Mask TextSpotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes

[1]Pengyuan Lyu*, [1]Minghui Liao*, [2]Cong Yao, [2]Wenhao Wu, [1]Xiang Bai

[1]School of Electronic Information and Communications,
Huazhong University of Science and Technology
[2]Megvii (Face++) Technology Inc.
lvpyuan@gmail.com, mhliao@hust.edu.cn, yaocong2010@gmail.com,
wwh@megvii.com, xbai@hust.edu.cn

Abstract. Recently, models based on deep neural networks have dominated the fields of scene text detection and recognition. In this paper, we investigate the problem of scene text spotting, which aims at simultaneous text detection and recognition in natural images. An end-to-end trainable neural network model for scene text spotting is proposed. The proposed model, named as Mask TextSpotter, is inspired by the newly published work Mask R-CNN. Different from previous methods that also accomplish text spotting with end-to-end trainable deep neural networks, Mask TextSpotter takes advantage of simple and smooth end-to-end learning procedure, in which precise text detection and recognition are acquired via semantic segmentation. Moreover, it is superior to previous methods in handling text instances of irregular shapes, for example, curved text. Experiments on ICDAR2013, ICDAR2015 and Total-Text demonstrate that the proposed method achieves state-of-the-art results in both scene text detection and end-to-end text recognition tasks.

Keywords: Scene Text Spotting, Neural Network, Arbitrary Shapes

1 Introduction

In recent years, scene text detection and recognition have attracted growing research interests from the computer vision community, especially after the revival of neural networks and growth of image datasets. Scene text detection and recognition provide an automatic, rapid approach to access the textual information embodied in natural scenes, benefiting a variety of real-world applications, such as geo-location, instant translation, and assistance for the blind.

Scene text spotting, which aims at concurrently localizing and recognizing text from natural scenes, have been previously studied in numerous works [4, 5]. However, in most works, except [2] and [3], text detection and subsequent recognition are handled separately. Text regions are first hunted from the original image by a trained detector and then fed into a recognition module. This procedure seems simple and natural, but might lead to sub-optimal performances for both detection and recognition, since these two tasks are highly correlated

* Authors contribute equally.

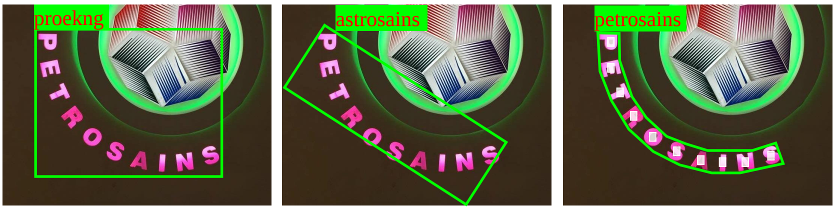

Fig. 1: Illustrations of different text spotting methods. The left presents horizontal text spotting methods [1, 2]; The middle indicates oriented text spotting methods [3]; The right is our proposed method. Green bounding box: detection result; Red text in green background: recognition result.

and complementary. On one hand, the quality of detections larges determines the accuracy of recognition; on the other hand, the results of recognition can provide feedback to help reject false positives in the phase of detection.

Recently, two methods [2, 3] that devise end-to-end trainable frameworks for scene text spotting have been proposed. Benefiting from the complementarity between detection and recognition, these unified models significantly outperform previous competitors. However, there are two major drawbacks in [2] and [3]. First, both of them can not be completely trained in an end-to-end manner. [2] applied a curriculum learning paradigm [6] in the training period, where the sub-network for text recognition is locked at the early iterations and the training data for each period is carefully selected. Busta *et al.* [3] at first pre-train the networks for detection and recognition separately and then jointly train them until convergence. There are mainly two reasons that stop [2] and [3] from training the models in a smooth, end-to-end fashion. One is that the text recognition part requires accurate locations for training while the locations in the early iterations are usually inaccurate.The other is that the adopted LSTM [7] or CTC loss [8] are difficult to optimize than general CNNs. The second limitation of [2] and [3] lies in that these methods only focus on reading horizontal or oriented text. However, the shapes of text instances in real-world scenarios may vary significantly, from horizontal or oriented, to curved forms.

In this paper, we propose a text spotter named as *Mask TextSpotter*, which can detect and recognize text instances of arbitrary shapes. Here, *arbitrary shapes* mean various forms text instances in real world might exhibit. Inspired by Mask R-CNN [9], which can generate shape masks of objects, we detect text by segment the instance text regions. Thus our detector is able to detect text of arbitrary shapes. Besides, different from the previous sequence-based recognition methods [10, 11, 12] which are designed for 1-D sequence, we recognize text via semantic segmentation in 2-D space, to solve the issues in reading irregular text instances. Moreover, another advantage of the method is that it does not require accurate locations for recognition. Therefore, the detection task and recognition task can be completely trained end-to-end, and benefiting from feature sharing and joint optimization.

We validate the effectiveness of our model on the datasets that include horizontal, oriented and curved text. The results demonstrate the advantages of the proposed algorithm in both text detection and end-to-end text recognition tasks. Specially, on ICDAR2015, evaluated at a single scale, our method achieves an F-Measure of 0.86 on the detection task and outperforms the previous top performers by 13.2% – 25.3% on the end-to-end recognition task.

The main contributions of this paper are four-fold. (1) We propose an end-to-end trainable model for text spotting, which enjoys a simple, smooth training scheme. (2) The proposed method can detect and recognize text of various shapes, including horizontal, oriented, and curved text. (3) In contrast to previous methods, precise text detection and recognition in our method are accomplished via semantic segmentation. (4) Our method achieves state-of-the-art performances in both text detection and text spotting on various benchmarks.

2 Related Work

2.1 Scene Text Detection

In scene text recognition systems, text detection plays an important role. A large number of methods have been proposed to detect scene text [13, 14, 15, 16, 17, 5, 18, 19, 20, 1, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]. Most conventional methods of text detection are designed for detecting horizontal text with handcraft features. Some methods based on SWT [13, 31] and MSER [14, 15, 16] first detect character candidates and then group character candidates to words or text lines. In [5], Jaderberg *et al.* use Edge Boxes [32] to generate proposals and refine candidate boxes by regression. Zhang *et al.* [19] detect scene text by exploiting the symmetry property of text. Recently, deep learning based methods have become the mainstream of scene text detection. Adapted from Faster R-CNN [33] and SSD [34] with well-designed modifications, [20, 1] are proposed to detect horizontal words directly.

Multi-oriented scene text detection has become a hot topic recently. Yao *et al.* [21] and Zhang *et al.* [22] detect multi-oriented scene text by semantic segmentation. Liu *et al.* [23] and He *et al.* [24] design multi-oriented anchors [33] or default boxes [34] to detect oriented text. Tian *et al.* [25] and Shi *et al.* [26] propose methods which first detect text segments and then link them into text instances by spatial relationship or link predictions. Zhou *et al.* [27] and He *et al.* [28] regress text boxes directly from dense segmentation maps. Lyu *et al.* [29] propose to detect and group the corner points of the text to generate text boxes. Rotation-sensitive regression for oriented scene text detection is proposed by Liao *et al.* [30].

Compared to the popularity of horizontal or multi-oriented scene text detection, there are few works focusing on text instances of arbitrary shapes. Recently, detection of text with arbitrary shapes has gradually drawn the attention of researchers due to the application requirements in the real-life scenario. In [35], Risnumawan *et al.* propose a system for arbitrary text detection based on text symmetry properties. In [36], a dataset which focuses on curve orientation text detection is proposed. Different from most of the above-mentioned methods, we propose to detect scene text by instance segmentation which can detect text with arbitrary shapes.

2.2 Scene Text Recognition

Scene text recognition aims at decoding the detected or cropped image regions into character sequences. The previous scene text recognition approaches can be roughly split into three branches: character-based methods, word-based methods, and sequence-based methods. The character-based recognition methods [37, 38] mostly first localize individual characters and then recognize and group them into words. In [39], Jaderberg *et al.* propose a word-based method which treats text recognition as a common English words (90k) classification problem. Sequence-based methods solve text recognition as a sequence labeling problem. In [11], Shi *et al.* use CNN and RNN to model image features and output the recognized sequences with CTC [8]. In [12, 10], Lee *et al.* and Shi *et al.* recognize scene text via attention based sequence-to-sequence model.

The proposed text recognition component in our framework can be classified as a character-based method. However, in contrast to previous character-based approaches, we use an FCN [40] to localize and classify characters simultaneously. Besides, compared with sequence-based methods which are designed for a 1-D sequence, our method is more suitable to handle irregular text (multi-oriented text, curved text *et al.*).

2.3 Scene Text Spotting

Most of the previous text spotting methods [5, 1, 41, 42] split the spotting process into two stages. They first use a scene text detector [5, 1, 42] to localize text instances and then use a text recognizer [39, 11] to obtain the recognized text. In [2, 3], Li *et al.* and Busta *et al.* propose end-to-end methods to localize and recognize text in a unified network, but require relatively complex training procedures. Compared with these methods, our proposed text spotter can not only be trained end-to-end completely, but also has the ability to detect and recognize arbitrary-shape (horizontal, oriented, and curved) scene text.

2.4 General Object Detection and Semantic Segmentation

With the rise of deep learning, general object detection and semantic segmentation have achieved great development. A large number of object detection and segmentation methods [43, 44, 33, 45, 46, 34, 47, 40, 48, 49, 9] have been proposed. Benefited from those methods, scene text detection and recognition have achieved obvious progress in the past few years. Our method is also inspired by those methods. Specifically, our method is adapted from a general object instance segmentation model Mask R-CNN [9]. However, there are key differences between the mask branch of our method and that in Mask R-CNN. Our mask branch can not only segment text regions but also predict character probability maps, which means that our method can be used to recognize the instance sequence inside character maps rather than predicting an object mask only.

3 Methodology

The proposed method is an end-to-end trainable text spotter, which can handle various shapes of text. It consists of an instance-segmentation based text detector and a character-segmentation based text recognizer.

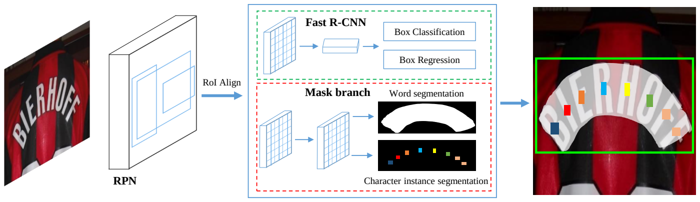

Fig. 2: Illustration of the architecture of the our method.

3.1 Framework

The overall architecture of our proposed method is presented in Fig. 2. Functionally, the framework consists of four components: a feature pyramid network (FPN) [46] as backbone, a region proposal network (RPN) [33] for generating text proposals, a Fast R-CNN [33] for bounding boxes regression, a mask branch for text instance segmentation and character segmentation. In the training phase, a lot of text proposals are first generated by RPN, and then the RoI features of the proposals are fed into the Fast R-CNN branch and the mask branch to generate the accurate text candidate boxes, the text instance segmentation maps, and the character segmentation maps.

Backbone Text in nature images are various in sizes. In order to build high-level semantic feature maps at all scales, we apply a feature pyramid structure [46] backbone with ResNet [50] of depth 50. FPN uses a top-down architecture to fuse the feature of different resolutions from a single-scale input, which improves accuracy with marginal cost.

RPN RPN is used to generate text proposals for the subsequent Fast R-CNN and mask branch. Following [46], we assign anchors on different stages depending on the anchor size. Specifically, the area of the anchors are set to $\{32^2, 64^2, 128^2, 256^2, 512^2\}$ pixels on five stages $\{P_2, P_3, P_4, P_5, P_6\}$ respectively. Different aspect ratios $\{0.5, 1, 2\}$ are also adopted in each stages as in [33]. In this way, the RPN can handle text of various sizes and aspect ratios. RoI Align [9] is adapted to extract the region features of the proposals. Compared to RoI Pooling [44], RoI Align preserves more accurate location information, which is quite beneficial to the segmentation task in the mask branch. Note that no special design for text is adopted, such as the special aspect ratios or orientations of anchors for text, as in previous works [1, 24, 23].

Fast R-CNN The Fast R-CNN branch includes a classification task and a regression task. The main function of this branch is to provide more accurate bounding boxes for detection. The inputs of Fast R-CNN are in 7×7 resolution, which are generated by RoI Align from the proposals produced by RPN.

Mask Branch There are two tasks in the mask branch, including a global text instance segmentation task and a character segmentation task. As shown in Fig. 3, giving an input RoI, whose size is fixed to $16 * 64$, through four convolutional layers and a de-convolutional layer, the mask branch predicts 38 maps (with $32 * 128$ size), including a global text instance map, 36 character maps, and a background map of characters. The global text instance map can give accurate localization of a text region, regardless of the shape of the text instance.

The character maps are maps of 36 characters, including 26 letters and 10 Arabic numerals. The background map of characters, which excludes the character regions, is also needed for post-processing.

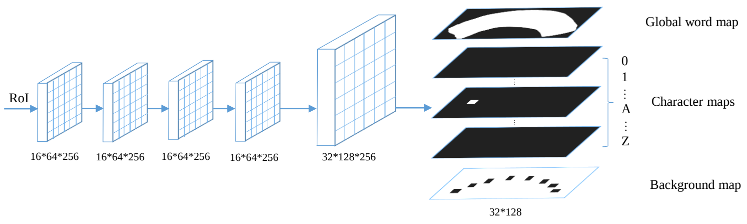

Fig. 3: Illustration of the mask branch. Subsequently, there are four convolutional layers, one de-convolutional layer, and a final convolutional layer which predicts maps of 38 channels (1 for global text instance map; 36 for character maps; 1 for background map of characters).

3.2 Label Generation

For a training sample with the input image I and the corresponding ground truth, we generate targets for RPN, Fast R-CNN and mask branch. Generally, the ground truth contains $P = \{p_1, p_2...p_m\}$ and $C = \{c_1 = (cc_1, cl_1), c_2 = (cc_2, cl_2), ..., c_n = (cc_n, cl_n)\}$, where p_i is a polygon which represents the localization of a text region, cc_j and cl_j are the category and location of a character respectively. Note that, in our method C is not necessary for all training samples.

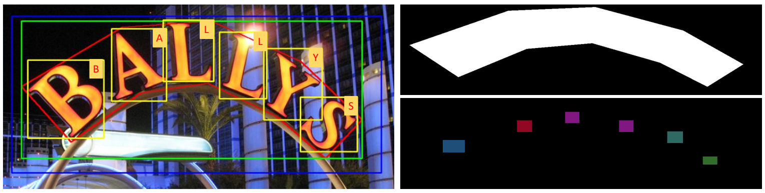

Fig. 4: Label generation of mask branch. Left: the blue box is a proposal yielded by RPN, the red polygon and yellow boxes are ground truth polygon and character boxes, the green box is the horizontal rectangle which covers the polygon with minimal area. Right: the global map (top) and the character map (bottom).

We first transform the polygons into horizontal rectangles which cover the polygons with minimal areas. And then we generate targets for RPN and Fast R-CNN following [44, 33, 46]. There are two types of target maps to be generated for the mask branch with the ground truth P, C (may not exist) as well as the proposals yielded by RPN: a global map for text instance segmentation and a character map for character semantic segmentation. Given a positive proposal r, we first use the matching mechanism of [44, 33, 46] to obtain the best matched horizontal rectangle. The corresponding polygon as well as characters (if any) can be obtained further. Next, the matched polygon and character boxes are shifted and resized to align the proposal and the target map of $H \times W$ as the following formulas:

$$B_x = (B_{x_0} - min(r_x)) \times W/(max(r_x) - min(r_x)) \tag{1}$$

$$B_y = (B_{y_0} - min(r_y)) \times H/(max(r_y) - min(r_y)) \tag{2}$$

where (B_x, B_y) and (B_{x_0}, B_{y_0}) are the updated and original vertexes of the polygon and all character boxes; (r_x, r_y) are the vertexes of the proposal r.

After that, the target global map can be generated by just drawing the normalized polygon on a zero-initialized mask and filling the polygon region with the value 1. The character map generation is visualized in Fig. 4. We first shrink all character bounding boxes by fixing their center point and shortening the sides to the fourth of the original sides. Then, the values of the pixels in the shrunk character bounding boxes are set to their corresponding category indices and those outside the shrunk character bounding boxes are set to 0. If there are no character bounding boxes annotations, all values are set to -1.

3.3 Optimization

As discussed in Sec. 3.1, our model includes multiple tasks. We naturally define a multi-task loss function:

$$L = L_{rpn} + \alpha_1 L_{rcnn} + \alpha_2 L_{mask}, \tag{3}$$

where L_{rpn} and L_{rcnn} are the loss functions of RPN and Fast R-CNN, which are identical as these in [33] and [44]. The mask loss L_{mask} consists of a global text instance segmentation loss L_{global} and a character segmentation loss L_{char}:

$$L_{mask} = L_{global} + \beta L_{char}, \tag{4}$$

where L_{global} is an average binary cross-entropy loss and L_{char} is a weighted spatial soft-max loss. In this work, the α_1, α_2, β, are empirically set to 1.0.

Text instance segmentation loss The output of the text instance segmentation task is a single map. Let N be the number of pixels in the global map, y_n be the pixel label ($y_n \in 0, 1$), and x_n be the output pixel, we define the L_{global} as follows:

$$L_{global} = -\frac{1}{N} \sum_{n=1}^{N} [y_n \times log(S(x_n)) + (1 - y_n) \times log(1 - S(x_n))] \tag{5}$$

where $S(x)$ is a sigmoid function.

Character segmentation loss The output of the character segmentation consists of 37 maps, which correspond to 37 classes (36 classes of characters and the background class). Let T be the number of classes, N be the number of pixels in each map. The output maps X can be viewed as an $N \times T$ matrix. In this way, the weighted spatial soft-max loss can be defined as follows:

$$L_{char} = -\frac{1}{N} \sum_{n=1}^{N} W_n \sum_{t=0}^{T-1} Y_{n,t} log(\frac{e^{X_{n,t}}}{\sum_{k=0}^{T-1} e^{X_{n,k}}}), \tag{6}$$

where Y is the corresponding ground truth of X. The weight W is used to balance the loss value of the positives (character classes) and the background class. Let the number of the background pixels be N_{neg}, and the background class index be 0, the weights can be calculated as:

$$W_i = \begin{cases} 1 & \text{if } Y_{i,0} = 1, \\ N_{neg}/(N - N_{neg}) & \text{otherwise} \end{cases} \tag{7}$$

Note that in inference, a sigmoid function and a soft-max function are applied to generate the global map and the character segmentation maps respectively.

3.4 Inference

Different from the training process where the input RoIs of mask branch come from RPN, in the inference phase, we use the outputs of Fast R-CNN as proposals to generate the predicted global maps and character maps, since the Fast R-CNN outputs are more accurate.

Specially, the processes of inference are as follows: first, inputting a test image, we obtain the outputs of Fast R-CNN as [33] and filter out the redundant candidate boxes by NMS; and then, the kept proposals are fed into the mask branch to generate the global maps and the character maps; finally the predicted polygons can be obtained directly by calculating the contours of text regions on global maps, the character sequences can be generated by our proposed *pixel voting* algorithm on character maps.

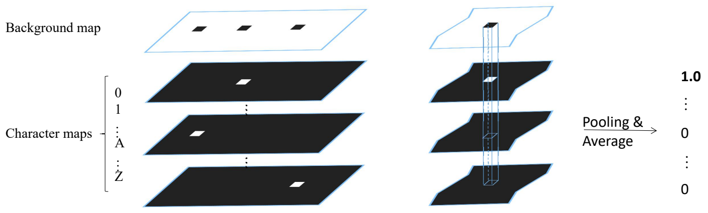

Fig. 5: Overview of the pixel voting algorithm. Left: the predicted character maps; right: for each connected regions, we calculate the scores for each character by averaging the probability values in the corresponding region.

Algorithm 1 Pixel Voting

```
Input: Background map $B$, Character maps $C$
1: Generating connected regions $R$ on the binarized background map
2: $S \leftarrow \varnothing$
3: for $r$ in $R$ do
4:     $scores \leftarrow \varnothing$
5:     for $c$ in $C$ do
6:         mean = Average(c[r])
7:         $scores \leftarrow$ scores + mean
8:     $S \leftarrow$ S + Argmax(scores)
9: return $S$
```

Pixel Voting We decode the predicted character maps into character sequences by our proposed pixel voting algorithm. We first binarize the background map, where the values are from 0 to 255, with a threshold of 192. Then we obtain all character regions according to connected regions in the binarized map. We calculate the mean values of each region for all character maps. The values can be seen as the character classes probability of the region. The character class with the largest mean value will be assigned to the region. The specific processes are shown in Algorithm 1. After that, we group all the characters from left to right according to the writing habit of English. The detailed calculation is described in the Algorithm 1 in the paper.

Weighted Edit Distance Edit distance can be used to find the best-matched word of a predicted sequence with a given lexicon. However, there may be multiple words matched with the minimal edit distance at the same time, and the algorithm can not decide which one is the best. The main reason for the above-mentioned issue is that all operations (delete, insert, replace) in the original edit distance algorithm have the same costs, which does not make sense actually.

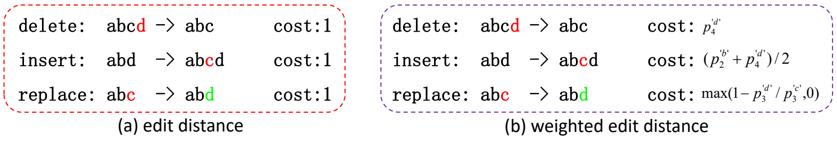
delete: abcd -> abc cost:1
insert: abd -> abcd cost:1
replace: abc -> abd cost:1
(a) edit distance
delete: abcd -> abc cost: $p_4^{'d'}$
insert: abd -> abcd cost: $(p_2^{'b'} + p_4^{'d'})/2$
replace: abc -> abd cost: $\max(1 - p_3^{'d'} / p_3^{'c'}, 0)$
(b) weighted edit distance

Fig. 6: Illustration of the edit distance and our proposed weighted edit distance. The red characters are the characters will be deleted, inserted and replaced. Green characters mean the candidate characters. p^c_{index} is the character probability, *index* is the character index and c is the current character.

Inspired by [51], we propose a weighted edit distance algorithm. As shown in Fig. 6, different from edit distance, which assign the same cost for different operations, the costs of our proposed weighted edit distance depend on the character probability p^c_{index} which yielded by the pixel voting. Mathematically, the weighted edit distance between two strings a and b, whose length are $|a|$ and $|b|$ respectively, can be described as $D_{a,b}(|a|, |b|)$, where

$$D_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0, \\ \min \begin{cases} D_{a,b}(i-1,j) + C_d \\ D_{a,b}(i,j-1) + C_i \\ D_{a,b}(i-1,j-1) + C_r \times 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases} \tag{8}$$

where $1_{(a_i \neq b_j)}$ is the indicator function equal to 0 when $a_i = b_j$ and equal to 1 otherwise; $D_{a,b}(i,j)$ is the distance between the first i characters of a and the first j characters of b; C_d, C_i, and C_r are the deletion, insert, and replace cost respectively. In contrast, these costs are set to 1 in the standard edit distance.

4 Experiments

To validate the effectiveness of the proposed method, we conduct experiments and compare with other state-of-the-art methods on three public datasets: a horizontal text set ICDAR2013 [52], an oriented text set ICDAR2015 [53] and a curved text set Total-Text [36].

4.1 Datasets

SynthText is a synthetic dataset proposed by [41], including about 800000 images. There are a large number of multi-oriented text instances, which are annotated detailedly with word-level, character-level rotated bounding boxes, as well as text sequences. All samples in this dataset are used for pre-training.
ICDAR2013 is a dataset proposed in Challenge 2 of the ICDAR 2013 Robust Reading Competition [52] which focuses on the horizontal text detection and recognition in natural images. There are 229 images in the training set and 233 images in the test set. Besides, the bounding box and the transcription are also provided for each word-level and character-level text instance.
ICDAR2015 is proposed in Challenge 4 of the ICDAR 2015 Robust Reading Competition [53]. Compared to ICDAR2013 which focuses on "focused text" in particular scenario, ICDAR2015 is more concerned with the incidental scene text detection and recognition. It contains 1000 training samples and 500 test images. All training images are annotated with word-level quadrangles as well as corresponding transcriptions. Note that, only localization annotations of words are used in our training stage.
Total-Text is a comprehensive scene text dataset proposed by [36]. Except for the horizontal text and oriented text, Total-Text also consists of a lot of curved text. Total-Text contains 1255 training images and 300 test images. All images are annotated with polygons and transcriptions in word-level. Note that, we only use the localization annotations in the training phase.

4.2 Implementation details

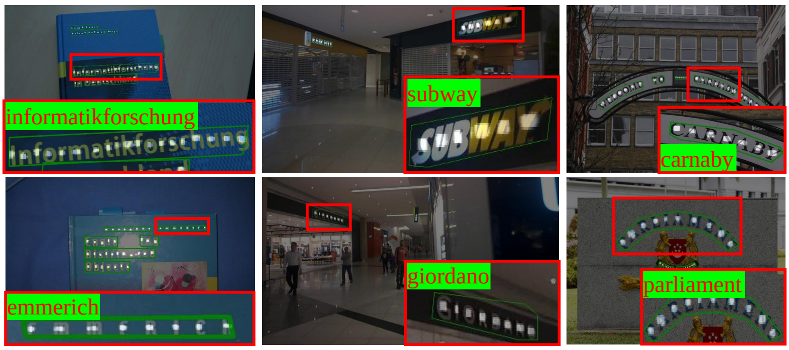

Fig. 7: Visualization results of ICDAR 2013 (the left), ICDAR 2015 (the middle) and Total-Text (the right).

Training Different from previous text spotting methods which use two independent models [38, 1] (the detector and the recognizer) or alternating training strategy [2], all subnets of our model can be trained synchronously and end-to-end. The whole training process contains two stages: pre-trained on SynthText and fine-tuned on the real-world data.

In the pre-training stage, we set the mini-batch to 8, and all the shorter edge of the input images are resized to 800 pixels while keeping the aspect ratio of the images. The batch sizes of RPN and Fast R-CNN are set to 256 and 512 per image with a 1 : 3 sample ratio of positives to negatives. The batch size of the mask branch is 16. In the fine-tuning stage, data augmentation and multi-scale training technology are applied due to the lack of real samples. Specifically, for data augmentation, we randomly rotate the input pictures in a certain angle range of $[-15°, 15°]$. Some other augmentation tricks, such as modifying the hue, brightness, contrast randomly, are also used following [34]. For multi-scale training, the shorter sides of the input images are randomly resized to three scales (600, 800, 1000). Besides, following [2], extra 1162 images for character detection from [20] are also used as training samples. The mini-batch of images is kept to 8, and in each mini-batch, the sample ratio of different datasets is set to $4:1:1:1:1$ for SynthText, ICDAR2013, ICDAR2015, Total-Text and the extra images respectively. The batch sizes of RPN and Fast R-CNN are kept as the pre-training stage, and that of the mask branch is set to 64 when fine-tuning.

We optimize our model using SGD with a weight decay of 0.0001 and momentum of 0.9. In the pre-training stage, we train our model for 170k iterations, with an initial learning rate of 0.005. Then the learning rate is decayed to a tenth at the 120k iteration. In the fine-tuning stage, the initial learning rate is set to 0.001, and then be decreased to 0.0001 at the 40k iteration. The fine-tuning process is terminated at the 80k iteration.

Inference In the inference stage, the scales of the input images depend on different datasets. After NMS, 1000 proposals are fed into Fast R-CNN. False alarms and redundant candidate boxes are filtered out by Fast R-CNN and NMS respectively. The kept candidate boxes are input to the mask branch to generate the global text instance maps and the character maps. Finally, the text instance bounding boxes and sequences are generated from the predicted maps.

We implement our method in Caffe2 and conduct all experiments on a regular workstation with Nvidia Titan Xp GPUs. The model is trained in parallel and evaluated on a single GPU.

4.3 Horizontal text

We evaluate our model on ICDAR2013 dataset to verify its effectiveness in detecting and recognizing horizontal text. We resize the shorter sides of all input images to 1000 and evaluate the results on-line.

Table 1: Results on ICDAR2013. "S", "W" and "G" mean recognition with strong, weak and generic lexicon respectively.

Method	Word Spotting			End-to-End			FPS
	S	W	G	S	W	G	
Jaderberg *et al.* [5]	90.5	-	76	86.4	-	-	-
FCRNall+multi-filt [41]	-	-	84.7	-	-	-	-
Textboxes [1]	93.9	92.0	85.9	91.6	89.7	83.9	-
Deep text spotter [3]	92	89	81	89	86	77	**9**
Li *et al.* [2]	**94.2**	**92.4**	**88.2**	91.1	89.8	84.6	1.1
Ours	92.5	92.0	**88.2**	**92.2**	**91.1**	**86.5**	4.8

The results of our model are listed and compared with other state-of-the-art methods in Table 1 and Table 3. As shown, our method achieves state-of-the-art results among detection, word spotting and end-to-end recognition. Specifically, for detection, though evaluated at a single scale, our method outperforms some previous methods which are evaluated at multi-scale setting [54, 28] (F-Measure: 91.7% *v.s.* 90.3%); for word spotting, our method is comparable to the previous best method; for end-to-end recognition, despite amazing results have been achieved by [1, 2], our method is still beyond them by 1.1% − 1.9%.

4.4 Oriented text

Table 2: Results on ICDAR2015. "S", "W" and "G" mean recognition with strong, weak and generic lexicon respectively.

Method	Word Spotting			End-to-End			FPS
	S	W	G	S	W	G	
Baseline OpenCV3.0 + Tesseract[53]	14.7	12.6	8.4	13.8	12.0	8.0	-
TextSpotter [55]	37.0	21.0	16.0	35.0	20.0	16.0	1
Stradvision [53]	45.9	-	-	43.7	-	-	-
TextProposals + DictNet [56, 39]	56.0	52.3	49.7	53.3	49.6	47.2	0.2
HUST_MCLAB [26, 11]	70.6	-	-	67.9	-	-	-
Deep text spotter [3]	58.0	53.0	51.0	54.0	51.0	47.0	**9.0**
Ours (720)	71.6	63.9	51.6	71.3	62.5	50.0	6.9
Ours (1000)	77.7	71.3	58.6	77.3	69.9	60.3	4.8
Ours (1600)	**79.3**	**74.5**	**64.2**	**79.3**	**73.0**	**62.4**	2.6

Table 3: The detection results on ICDAR2013 and ICDAR2015. For ICDAR2013, all methods are evaluated under the "DetEval evaluation protocol. The short sides of the input image in "Ours (det only)" and "Ours" are set to 1000.

Method	ICDAR2013			FPS	ICDAR2015			FPS
	Precision	Recall	F-Measure		Precision	Recall	F-Measure	
Zhang *et al.* [22]	88.0	78.0	83.0	0.5	71.0	43.0	54.0	0.5
Yao *et al.* [21]	88.9	80.2	84.3	1.6	72.3	58.7	64.8	1.6
CTPN [25]	93.0	83.0	88.0	7.1	74.0	52.0	61.0	-
Seglink [26]	87.7	83.0	85.3	**20.6**	73.1	76.8	75.0	-
EAST [27]	-	-	-	-	83.3	78.3	80.7	-
SSTD [24]	89.0	86.0	88.0	7.7	80.0	73.0	77.0	**7.7**
Wordsup [54]	93.3	87.5	90.3	2	79.3	77.0	78.2	2
He *et al.* [28]	92.0	81.0	86.0	1.1	82.0	80.0	81.0	1.1
Ours (det only)	94.1	88.1	91.0	4.6	85.8	**81.2**	83.4	4.8
Ours	**95.0**	**88.6**	**91.7**	4.6	**91.6**	81.0	**86.0**	4.8

We verify the superiority of our method in detecting and recognizing oriented text by conducting experiments on ICDAR2015. We input the images with three different scales: the original scale (720×1280) and two larger scales where shorter sides of the input images are 1000 and 1600 due to a lot of small text instance in ICDAR2015. We evaluate our method on-line and compare it with other methods in Table 2 and Table 3. Our method outperforms the previous methods by a large margin both in detection and recognition. For detection, when evaluated at the original scale, our method achieves the F-Measure of 84%, higher than the current best one [28] by 3.0%, which evaluated at multiple scales. When evaluated at a larger scale, a more impressive result can be achieved (F-Measure: 86.0%), outperforming the competitors by at least 5.0%. Besides, our method also achieves remarkable results on word spotting and end-to-end recognition. Compared with the state of the art, the performance of our method has significant improvements by 13.2% – 25.3%, for all evaluation situations.

4.5 Curved text

Detecting and recognizing arbitrary text (e.g. curved text) is a huge superiority of our method beyond other methods. We conduct experiments on Total-Text to verify the robustness of our method in detecting and recognizing curved text. Similarly, we input the test images with the short edges resized to 1000. The evaluation protocol of detection is provided by [36]. The evaluation protocol of end-to-end recognition follows ICDAR 2015 while changing the representation of polygons from four vertexes to an arbitrary number of vertexes in order to handle the polygons of arbitrary shapes.

Table 4: Results on Total-Text. "None" means recognition without any lexicon. "Full" lexicon contains all words in test set.

Method	Detection			End-to-End	
	Precision	Recall	F-Measure	None	Full
Ch'ng *et al.* [36]	40.0	33.0	36.0	-	-
Liao *et al.* [1]	62.1	45.5	52.5	36.3	48.9
Ours	**69.0**	**55.0**	**61.3**	**52.9**	**71.8**

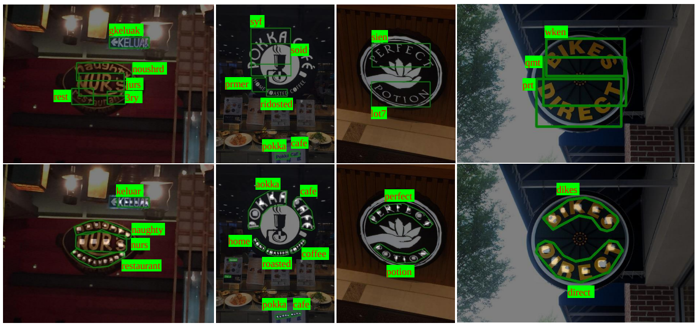

Fig. 8: Qualitative comparisons on Total-Text without lexicon. Top: results of TextBoxes [1]; Bottom: results of ours.

To compare with other methods, we also trained a model [1] using the code in [1] [1] with the same training data. As shown in Fig. 8, our method has a large superiority on both detection and recognition for curved text. The results in Table 4 show that our method exceeds [1] by 8.8 points in detection and at least 16.6% in end-to-end recognition. The significant improvements of detection mainly come from the more accurate localization outputs which encircle the text regions with polygons rather than the horizontal rectangles. Besides, our method

[1] https://github.com/MhLiao/TextBoxes

is more suitable to handle sequences in 2-D space (such as curves), while the sequence recognition network used in [1, 2, 3] are designed for 1-D sequences.

4.6 Speed

Compared to previous methods, our proposed method exhibits a good speed-accuracy trade-off. It can run at 6.9 FPS with the input scale of 720×1280. Although a bit slower than the fastest method [3], it exceeds [3] by a large margin in accuracy. Moreover, the speed of ours is about 4.4 times of [2] which is the current state-of-the-art on ICDAR2013.

4.7 Ablation Experiments

Table 5: Ablation experimental results. "Ours (a)": without character annotations from the real images; "Ours (b)": without weighted edit distance.

Method	ICDAR2013						ICDAR2015					
	Word Spotting			End-to-End			Word Spotting			End-to-End		
	S	W	G	S	W	G	S	W	G	S	W	G
Ours(a)	91.8	90.3	85.9	90.7	89.4	84.6	76.9	71.6	61.6	76.6	69.9	59.8
Ours(b)	91.4	90.5	84.3	91.3	89.9	83.8	75.9	67.5	56.8	76.1	67.1	56.7
Ours	**92.5**	**92.0**	**88.2**	**92.2**	**91.1**	**86.5**	**79.3**	**74.5**	**64.2**	**79.3**	**73.0**	**62.4**

With or without character maps We train a model named "Ours(det only)" which removes the subnet of the character maps from the original network to explore the effect of training detection and recognition jointly. As shown in Table 3 , the detection results of "Ours" exceed "Ours(det only)" by 0.7% and 2.6% on ICDAR2013 and ICDAR2015 respectively, which demonstrate that the detection task can benefit from the recognition task when jointly training.
With or without real-world character annotation The experiment without real-world character annotations is also conducted. As shown in Table 5, although "Ours(a)" is trained without any real-world character annotation, it still achieves competitive performances. More specifically, for horizontal text (ICDAR2013), it decrease "Ours", which is trained with a few real-world character annotations, by $0.7\% - 2.3\%$ on various settings; on ICDAR2015, "Ours(a)" still outperforms all other previous methods by a large margin.
With or without weighted edit distance We conduct experiments to verify the effectiveness of our proposed weighted edit distance. The method of using original edit distance is named "Ours(b)" and the results are shown in Table 5. As shown, weighted edit distance can boost the performance by at most 7.4 points of all experiments.

5 Conclusion

In this paper, we propose a text spotter, which detects and recognizes scene text in a unified network and can be trained end-to-end completely. Comparing with previous methods, our proposed network is very easy to train and has the ability to detect and recognize irregular text (e.g. curved text). The impressive performances on all the datasets which includes horizontal text, oriented text and curved text, demonstrate the effectiveness and robustness of our method for text detection and end-to-end text recognition.

References

1. Liao, M., Shi, B., Bai, X., Wang, X., Liu, W.: Textboxes: A fast text detector with a single deep neural network. In: Proc. AAAI. (2017) 4161–4167
2. Li, H., Wang, P., Shen, C.: Towards end-to-end text spotting with convolutional recurrent neural networks. In: Proc. ICCV. (2017) 5248–5256
3. Busta, M., Neumann, L., Matas, J.: Deep textspotter: An end-to-end trainable scene text localization and recognition framework. In: Proc. ICCV. (2017) 2223–2231
4. Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition. In: Proc. ICCV. (2011) 1457–1464
5. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. International Journal of Computer Vision **116**(1) (2016) 1–20
6. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: Proc. ICML. (2009) 41–48
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation **9**(8) (1997) 1735–1780
8. Graves, A., Fernández, S., Gomez, F.J., Schmidhuber, J.: Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proc. ICML. (2006) 369–376
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: Proc. ICCV. (2017) 2980–2988
10. Shi, B., Wang, X., Lyu, P., Yao, C., Bai, X.: Robust scene text recognition with automatic rectification. In: Proc. CVPR. (2016) 4168–4176
11. Shi, B., Bai, X., Yao, C.: An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11) (2017) 2298–2304
12. Lee, C., Osindero, S.: Recursive recurrent nets with attention modeling for OCR in the wild. In: Proc. CVPR. (2016) 2231–2239
13. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: Proc. CVPR. (2010) 2963–2970
14. Neumann, L., Matas, J.: A method for text localization and recognition in real-world images. In: Proc. ACCV. (2010) 770–783
15. Neumann, L., Matas, J.: Real-time scene text localization and recognition. In: Proc. CVPR. (2012) 3538–3545
16. Huang, W., Qiao, Y., Tang, X.: Robust scene text detection with convolution neural network induced MSER trees. In: Proc. ECCV. (2014) 497–511
17. Kang, L., Li, Y., Doermann, D.S.: Orientation robust text line detection in natural images. In: Proc. CVPR. (2014) 4034–4041
18. Tian, S., Pan, Y., Huang, C., Lu, S., Yu, K., Tan, C.L.: Text flow: A unified text detection system in natural scene images. In: Proc. ICCV. (2015) 4651–4659
19. Zhang, Z., Shen, W., Yao, C., Bai, X.: Symmetry-based text line detection in natural scenes. In: Proc. CVPR. (2015) 2558–2567
20. Zhong, Z., Jin, L., Zhang, S., Feng, Z.: Deeptext: A unified framework for text proposal generation and text detection in natural images. CoRR **abs/1605.07314** (2016)
21. Yao, C., Bai, X., Sang, N., Zhou, X., Zhou, S., Cao, Z.: Scene text detection via holistic, multi-channel prediction. CoRR **abs/1606.09002** (2016)

22. Zhang, Z., Zhang, C., Shen, W., Yao, C., Liu, W., Bai, X.: Multi-oriented text detection with fully convolutional networks. In: Proc. CVPR. (2016) 4159–4167
23. Liu, Y., Jin, L.: Deep matching prior network: Toward tighter multi-oriented text detection. In: Proc. CVPR. (2017) 3454–3461
24. He, P., Huang, W., He, T., Zhu, Q., Qiao, Y., Li, X.: Single shot text detector with regional attention. In: Proc. ICCV. (2017) 3066–3074
25. Tian, Z., Huang, W., He, T., He, P., Qiao, Y.: Detecting text in natural image with connectionist text proposal network. In: Proc. ECCV. (2016) 56–72
26. Shi, B., Bai, X., Belongie, S.J.: Detecting oriented text in natural images by linking segments. In: Proc. CVPR. (2017) 3482–3490
27. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J.: EAST: an efficient and accurate scene text detector. In: Proc. CVPR. (2017) 2642–2651
28. He, W., Zhang, X., Yin, F., Liu, C.: Deep direct regression for multi-oriented scene text detection. In: Proc. ICCV. (2017) 745–753
29. Lyu, P., Yao, C., Wu, W., Yan, S., Bai, X.: Multi-oriented scene text detection via corner localization and region segmentation. In: Proc. CVPR. (2018) 7553–7563
30. Liao, M., Zhu, Z., Shi, B., Xia, G.s., Bai, X.: Rotation-sensitive regression for oriented scene text detection. In: Proc. CVPR. (2018) 5909–5918
31. Huang, W., Lin, Z., Yang, J., Wang, J.: Text localization in natural images using stroke feature transform and text covariance descriptors. In: Proc. ICCV. (2013) 1241–1248
32. Zitnick, C.L., Dollár, P.: Edge boxes: Locating object proposals from edges. In: Proc. ECCV. (2014) 391–405
33. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6) (2017) 1137–1149
34. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C.: SSD: single shot multibox detector. In: Proc. ECCV. (2016) 21–37
35. Risnumawan, A., Shivakumara, P., Chan, C.S., Tan, C.L.: A robust arbitrary text detection system for natural scene images. Expert Syst. Appl. **41**(18) (2014) 8027–8048
36. Chng, C.K., Chan, C.S.: Total-text: A comprehensive dataset for scene text detection and recognition. In: Proc. ICDAR. (2017) 935–942
37. Bissacco, A., Cummins, M., Netzer, Y., Neven, H.: Photoocr: Reading text in uncontrolled conditions. In: Proc. ICCV. (2013) 785–792
38. Jaderberg, M., Vedaldi, A., Zisserman, A.: Deep features for text spotting. In: Proc. ECCV. (2014) 512–528
39. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Synthetic data and artificial neural networks for natural scene text recognition. CoRR **abs/1406.2227** (2014)
40. Shelhamer, E., Long, J., Darrell, T.: Fully convolutional networks for semantic segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **39**(4) (2017) 640–651
41. Gupta, A., Vedaldi, A., Zisserman, A.: Synthetic data for text localisation in natural images. In: Proc. CVPR. (2016) 2315–2324
42. Liao, M., Shi, B., Bai, X.: Textboxes++: A single-shot oriented scene text detector. IEEE Transactions on Image Processing **27**(8) (2018) 3676–3690
43. Girshick, R.B., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proc. CVPR. (2014) 580–587
44. Girshick, R.B.: Fast R-CNN. In: Proc. ICCV. (2015) 1440–1448

45. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: Proc. NIPS. (2016) 379–387
46. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: Proc. CVPR. (2017) 936–944
47. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proc. CVPR. (2016) 779–788
48. Dai, J., He, K., Li, Y., Ren, S., Sun, J.: Instance-sensitive fully convolutional networks. In: Proc. ECCV. (2016) 534–549
49. Li, Y., Qi, H., Dai, J., Ji, X., Wei, Y.: Fully convolutional instance-aware semantic segmentation. In: Proc. CVPR. (2017) 4438–4446
50. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. CVPR. (2016) 770–778
51. Yao, C., Bai, X., Liu, W.: A unified framework for multioriented text detection and recognition. IEEE Trans. Image Processing **23**(11) (2014) 4737–4749
52. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., i Bigorda, L.G., Mestre, S.R., Mas, J., Mota, D.F., Almazán, J., de las Heras, L.: ICDAR 2013 robust reading competition. In: Proc. ICDAR. (2013) 1484–1493
53. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S.K., Bagdanov, A.D., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., Shafait, F., Uchida, S., Valveny, E.: ICDAR 2015 competition on robust reading. In: Proc. ICDAR. (2015) 1156–1160
54. Hu, H., Zhang, C., Luo, Y., Wang, Y., Han, J., Ding, E.: Wordsup: Exploiting word annotations for character based text detection. In: Proc. ICCV. (2017) 4950–4959
55. Neumann, L., Matas, J.: Real-time lexicon-free scene text localization and recognition. IEEE Trans. Pattern Anal. Mach. Intell. **38**(9) (2016) 1872–1885
56. Gómez, L., Karatzas, D.: Textproposals: a text-specific selective search algorithm for word spotting in the wild. Pattern Recognition **70** (2017) 60–74