text regions are first extracted using a color histogram or threshold-based methods [7, 4] and then pixel similarity is calculated to replace the text area with the background. These methods are limited to complex scenarios. With the advancement of deep learning, methods based on deep inpainting and text detection are designed to handle scene text images.

Scene Text Eraser [13] is the first method that addresses the scene text erasing with the CNN. The method divides the image into small patches and uses a U-Net shaped neural network to erase the text. This patch-based processing fails to localize text with complex shapes and inevitably damaged the consistency of erased results. [19] adopts a separately trained scene text detector and an inpainting network. The text detector predicts segmentation maps of text instances that are fed into the inpainting network. The inpainting network fills in text regions and generates a final image. At inference time, the two modules are connected into one model for scene text erasing. [3] also adopts a two-stage approach and optimizes the mask generator and inpainting model separately. [17] introduces MTRNet that considers text region mask as an additional input into the network. However, the disadvantage of this method is that it requires text region masks as auxiliary information, which means that text removal cannot be guaranteed without additional information. There are also one-stage approaches that combine the two subtasks into a unified network. [20] designs an end-to-end architecture that employs a GAN with a UNet shaped encoder-decoder generator, and trains it with several losses. [10] proposes an end-to-end network that consists of a generator as well as a discriminator conditioned on the text regions. [18] introduces PERT that explicitly erases with a progressive strategy to remove texts accurately. [16] extends MTRNet, which is free from text location methods. [11] introduces the GAN-based EraseNet which adopts a coarse-to-fine erasure architecture with an additional segmentation head for learning text mask to help with text region localization. [2] proposes end-to-end framework consists of a text stroke detection network and a text removal generation network. [9] uses text synthesis module for text erasing task. The synthesis module aims to generate training samples and learn a policy network by selecting more realistic training data.

In this paper, we also employ the one-stage strategy to avoid the complex training process for separate text detection and inpainting networks. Different from previous one-stage approaches, we focus on modeling the text stroke masks and finally achieve outstanding results. Very recently, [15] also utilizes the text stroke for erasing, where the text regions are firstly extracted by an off-the-shelf text detector [1]. [15] focuses the text erasing on the cropped text regions. The text detector and text eraser are trained separately, and the text erasing depends on the results of text detectors learned from extra datasets. In contrast, our method adopts an end-to-end framework, which reduces the complexity of the text erasure algorithm. Without auxiliary images and annotations to train the text detector. Moreover, [15] needs the ground truth of the text stroke in the erasing scheme, which is a huge challenge for real-world datasets. In our framework, the text stroke is calculated from the original image and the ground truth without manual annotations. In addition, our method outperforms these state-of-the-art methods as shown in the experiments.

3. Method

In this section, we give the details of the proposed SAEN. As shown in Figure 2, the stroke-aware erasing network consists of a basic network, text stroke generation decoder, and the stroke-aware module. Here we focus on pixel-level text erasing, and introduce accurate text stroke masks to help to improve the performance of the process. The basic network is designed to generate initial erased images. Predicted text stroke can be fully exploited by merging with initial erase results. The stroke-aware module aims to erase the residual text and mask the text region more naturally. We start by introducing the basic network for stroke and initial erased image generation.
3.1. Basic Encoder-Decoder Structure

The basic network (BasicNet) employs the U-Net-like architecture composed of an encoder and decoder. The encoder follows the typical architecture of a convolutional network, composed of repeated convolutional layers and residual basic blocks. We control the stride parameter to reduce the resolution of the feature map. At each downsampling step, we double the number of feature channels and half the size of feature maps. The residual basic blocks follow ResNet [6] to retain more shallow information. The decoder consists of five up-sampling transposed convolutional layers with kernel size, stride step, and padding size set to 3, 2, 1, respectively. Additionally, the features from shallow layers are concatenated with the high-level ones to recover the structure and the texture details. The architecture of the BasicNet is shown in Figure 3(a). Given the original image \( I \), the initial erased image \( I_1 \) is obtained by going through the basic network \( \Theta_{bn} \), i.e.,

\[
I_1 = \Theta_{bn}(I)
\]  

(1)

3.2. Stroke Decoder

To generate the text stroke mask, a second stroke decoder is connected after the basic encoder and designed with the similar structure of the BasicNet decoder. The learning target of this module, i.e., the ground-truth text stroke mask \( S_{gt} \), is computed by subtracting original image \( I \) from ground-truth image \( I_{gt} \) and then thresholding. Specifically, when the absolute value of the difference is less than the set threshold, we set the value of \( S_{gt} \) at the corresponding position to 1; otherwise, the value is 0. Different threshold values result in generating different strokes (see Figure 4), which will further affect the erasing result directly; we will discuss the choice of this parameter in the evaluations.

The inputs of the stroke decoder are the output feature maps from the 6-th residual block. We introduce \( 3 \times 3 \) convolutional operation between each transposed convolutional layer in the stroke decoder to reduce dimension and improve the representation ability of the features. The up-sampled features maps are concatenated with the corresponding ones from the down-sampling step. With the combination of BasicNet encoder and stroke decoder \( (\Theta_{sd}) \), the text stroke \( S_p \) can be predicted:

\[
S_p = \Theta_{sd}(I)
\]  

(2)

3.3. Modeling Stroke Mask

The initial erased image \( I_1 \) has an approximate result, but sometimes retains slight text residue and the whole image is intuitively unnatural. The factors leading to this situation can be attributed to the lack of sufficient supervision. To address the issue and improve the erasing performance, we jointly model stroke mask \( S_p \) and initial output \( I_1 \).

As shown in Figure 2, we design a stroke-aware encoder-decoder module as the second stage. The stroke obtained from the stroke decoder assists in text localization for high-quality erased results. The encoder receives the initial erased results \( I_1 \) concat with the text stroke \( S_p \) as input. The encoder obtains more accurate text stroke distribution...
in an incremental manner by combining the predicted stroke masks and thus can remove residual text effectively. The stroke-aware module $\Theta_{sam}$ returns the final erased image $I_2$, which has the same resolution as $I_1$, i.e.,

$$I_2 = \Theta_{sam}(I_1, S_p)$$  (3)

### 3.4. Model Training

Several loss terms are employed to train a robust erasing model. We adopt the reconstruction loss to measure pixel-wise similarity between the ground-truth and predicted erased images. Since the task focuses on the text regions, we set different weights for text stroke pixels and background pixels and formulate the reconstruction loss as

$$L_r = \frac{1}{N} \sum_{k=1}^{N} \left\{ \| (I_{gt}^k - I_2^k) \ast (1 - S_{gt}^k) \|_1 + \lambda_r \| (I_{gt}^k - I_2^k) * S_{gt}^k \|_1 \right\}$$  (4)

where $I_{gt}^k$ and $S_{gt}^k$ are the $k$-th ground-truth erased image and stroke mask, $I_2^k$ is the $k$-th predicted erased image, $N$ is the image number, $\ast$ denotes element-wise multiplication, and $\lambda_r$ is the weights for the non-stroke pixels.

The stroke loss is then designed for the predictions of the stroke mask. Text stroke usually occupies a small proportion in entire scene images. Therefore, to reduce the imbalance issue, we adopt the Dice [12] between the ground-truth stroke $S_{gt}$ and prediction stroke $S_p$ are computed. The stroke loss is calculated as

$$L_s = \frac{1}{N} \sum_{k=1}^{N} \{1 - \text{Dice}(S_p^k, S_{gt}^k)\}$$  (5)

$$\text{Dice}(S_p^k, S_{gt}^k) = \frac{2 \cdot \sum_{i,j} (S_p^k[i,j] \cdot S_{gt}^k[i,j])}{\sum_{i,j} (S_p^k[i,j])^2 + \sum_{i,j} (S_{gt}^k[i,j])^2}$$  (6)

where $S_p^k[i,j]$ denotes the value of stroke mask $S_p^k$ at the coordinate $(i,j)$.

In addition, we utilize the adversarial loss $L_a$ to generate realistic and natural erased images, which is formulated as

$$L_a = -\frac{1}{N} \sum_{k=1}^{N} D(I_2^k)$$  (7)

where $D(I)$ is an one-dimensional output of discriminator $D$ denoting the probability of an erased image $I$ as fake generated images. We follow the design in [11] and $D$ is an extra classification network with VGG structure. We train the discriminator iteratively together with the proposed erasing network with a loss function $L_D = \frac{1}{N} \sum_k \{\text{ReLU}(1 - D(I_{gt}^k)) + \text{ReLU}(1 + D(I_2^k))\}$.

Finally, the overall loss function $L$ is expressed as a combination of these losses, i.e.,

$$L = L_r + \lambda_s L_s + \lambda_a L_a$$  (8)

### Implementation Details

Our text-erased model is implemented using PyTorch [14]. We resize images to 512 × 512 and normalize the pixel values to the range of (0, 1) with random horizontal flip and rotation. The default threshold is 35 for ground-truth stroke generation. For the loss function, we set $\lambda_r = 0.2$, $\lambda_s = 0.2$, and $\lambda_a = 0.01$. The network is initialized using a Gaussian distribution and the Adam optimizer is used with $\beta = (0.5, 0.9)$. The initial learning rate is set to $10^{-4}$. We set the $\beta$ of the discriminator $D$ to $(0.0, 0.9)$ with learning rate of $10^{-5}$. For other settings of $D$, we follow [11]. The training process of the proposed stroke-aware network reaches convergence after 300 epochs.

### 4. Experiment

#### Datasets

We evaluate the proposed stroke-aware network on both the synthesis and real-world scene image datasets. For the synthesis data, we use the SCUT-Syn dataset [11], which is generated by text synthesis technology [5]. The
Table 1. Comparisons with previous approaches on the SCUT-Syn and SCUT-EnsText datasets. The bold and underline results denote the best performer and the runner-up in each metric, respectively.

<table>
<thead>
<tr>
<th>SCUT-EnsText</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
<th>MSE↓</th>
<th>AGE↓</th>
<th>pEPS↓</th>
<th>pCEPS↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>SceneTextEraser [13]</td>
<td>25.47</td>
<td>90.14</td>
<td>0.0047</td>
<td>6.01</td>
<td>0.0533</td>
<td>0.0296</td>
</tr>
<tr>
<td>Pix2Pix [8]</td>
<td>26.70</td>
<td>88.56</td>
<td>0.0037</td>
<td>6.09</td>
<td>0.0480</td>
<td>0.0270</td>
</tr>
<tr>
<td>EnsNet [20]</td>
<td>29.54</td>
<td>92.47</td>
<td>0.0024</td>
<td>4.16</td>
<td>0.0307</td>
<td>0.0136</td>
</tr>
<tr>
<td>EraseNet [11]</td>
<td>32.30</td>
<td>95.42</td>
<td>0.0015</td>
<td>3.02</td>
<td>0.0160</td>
<td>0.0090</td>
</tr>
<tr>
<td>CRAFT+SBSTE [15]</td>
<td><strong>35.34</strong></td>
<td>96.24</td>
<td>0.0009</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PERT [18]</td>
<td>33.25</td>
<td><strong>96.95</strong></td>
<td>0.0014</td>
<td>2.18</td>
<td>0.0136</td>
<td>0.0088</td>
</tr>
<tr>
<td>SAEN [This paper]</td>
<td>34.75</td>
<td>96.53</td>
<td><strong>0.0007</strong></td>
<td><strong>1.98</strong></td>
<td><strong>0.0125</strong></td>
<td><strong>0.0073</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCUT-Syn</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
<th>MSE↓</th>
<th>AGE↓</th>
<th>pEPS↓</th>
<th>pCEPS↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>SceneTextEraser [13]</td>
<td>14.68</td>
<td>46.13</td>
<td>0.7148</td>
<td>13.29</td>
<td>0.1859</td>
<td>0.0936</td>
</tr>
<tr>
<td>Pix2Pix [8]</td>
<td>25.60</td>
<td>89.86</td>
<td>0.2465</td>
<td>5.60</td>
<td>0.0571</td>
<td>0.0423</td>
</tr>
<tr>
<td>EnsNet [20]</td>
<td>37.36</td>
<td>96.44</td>
<td>0.0021</td>
<td>1.73</td>
<td>0.0069</td>
<td>0.0020</td>
</tr>
<tr>
<td>MTRNet [17]</td>
<td>29.71</td>
<td>94.43</td>
<td>0.0004</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MTRNet++ [16]</td>
<td>34.55</td>
<td><strong>98.45</strong></td>
<td>0.0004</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EraseNet [11]</td>
<td>38.32</td>
<td>97.67</td>
<td><strong>0.0002</strong></td>
<td>1.60</td>
<td>0.0048</td>
<td><strong>0.0004</strong></td>
</tr>
<tr>
<td>Weak Supervision [19]</td>
<td>37.46</td>
<td>93.64</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PERT [18]</td>
<td><strong>39.40</strong></td>
<td>97.87</td>
<td><strong>0.0002</strong></td>
<td>1.41</td>
<td>0.0045</td>
<td>0.0006</td>
</tr>
<tr>
<td>SAEN [This paper]</td>
<td>38.63</td>
<td>98.27</td>
<td><strong>0.0003</strong></td>
<td><strong>1.39</strong></td>
<td><strong>0.0043</strong></td>
<td><strong>0.0004</strong></td>
</tr>
</tbody>
</table>

Table 2. Comparison between different settings of the network on SCUT-EnsText. (a) The effect of different stages. (b) The effect of stroke mask modeling, where Box-aware indicates the network replacing the stroke units with text box mask. (c) Different terms in loss functions. (d) The results of the model trained with different thresholds for the stroke generation.

<table>
<thead>
<tr>
<th>Settings</th>
<th>PSNR↑</th>
<th>SSIM↑</th>
<th>MSE↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) BasicNet only</td>
<td>29.18</td>
<td>94.54</td>
<td>0.0026</td>
</tr>
<tr>
<td>Stage-I only</td>
<td>32.27</td>
<td>95.07</td>
<td>0.0017</td>
</tr>
<tr>
<td>Stage-I+Stage-II</td>
<td>34.75</td>
<td>96.53</td>
<td>0.0007</td>
</tr>
<tr>
<td>SAEN+</td>
<td>34.91</td>
<td>96.64</td>
<td>0.0006</td>
</tr>
<tr>
<td>(b) Box-aware</td>
<td>33.26</td>
<td>95.81</td>
<td>0.0008</td>
</tr>
<tr>
<td>Stroke-aware</td>
<td>34.75</td>
<td>96.53</td>
<td>0.0007</td>
</tr>
<tr>
<td>(c) $L_r$</td>
<td>39.74</td>
<td>95.85</td>
<td>0.0019</td>
</tr>
<tr>
<td>$L_r$, $L_s$</td>
<td>33.94</td>
<td>95.76</td>
<td>0.0009</td>
</tr>
<tr>
<td>$L_r$, $L_s$, $L_a$</td>
<td>34.75</td>
<td>96.53</td>
<td>0.0007</td>
</tr>
<tr>
<td>Threshold=20</td>
<td>33.27</td>
<td>95.39</td>
<td>0.0010</td>
</tr>
<tr>
<td>Threshold=25</td>
<td>33.86</td>
<td>95.87</td>
<td>0.0008</td>
</tr>
<tr>
<td>Threshold=30</td>
<td>34.40</td>
<td>96.15</td>
<td>0.0008</td>
</tr>
<tr>
<td>Threshold=35</td>
<td>34.75</td>
<td>96.53</td>
<td>0.0007</td>
</tr>
<tr>
<td>Threshold=40</td>
<td>34.56</td>
<td>96.27</td>
<td>0.0007</td>
</tr>
<tr>
<td>Threshold=45</td>
<td>34.38</td>
<td>96.33</td>
<td>0.0009</td>
</tr>
</tbody>
</table>

**Evaluation metrics** We adopted the image inpainting metrics to evaluate our method. The first set of metrics measured the statistics between the ground-truth and predicted image and is widely used in image processing tasks, including PSNR (peak signal-to-noise ratio), MSE (mean squared error), and SSIM (structural similarity). We also employ three metrics that focus on pixel-level information: AGE that computes the average of gray-level absolute difference, pEPS to calculate the percentage of error pixels, and pCEPS to calculate percentage of four-connected neighbors error.

For the real-world scene images, we evaluate the performance on the SCUT-EnsText dataset [11]. The dataset contains 2,749 training and 813 testing images, which are selected from public scene text detection benchmarks. The dataset takes into account the diversity of text types, including text fonts, text orientations, text shapes and scene diversity. The erased ground-truth is computed by manually erasing all text instances using Adobe Photoshop. Figure 5(b) shows some examples of the dataset.

Figure 5. Sample images and the ground-truth erased images.
pixels. A higher value of PSNR and SSIM or lower MSE, AGE, pEPs, and pCEPS represent better results.

4.1. Comparison with State-of-the-arts

Table 1 summarizes all metrics for both datasets. We compare SAEN with several baselines and state-of-the-art approaches. Among them, the first group contains the well-known general image inpainting approach (Pix2Pix). We also compare ours with the two-stage methods (SceneTextEraser, CRAFT+SBSTE), the one-stage methods (EnsNet, MTRNet, MTRNet++, EraseNet), as well as the recent progressive network (PERT).

For both the synthesis data and real-world images, SAEN can reach a comparable results under different metrics. Compared with the other one-stage erasing methods, SAEN shows very competitive performance with PSNR of 34.75 (SCUT-EnsText) and 38.63 (SCUT-Syn), which outperform the other one-stage erasing methods and is significantly better for the real-world images. Compared with the two-stage method [15] that also employ the stroke information, our method performs better five metrics and achieves comparable results on PSNR, while ours do not rely on the text detector trained from auxiliary data. We also compare SAEN with recent progressive method [18]. Our network is more compact and we can achieve a comparable performance. The results confirm the effectiveness of the proposed text stroke modeling approach for erasing texts from scene images. Figure 6 shows some text erased images from different methods from the SCUT-EnsText dataset.

4.2. Ablation Study

In this subsection, we evaluate the alternative implementation and parameters for constructing the stroke-aware erasing network. We report results on the SCUT-EnsText dataset as it is more challenging than the other.

Network components. In Table 2(a), we first evaluate the effect of different network components. Using the baseline encoder-decoder structure without stroke module, the eraser reaches a PSNR of 29.18 and SSIM of 94.93. Incorporating the stroke decoder boosts the PSNR with a margin of 3.09, indicating that the representation ability of the BasicNet encoder is improved with the help of stroke decoder.
When we leverage the stroke-aware encoder and decoder in stage-II, we obtain a PSNR of 34.75 and SSIM of 96.53. To understand the effect of stage-II, we conduct qualitative comparisons on Figure 7 and observe that the erased images by full model are more clean and natural than the results by the model with only stage-I. Moreover, we also add another refinement network (with the same structure of stage-II) and denote it as SAEN+. The images from SAEN+ reach slightly better erasing results than those from stage-II, while the overall performance tends to be saturated. In order to make a good trade-off between performance and model efficiency, we adapt the two-stage model in this work.

The importance of stroke mask. The outputs from different stages in SAEN are further visualized in Figure 8. We can see that it is hard for a single network to erase the text completely since the stroke information is embedded implicitly. The stroke decoder can generate masks that consist of most real text strokes as well as some text-like edges and symbols. With the stroke-aware module, the residual texts are then erased and the refined results can be obtained.

Recall that previous methods [19, 15] utilize text box masks to locate text and then enhance the performance. We believe the stroke generator helps to provide clues to the erased texts. Therefore, we quantitatively compared the results of box mask aware and text stroke aware on text erasing. Table 2(b) shows that the stroke-aware erasing model has better performance than the box mask aware method. It has gained 1.49 and 0.72 improvements on PSNR and SSIM compared with box-aware network. Some qualitative erased results are illustrated in Figure 7(c,d).

Loss function. The weights of the full model are learned by three terms, i.e., the reconstruction loss, the stroke loss, and the adversarial loss, as described in Section 3.4. Adding the adversarial loss to the loss function leads to a slight improvement of the metrics (e.g., 0.81 for PSNR and 0.77 for SSIM, see Table 2(c)). More importantly, from the qualita-
Figure 9. Qualitative comparisons with the model trained with different thresholds for the stroke generation.

In Figure 9, we show qualitative comparisons with the model trained with different thresholds for the stroke generation. We can see that the erased images are more realistic and the generated regions are more smooth. When considering the reconstruction loss only, the network degenerates to a stack of two encoder-decoder modules. The results (PSNR 30.74, SSIM 94.85) are much lower than the full model but close to the result of BasicNet only, which also validate the importance of stroke in our framework.

Threshold for generating stroke mask. We compare the performance of our frameworks with different settings of the threshold for generating stroke masks. As illustrated in Figure 4, a low threshold usually contains redundant background, while some text strokes are filtered out with a high threshold. According to the results shown in Table 2(d), under the same experimental setting, threshold=35 is a suitable trade-off for the erasing. Figure 9 shows the outputs of SAEN with different thresholds. With the appropriate threshold, SAEN can restore a natural background.

Failure case analysis. There are generally two common types of failure text erasing. The first is the false-erased text-like symbols, as illustrated in the top row of Figure 11. We also observe that some small curve text regions are ignored, due to failure of the stroke prediction.

Figure 10. Erased images with different loss functions.

Figure 10 shows the outputs of SAEN with different loss functions. We observe that the erased images are more realistic when using the full loss function ($L_r + L_s + L_a$).

Figure 11. Samples of failure cases of our method. The failures are highlighted with red bounding box.

5. Conclusions

We introduced SAEN, a simple yet effective framework for scene text erasing. SAEN is an end-to-end framework that consists of a basic erasing network and a text stroke decoder, as well as the stroke-aware module. In particular, the basic network first uses a shared encoder to learn the representation and feed into two decoders that generate the predicted stroke and initial erased image, respectively. The stroke-aware module aims to repair the imprecise initial erased image with more discriminative features. This is achieved by concatenating the initial erased image and predicted stroke and then sending them to the module. We conduct extensive experiments on SCUT-Syn and SCUT-EnsText for both synthesis and real-world scene text erasing. The results demonstrate the effectiveness of the proposed approach.

Acknowledgement. This work was supported by National Archives Administration of China Research Program (2021-X-25), Inovation Program of Ministry of Culture and Tourism (CXGCXM2021-039), and Shanghai Archives Research Program (2108).
References


