Boundary TextSpotter: Toward Arbitrary-Shaped Scene Text Spotting

Pu Lu, Hao Wang, Shenggao Zhu, Jing Wang, Xiang Bai, *Senior Member, IEEE*, and Wenyu Liu, *Senior Member, IEEE*

***Abstract*— Reading arbitrary-shaped text in an end-to-end fashion has received particularly growing interested in computer vision. In this paper, we study the problem of scene text spotting, which aims to detect and recognize text from cluttered images simultaneously and propose an end-to-end trainable neural network named Boundary TextSpotter. Different from existing methods that describe the shape of text instance with bounding box or shape mask, Boundary TextSpotter formulates it as a set of boundary points. Besides, the representation of such boundary points provides the order of reading text. Benefiting from the representation on both detection and recognition, Boundary TextSpotter can easily deal with the text of arbitrary shapes. Further, to efficiently detect the boundary points of the text, a single-stage text detector is proposed, which can almost perform at a real-time speed. Experiments on three challenging datasets, including ICDAR2015, Total-Text and CTW1500 demonstrate that the proposed method achieves state-of-the-art or competitive results, meanwhile significantly improving the inference speed.**

***Index Terms*— Scene text spotting, scene text detection, scene text recognition.**

I. Introduction

AUTOMATIC reading text from natural images is of great value for many real-world applications such as image retrieval [1], visual search [2], network content security, office automation and intelligent transportation system [3], [4], as scene text is very common and contains quite meaningful semantics to understand the real-world.

Scene text spotting, which aims at concurrently localizing and recognizing texts from scene images, has been extensively studied by previous methods [5], [6], [7], [8], [9]. Those methods [7], [10], [11], [12] treat scene text detection and recognition as two separated sub-tasks, in which text proposals are first hunted by a trained text detector, then fed to a learned text recognition model. But in fact, the two sub-tasks are highly relevant and complementary to each other, which is confirmed by the recent end-to-end text spotting methods [5], [6], [7], [8], [9], [11], [13]. Those methods combine the detection and recognition stages in an end-to-end trainable neural network, and follow a similar pipeline. First, the detection stage localizes the bounding box of each text instance. Then, CNN features inside the detected bounding boxes are extracted and fed to the text recognition stage. Benefiting from feature sharing and joint optimization, the performances of detection and end-to-end recognition can be enhanced at the same time.

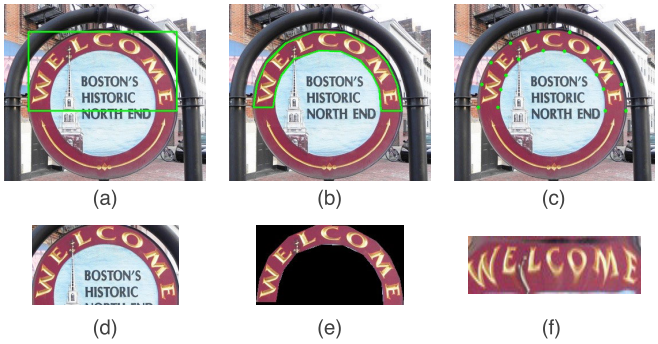


Fig. 1. Illustrations of three kinds of methods for text region representation. (a) A rectangular box is used to represent the text region and cropped as (d); (b) A shape mask is employed to formulate text region, and background noisy in (e) can be suppressed; (c) The text region is represented as a set of boundary points, with which the text can be transformed into a horizontal region like (d).

Despite promising progress, most previous spotting methods [5], [6], [7], [8], [9], [11] suffer from dealing with text of irregular shapes, such as curved text. For a general end-to-end OCR system, it is inevitable to handle the text with arbitrary shapes, as irregular texts are very common in our real world. In [9], [8], [6], [5], [14], and [15], the shape of each detected text instance is represented with a rectangle, which only can tightly cover straight text instances. Rectangular boxes are highly limited to describing irregular text since it more or less contains background information which brings difficulties to the text recognition stage, as shown in Fig. 1(d). Recent text spotting methods [16], [17] use shape masks to describe irregular texts. With the shape mask of each text, background information can be suppressed, as illustrated in Fig. 1(e), but the complex layout of irregular text is still challenging for the text recognition stage.

Manuscript received 19 October 2021; revised 22 June 2022; accepted 22 August 2022. Date of publication 20 September 2022; date of current version 29 September 2022. This work was supported by the National Natural Science Foundation of China under Grant 61733007. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiangqian Wu. *(Pu Lu and Hao Wang contributed equally to this work.) (Corresponding author: Wenyu Liu.)*

Pu Lu, Shenggao Zhu, and Jing Wang are with Huawei Cloud, Shenzhen 518129, China (e-mail: lupu1@huawei.com; zhushenggao@huawei.com; wangjing105@huawei.com).

Hao Wang, Xiang Bai, and Wenyu Liu are with the School of Electronic Information and Communications, Huazhong University of Science and Technology (HUST), Wuhan 430074, China (e-mail: wanghao4659@hust.edu.cn; xbai@hust.edu.cn; liuwy@hust.edu.cn).

Digital Object Identifier 10.1109/TIP.2022.3206615

In this paper, we propose an end-to-end trainable network for text spotting, named Boundary TextSpotter, which can read text of arbitrary shapes. Instead of detecting a rectangle bounding box or segmenting a shape mask, our detection stage aims at localizing the boundary points of a text instance. Specifically, the goal of our detection is to predict a set of boundary points, which are more flexible for describing various shapes of text, often embodied in two-dimensional space, as shown in Fig. 1(c). For an end-to-end OCR system, the usage of boundary points has three advantages: 1) The irregular text region can be covered tightly with boundary points, contributing to effectively eliminating the disturb of background noise for the subsequent recognition; 2) Following the direction of boundary points, irregular text can be easily read as a regular one, as described in Fig. 1(f), which is realistic input for a sequence recognition model. 3) The position of boundary points can be easily refined through backpropagation when training a recognition model, fully enjoying the improvement of detection performance from the recognition stage. Therefore, boundary points appear to be a reasonable representation that can smoothly and effectively bridge text detection and recognition modules.

In this paper, the major extension over its conference version [18], lies in the detection part of the network. In [18], boundary point detection is performed in a two-stage way, in which the minimum oriented rectangular box of each text instance is first detected, then boundary points are predicted within the detected minimum oriented rectangular box. Such a complex detection process results in the loss of inference speed. In this improved Boundary TextSpotter, a single stage text detector is designed to achieve the direct regression of boundary points in each location. However, limited to receptive field of convolutional neural network, it is challenging to directly detect boundary points of text due to the diversity of text shape and scale. To effectively localize boundary points of text instances, a lightweight boundary refinement module is proposed to iteratively refine the positions of detected boundary points, which can dynamically adjust the receptive field. The process of refining boundary points has two advantages: 1) As the refinement time increases, the text region that boundary refinement module can observe is more complete and larger, contributing to predicting precise boundary points. 2) Benefiting from the simplicity of this boundary refinement module, the detection stage is efficient, resulting in improving the inference speed of the whole spotting network.

The main contributions in this work are three-fold: 1) We recommend the representation of boundary points for end-to-end text spotting, which is more suitable than a rectangular box or shape mask for connecting detection and recognition modules; 2) A single-stage boundary predictor accompanied with the boundary refinement module is proposed to detect boundary points. Its simplicity allows the whole network to perform inference at an almost real-time speed; 3) We design a novel end-to-end trainable network for joint optimizing boundary point detection and text recognition, which can read both straight and curve text.

II. Related Work

A. Scene Text Spotting

Most of the previous text spotting methods [7], [10], [11], [12] use two separate networks for detection and recognition. Wang *et al*. [19] attempt to detect and classify characters with CNNs. Jaderberg *et al*. [7] combine a word detector and a word classifier for spotting. Liao *et al*. propose TextBoxes [11] and TextBoxes++ [10] to combine their proposed text detectors with CRNN [20]. This separate pipeline results in unsatisfactory performance on both tasks due to ignoring the complementarity of text detection and recognition.

To share visual cues between text detection and recognition tasks, the recent text spotting methods [5], [6], [8], [9], [21] integrate the two sub-tasks into a unified end-to-end trainable neural network. Those methods follow a similar pipeline: text regions are first detected, then features within them are extracted for subsequent recognizer. In [8], text region is formulated as a horizontal rectangular box, which can only support horizontal text instance. Methods in [9], [6], [5], and [21] extract text features in the minimum oriented rectangular box, which can tackle oriented text. But those works suffer from handling curved text.

Recently, some text spotters are proposed to read arbitrarily shaped texts. Segmentation-based methods [16], [17], [22] detect arbitrary shaped texts by segmenting their shape masks. In [16] and [17], shape masks of text instances are further exploited to surpass background noise for better recognition by multiplying each shape mask and the RoI feature. But the problem of complex layout of the arbitrary shaped text remains to be solved for the subsequent recognizer. At the same time, methods [18], [23], [24], [25] share a common idea: text regions are formulated as a serial of boundary points, with which irregular texts could be transformed as regular ones to reduce the burden of recognizer. In [23] and [25], complex post-processing is required to generate boundary points, resulting in inefficient pipelines. Boundary TextSpotter [18] directly predicts boundary points by a two-stage detector progressively. Instead, the proposed method in this paper designs a single-stage detector to improve the inference speed.

B. Scene Text Detection

In the past decade, deep learning based scene text detection methods mainly focus on multi-oriented text. Zhang *et al*. [26] detect multi-oriented scene text by semantic segmentation. Methods [27], [28] have a common idea that text components are first detected and then linked into text instances. On the basis of SSD [29], Liao *et al*. propose TextBoxes++ [10] in which the network parameters are designed for multi-oriented text. Similar to Densebox [30], Zhou *et al*. [31] and He *et al*. [15] regress text boxes directly from dense segmentation maps. Lyu *et al*. [32] propose to detect and group the corners of each text instance into text box. Though these methods make great progress for multi-oriented texts, they suffer from handling arbitrarily shaped texts.

Recently, many novel representations are proposed to formulate the shape of the irregular text. Some methods

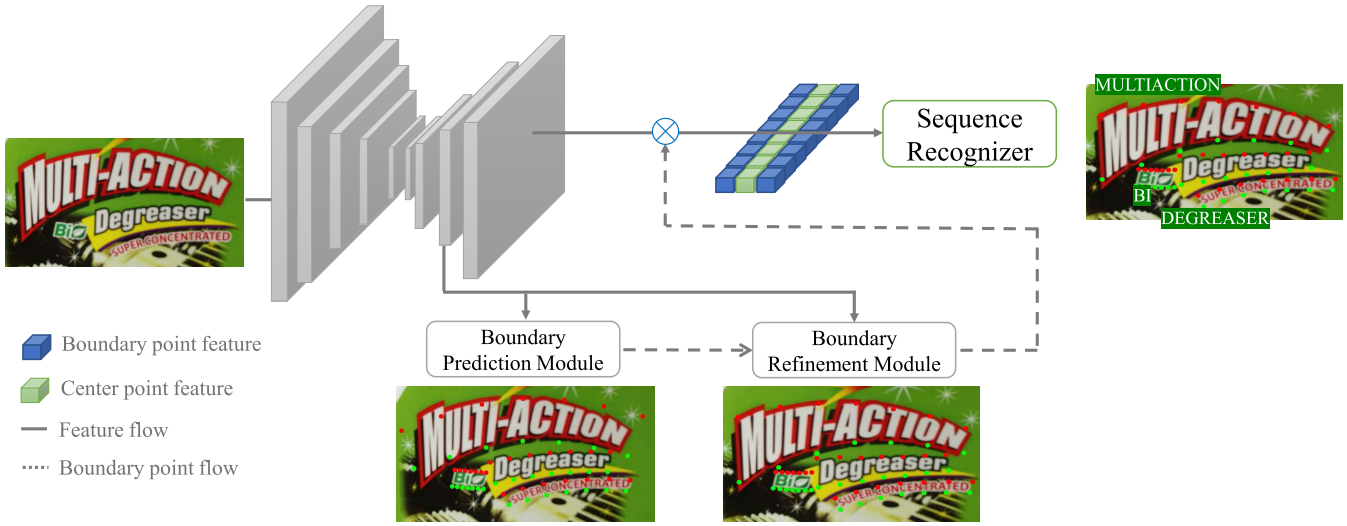


Fig. 2. The overview of the proposed framework. The solid lines mean the feature flow, the dashed lines mean the predicted boundary points flow. The predictions of the different modules are also visualized in the origin image.

[25], [33], [34], [35] detect local text components first and group them into text boundary. Specifically, text components are defined as a series of ordered disks, character regions and local text boxes in TextSnake [33], CRAFT [25] and Zhang *et al*. [34], respectively. Segmentation-based methods [36], [37], [38], [39] detect arbitrary shaped text by instance segmentation, and propose different schemes to separate adjacent text instances. In [38] and [39], pixels belonging to identical text instance are embedded into the same space. PSENet [36] segments different scale kernels and merges them progressively. DB [37] separate different text instances with learnable adaptive thresholds. Besides, TextField [40] predicts the direction field pointing away from the nearest text boundary to each text point within the text region. Wang *et al*. [41] detect boundary points by a recurrent neural network adaptively.

Different from most of the above-mentioned methods, we formulate text region as a serial of boundary points and design a single-stage detector to localize them.

C. Scene Text Recognition

Traditional methods [42], [43] recognize the character based on text segment or character candidate, and group the character to text instance. The recognition is easily influenced by character location precision and is hard to model the semantic information of text. Methods [20], [44] directly extract features from the image of text line through CNN and RNN, and use CTC-based decoder to align the prediction with word-level annotation. These methods transform the character classification to image-based sequence recognition.

The above methods design for straight text recognition and suffer from dealing with arbitrary-shaped text that is very common in nature scenes. Shi *et al*. [45] propose an end-to-end model composed of a rectification network that can rectify the irregular text to a straight one and an attention-based recognition network that predicts the character sequence. Some other methods process irregular text from different perspectives. [46] extracts visual features of character in four directions and combine these features for recognition. [47] presents a 2D tailored attention module which can select feature for decoding characters. Liao *et al*. [48] handle the irregular text in a two-dimensional fashion through semantic segmentation with the help of character-level localization.

The recognition module of our method is most similar with [45]. Benefiting from the advantage that the boundary points can provide the reading order of irregular text, the recognition module can handle arbitrarily shaped texts without the help of the rectification network, which further promotes the inference speed.

III. Methodology

A. Overview

The overall architecture is presented in Fig. 2. The framework consists of four components: feature extraction module, boundary prediction module, boundary refinement module, and text recognition module.

The feature extraction module aims at extracting features for the text detection and recognition tasks from input images. Then, the boundary prediction module densely detects the boundary points, but suffers from regressing their precise locations due to various directions and shapes of texts. To alleviate this effect, a lightweight boundary refinement module is proposed to obtain more precise boundary points with very little cost of computation. After obtaining the boundary points of the text, the features of text instances are sampled directly and are fed into the subsequent text recognition module. The recognition is only performed based on final boundary points.

B. Feature Extraction Module

The goal of feature extraction module is to provide features to text detection and recognition tasks. To better handle texts of various sizes, we apply a feature pyramid network

Fig. 3. The schematic of boundary points. The boundary points on the up line or bottom line of text instance are in red or green respectively. The yellow polygons are the original annotations.

(FPN) [49] equipped with ResNet-50 [50] as the feature extraction module. After this module, the features are not only rich in details but also contain high-level semantic information.

Different from FPN, we only conduct tasks of text detection and recognition on a single feature map to improve the inference speed. As illustrated in [51], the two tasks have different requirements on feature maps. Specifically, the recognition task needs more detailed information for character sequence prediction while the detection task mainly focuses on the whole text instance. So we conduct text detection on the feature map of 1/8 resolution of the input image, then features of the detected text instances are sampled on the feature map with 1/4 size for recognition.

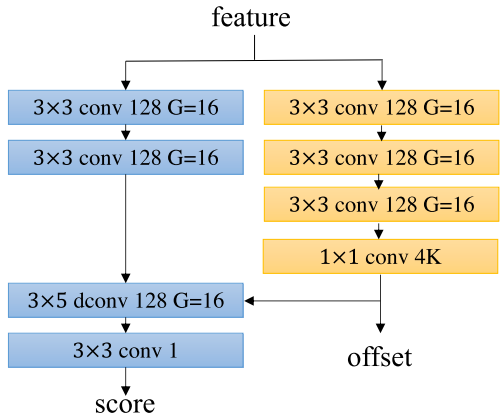


Fig. 4. Architecture of the boundary predictor module. dconv means the deformable convolution.

Fig. 5. The schematic of boundary points regression. The boundary points with red color are directly regressed from the reference point with blue color.

C. Boundary Prediction Module

Following the conference version [18], we describe the shape of text instance by formulating its two long sides with a set of points. As shown in Fig. 3, each long side of arbitrary-shaped text (*e.g.*, multi-oriented text or curved text) is represented as K points that are equidistantly distributed on it. Namely, the shape of text instance is finally formulated as $2K$ boundary points.

Inspired by [52], [53], and [54], the boundary prediction module predicts the boundary points directly in an anchor-free manner. As shown in Fig. 4, this module consists of two heads, *i.e.*, boundary regression head and classification head. The boundary regression head densely predicts $2K$ boundary points on the feature map. Then, the classification head evaluates the quality of each detected text region that is formulated by the boundary points. And only those text regions of which overlap with ground truths is over the threshold are selected for the subsequent modules.

1) Boundary Regression Head: As illustrated in Fig. 5, given the feature map, $2K$ coordinate offsets $(\Delta x', \Delta y')$ to the boundary points are predicted at each location (x, y). Then, the predicted boundary points T can be calculated as

$$T = \{(x_{ref} + \Delta x'_i, y_{ref} + \Delta y'_i)\}_i^{2K}, \tag{1}$$

where the coordinates of reference points (x_{ref}, y_{ref}) are the location that maps (x, y) to the input image. They can be calculated as $([S/2] + x * S, [S/2] + y * S)$, and S is the stride size of this feature map. The architecture of boundary regression head is given in Fig. 4, which consists of three stacked 3×3 convolution layers and one 1×1 convolution layers that outputs $4K$-dimensional offsets.

The objective function in this head is to close the distance between the predicted boundary points T and the target boundary points $\hat{T} = \{(\hat{x}_i, \hat{y}_i)\}_i^{2K}$. However, during the training stage, the common distance loss (*e.g.*, L1 or L2) is sensitive to the scale of object. The negative impact is further enlarged on scene text due to its large variations in scale and aspect ratio. To solve the scale sensitivity, we normalize the distance between T and $\hat{T}$ via dividing it with a scale-sensitive factor. The normalized distance is defined as

$$\begin{aligned} \Delta x''_i &= \frac{x_{ref} + \Delta x'_i - \hat{x}_i}{(\log_2(|\hat{x}_i - \hat{x}_{center}| + 2)) * S} \\ \Delta y''_i &= \frac{y_{ref} + \Delta y'_i - \hat{y}_i}{(\log_2(|\hat{y}_i - \hat{y}_{center}| + 2)) * S} \end{aligned} \tag{2}$$

where the coordinate $(\hat{x}_{center}, \hat{y}_{center})$, formulated as Eq. 3, means the center points of text instance, S means the stride of the input feature map. After normalized by the distance between boundary points with the center point, $(\Delta x''_i, \Delta y''_i)$ is not sensitive to the scale of text.

$$(\hat{x}_{center}, \hat{y}_{center}) = (\frac{1}{2K}\sum_{i=1}^{2K}\hat{x}_i, \frac{1}{2K}\sum_{i=1}^{2K}\hat{y}_i) \tag{3}$$

With the normalized distance $(\Delta x''_i, \Delta y''_i)$, the boundary prediction loss function (L_{bp}) can be formulated as

$$L_{bp} = \frac{1}{2K}\sum_{i=1}^{2K}(Smooth_{L1}(\Delta x''_i) + Smooth_{L1}(\Delta y''_i)). \tag{4}$$

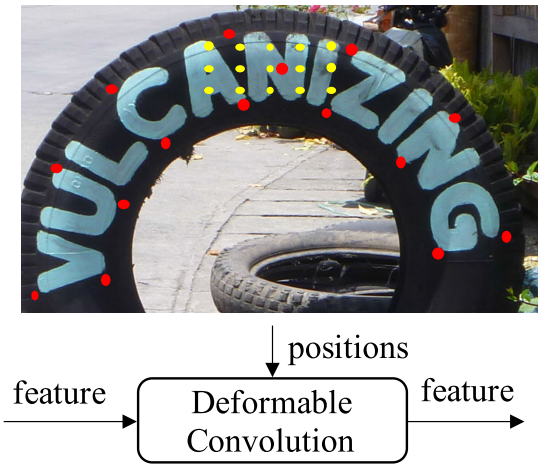


Fig. 6. The red points denote the adaptive sampling position of deformable convolution. The yellow points indicate the sample location of conventional convolution.

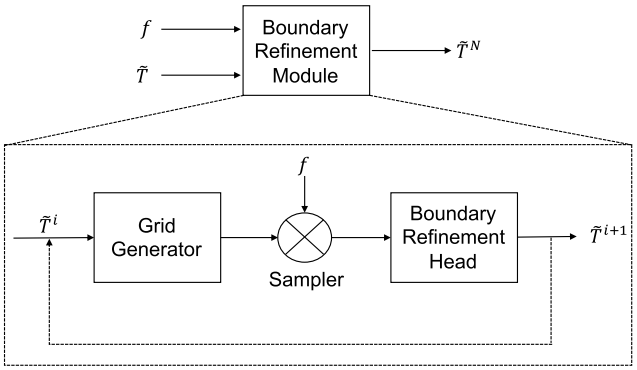


Fig. 7. The process of boundary refinement module. The precise predictions are obtained through this module.

2) Classification Head: As discussed above, the region of text instance is represented as $2K$ boundary points. At each location of the input feature map, the boundary regression head densely predicts the corresponding target text region. Among those detected text regions, many of them have low overlap with target text region, (*i.e.*, false positives). To further remove the false positives, the classification head classifies each detected text region as positive or negative region. For each predicted text region T, we attach it with a classification label defined as

$$label(T) = \begin{cases} 1, & \text{if} \quad IOU(T, \hat{T}) > \tau_h \\ 0, & \text{if} \quad IOU(T, \hat{T}) < \tau_l \\ \text{ignored}, & \text{otherwise}, \end{cases} \tag{5}$$

where IOU means the function that calculates the intersection over union between T and $\hat{T}$. The classification labels 1 and 0 denote positive and negative text regions. The label $ignored$ means that the text region can be ignored during the training stage in the classification head. In this work, the τ_h, τ_l are empirically set to 0.5 and 0.3.

Unlike general objects (*e.g.*, cars and persons) that are represented with horizontal bounding boxes, text regions are formulated as $2K$ boundary points. As illustrated in Fig. 6, to extract purified visual features for better classification, the features at those positions of $2K+1$ points, (*i.e.*, $2K$ boundary points T and its center point), are extracted to represent the text region. The center point $(\hat{x}'_{center}, \hat{y}'_{center})$ of T is defined as

$$(\hat{x}'_{center}, \hat{y}'_{center}) = (\frac{1}{2K}\sum_{i=1}^{2K}(x_{ref} + \Delta x'_i), \frac{1}{2K}\sum_{i=1}^{2K}(y_{ref} + \Delta y'_i)). \tag{6}$$

Specifically, as shown in Fig. 4, the classification head takes the feature map and the predicted $4K$-dimensional offsets from the boundary regression head as inputs. Inspired by [54], the features used for classification are extracted by a deformable convolution layer that adaptively adjusts its convolutional kernel to the positions of $2K + 1$ points. Then, the extracted features are fed to a 3×3 convolution layer to output the binary score that indicates the text region is positive or negative.

Considering the imbalance between positive and negative samples, we use focal loss [55] for optimization, which is formulated as

$$L_{cls} = -\hat{y}p^{\gamma}\log(p) - \alpha(1-\hat{y})(1-p)^{\gamma}\log(1-p), \tag{7}$$

where p is the estimated probability of the text region, and $\hat{y}$ is the corresponding classification label. The factor α is applied to balance the negative and positive sample, and γ is a tunable parameter that controls the learning weight on hard examples.

D. Boundary Refinement Module

The boundary prediction module generates many positive text regions $\widetilde{T}$ of which the overlap with their corresponding target text regions is larger than τ_h. However, suffering from various directions and shapes of texts, it is difficult to directly regress their precise locations. Consequently, most positive text regions can only cover part of text instances, resulting in inevitably wrong recognition results due to incomplete text regions. Inspired by [56] and [57], we propose the boundary refinement module that refines each positive text region $\widetilde{T}$ in a cascaded manner.

As illustrated in Fig. 7, the boundary refinement module inputs feature map f and each positive text region $\widetilde{T}$ and outputs $\widetilde{T}^N$ that is iteratively refined by N times. For the $i+1$-th refinement, we first extract the features on the locations of $2K$ boundary points of the former refinement result $\widetilde{T}^i$ ($\widetilde{T}^0 = \widetilde{T}$) by a grid generator generating sampling grid and a sampler obtaining feature from f. We refer the readers to [58] for more details on grid generator and sampler. Then the features are fed to the boundary refinement head that consists of single full-connect layer to estimate the $4K$-dimensional offsets relevant to $\widetilde{T}^i$. The $i+1$-th refined result $\widetilde{T}^{i+1}$ that is calculated by adding the estimated offset to the $\widetilde{T}^i$ is inputted for the next refinement. After refined for N times, each inputted positive text region $\widetilde{T}$ is adjusted as more precise text region $\widetilde{T}^N$.

We optimize this module by minimizing the average L1 distance between each refined text region $\widetilde{T}^i$ and its corresponding target $\hat{T}$. Specifically, the loss function L_{rf} of this

TABLE I
THE ARCHITECTURE OF THE RECOGNITION BRANCH

	Type	Configurations	Out Channels
Encoder	conv_bn_relu	[3,1,(1,0)]	128
	conv_bn_relu	[3,1,(1,1)]	128
	conv_bn_relu	[3,1,(0,1)]	128
	BLSTM		128
Decoder	Att. BLSTM		128
	FC		$\|S\|$

module is formulated as

$$L_{rf} = \frac{1}{N}\sum_{i=1}^{N} Smooth_{L1}(\hat{T} - \widetilde{T}^{i}), \tag{8}$$

where the difference $\hat{T} - \widetilde{T}^{i}$ is the mean difference of their $2K$ boundary points.

E. Text Recognition

With precise boundary points, features for recognition can be extracted easily. [59] extracts features by RoISlide and use character classifier. [18], [22], [24] extract whole region of text instance and use sequence-to-sequence recognizer. Similar to the boundary refinement module, features for recognition are sampled along boundary and center line of text. We concatenate the upper boundary feature $F_{up} \in R^{n\times C}$, center line feature $F_{cl} \in R^{n\times C}$ and bottom boundary feature $F_{bt} \in R^{n\times C}$, obtaining the visual feature $F_v = concat(F_{up}, F_{cl}, F_{bt}) \in R^{3\times n\times C}$. The features are fed into the recognition branch of which architecture is given in Tab. I.

Firstly, we apply an encoder consisting of convolutional layers and Bidirectional LSTM [60], [61] layers to extract feature sequence with high level semantic, denoted by $H = [h_1 \ldots, h_n]$, from the input feature F_v. Then an attentional sequence-to-sequence model translates the feature sequence into a character sequence. The sequence-to-sequence model works iteratively for M steps, producing a character sequence of length M, $(y_1, \ldots, y_M)$. At step m, the prediction is based on H, the internal state s_{m-1} of decoder BLSTM, and the last step predicted symbol y_{m-1}. At this step, a vector of attentional weights, α_m, is computed to indicate the importance of encoder output. which can be formulated as

$$\begin{aligned} e_m &= W_m \times \tanh\left(W_s \times S_{m-1} + W_f \times F + b\right) \\ \alpha_{m,i} &= \exp\left(e_{m,i}\right) / \sum_{i=0}^{n} \exp\left(e_{m,i'}\right) \end{aligned} \tag{9}$$

where w,W,V are trainable weights. Then, the weighted feature g_m is calculated with attentional vector.

$$g_m = \sum_{i=1}^{n} \alpha_{m,i} h_i \tag{10}$$

Then BLSTM calculates the output vector x_m and new state vector s_m, applying a linear unit and softmax to the x_m, we can get the distribution of the current-step symbol. These process can be formulated as

$$\begin{aligned} (x_m, s_m) &= RNN(s_{m-1}, (g_t, onehot(y_{m-1}))) \\ p(y_m) &= softmax(W_o x_m + b_o) \end{aligned} \tag{11}$$

The recognition loss can be formulated as

$$L_{recog} = -\frac{1}{M}\sum_{m=1}^{M} \log p(y_m) \tag{12}$$

During inference, we take the symbol with the highest $softmax$ score adopting a greedy decoding scheme. When the lexicon is available, we use the weighted edit distance proposed by [22] to find the best-matched word of a predicted sequence.

F. Optimization

To train the model in an end-to-end manner, the objective function is combine with losses of modules mentioned above, which is formulated as

$$L = L_{bp} + L_{cls} + L_{rf} + \lambda_{recog} L_{recog} \tag{13}$$

where λ_{recog} is the hyper-parameter to balance the text recognition loss. In this work, the λ_{recog} is empirically set to 2.0.

IV. EXPERIMENTS

First, we introduce the datasets used in our experiments. Then, the implementation details are given. Third, we evaluate the spotting/detection tasks of our method and make comparisons with the state-of-the-art scene text spotting/detection approaches. Last, ablation studies are conducted to discuss the effectiveness of each proposed module.

A. Datasets

We evaluate the performance of our method on three popular benchmarks, including a multi-oriented text dataset ICDAR2015, two curved text datasets Total-Text and CTW1500. The details about datasets used in our experiment are as follows.

SynthText [62] is a synthetic dataset with around 800000 images. The text instances are annotated with word-level, character-level rotated bounding boxes and transcriptions. This dataset is widely adopted for pretraining the model. In our experiment, we only use word-level annotations.

ICDAR2015 [63] is a multi-oriented text detection and recognition scene dataset, including 1000 training images and 500 test images. The text instances are annotated with word-level quadrangles and corresponding transcriptions. This dataset includes many incidental scene text instances such as blur or small text instances, which bring challenge to text detection and recognition.

Total-Text [64] is a scene text dataset including the horizontal, oritented and curved text. Total-Text contains 1255 training images and 300 test images. The text instances are annotated with word-level polygons and corresponding transcriptions.

CTW1500 [65] includes 1000 training images and 500 test images. This dataset contains horizontal, multi-oritented and curved text instances. Different from datasets mentioned above, the text instances are annotated with line-level polygons with 14 vertexes.

B. Implementation

We implement our method based on mmdetecion[1] in Pytorch [66] and train our models on a workspace with 4 Nvidia V100 GPUs.

1) Boundary Text Spotter: The tasks of text detection and recognition are optimized jointly in an end-to-end manner. The whole training process including two stages: first using SynthText to pretrain the model, then finetuning with the real data.

During the pretraining stage, the mini-batch is set to 24. For data augmentation, images are randomly resized in range of (480,800), then randomly rotated with a certain angle of $(0, \pm 15, \pm 30, \pm 35)$ degree, finally applied with color transform following [29]. During the finetuning stage, the mini-batch is set to 12. Besides, following [8], extra 1162 images (SCUT) from [67] are used for training. For each mini-batch, we sample the image from the SynthText, ICDAR2015, Total-Text, and SCUT with a fixed sample ratio, which is set to $1:1:1:0.8$ respectively. Except the real scene data are randomly resized in the range (320,1600), we follow the same augmentation as the one in the pretraining stage.

We optimize our model using SGD with a weight decay of 0.0001 and momentum of 0.9. For the recognition subnetwork, we multiply the learning rate by a factor of 10. We train our model for 220k iterations for pretraining with an initial learning rate of 0.005, and the learning rate is decayed at the 100k and 200k iteration. During the finetuning, the initial learning rate is set to 0.005 and is decayed at the 100k and 200k iteration. For ICDAR2015 and Total-Text, the model is finetuned for 220k iterations. CTW1500 is annotated with line-level polygons, the spotting results of the finetuned model are inconsistent with annotations, so we finetune the model with CTW1500 for another 50k iterations.

2) Standalone Detection Model: For a better comparison of the detection performance with other methods, we follow a similar training process and data argumentation used in other text detectors. We first pre-train the model with SynthText for 130k iterations. Then we finetune the models on target datasets for 80k iterations. The training batch is set to 16. During pre-training, the initial learning rate is set to 0.01 and is decayed at 50k and 100k iteration. During finetuning stage, the learning rate is set to 0.005 and is decayed to a tenth at the 10k,30k, and 40k iteration. For data augmentations, during the pre-training period, the images are simply recalled to 800×800 pixels. During finetuning, the augmentation includes: (1) random rotation with certain angle of $(0, \pm 10, \pm 15)$; (2) random cropping; (3) rescaling to 960×960 pixels; (4) color transform.

3) Label Generation: During training, we need equidistantly spaced boundary points to construct the target boundary points. However, only corner points are given in the ground truth. So we need to sample points on the longer sides of text boundary by Algorithm 1. In our experiments, K is set to 7.

[1]https://github.com/open-mmlab/mmdetection

Algorithm 1 Generate Target Boundary Points

Require:
Points on each long side: $P = \{p_0, p_1, ..., p_{N-1}\}$.
The excepted number of sampled points: K.
Ensure:
The generated target boundary points:
$Q = \{q_0, q_1, ..., q_{K-1}\}$.

$distance[0]$ =0; $len[0] = 0$;
for i=0; i < N-1; i++ **do**
  // calculate distance between two adjacent points
  $distance[i+1]$ = distance(p_i, p_{i+1});
end for
for j=1; j<N; j++ **do**
  $len[j] = len[j-1] + distance[j]$;
end for
$average\_distance = \frac{1}{K-1} * \sum distance[i]$;
for i=0; i < K; i++ **do**
  $cur\_pos = average\_distance * i$;
  for j=0; j < N-1; j++ **do**
    if $len[j] \leqslant cur\_pos < len[j+1]$ **then**
      $q_i = \frac{(p_{j+1}-p_j)*(cur\_pos-len[j])}{len[j+1]-len[j]} + p_j$;
    end if
  end for
end for

TABLE II
RESULT ON TOTAL-TEXT. "P", "R" AND "F" MEAN PRECISION, RECALL AND F-MEASURE IN DETECTION TASK RESPECTIVELY. "NONE" MEANS RECOGNITION WITHOUT ANY LEXICON, "FULL" LEXICON CONTAINS ALL WORDS IN TEST SET. RESULTS WITH * DENOTE THAT THE EVALUATED MODELS ARE TRAINED WITH MORE EXTERNAL DATASETS

Method	Detection			End-to-end		FPS
	P	R	F	None	Full	
Total-Text [64]	40.0	33.0	36.0	-	-	-
TextBoxes [11]	62.1	45.5	52.5	36.3	48.9	-
Mask TextSpotter [13]	81.8	75.4	78.5	52.9	71.8	-
Mask TextSpotter v2 [22]	87.0	80.2	83.4	65.3	77.4	-
Mask TextSpotter v3 [17]	-	-	-	71.2	78.4	-
TextDragon [68]	85.6	75.7	80.3	48.8	74.8	-
Feng et al. [59]	87.1	80.3	83.5	54.6	78.6	7.2
ABCNet* [24]	-	-	-	64.2	75.7	17.9
Qin et al. R50* [68]	83.3	83.4	83.3	67.8	-	-
CharNet R-50* [69]	81.0	88.6	84.6	63.6	-	-
PGNet [70]	85.5	86.8	86.1	61.7	-	38.2
PAN++* [71]	89.9	81.0	85.3	68.6	78.6	21.1
Conference version [18]	88.9	85.0	87.0	65.0	76.1	3.0
Ours(S600)	91.0	77.6	83.7	63.0	76.4	19.7
Ours(S800)(det only)	84.6	82.4	83.5	-	-	14.7
Ours(S800)	89.6	81.2	85.2	66.2	78.4	13.4

C. Results on Curved Text Spotting

We conduct experiments on two curved datasets (*i.e.*, Total-Text and CTW1500) to verify the superiority of Boundary TextSpotter.

1) Experiment Results on Total-Text: As shown in Tab. II, our method achieves competitive end-to-end recognition performance. Meanwhile, we have a significant speed advantage over the recent method. The conference version [18] has shown its superiority on curved text, and the proposed method furthermore enhances the performance and efficiency. Though the detection performance is relatively low, benefiting from the precise predicted boundary points, the proposed method

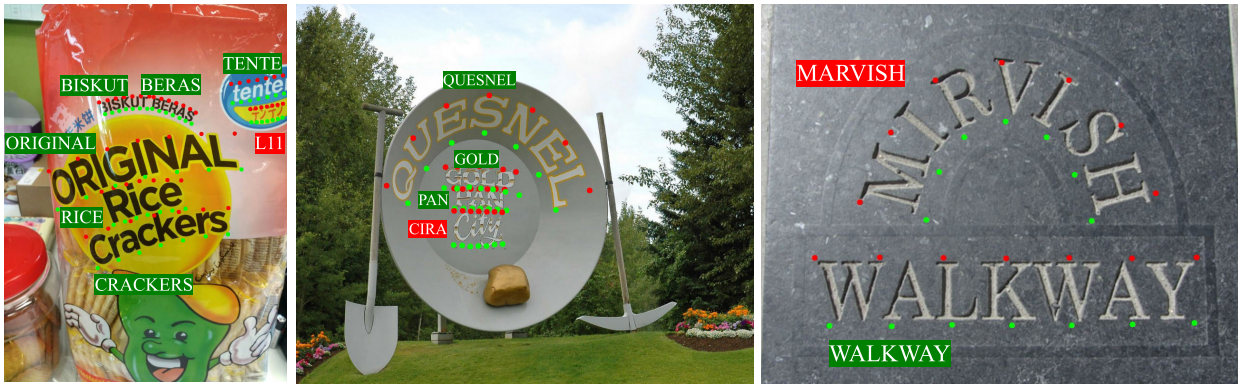


Fig. 8. Examples of text spotting results on Total-Text.

TABLE III

RESULT ON CTW1500. "P", "R" AND "F" MEAN PRECISION, RECALL AND F-MEASURE IN DETECTION TASK RESPECTIVELY. "NONE" MEANS RECOGNITION WITHOUT ANY LEXICON, "FULL" LEXICON CONTAINS ALL WORDS IN TEST SET

Method	Detection			End-to-end		FPS
	P	R	F	None	Full	
FOTS* [9]	79.5	52.0	62.8	21.1	39.7	
TextDragon [68]	84.5	82.8	83.6	39.7	72.4	
Feng et al. [59]	87.3	81.8	84.5	41.4	74.3	7.2
ABCNet* [24]				45.2	74.1	
Ours(det only)	86.0	78.9	82.3	-	-	23.5
Ours(S600)	88.1	78.2	82.9	46.1	73.0	18.5

TABLE IV

RESULT ON ICDAR 2015. "P", "R" AND "F" MEAN PRECISION, RECALL AND F-MEASURE IN DETECTION TASK RESPECTIVELY. "S", "W" AND "G" MEAN RECOGNITION WITH STRONG, WEAK AND GENERIC LEXICON RESPECTIVELY. RESULTS WITH * DENOTE THAT THE EVALUATED MODELS ARE TRAINED WITH MORE EXTERNAL DATASETS

Method	Detection			End-to-end			FPS
	P	R	F	S	W	G	
DeepTextSpotter [5]				54.0	51.0	47.0	-
TextBoxes++ [10]	87.2	76.7	81.7	73.3	65.9	51.9	-
He* et al [6]	87.0	86.0	87.0	82.0	77.0	63.0	-
FOTS * [9]	91.0	85.2	88.0	81.1	75.9	60.8	7.5
Mask TextSpotter [13]	91.6	81.0	86.0	79.3	73.0	62.4	-
Mask TextSpotter v2 [22]	86.6	87.3	87.0	83.0	77.7	73.5	3.0
Mask TextSpotter v3 [17]	-	-	-	83.3	78.1	74.2	2.5
TextDragon [68]	92.5	83.8	87.9	82.5	78.3	65.2	-
Feng et al. [59]	91.4	85.3	88.2	83.7	79.2	65.8	6.8
Qin et al. R50* [68]	89.3	85.7	87.5	83.3	79.9	67.9	-
CharNet R-50* [69]	91.1	88.3	89.7	80.1	74.4	62.1	-
PGNet [70]	91.8	84.8	88.2	82.9	77.7	62.3	-
PAN++* [71]	91.4	83.9	87.5	82.7	78.2	69.2	13.8
Conference version [18]	89.8	87.5	88.6	79.7	75.2	64.1	2.0
Ours(S720)	87.2	81.6	84.3	79.3	75.3	65.0	13.5
Ours(S1080)(det only)	85.8	84.2	85.0	-	-	-	-
Ours(S1080)	88.7	84.6	86.6	82.5	77.4	71.7	6.1

outperforms conference version with improvements 1.2% and 2.3% respectively on end-to-end recognition without lexicon and recognition with the full lexicon. Compared to ABCNet [24], our method with an input image of which short side is 600 pixels has fast speed and better end-to-end recognition without lexicon. Noted that the speed of ABCNet is evaluated with Tesla V100 while ours is evaluated with Titan X. Some qualitative results are shown in Fig. 8.

2) Experiment Results on CTW1500: The text instances in CTW1500 dataset are annotated with line level, most of which belong to long curved texts. This dataset is usually used for evaluating the performance that algorithms deal with long text instances. As shown in Tab. III, our method achieves the state-of-the-art end-to-end recognition performance without lexicon, which demonstrates that our method can handle long text instances of arbitrary shapes well. Besides, our method runs in the fastest inference speed among existing methods while keeping comparable end-to-end recognition performance when full lexicon is used for evaluation, achieving a better trade-off between precision and speed. Some examples in Fig. 9 show that our method can predict precise boundary points of the long curved text.

D. Results on Multi-Oriented Text Spotting

To verify the effectiveness of our method on multi-oriented scene text, we conduct experiments on ICDAR2015. We use two different resolutions as input, the small size 768×1280 and the big size 1920×1280. As shown in Tab. IV, although our method is not best in end-to-end recognition performance, we achieve a good trade-off between performance and speed. Meanwhile, we should notice that Mask TextSpotter v2 and v3 use character-level annotations and use ensemble results of two recognition branches for better recognition. Compared to the conference version, our method has greater improvement by 2.8%, 2.2%, and 7.6% with strong, weak, and generic lexicons respectively.

With a smaller input size, our method has comparable performance with generic lexicons while is far faster than other methods except for Mask TextSpotter v2 and v3. Noted that the F-measure of detection task is relatively low, we extrapolate that this mainly results from the lack of RPN. Unlike Boundary TextSpotter v1 and Mask Textspotter v2 generating massive proposals by RPN, our method is anchor-free and directly outputs boundary points on a single level feature map, harming the recall of text detector. Some qualitative results are shown in Fig. 10.

E. Standalone Detection Performance

We train a model which removes the recognition part from the original network to explore the superiority of

Fig. 9. Examples of text spotting results on CTW1500.

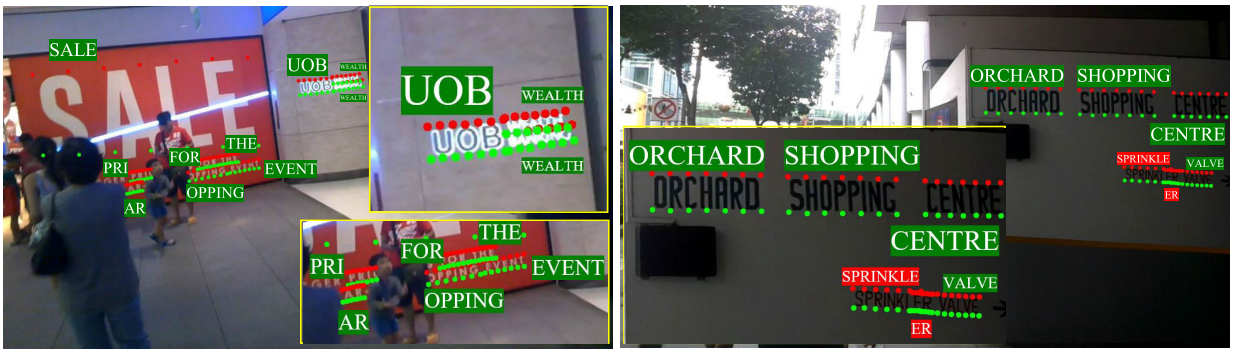


Fig. 10. Examples of text spotting results on ICDAR2015.

the detection part of our method. We conduct experiments on the three datasets including ICDAR2015, Total-Text and CTW1500. Some recent representative text detection methods [25], [36], [37], [72], [73] are selected for comparison.

As shown in Tab. V, the competitive performance and speed demonstrate the overall superiority of our model against recent methods. Though our method is based on regression, the performance on two curved datasets is comparable to the methods [37] based on segmentation. Meanwhile, our method has a much simple post-process compared to CRAFT [25], GNNets [73], and PSENet [36], which contributes to the efficiency improvement.

TABLE V
THE PERFORMANCE OF STANDALONE DETECTION NETWORK

Datasets	Methods	P	R	F	FPS
Total-Text	TextField [40]	81.2	79.9	80.6	-
	CRAFT [25]	87.6	79.9	83.6	-
	PSE-1s [36]	84.0	78.0	80.9	3.9
	DB-ResNet-50 [37]	87.1	82.5	84.7	32.0
	LOMO [72]	88.6	75.7	81.6	-
	DRRG [34]	86.5	84.9	85.7	-
	Ours(640)	87.5	77.3	82.1	24.0
	Ours(860)	88.1	83.9	86.0	14.2
CTW1500	TextFied [40]	83.0	79.8	81.4	-
	PSE-1s [36]	84.8	79.7	82.2	3.9
	DB-ResNet50 [37]	86.9	80.2	83.4	22.0
	LOMO [72]	89.2	69.6	78.4	-
	DRRG [34]	83.0	85.9	84.5	-
	Ours(S600)	86.0	80.3	83.1	26.0
ICDAR2015	TextField [40]	84.3	80.1	82.4	6.0
	GNNets [73]	90.4	86.7	88.5	2.1
	CRAFT [25]	89.8	84.3	86.9	-
	DB-ResNet-50 [37]	91.8	83.2	87.3	12.0
	LOMO [72]	91.3	83.5	87.2	-
	MOST [74]	89.1	87.3	88.2	10.0
	Ours(S736)	88.6	78.8	83.4	16.8
	Ours(S1300)	89.3	85.3	87.2	6.9

F. Ablation Studies

First, we discuss the impact of the recognition module to the detection module. Then, we make some quantitative analysis of the hyperparameter K to the performance. Finally, an ablation study on the boundary refinement module is conducted to verify its effectiveness.

1) The Impact of the Recognition Module to the Detection Module: We train a model named "Ours (det only)" which removes the recognition module from the original network to discuss the advantage of training detection and recognition jointly. The experimental setting (*i.e.*, the data augmentation and the training datasets) is the same as the ones used for training Boundary TextSpotter, which is introduced in Sec. IV-B.1. As shown in Tab. II, III, IV, the detection results of "Ours" exceed "Ours (det only)" by 1.7%, 0.6% and 1.6% on Total-Text, CTW1500 and ICDAR2015 respectively, which demonstrate that the detection task can benefit from the recognition task when jointly training. Those improvements mainly come from eliminating false positives by optimizing the features of the backbone with the help of the recognition task.

2) Discussion on the Number of Boundary Points: In our experiments, we use fixed number boundary points to represent the text instance. In this situation, the number of boundary

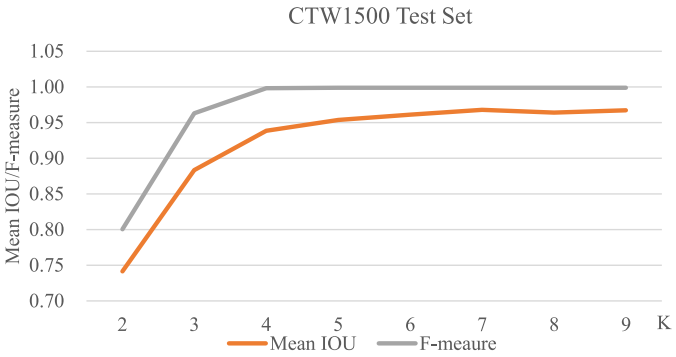


Fig. 11. Upper bound of boundary point presentation on CTW1500 train set.

TABLE VI
THE IMPACT OF HYPER PARAMERTER K ON TOTALTEXT

K	3	5	7	9
Detection	75.8	77.1	85.2	85.1
Recognition	33.5	67.6	78.4	78.1

points K is an important hyperparameter that needs to be discussed. For keeping the whole network simple and effective, the hyperparameter K should have two properties: 1) the value of K should be as small as possible for a lightweight architecture of the boundary regression head; 2) the $2K$ (K boundary points on each long side of text instance) boundary points can cover text instances of arbitrary shapes precisely.

Ideally, the overlap between the original ground-truth polygon and its corresponding $2K$ generated boundary points is higher, the $2K$ boundary points cover this text instance more precisely. To evaluate the precision of the generated boundary points, the IOU and F-measure between the $2K$ boundary points and its corresponding ground-truth are used. We collect all ground-truth polygons and their corresponding boundary points generated by Algorithm 1 with different values of the hyperparameter K from CTW1500 test dataset, and calculate IOU and F-measure between them.

As shown in Fig. 11, both the values of IOU and F-measure increase as more boundary points are used to represent the shape of text instance. In particular, the mean IOU is greater than 0.95 and the F-measure approaches to 1.0 when K equals to 7. This shows that the limited number boundary points can depict the curved text well. Furthermore, we conduct experiments on TotalText to discuss the impact of K to the performance under the value 3, 5, 7, 9. From Tab. VI, we can conclude that detection and recognition performances increase when k varies from 3 to 7. However, as K reaches 7, the performances become stable despite larger K. So, we set K to 7 in this work.

3) The Boundary Refinement Module: As we mentioned above, directly predicting the boundary points is challenging. The boundary refinement module is proposed to refine the boundary points. Through this module, the detected boundary points can have a high overlap with groundtruth, resulting in completely extracting the feature of the whole text region to the subsequent recognition. Especially for recognition, the slight perturbation on the predicted boundary points wouldn't cause the noticeable change of the overlap with the groundtruth but can result in wrong recognition due to the dramatic deformation of the extracted text region. Specifically, we compare the recognition result of text instances described by each round of detected boundary points. Note that we only perform text recognition on the final detection results during inference. As shown in Fig. 12, the initial boundary points predicted by the boundary regression head suffer from fitting the text instance properly, resulting in incorrect recognition. After the process of refining, the predicted boundary points gradually become accurate and the recognition is simultaneously corrected.

TABLE VII
ABLATION STUDY OF BOUNDARY REFINEMENT MODULE. THE NUMBER IN THIS TABLE IS THE VALUE OF F-MEASURE. "BASE" MEANS NO REFINE MODULE USED, "REFINE x" STANDS FOR REFINING FOR x TIMES. "IOU@0.5" AND "IOU@0.75" MEAN THAT EVALUATION METRIC USE IOU THRESHOLD OF 0.5 AND 0.75 SEPARATELY

	TotalText	Base	Refine 1	Refine 2
IOU@0.5	Detection	84.5	85.2	85.6
	Recognition	53.5	64.7	64.9
IOU@0.75	Detection	47.8	66.8	67.0
	Recognition	35.5	54.0	54.0

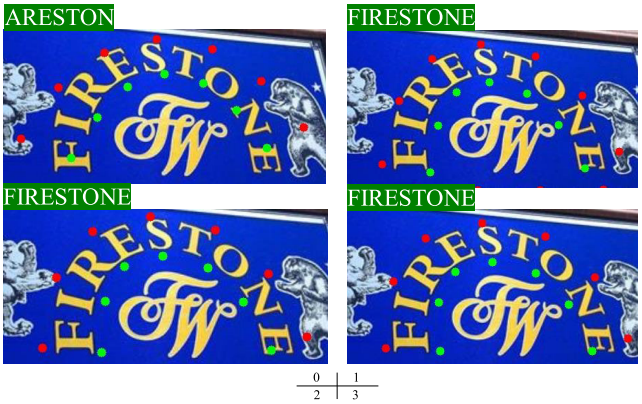


Fig. 12. The process of boundary points refinement. The predictions of different stages are visualized. The numbers below the picture indicate how many refine modules are used.

To further verify the effectiveness of this module quantitatively, we discuss the influence of the refinement times on both text detection and recognition tasks. We train two models of our method for comparison with Total-Text dataset: 1) "Base" model is trained by removing the boundary refinement module from the whole network; 2) the complete model.

Noted that the evaluation metric is base on PASCAL with a polygon IOU threshold of 0.5, which is not proper to describe the precision of boundary points for a text spotter, as the text region satisfying IOU@0.5 is positive for detection task but not for the recognition task. So we also evaluate the performance with IOU threshold of 0.75. As shown in Tab. VII, the detection performance of "Base" model decreases a lot especially under IOU@0.75. The performance of end-to-end recognition drops dramatically on both IOU settings.

Compared to refinement once, the second refinement on boundary points only promotes a little both performances of

Fig. 13. Failure examples of Boundary TextSpotter.

detection and end-to-end recognition, which demonstrates that it is not necessary to refine the boundary points too many times. In our experiments, the boundary points are only refined twice for efficiency.

G. Speed Analysis

Since the inference speed varies depending on the image resolution and hardware. For a fair comparison, we use GTX Titan X GPU for speed testing, and evaluation with different resolutions. As shown in Tab. II-IV, our method has a very competitive speed compared to other methods. To further investigate the efficiency of the model, we analyze the time percentage of the four modules: feature extraction module, boundary prediction module, boundary refinement module and text recognition module. With an input image of 600×600 on Total-Text, the four modules consume around 15ms, 7ms, 2ms, 16ms. It is concluded that one boundary refinement module only takes around 4.76% of the whole inference time, and the duration mainly comes from the feature extraction module and the text recognition module.

However, the inference speed does not have a significant advantage on ICDAR2015. The main reason is that most text instances in this dataset are small, and the network requires a high resolution (the short side is set to 1080) of the input image to keep recognition accuracy. Consequently, the high-resolution input images significantly enlarge the running time of the feature extraction module, as this module extracts feature maps from the raw pixels of images. In this case, the time proportion saved by our boundary refinement module is minimal, weakening the advantage of our method in terms of inference time.

H. Weakness

The proposed method achieves a good balance between efficiency and performance for arbitrary-shaped text spotting in most cases. It is still difficult to handle some special cases, such as circular text demanding more boundary points, the long text beyond the receptive field of the network, and artistic text with a complex appearance and unbalanced distribution among datasets. These special cases are also challenging to other methods and have a relatively small proportion in the real scene.

The specific failure examples of Boundary TextSpotter are shown in Fig. 13. The two wrong results in the first column are caused by the complex font style of artistic text (the up case) and occluded characters (the below case), both attributable to the text recognition module despite the tight boundary points. The incorrect results ascribe to the under-surrounding/over-surrounding text instances by boundary points in the second column. Finally, reversing the two group boundary points results in the wrong sequence translation for the case in the last column.

V. Conclusion

In this paper, we further explore the boundary points representation in arbitrary shape text spotting and exploit its flexibility, accuracy, and efficiency. We proposed a novel boundary point detector with a refinement module, which can obtain accurate boundary points with competitive inference speed. With the feature extracted based on boundary points, an attention-based sequence recognizer is applied to make the framework more precise and free from character-level annotations. Based on boundary points, our method achieves a better trade-off between precision and efficiency and has competitive performance compared to recent methods on several datasets.

References

[1] H. Wang, X. Bai, M. Yang, S. Zhu, J. Wang, and W. Liu, "Scene text retrieval via joint text detection and similarity learning," in *Proc. CVPR*, 2021, pp. 4558–4567.

[2] X. Bai, M. Yang, P. Lyu, Y. Xu, and J. Luo, "Integrating scene text and visual appearance for fine-grained image classification," *IEEE Access*, vol. 6, pp. 66322–66335, 2018.

[3] Y. Zhu, M. Liao, W. Liu, and M. Yang, "Cascaded segmentation-detection networks for text-based traffic sign detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 209–219, Jan. 2018.

[4] X. Rong, C. Yi, and Y. Tian, "Recognizing text-based traffic guide panels with cascaded localization network," in *Proc. ECCV*, 2016, pp. 109–121.

[5] M. Busta, L. Neumann, and J. Matas, "Deep TextSpotter: An end-to-end trainable scene text localization and recognition framework," in *Proc. ICCV*, Oct. 2017, pp. 2204–2212.

[6] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, "An end-to-end textspotter with explicit alignment and attention," in *Proc. CVPR*, Jun. 2018, pp. 5020–5029.

[7] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *Int. J. Comput. Vis.*, vol. 116, no. 1, pp. 1–20, 2016.

[8] H. Li, P. Wang, and C. Shen, "Towards end-to-end text spotting with convolutional recurrent neural networks," in *Proc. ICCV*, 2017, pp. 5238–5246.

[9] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, "FOTS: Fast oriented text spotting with a unified network," in *Proc. CVPR*, Jun. 2018, pp. 5676–5685.

[10] M. Liao, B. Shi, and X. Bai, "TextBoxes++: A single-shot oriented scene text detector," *IEEE Trans. Image Process.*, vol. 27, no. 8, pp. 3676–3690, Aug. 2018.

[11] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "TextBoxes: A fast text detector with a single deep neural network," in *Proc. AAAI*, 2017, pp. 4161–4167.

[12] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. ICCV*, Nov. 2011, pp. 1457–1464.

[13] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai, "Mask TextSpotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," in *Proc. ECCV*, 2018, pp. 67–83.

[14] Y. Liu and L. Jin, "Deep matching prior network: Toward tighter multi-oriented text detection," in *Proc. CVPR*, Jul. 2017, pp. 1962–1969.

[15] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep direct regression for multi-oriented scene text detection," in *Proc. ICCV*, 2017, pp. 745–753.

[16] S. Qin, A. Bissacco, M. Raptis, Y. Fujii, and Y. Xiao, "Towards unconstrained end-to-end text spotting," in *Proc. ICCV*, Oct. 2019, pp. 4704–4714.

[17] M. Liao, G. Pang, J. Huang, T. Hassner, and X. Bai, "Mask TextSpotter v3: Segmentation proposal network for robust scene text spotting," in *Proc. ECCV*, 2020, pp. 706–722.

[18] H. Wang *et al.*, "All you need is boundary: Toward arbitrary-shaped text spotting," in *Proc. AAAI*, 2020, pp. 12160–12167.

[19] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proc. ICPR*, 2012, pp. 3304–3308.

[20] B. Shi, X. Bai, and C. Yao, "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.*, vol. 39, no. 11, pp. 2298–2304, Nov. 2016.

[21] Y. Sun, C. Zhang, Z. Huang, J. Liu, J. Han, and E. Ding, "TextNet: Irregular text reading from images with an end-to-end trainable network," in *Proc. ACCV*, 2018, pp. 83–99.

[22] M. Liao, P. Lyu, M. He, C. Yao, W. Wu, and X. Bai, "Mask TextSpotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 532–548, Feb. 2021.

[23] L. Qiao *et al.*, "Text perceptron: Towards end-to-end arbitrary-shaped text spotting," in *Proc. AAAI*, 2020, pp. 11899–11907.

[24] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, "ABCNet: Real-time scene text spotting with adaptive Bezier-curve network," in *Proc. CVPR*, Jun. 2020, pp. 9809–9818.

[25] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proc. CVPR*, 2019, pp. 9365–9374.

[26] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. CVPR*, Jun. 2016, pp. 4159–4167.

[27] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proc. CVPR*, Jul. 2017, pp. 2550–2558.

[28] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. ECCV*, 2016, pp. 56–72.

[29] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. ECCV*, 2016, pp. 21–37.

[30] L. Huang, Y. Yang, Y. Deng, and Y. Yu, "DenseBox: Unifying landmark localization with end to end object detection," *CoRR*, vol. abs/1509.04874, pp. 1–13, Sep. 2015.

[31] X. Zhou *et al.*, "East: An efficient and accurate scene text detector," in *Proc. CVPR*, 2017, pp. 5551–5560.

[32] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, "Multi-oriented scene text detection via corner localization and region segmentation," in *Proc. CVPR*, Jun. 2018, pp. 7553–7563.

[33] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, "TextSnake: A flexible representation for detecting text of arbitrary shapes," in *Proc. ECCV*, 2018, pp. 20–36.

[34] S.-X. Zhang *et al.*, "Deep relational reasoning graph network for arbitrary shape text detection," in *Proc. CVPR*, Jun. 2020, pp. 9696–9705.

[35] J. Tang, Z. Yang, Y. Wang, Q. Zheng, Y. Xu, and X. Bai, "SegLink++: Detecting dense and arbitrary-shaped scene text by instance-aware component grouping," *Pattern Recognit.*, vol. 96, Dec. 2019, Art. no. 106954.

[36] W. Wang *et al.*, "Shape robust text detection with progressive scale expansion network," in *Proc. CVPR*, Jun. 2019, pp. 9328–9337.

[37] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, "Real-time scene text detection with differentiable binarization," in *Proc. AAAI*, 2020, pp. 11474–11481.

[38] W. Wang *et al.*, "Efficient and accurate arbitrary-shaped text detection with pixel aggregation network," in *Proc. ICCV*, Oct. 2019, pp. 8439–8448.

[39] Z. Tian *et al.*, "Learning shape-aware embedding for scene text detection," in *Proc. CVPR*, Jun. 2019, pp. 4234–4243.

[40] Y. Xu, Y. Wang, W. Zhou, Y. Wang, Z. Yang, and X. Bai, "TextField: Learning a deep direction field for irregular scene text detection," *IEEE Trans. Image Process.*, vol. 28, no. 11, pp. 5566–5579, Nov. 2019.

[41] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, "Arbitrary shape scene text detection with adaptive text region representation," in *Proc. CVPR*, Jun. 2019, pp. 6449–6458.

[42] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proc. ICCV*, Dec. 2013, pp. 785–792.

[43] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Deep features for text spotting," in *Proc. ECCV*, 2014, pp. 512–528.

[44] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang, "Reading scene text in deep convolutional sequences," in *Proc. AAAI*, 2016, pp. 1–8.

[45] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "ASTER: An attentional scene text recognizer with flexible rectification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2035–2048, Sep. 2019.

[46] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou, "AON: Towards arbitrarily-oriented text recognition," in *Proc. CVPR*, Jun. 2018, pp. 5571–5579.

[47] H. Li, P. Wang, C. Shen, and G. Zhang, "Show, attend and read: A simple and strong baseline for irregular text recognition," in *Proc. AAAI*, 2019, pp. 8610–8617.

[48] M. Liao *et al.*, "Scene text recognition from two-dimensional perspective," in *Proc. AAAI*, 2019, pp. 8714–8721.

[49] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. CVPR*, Jul. 2017, pp. 2117–2125.

[50] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, Jun. 2016, pp. 770–778.

[51] M. Liao, Z. Zhu, B. Shi, G.-S. Xia, and X. Bai, "Rotation-sensitive regression for oriented scene text detection," in *Proc. CVPR*, Jun. 2018, pp. 5909–5918.

[52] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. ICCV*, Oct. 2019, pp. 9627–9636.

[53] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *Proc. ICCV*, 2019, pp. 6569–6578.

[54] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, "RepPoints: Point set representation for object detection," in *Proc. ICCV*, Oct. 2019, pp. 9657–9666.

[55] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. ICCV*, 2017, pp. 2980–2988.

[56] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. CVPR*, 2018, pp. 6154–6162.

[57] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proc. CVPR*, 2018, pp. 4203–4212.

[58] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proc. NIPS*, 2015, pp. 2017–2025.

[59] W. Feng, F. Yin, X.-Y. Zhang, W. He, and C.-L. Liu, "Residual dual scale scene text spotting by fusing bottom-up and top-down processing," *Int. J. Comput. Vis.*, vol. 129, no. 3, pp. 619–637, Mar. 2021.

[60] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

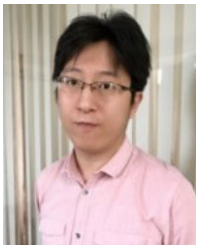
[61] P. Zhou *et al.*, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proc. ACL*, 2016, pp. 207–212.
[62] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. CVPR*, Jun. 2016, pp. 2315–2324.
[63] D. Karatzas *et al.*, "ICDAR 2015 competition on robust reading," in *Proc. 13th ICDAR*, 2015, pp. 1156–1160.
[64] C.-K. Ch'ng, C. S. Chan, and C.-L. Liu, "Total-text: Toward orientation robustness in scene text detection," *Int. J. Document Anal. Recognit.*, vol. 23, pp. 31–52, Aug. 2019.
[65] L. Yuliang, J. Lianwen, Z. Shuaitao, and Z. Sheng, "Detecting curve text in the wild: New dataset and new solution," 2017, *arXiv:1712.02170*.
[66] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. 31st Conf. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, 2017, pp. 1–4.
[67] Z. Zhong, L. Jin, S. Zhang, and Z. Feng, "DeepText: A unified framework for text proposal generation and text detection in natural images," 2016, *arXiv:1605.07314*.
[68] W. Feng, W. He, F. Yin, X.-Y. Zhang, and C.-L. Liu, "TextDragon: An end-to-end framework for arbitrary shaped text spotting," in *Proc. ICCV*, vol. 2019, pp. 9076–9085.
[69] L. Xing, Z. Tian, W. Huang, and M. R. Scott, "Convolutional character networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9126–9136.
[70] P. Wang *et al.*, "PGNet: Real-time arbitrarily-shaped text spotting with point gathering network," in *Proc. AAAI*, 2021, pp. 2782–2790.
[71] W. Wang *et al.*, "PAN++: Towards efficient and accurate end-to-end spotting of arbitrarily-shaped text," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5349–5367, Sep. 2021.
[72] C. Zhang *et al.*, "Look more than once: An accurate detector for text of arbitrary shapes," 2019, *arXiv:1904.06535*.
[73] J. Duan *et al.*, "Geometry normalization networks for accurate scene text detection," in *Proc. ICCV*, Oct. 2019, pp. 9136–9145.
[74] M. He *et al.*, "MOST: A multi-oriented scene text detector with localization refinement," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8813–8822.

Pu Lu received the bachelor's and B.S. degrees from the School of Electronic Information and Communications, Huazhong University of Science and Technology (HUST), Wuhan, China, in 2018 and 2021, respectively. His research interests include scene text detection and text spotting.

Hao Wang received the B.S. degree from the School of Computer and Information, China Three Gorges University (CTGU), Yichang, China, in 2017. He is currently pursuing the Ph.D. degree with the School of Electronic Information and Communications, Huazhong University of Science and Technology (HUST), Wuhan, China. His current research interests include scene text spotting and retrieval, and multi-modal analysis.

Shenggao Zhu received the bachelor's degree in electronic engineering and information science from the University of Science and Technology of China (USTC) in 2011, and the Ph.D. degree in computer science from the National University of Singapore (NUS) in 2017. He joined Huawei Cloud in 2017, where he is currently a Technical Expert. His research interests include computer vision and AI applications.

Jing Wang received the bachelor's degree from the University of Science and Technology of China in 2010, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2016. He has published more than ten articles related to artificial intelligence and data analysis, and provides more than 30 suggestions on CVE security vulnerabilities. His research interests include anomaly detection, machine learning related to finance, and deep learning-based image detection and recognition.



Xiang Bai (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronics and information engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 2003, 2005, and 2009, respectively. He is currently a Professor with the School of Electronic Information and Communications, HUST. He is also the Vice-Director of the National Center of Anti-Counterfeiting Technology, HUST. His research interests include object recognition, shape analysis, scene text recognition, and intelligent systems. He serves as an Associate Editor for *Pattern Recognition*, *Pattern Recognition Letters*, *Neurocomputing*, and *Frontiers of Computer Science*.

Wenyu Liu (Senior Member, IEEE) received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 1986, and the M.S. and Ph.D. degrees in electronics and information engineering from the Huazhong University of Science and Technology (HUST), Wuhan, China, in 1991 and 2001, respectively. He is currently a Professor and an Associate Dean of the School of Electronic Information and Communications, HUST. His current research interests include computer vision, multimedia, and machine learning.