Deep Multi-Scale Context Aware Feature Aggregation for Curved Scene Text Detection

Pengwen Dai, Hua Zhang, and Xiaochun Cao, *Senior Member, IEEE*

***Abstract*—Scene text plays a significant role in image and video understanding, which has made great progress in recent years. Most existing models on text detection in the wild have the assumption that all the texts are surrounded by a rotated rectangle or quadrangle. While there also exist lots of curved texts in the wild, which would not be bounded by a regular bounding box. In this paper, we develop a novel architecture to localize the text regions, which can deal with curved-shape scene texts. Specifically, we first design a text-related feature enhancement module by incorporating the prior knowledge of the text shape to enhance the feature representations. After that, based on the enhanced features, we employ a region proposal network to generate the candidate boxes of scene texts. For each text candidate, a pyramid region-of-interest pooling attention module is utilized to extract the fixed-size features. Finally, we exploit the box-aware context-based text segmentation module and box refinement network to obtain the location of scene text. Experiments are conducted on four challenging benchmarks *CTW1500*, *totalTEXT*, *ICDAR-2015* and *MLT*, and the experimental results have demonstrated the superiority of our model.**

***Index Terms*—Curved text detection, text-related feature enhancement, pyramid pooling attention, box-aware segmentation.**

Manuscript received July 20, 2018; revised February 20, 2019, May 14, 2019, and September 17, 2019; accepted November 3, 2019. Date of publication November 11, 2019; date of current version July 24, 2020. This work was supported in part by the National Key R&D Program of China under Grant 2018YFB0803701, in part by the Beijing Natural Science Foundation under Grant 4172068, in part by the Key Program of the Chinese Academy of Sciences under Grant QYZDB-SSW-JSC003, in part by The Open Research Fund from Shenzhen Research Institute of Big Data, under Grant 2019ORF01010, in part by the National Natural Science Foundation of China under Grant U1636214, in part by the Peng Cheng Laboratory Project of Guangdong Province under Grant PCL2018KP004, and in part by the CCF-Tencent Open Research Fund and National Natural Science Foundation of China under Grant 61602464. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Benoit Huet *(Corresponding author: Xiaochun Cao.)*

P. Dai is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100195, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: daipengwen@iie.ac.cn).

H. Zhang is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100195, China (e-mail: zhanghua@iie.ac.cn).

X. Cao is with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100195, China, the Peng Cheng Laboratory, Cyberspace Security Research Center, Shenzhen 518055, China, the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China, and also with the Shenzhen Research Institute of Big Data, Shenzhen 518000, China (e-mail: caoxiaochun@iie.ac.cn).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2019.2952978

I. Introduction

TEXT in the natural scene image contains valuable semantic information, which can directly transmit text information or indirectly contribute to scene understanding [3], [4]. Reading scene text is thus related to many practical applications, e.g., autonomous driving, robot navigation, video analysis, etc. Scene text detection, as a crucial step in the reading text system, becomes a hot topic in the field of multimedia and document analysis.

Considering the effects of scene factors (e.g., various illumination, perspective distortion, blur, and low-contrast) and the specific characteristics of scene texts (e.g., various aspect ratios, ambiguous boundaries, various fonts, and similar texture with backgrounds), scene text detection has become a challenging task. Recently, there are many methods to detect the horizontal or multi-oriented scene text [1], [5]–[7]. However, some texts distribute along a curved line in the natural scene image, which could not be enclosed well by a regular bounding box based on existing models. Therefore, most of the horizontal and multi-oriented scene text detectors are not suitable for curved scene text detection.

Curved scene text needs more points to annotate its bounding box, compared with the horizontal scene text annotated by two points (e.g., top-left and bottom-right points) and the multi-oriented scene text annotated by four corner points of the quadrangle. The multi-oriented detectors do not accurately localize the bounding box of curved scene text, as shown in Fig. 1(a). Although a few curved scene text detection methods have been proposed, they usually regress multiple key points [2] or directly segment the scene text [8] from the entire image based on local cues. The regression of points is sensitive to the location offset of each position. The segmentation performing on the entire image is hard to determine the boundaries when the text instances are very close. Therefore, they generate imprecise boundaries for each text instance. Besides, existing methods [2], [8] for the curved scene text detection utilize single-level feature representations that also do not encode the prior knowledge of the text shape. Such features do not distinguish the scene text from cluttered backgrounds well, as shown in Fig. 1(c).

To achieve more precise boundaries of curved scene text, one of the intuitive way is to utilize the instance segmentation on the candidate boxes. It provides the pixel-level supervision to suppress the effect of background in the candidate boxes. However, it is easy to result in more than one segmented text part in one bounding box. And it also tends to generate under-segmentation

Fig. 1. Illustrations of different text detection methods. (a) The text detection results achieved by the multi-oriented text detection method [1]. (c) The results obtained by the existing curved text detector [2]. (b) (d) The results detected by our proposed model. Green bounding boxes denote the ground truth, and red bounding boxes represent the detected results.

or over-segmentation text instances, because of the ambiguous boundaries of the scene text or the low-contrast differences between the scene text texture and backgrounds. To solve the limitations, we develop the box-aware context-based text segmentation module that encodes the global information of the bounding box.

To promote the ability to discriminate the curved scene text from cluttered backgrounds, we exploit the multi-scale feature aggregation techniques. They are utilized to not only enhance the feature representations with the irregular kernel filters but also fuse different kinds of pyramid pooling features based on the learned weights. As illustrated in Fig. 1(b) and (d), our proposed method can alleviate the above-mentioned limitations. The detected bounding box encloses the arbitrary-shape scene text well, and the discrimination between scene text and background has also been improved.

In this paper, we illustrate our proposed architecture in Fig. 2. Given an input image, the backbone network is utilized to extract the feature representations. Then, we develop a text-related feature enhancement (TFE) module that is composed of a few TFE units (TFEU), to enhance the representations of extracted features. After that, based on the enhanced features, we use a region proposal network (RPN) to generate text proposals. For each proposal, it is correlated with the corresponding enhanced features based on the size of the proposal via the feature assignment gate (FAG), before utilizing the pyramid region-of-interest (ROI) pooling attention (PRPA) module to extract and aggregate the different fixed-size features with the learned weights. Finally, the aggregated pooling features are fed into a box refinement network (BRN) to refine the confidence scores and location offsets of the candidate proposals. And they are also employed to perform text segmentation via the box-aware context-based text segmentation (BCTS) module. We train the BRN and BCTS modules simultaneously. In the testing stage, BCTS will be utilized to segment the scene text based on the refined proposals generated by BRN for more accurate segmentation.

The contributions of our work are summarized as follows:

i) We propose a novel text-related feature enhancement module incorporated with irregular filters, which can improve the discrimination of text feature representations.
ii) A novel attention-based pyramid ROI pooling mechanism is developed to adaptively learn the discriminative features with different pooling windows.
iii) A novel box-aware context-based text segmentation is exploited, which can generate more accurate text boundaries.

The rest of the paper is organized as follows. Section II introduces the related work on scene text detection in details. Then, the proposed method is presented in Section III. Numerous experiments are conducted and the results are described in Section IV. Section V finally concludes the paper.

II. Related Work

Scene text detection has been widely studied for a long period [9], [10]. In this section, we will review the related approaches on this topic.

A. Horizontal Scene Text Detection

Horizontal scene text detector aims to localize the word-level or line-level text with horizontal bounding boxes. Traditional methods could be divided into two types: sliding window and connected component based methods.

The sliding window based methods [11]–[13] usually involve four steps: sub-text generating, sub-text candidates filtering, word-level or line-level text constructing, and text candidates filtering. For example, the authors [11] utilized multi-scale sliding windows to generate sub-text candidates. Then the hand-crafted features extracted from the sub-text candidates were fed into a cascade AdaBoost classifier to remove the windows that do not contain text patterns. Finally, the word-level or line-level text candidates generated by heuristic rules were distinguished by a multilayer perceptron classifier.

The connected component based methods [14]–[18] have a similar process with the sliding window based methods, except that they utilize Stroke Width Transform (SWT) [14], Maximally Stable Extremal Regions (MSER) [15] or their extensions [16]–[18] to generate sub-text candidates. In [18], the authors proposed edge-preserving MSER via employing characterness cues to generate character-level candidates. Then distinguishing characters from the non-characters was performed by modeling a binary label problem with a learning-based energy function. Finally, the remaining characters were grouped into clusters based on the mean shift algorithm.

However, the hand-crafted features are not robust to the complex backgrounds, which limit the performance of detection. Recently, deep learning based methods play significant roles in scene text detection, which could be roughly divided into three categories: classification-based, box-based and segmentation-based methods.

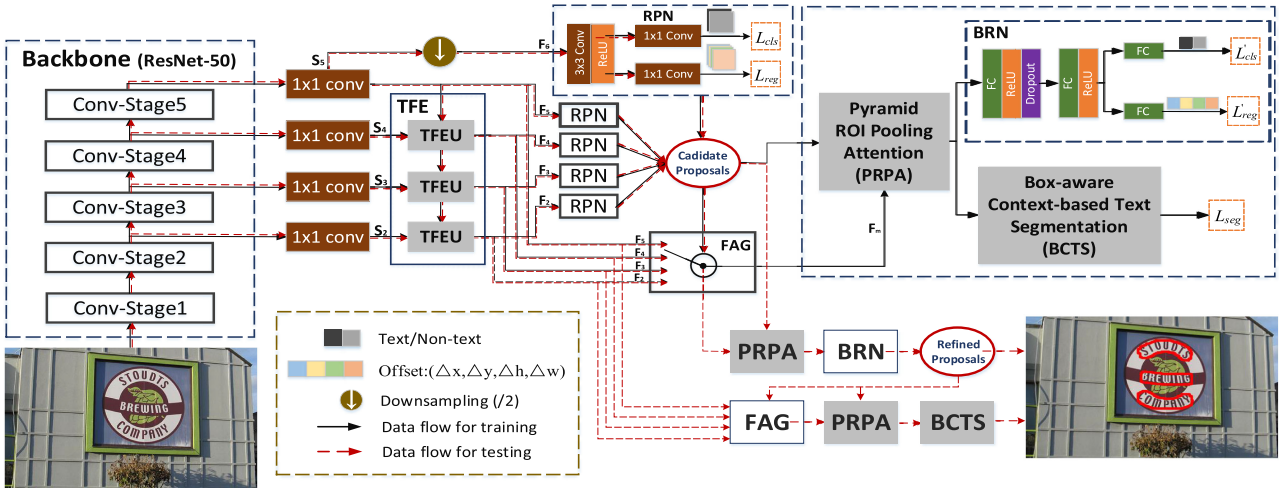

Fig. 2. The framework of our proposed method. The backbone network is firstly employed to extract features of the input image, and then the TFE module enhances these extracted features. Next, RPN is utilized to generate candidate proposals based on these enhanced features. After FAG is employed to assign the corresponding enhanced features for the proposals based on their scales, the PRPA module projects the proposals into fixed-size features. Finally, the fixed-size features are employed for BRN and BCTS simultaneously when training. Note that the BCTS module performs segmentation on the refined proposals generated by BRN in the testing stage. Best view in color.

Specifically, for classification based methods [19]–[23], they employ Convolutional Neural Network (CNN) to extract more representative features and then distinguish text from non-text regions, instead of utilizing artificially-designed features. In [21], He *et al.* used CNN to focus on text-related regions with the multi-level supervised information, instead of directly discriminating text/non-text with CNN in [19], [20]. Additionally, the authors [23] also combined the high-level features extracted by CNN and some manually-designed features to remove false positives.

Differently, bounding box based methods [24]–[26] also employ CNN to regress the location of scene text. A unified framework for scene text detection, that inherited the framework of general object detection [27], [28], was proposed [5], [24]–[26]. In [24], the authors inherited the Single Short Detection (SSD) [28] framework to regress and classify the prior boxes, incorporating with irregular filters. In [25], [26], the fixed-width fine-scale sequential text proposals were connected by Bidirectional Long Short-Term Memory (BLSTM). Additionally, the weakly-supervised learning strategy was also explored for detecting the horizontal scene text in [29].

While for segmentation-based methods, Tang *et al.* [30] proposed a cascaded CNN, which was composed of the detection, segmentation and classification network, to detect text-aware regions, refine these candidate text regions and remove false positives, respectively.

Different from these horizontal scene text detection methods, our curved scene text detector not only could detect the curved-shape scene text but also was fit for detecting the horizontal scene text, since the horizontal-line text can be regarded as a special curved-line text that the curvature is zero.

B. Multi-Oriented Scene Text Detection

Multi-oriented scene text detector attempts to localize the arbitrarily-oriented text by utilizing the rotated rectangular bounding box with angle or the arbitrary quadrangle with four clockwise points. Such detector is an extension from the horizontal scene text detector, which is also fit for detecting the horizontal text.

In traditional multi-oriented scene text detection methods [31]–[34], Yao *et al.* [31] extracted manually-designed features for each component generated by SWT, and then employed a hierarchical agglomerative clustering method to create component chains. Random Forest (RF) classifier was adopted to remove the non-text components and lines. Similarly, in [32], before distinguishing the line-level text from candidates with a classifier, the authors exploited a high-order correlation clustering algorithm to generate the text-line candidates based on a graph that took MSERs as the nodes.

Recently, a remarkable improvement of performance for multi-oriented scene text detection has been achieved by the deep learning based methods [1], [6], [35]–[48].

Before employing a CNN-based classifier to remove false positives from character-level or line-level text candidates, Huang *et al.* [35] utilized a modified MSER approach [15] to generate character candidates and exploited an attention model based on Recurrent Neural Network (RNN) to obtain the line-level text candidates.

Besides, the box-based methods combine the regression and classification task in the CNN framework. Liu *et al.* [36] and Ma *et al.* [37] utilized inclined anchors to facilitate the regression of arbitrarily-oriented bounding boxes. Instead of regressing the whole bounding box, Shi *et al.* [38] predicted the text parts and the links among them with an end-to-end framework. While a weakly-supervised method [39] was proposed to detect text parts, and then they were grouped into text lines with arbitrary orientation. Different from regressing the bounding box, Zhou *et al.* [40] and He *et al.* [41] regressed and classified the pixel in the bounding box to detect the multi-oriented scene text. Besides, in [1], [42], [43], detection and recognition were integrated into a unified framework by a shared network, in which

the learning of recognition could contribute to the scene text detection.

Different from regressing the location coordinates of the bounding box or pixel, segmentation-based methods [44]–[47] take pixels in the ground-truth bounding box as texts and distinguish them from cluttered backgrounds. In [44], the Fully Convolutional Network (FCN) [49] for the semantic segmentation of general object was extended to detect the multi-oriented scene text. Instead of performing segmentation on the entire image, He *et al.* [47] performed the text instance segmentation on the bounding boxes based on a cascaded network. Moreover, the pixel-wise segmentation task was embedded into the framework of box-based methods, forming an end-to-end multi-task learning architecture. He *et al.* [45] employed the output of segmentation as an attention map to help the classification and regression of the bounding box. In [46], the authors assigned a confidence score to each bounding box based on the segmentation map.

Our proposed model is also suitable for detecting the multi-oriented scene text. On the contrary, most of outstanding multi-oriented scene text detectors fail to localize the curved scene text, since the detected bounding box cannot surround the curved-shape bounding box well.

C. Curved Scene Text Detection

In the field of scene text detection, previous methods mainly concentrate on horizontal or multi-oriented scene text detection. Curved scene text detection, as a more general form of horizontal and multi-oriented scene text detection, draws a little attention of researchers.

In [50], the authors proposed a traditional method to detect the curved text in the wild. In this method, a connected component based approach was utilized to generate character-level candidates that would be linked by a graph traversal algorithm, and then the convex hulls of successive pairs in each generated chain were joined together to form the arbitrary-shape text region. In [2], Liu *et al.* presented a box-based approach that not only regressed the circumscribed rectangle of the curved scene text, but also learned the offsets between some key points and the top-left point of the circumscribed rectangle, and then they integrated BLSTM into the framework to smooth the offsets. Recently, segmentation-based methods, that perform segmentation on the entire image, are explored to detect the curved scene text in [8], [51], [52]. The authors [8] directly customized DeconvNet [53] for curved text detection. In [51], Lin *et al.* inherited the FPN framework [54] to generate multiple segmentation maps. After that, a progressive scale expansion algorithm was used to fuse these segmentation maps to produce final results. In [52], an arbitrary-shape text instance was expressed as a sequence of overlapping disks. The authors segmented text regions and text center lines, and predicted the geometry attributes of the disk. After that, a heuristic post-processing algorithm was designed to extract text instances. Besides, the segmentation-based methods that perform segmentation on each bounding box also draw the attention of scholars [55]–[57]. Employing the FCIS framework [58], Dai *et al.* [55] learned the position-sensitive inside/outside score maps based on the fused features. Then a position-sensitive ROI pooling technique was employed to project each proposal to fixed-size inside/outside score maps for the succeeding classification and mask generation of scene text. In [56], for detecting the scene text, the authors employed the pyramid attention network (PAN) [59] as the backbone network and made some modifications in the feature pyramid attention module and global attention module of PAN. In [57], the authors inherited the Mask RCNN [60] framework not only to segment the entire text region, but also to achieve the individual character segmentation in the mask branch. And they performed the character recognition based on the segmented character maps for more accurate detection.

Different from the above-mentioned curved scene text detection methods, we propose a segmentation-based method that performs scene text segmentation on each bounding box based on the context of the whole bounding box. According to the characteristics of scene text, the multi-scale text-related feature aggregation technique and the attention mechanism of pyramid ROI pooling are also proposed to improve the discrimination of text/non-text.

III. Methodology

The framework of our proposed method is illustrated in Fig. 2. We employ ResNet-50 as the backbone network. The features extracted from the backbone network are first enhanced by the text-related feature enhancement (TFE) module. Next, the region proposal network (RPN) [27] is utilized to generate candidate proposals that will be assigned to the corresponding enhanced features via the feature assignment gate (FAG) [60]. Then, the pyramid region-of-interest pooling attention (PRPA) module extracts fixed-size features from the enhanced features for the proposals. In the step of training, these fixed-size features of proposals will be simultaneously fed into the box refinement network (BRN) [27] and box-aware context-based text segmentation (BCTS) module based on the multi-task learning strategy. In the inference stage, instead of performing segmentation on the proposals generated by RPN, BCTS will be used to segment the scene text on the refined proposals generated by BRN. Before introducing the proposed modules in details, some common notations used in the following sections are defined as shown in Table I.

A. Text-Related Feature Enhancement

In the text-related feature enhancement (TFE) module, the irregular convolutional kernels are embedded into each TFE unit (TFEU). Different TFEUs will generate features of different scales, and these multi-scale features are aggregated by a top-down strategy. The structure of TFEU is illustrated in Fig. 3, where $\mathbf{S}_m$ denotes the m-th side-output feature map generated by 256 filters with the 1×1 kernel size. $\mathbf{F}_{m+1}$ and $\mathbf{F}_m$ are the input features and output features, respectively. The spatial dimension of $\mathbf{S}_m$ is formulated as,

$$\mathrm{h}_{\mathbf{S}_m} = \lfloor \mathrm{h}_I / 2^m \rfloor,\ \mathrm{w}_{\mathbf{S}_m} = \lfloor \mathrm{w}_I / 2^m \rfloor,\ m = 2, 3, 4, 5, \quad (1)$$

where $\lfloor \cdot \rfloor$ is the floor operation. $\mathrm{h}_{\mathbf{S}_m}$ and $\mathrm{w}_{\mathbf{S}_m}$ indicate the height and width of $\mathbf{S}_m$. h_I and w_I are the height and width of the input

TABLE I
DEFINITION OF MAIN NOTATIONS

Notation	Description
$\mathbf{S}_m$	The m-th side-output feature map
$\mathbf{F}_m$	The m-th enhanced feature map
ρ_n	The size of the n-th kind of pooling window
$\mathbf{H}_n$	The n-th kind of pooling feature map
$\mathbf{U}_n$	The n-th kind of upsampled feature map
$\mathbf{W}_n$	The attention map for the n-th pooling window
$\mathbf{V}_d$	The weighted feature map of the d-th channel
$\hat{\mathbf{M}}$	The predicted text instance mask
$\mathbf{s}_l$	The confidence score belonging to text or non-text
$\mathbf{a}$	The location parameter vector of anchor
$\hat{\mathbf{t}}$	The predicted location parameter vector
$\mathbf{t}^*$	The location parameter vector of ground truth
$\mathbf{R}_k$	The location parameter vector of the k-th ROI
$\mathbf{O}_k$	The location offsets of the k-th ROI
$\mathbf{C}_k$	The confidence scores of the k-th ROI
$\mathbf{B}_k$	The binary mask of the k-th text instance
K	The number of candidate proposals

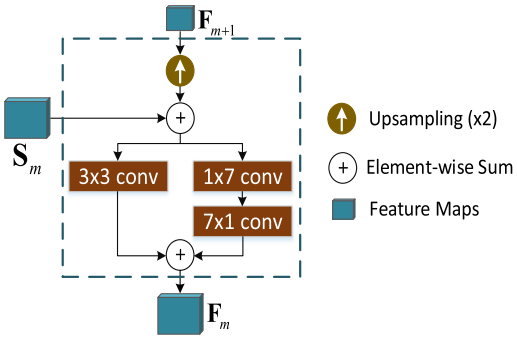

Fig. 3. Architecture of TFEU. Each convolutional layer has 256 filters. The feature map $\mathbf{F}_{m+1}$ is firstly upsampled and then fused with the side-output features $\mathbf{S}_m$. These fused features are enhanced based on the irregular and regular convolutional kernels that capture the text-related characteristics. All feature maps have D channels and their height and width depend on the size of the input image and the value of m.

image I. The output of TFEU is denoted as,

$$\mathbf{F}_m = \varphi(\mathbf{S}_m, \mathbf{F}_{m+1}; \Theta_m), \tag{2}$$

where $\varphi(\cdot)$ is the function of enhancing text features. Θ_m denotes the parameters to be learned in the m-th TFEU. The irregular 1×7 and 7×1 convolutional filters are utilized to generate wider or higher rectangular receptive fields, which are more suitable for scene text with various aspect ratios. Meanwhile, 256 convolutional filters with 3×3 kernel size are also integrated into TFEU to generate square receptive fields, which are suitable for slight-change aspect ratios of scene text.

In the TFE module, the input of the top-level TFEU is assigned with the high-level side-output features, namely, $\mathbf{F}_5 = \mathbf{S}_5$. With the guidance of the high-level semantic features, the top-down aggregation strategy makes the features $\mathbf{F}_2$, $\mathbf{F}_3$ and $\mathbf{F}_4$ combine the high-level features and the low-level features. These enhanced features at each level have stronger semantics and are more representative, compared with $\mathbf{S}_2$, $\mathbf{S}_3$ and $\mathbf{S}_4$. They are significantly helpful to distinguish the scene text from the cluttered backgrounds.

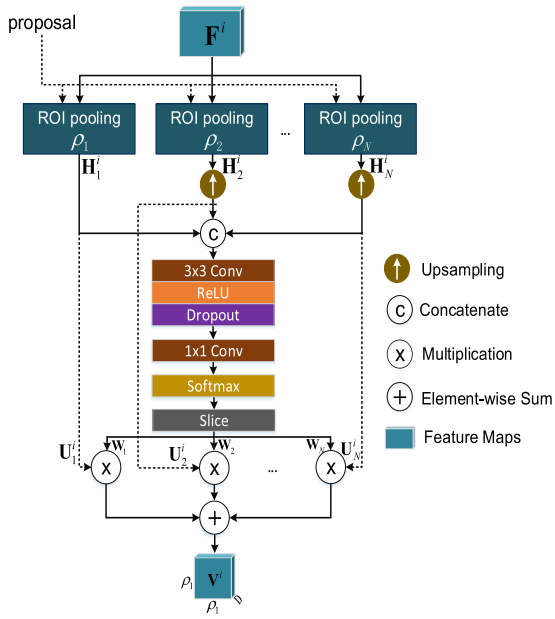

Fig. 4. Architecture of PRPA. The convolutional layer with 3×3 kernel size has the parameters 256/1/1 (filters/stride/padding), while the parameters of the convolutional layer with 1×1 kernel size are N/1/0. The dropout probability is 0.5 in the Dropout layer.

B. Pyramid ROI Pooling Attention

In this subsection, the feature assignment gate (FAG) module, which assigns the proposals generated by RPN to the corresponding enhanced feature maps based on the scale of proposals, is first formulated in detail. Then the proposed pyramid ROI pooling attention (PRPA) module is introduced to project the corresponding features of the proposals into different kinds of fixed-size features, and focus on the suitable features via the learned weights. The architecture of PRPA is shown in Fig. 4.

For the FAG module, the assignment rule follows,

$$\mathbf{F}^i = \mathbb{1}(A_{m-1}^{\tau} \le A^i \le A_m^{\tau}) \cdot \mathbf{F}_m, \quad m = 2, 3, 4, 5, \tag{3}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function. $\mathbf{F}^i$ is the corresponding features of the i-th proposal. A^i denotes the area of the i-th proposal, and A_m^{τ} is the m-th threshold of area. In experiments, we set $A_1^{\tau} = 0$, $A_2^{\tau} = 112^2$, $A_3^{\tau} = 224^2$, $A_4^{\tau} = 448^2$, and $A_5^{\tau} = \text{INF}$.

For a proposal P_i, we utilize ROIAlign [60] operation to perform ROI pooling, and the pooling features are defined as,

$$\mathbf{H}_n^i = ROIpooling\ (\mathbf{F}^i, P_i, \rho_n),\ n = 1, 2, \ldots, N, \tag{4}$$

where ρ_n denotes the size of pooling window. N indicates the number of pooling windows. In experiments, we set $N = 3$, $\rho_1 = 14$, $\rho_2 = 7$, and $\rho_3 = 3$.

Next, the pooling features are resized by carrying out the upsampling operation:

$$\mathbf{U}_n^i = Upsampling(\mathbf{H}_n^i, \lceil \rho_1/\rho_n \rceil), \tag{5}$$

where $\lceil \cdot \rceil$ denotes the ceiling operation. These upsampled features are cropped to identically spatial size and concatenated along the axis of channels. Next, the concatenated features are

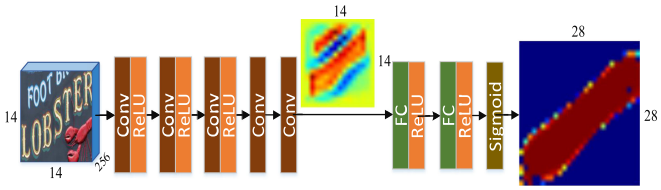

Fig. 5. Architecture of BCTS. The convolutional layers have 256 filters with 3×3 kernel size, except that the last convolutional layer has only 2 filters with 1×1 kernel size. The two fully connected layers own the hidden states of 512 and 1568, respectively. Here we show one proposal to illustrate BCTS, but it actually performs on feature maps.

fed into a convolutional layer with 3×3 kernel size, and activated by ReLU and Dropout layers. Then a convolutional layer with 1×1 kernel size is utilized to learn the weights for different kinds of pooling features, and the generated tensor $\mathbf{T} \in \mathbb{R}^{\rho_1 \times \rho_1 \times N}$ would be normalized by the $softmax$ operation as,

$$\mathbf{W}_n^j = \frac{exp(\mathbf{T}_n^j)}{\sum_{n=1}^{N} exp(\mathbf{T}_n^j)}, \tag{6}$$

where j indicates the index of the spatial positions. $\mathbf{W}_n$ denotes the attention map for the n-th kind of pooling window, and it is generated by the slice layer that divides the tensor $\mathbf{W} \in \mathbb{R}^{\rho_1 \times \rho_1 \times N}$ into N maps along the axis of channels. Each generated map has the spatial resolution of $\rho_1 \times \rho_1$. Thus, the weighted feature map $\mathbf{V} \in \mathbb{R}^{\rho_1 \times \rho_1 \times D}$ for the i-th proposal satisfies that,

$$\mathbf{V}_d^i = \sum_{n=1}^{N} \mathbf{W}_n \odot \mathbf{U}_n^d, \quad d = 1, 2, \ldots, D, \tag{7}$$

where D denotes the dimension of features, $\odot$ indicates the element-wise multiplication. In experiments, D is set to 256.

The attention map $\mathbf{W}$ not only implies the importance of features generated by different pooling windows but also reflects the effects of features at different positions. Therefore, this PRPA module can adaptively focus on suitable features.

C. Box-Aware Context-Based Text Segmentation

Since the curved text has arbitrary shapes, it is hard to precisely localize the position of the text instance only with the regression of limited points. Thus, we introduce the box-aware context-based text segmentation (BCTS) module to segment the arbitrary-shape text instance in each proposal. The architecture of BCTS is presented in Fig. 5. After the pooling features go through several convolutional layers and ReLU operations, a convolutional layer with 1×1 kernel size is utilized to generate the text/non-text mask $\mathbf{M} \in \mathbb{R}^{14 \times 14 \times 2}$. Then the mask $\hat{\mathbf{M}} \in \mathbb{R}^{28 \times 28 \times 2}$ encodes the global information of the proposal, which is formulated as,

$$\hat{\mathbf{M}} = \Omega^{-1}(\Phi(\Omega(\mathbf{M}); \Theta)), \tag{8}$$

where $\Omega(\cdot)$ denotes the vectorization operation that flattens a tensor into a vector, and Ω^{-1} is the inverse process of Ω, as reshaping a vector into a tensor. $\Phi(\cdot)$ is a nonlinear function which is simulated by two fully connected (FC) layers together with the nonlinear operator ReLU and sigmoid layer. Θ indicates the parameters to be learned in FC layers.

For the traditional fully convolutional network (FCN) in the mask branch [60], it generates the pixel-wise prediction based on the local receptive field. However, our BCTS module can take full advantage of the global information of the proposal to differentiate the dominant text region that has the largest overlap with the text instance. Moreover, it is also location-sensitive, which makes different spatial locations correlate with different parameters, compared with the shared parameters in FCN. Therefore, this location-sensitive property can highlight the dominant text region and generate more accurate boundaries of scene text. As shown in Fig. 5, the text mask generated by FCN contains parts of different text instances, some of which are not the dominant one. While the mask generated by FC layer can suppress the nondominant text regions.

D. Optimization

The standard SGD [61] algorithm is utilized to optimize our proposed network, which is trained by the alternating strategy [27]. In the first step, we train RPN via initializing the backbone network with the ImageNet-pretrained model and other network layers with the Gaussian distribution $\mathcal{N}(0, 0.01)$. In the second step, employing the proposals generated by the step-1 RPN, BRN and BCTS are trained together, which also adopt the ImageNet-pretrained model to initialize the backbone network. In the third step, we utilize the step-2 trained parameters to initialize all the layers of the backbone network and fix these layers to finetune the other layers of RPN. Finally, BRN and BCTS are finetuned by fixing the parameters shared with the step-3 RPN and utilizing the proposals generated by the step-3 RPN.

When training RPN, a multi-task learning strategy is employed, and the loss function is defined as,

$$L_{rpn} = \frac{1}{N_1} \sum_{i}^{N_1} L_{cls} + \frac{\lambda_1}{N_2} \sum_{i}^{N_2} l_i L_{reg}, \tag{9}$$

where N_1 and N_2 are the number of all the samples and positive samples in a minibatch, respectively. λ_1 is the trade-off factor between the two tasks, which is fixed to 1 in our experiments. l_i indicates the label of the i-th positive sample. L_{cls} and L_{reg} denote the loss function of classification and regression, respectively.

For one sample in a minibatch, the classification loss is calculated as,

$$L_{cls}(\mathbf{s}, l) = -log(\mathbf{s}_l), \tag{10}$$

where l is the label of the sample ($l = 0$ denotes the background while $l = 1$ denotes the text). $\mathbf{s}_l$ denotes the confidence of the sample belonging to background or text.

Meanwhile, the smooth-L_1 [27] function is adopted to formulate the regression loss as,

$$L_{reg}(\hat{\mathbf{t}}, \mathbf{t}^*, \mathbf{a}) = \sum_{q \in \mathbf{Q}} smooth_{L1}(\psi(\hat{\mathbf{t}}_q, \mathbf{a}_q) - \psi(\mathbf{t}_q^*, \mathbf{a}_q)), \tag{11}$$

where $\mathbf{Q} = \{\mathrm{x}, \mathrm{y}, \mathrm{h}, \mathrm{w}\}$ denotes the parameters of the bounding box (x and y are the center coordinates. h and w are the height and width). $\hat{\mathbf{t}}$, $\mathbf{t}^*$ and $\mathbf{a}$ are the parameter vector of the predicted box, ground truth and designed anchor, respectively. The geometry offset function $\psi(\cdot)$ is defined as,

$$\psi(\mathbf{t}_q, \mathbf{a}_q) = \begin{cases} \dfrac{\mathbf{t}_q - \mathbf{a}_q}{\mathbf{a}_\mathrm{w}} & q = \mathrm{x}, \\ \dfrac{\mathbf{t}_q - \mathbf{a}_q}{\mathbf{a}_\mathrm{h}} & q = \mathrm{y}, \\ log\left(\dfrac{\mathbf{t}_q}{\mathbf{a}_q}\right) & q \in \{\mathrm{w}, \mathrm{h}\}, \end{cases} \tag{12}$$

where $\mathbf{t} \in \{\hat{\mathbf{t}}, \mathbf{t}^*\}$. The smooth-$L_1$ function is defined as,

$$smooth_{L_1}(x) = \begin{cases} \frac{(\sigma x)^2}{2} & \text{if } |x| < \frac{1}{\sigma^2}, \\ |x| - \frac{1}{2\sigma^2} & \text{otherwise}, \end{cases} \tag{13}$$

where σ is set to 3 in the experiments.

When training BRN and BCTS, it also employs the multi-task learning manner, formulating the loss function as,

$$L = \frac{1}{N_1}\sum_i^{N_1} L_{cls}^{'} + \frac{\lambda_1}{N_2}\sum_i^{N_2} l_i L_{reg}^{'} + \frac{\lambda_2}{N_2}\sum_i^{N_2} l_i L_{seg}, \tag{14}$$

where $L_{cls}^{'}$ and $L_{reg}^{'}$ have the same definition with Eq. (10) and Eq. (11), except that here σ is set to 1 and $\mathbf{a}$ denotes the parameter vector of the proposal generated by RPN. λ_2 is a control factor of the segmentation loss L_{seg}, which is fixed to 1 in the experiments.

The cross entropy loss is employed to formulate the segmentation loss as,

$$\begin{aligned} &L_{seg}(\mathbf{M}^*, \hat{\mathbf{M}}) \\ &\quad = avg(-\mathbf{M}^* \odot log(\hat{\mathbf{M}}) - (1 - \mathbf{M}^*) \odot log(1 - \hat{\mathbf{M}})), \end{aligned} \tag{15}$$

where $\mathbf{M}^*$ and $\hat{\mathbf{M}}$ denote the ground truth and the predicted mask, respectively. $avg(\cdot)$ is the operation that calculates the mean value of all the positions in a tensor.

E. Post Processing

Since the devised network cannot produce final detection results, the post-processing is exploited to form ultimate results based on the outputs of the network. The main idea of the post-processing algorithm is to recover the actual locations of text instances and remove the densely overlapped text instances based on the mask-level NMS. We first obtain the absolute locations of the refined proposals based on the candidate proposals generated by RPN and the geometry offsets predicted by BRN. Then the refined proposals with low predicted scores are filtered. Next, we project the predicted masks of the remaining proposals to the binary maps. Finally, the mask-level NMS is employed to suppress those densely overlapped binary maps. Concretely, the network outputs four parts: the candidate proposals $\{\mathbf{R}_k\}_{k=1}^K$ generated by RPN, the confidence scores $\{\mathbf{C}_k\}_{k=1}^K$ generated by BRN, the location offsets $\{\mathbf{O}_k\}_{k=1}^K$ generated by BRN, and the text instance masks $\{\hat{\mathbf{M}}_k\}_{k=1}^K$ generated by BCTS. We first calculate the absolute locations $\{\mathbf{L}_k\}_{k=1}^K$ of the bounding boxes of text regions based on $\{\mathbf{R}_k\}_{k=1}^K$ and $\{\mathbf{O}_k\}_{k=1}^K$ via Eq. (12), which will cause dense overlaps among the text instance masks.

Algorithm 1: Post Processing

Input: $\{\mathbf{R}_k\}_{k=1}^K$, $\{\mathbf{O}_k\}_{k=1}^K$, $\{\mathbf{C}_k\}_{k=1}^K$, $\{\hat{\mathbf{M}}_k\}_{k=1}^K$, h_I, w_I, τ_c, τ_b, τ_n.
Output: $\mathcal{S}$.

```
{L_k}_{k=1}^K are generated via Eq. (12).
Y is obtained based on {C_k}_{k=1}^K and τ_c.
for i = 1 to len(Y) do
    k = Y_i.
    E_k is obtained via resizing M̂_k to the size of L_k.
    Ẽ_k is generated via binarizing E_k with τ_b.
    B_k = zeros(h_I, w_I).
    Recording the top-left coordinates of L_k as L_k^{xy}.
    for pixel in Ẽ_k do
        if Ẽ_{pixel} == 1 then
            Recording the coordinates of this pixel as p^{xy}.
            B_k[L_k^{xy} + p^{xy}] = 1.
        end if
    end for
    L̃_i = L_k, C̃_i = C_k, B̃_i = B_k.
end for
Z is obtained via MNMS based on Eq. (16), B̃ and τ_n.
S ← ∅.
for z in Z do
    L̃_z, C̃_z and B̃_z are appended into S.
end for
Return S
```

To remove the redundant text instance masks, we first filter some detected instances with low confidence scores based on the confidence threshold τ_c, and the index set of the remaining instances is denoted as $\mathbf{Y}$ with the length of K'. Then the remaining masks are resized to the same spatial resolutions with their corresponding bounding boxes, termed as $\{\mathbf{E}_k\}_{k=1}^{K'}$. These resized masks are binarized by a threshold τ_b, termed as $\{\tilde{\mathbf{E}}_k\}_{k=1}^{K'}$. Furthermore, the binary mask is projected into the full map with the same spatial resolution ($\mathrm{h}_I \times \mathrm{w}_I$) of the input image I, termed as $\{\mathbf{B}_k\}_{k=1}^{K'}$. Finally, the mask-level non-maximum suppression (MNMS) is employed to filter the overlapped text instances. The indexes of the remaining text instances are denoted as the vector $\mathbf{Z}$. In MNMS, the Intersection-over-Union (IoU) overlap is calculated as,

$$\mathbf{G}_{ij} = \frac{\sum_p (\mathbf{B}_i^p \odot \mathbf{B}_j^p)}{\sum_p \mathbf{B}_i^p + \sum_p \mathbf{B}_j^p - \sum_p (\mathbf{B}_i^p \odot \mathbf{B}_j^p)}, \tag{16}$$

where $i, j \in \{1, 2, \ldots, K'\}$. $\mathbf{G}_{ij}$ denotes the IoU overlap between the i-th and the j-th mask. p is the index of the pixel in each binary mask. The threshold in MNMS is defined as τ_n. The whole post-processing operation is summarized in Algorithm 1. The settings of these hyper-parameters are detailedly discussed in Section IV-C.

IV. Experiments

A. Implementation Details

For the datasets *CTW1500* and *totalTEXT*, in each RPN phase, we train 8 epochs with the initial learning rate of 0.004 and the learning rate multiplies 0.1 after 6 epochs. While BRN together with BCTS is trained 24 epochs at the first stage and 12 epochs at the second stage with the initial learning rate as the same as that in RPN, and the learning rate is decayed to its 0.1 after 20 epochs for the first phase. For the dataset *ICDAR-2015*, it is nearly the same with the training settings mentioned above, except that the step-2 BRN together with BCTS is trained 16 epochs. For the dataset *MLT*, we train 10 epochs, and the learning rate is decreased by 0.1 after 8 epochs in the two RPN phases. While the step-2 BRN together with BCTS is trained 24 epochs.

Additionally, our model is only trained on the training set and tested on the testing set for *CTW1500*, *totalTEXT* and *ICDAR-2015*. While for *MLT*, the training set and the validation set are merged to train the model, and the testing set is provided for evaluation [43], [48]. No extra data augmentation, except for horizontally flipping images, is employed. The batch size is fixed to 256 in RPN, while 128 in BRN and BCTS. Besides, in RPN, the base scales of anchors on feature maps ($\mathbf{F}_2$, $\mathbf{F}_3$, $\mathbf{F}_4$, $\mathbf{F}_5$ and $\mathbf{F}_6$) are set to 8×8 with three kinds of aspect ratios (0.5, 1 and 2). We also utilize 0.9 momentum and a weight decay of 0.0001 in the SGD algorithm.

The proposed network is implemented with the deep learning framework MXNet, and we conduct all experiments on a server with the Intel i7 CPU and one Nvidia Titan X GPU.

B. Datasets and Evaluation Metrics

Our proposed method is mainly evaluated on two public curved text datasets (*CTW1500* and *totalTEXT*). Additionally, to further validate the ability of our method, we also evaluate on two multi-oriented text datasets (*ICDAR-2015* and *MLT*).

CTW1500[1] consists of 1,000 images for training and 500 images for testing. There are 10,751 text instances, and each image has one curved text at least. The scenes in this dataset are challenging and of diversity, involving blur, low resolution and perspective distortion. Besides, the scene text instance is annotated by 14 key points.

totalTEXT[2] contains 1,555 images (1,255 images for training and 300 images for testing). It consists of 11,459 text bounding boxes, including 3,936 and 971 curved text instances in the training and testing set, respectively. Similarly, the text instances in this dataset are also distorted by various scenes (e.g., perspective, blur, etc.). The number of annotated clockwise points is unfixed for each text instance.

ICDAR-2015[3] contains 1,500 images in the whole dataset, in which 1,000 images are used for training and the rest serves as the testing set. Four corner points are utilized to annotate the text instance in a clockwise way, forming the quadrangular bounding box for each text instance.

MLT [4] contains 7,200 images for training, 1,800 images for validation and 9,000 images for testing. This dataset involves multi-lingual and multi-oriented scene texts. It includes 9 languages, and the annotations are quadrangles.

For evaluating the performance of detection, two evaluation criteria are employed for *CTW1500* and *totalTEXT*, including IoU and $DetEval$. For *ICDAR-2015* and *MLT*, we submit our results to the websites to evaluate them online.

IoU protocol is employed to evaluate the performance of curved scene text detection as [2], which is defined as,

$$IoU_i^j = \max_{k=1\ldots|G_i|} \frac{Area(D_i^j \bigcap G_i^k)}{Area(D_i^j \bigcup G_i^k)}, \quad j = 1\ldots|D_i|, \tag{17}$$

where D_i^j denotes the j-th detected bounding box of the i-th image, and G_i^k means the k-th ground truth of the i-th image. $|D_i|$ and $|G_i|$ are the number of detected bounding boxes and ground truth of the i-th image. $Area(\cdot)$ denotes the area. Thus the true positives (TP) of the i-th image can be formulated as,

$$TP_i = \sum_j^{|D_i|} \mathbb{1}(IoU_i^j > \tau), \tag{18}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function, and τ is a threshold (default value is 0.5) that controls the matching quality. Thus $recall$ (R), $precision$ (P) and *F-measure* (F) are calculated as,

$$recall = \frac{\sum_i^N TP_i}{\sum_i^N |G_i|}, \tag{19}$$

$$precision = \frac{\sum_i^N TP_i}{\sum_i^N |D_i|}, \tag{20}$$

$$\textit{F-measure} = 2 \times \frac{recall \times precision}{recall + precision}, \tag{21}$$

where N is the number of images in the testing set.

DetEval protocol is utilized for evaluating the performance of curved text detection as [8]. This metric could highlight the performance even when a text line contains many individual words, via considering one-to-one, one-to-many and many-to-one matching. The $recall'$ (R′) and $precision'$ (P′) can be computed as,

$$recall' = \frac{\sum_i^N \sum_k^{|G_i|} Match_G(G_i^k, D_i, \tau_r, \tau_p)}{\sum_i^N |G_i|}, \tag{22}$$

$$precision' = \frac{\sum_i^N \sum_j^{|D_i|} Match_D(D_i^j, G_i, \tau_r, \tau_p)}{\sum_i^N |D_i|}, \tag{23}$$

where τ_r and τ_p are the restraining threshold of matching, and they are fixed to 0.8 and 0.4 based on [62]. $Match_G$ and $Match_D$ denote the matching scores [62]. The calculation of *F-measure*′ (F′) is similar to Eq. (21).

[1][Online]. Available: https://github.com/Yuliang-Liu/Curve-Text-Detector
[2][Online]. Available: https://github.com/cs-chan/Total-Text-Dataset
[3][Online]. Available: http://rrc.cvc.uab.es/?ch=4
[4][Online]. Available: http://rrc.cvc.uab.es/?ch=8

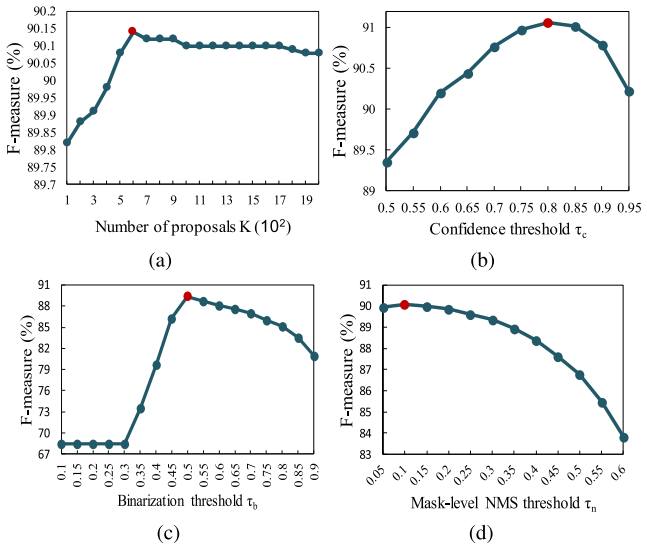

Fig. 6. The performances against different values of hyper-parameters on the training set of *CTW1500*. (a) The number of candidate proposals K. (b) The confidence threshold τ_c. (c) The binarization threshold τ_b. (d) The mask-level NMS threshold τ_n. The red dot denotes the performance under the optimal setting.

C. Hyper-Parameters Selection

In the inference stage, it involves several hyper-parameters in the post-processing Algorithm 1, i.e., the number of candidate proposals K, the confidence threshold τ_c, the binarization threshold τ_b, and the mask-level NMS threshold τ_n. In the exploration experiments (Section IV-D and Section IV-E), for a fair comparison, the same hyper-parameter settings are employed in all experiments. We empirically set K, τ_b and τ_n to 1000, 0.5 and 0.3, respectively, as the same with [60]. Besides, there are only two classes for the bounding box, and their predicted scores are handled by the *softmax* operation. Hence we empirically set τ_c to the marginal value 0.5. When we compare our model with other related methods (Section IV-F), to achieve better performances of our method, these hyper-parameters are tuned on the training set of CTW1500 to obtain the optimal settings. For the number of candidate proposals K, we exhaustively search on the range of [100, 2000] with an interval of 100. Fig. 6(a) has shown the changes of *F-measure* against the number of candidate proposals K. When K is 600, it achieves the optimal performance. Hence we set $\mathrm{K} = 600$. For the confidence threshold τ_c, we exhaustively search on the range of [0.5, 0.95] with an interval of 0.05. Fig. 6(b) has illustrated that the optimal *F-measure* is generated at the threshold of 0.8. Hence we set $\tau_c = 0.8$. For the binarization threshold τ_b, we carry out the exhaustive search on [0.1, 0.9] with the step size of 0.05. Fig. 6(c) indicates that the threshold 0.5 is the best choice. Hence we set $\tau_b = 0.5$. For the mask-level NMS threshold τ_n, we carry out the exhaustive search on [0.05, 0.6] with the step size of 0.05. Fig. 6(d) reveals that the threshold 0.1 yields the optimal performance. Hence we set $\tau_n = 0.1$.

D. Explorations of Proposed Modules

In this subsection, we conduct exploration experiments on the dataset *CTW1500* to illustrate the effectiveness of the proposed modules. We train the model on the training set of *CTW1500* where the short side of the image is set to 800 and the long side is not more than 1,000. And our model is tested on the testing set of *CTW1500*, where the short side of the image is set to 500 and the long side is not more than 700.

For the sake of simplicity, we define a few abbreviations and explain the meaning of them:

- **TFE-TD** indicates our proposed text-related feature enhancement via a top-down strategy.
- **TFE-BU** is a similar architecture with TFE-TD, expect that it adopts the bottom-up strategy that takes $\mathbf{F}_2$ as the initializing input of TFE.
- **PRPA** stands for our proposed pyramid ROI pooling attention module.
- **PRPS** is similar to PRPA, except that it utilizes the pyramid pooling features for summation instead of attention.
- **PRPC** is defined as the concatenation of pyramid ROI pooling features. The dimension of the combined features will be reduced by a convolutional layer with 1×1 kernel size and 256 filters.
- **BCTS** denotes our proposed box-aware context-based text segmentation module.
- **BLTS** stands for the box-aware local-based text segmentation, whose architecture is the same as the mask subnetwork in [60].

Actually, we also reimplement the whole framework [60] to detect the curved scene text. The performance is shown in Table II,[5] and it is obvious that our method (TFE-TD+PRPA+BCTS) achieves better performances. In the following subsection, our exploration experiments are also related with [60], since BLTS is directly from [60]. And when the module PRPA/PRPC/PRPS is not selected, it is equivalent to the single-scale ROI pooling in [60]. When the TFE module is not integrated into the proposed framework, it means that the side-output features $\mathbf{S}_m$ instead of the enhanced features $\mathbf{F}_m$ are utilized to the subsequent modules.

1) Exploration of the TFE Module: As shown in Table II, compared with the individual BCTS (row 2) module, when TFE-BU is utilized (row 5), its *F-measure* increases $\mathbf{4.73\%}$ and $\mathbf{1.39\%}$ under the evaluation protocol $IoU@0.5$[6] and $DetEval$, respectively. Contrastively, When TFE-TD is adopted, it results in a fantastic improvement of performance in terms of $IoU@0.5$ metric. For exmple, the *F-measure* has increased $\mathbf{26.24\%}$, $\mathbf{22.51\%}$ and $\mathbf{7.69\%}$, compared with BCTS, BCTS+TFE-BU and CTD-TLOC [2], respectively.

The increase of performance mostly ascribes to the enhanced text-related features. Fig. 7 illustrates these enhanced features

[5]For fairly comparing with [2], we utilize 14 sampled points from the boundary of text region when evaluating, since [2] only outputs 14 points for each text instance.

[6]$IoU@0.5$ denotes that the threshold τ of Eq. (18) is set to 0.5 in **IoU** evaluation protocol.

TABLE II
EXPLORATIONS OF THE PROPOSED MODULES ON THE DATASET CTW1500

	Protocol							IoU@0.5			$DetEval$		
	BCTS	BLTS	TFE-TD	TFE-BU	PRPA	PRPS	PRPC	R(%)	P(%)	F(%)	R′(%)	P′(%)	F′(%)
CTD-TLOC [7]								69.75	77.37	73.36	55.16	62.36	58.54
Mask-RCNN [60]								80.86	79.23	80.04	66.57	66.68	66.62
1		√						78.55	37.79	51.03	64.63	62.27	63.43
2	√							77.80	41.13	53.81	66.77	64.98	65.86
3	√				√			77.22	43.29	55.47	68.01	66.60	67.29
4		√	√		√			80.54	78.22	79.36	65.83	65.66	65.74
5	√			√				77.22	47.13	58.54	67.47	67.02	67.25
6	√		√					81.91	80.21	81.05	69.53	70.36	69.94
7	√		√			√		**83.60**	74.05	78.54	**72.98**	68.38	70.60
8	√		√				√	81.52	78.72	80.10	65.85	65.89	65.87
9	√		√		√			81.81	**82.51**	**82.16**	69.47	**72.35**	**70.88**

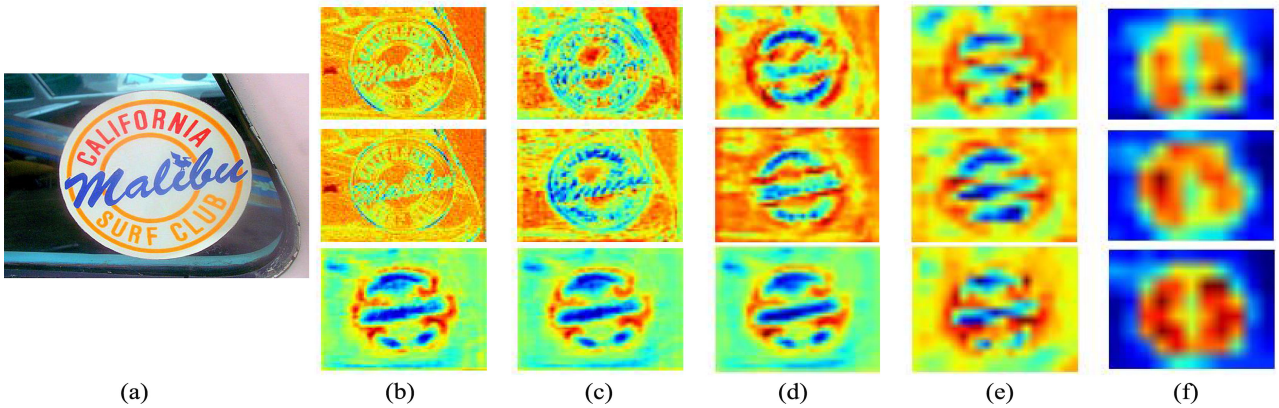

Fig. 7. Visualization of the activations. The shown activations are the sum of activations of all channels and then are normalized. (a) is the input image, and (b)∼(f) denote the activations of features $\mathbf{F}_2$, $\mathbf{F}_3$, $\mathbf{F}_4$, $\mathbf{F}_5$ and $\mathbf{F}_6$, respectively. The activations of BCTS, BCTS+TFE-BU and BCTS+TFE-TD are shown in the first, second and third row, respectively. For better visualization, the shown activation maps are resized to the same spatial dimensions.

($\mathbf{F}_2$, $\mathbf{F}_3$, $\mathbf{F}_4$, $\mathbf{F}_5$ and $\mathbf{F}_6$). TFE-TD has highlighted the line-level or word-level text boundaries in terms of shallow features (e.g., $\mathbf{F}_2$, $\mathbf{F}_3$, and $\mathbf{F}_4$) and enhanced semantic text regions according to the high-level features (e.g., $\mathbf{F}_6$).

When the TFE module is integrated into BCTS, it can remarkably reduce false positives. TFE-TD initializes the TFE module with the high-level features, which could reinforce the semantic representation of the features of shallow layers. Therefore, these features with semantic information would help distinguish text/non-text, significantly elevating the $precision$. On the contrary, TFE-BU cannot strengthen the semantic representation with the initialization of shallow features, consequently causing many false positives compared with TFE-TD. Some examples are shown in Fig. 8.

The irregular convolutional filters in the TFEU can choose different sizes. When the TFEU is without irregular convolutional filters, it means the TFEU only utilizes the 3×3 regular filters. As shown in Table III, the experimental results indicate that the filters with the size of 1×7 and 7×1 are more effective. Maybe they are beneficial to cover most of the scene text in the valid receptive fields, which can learn more discriminative representations for the scene text.

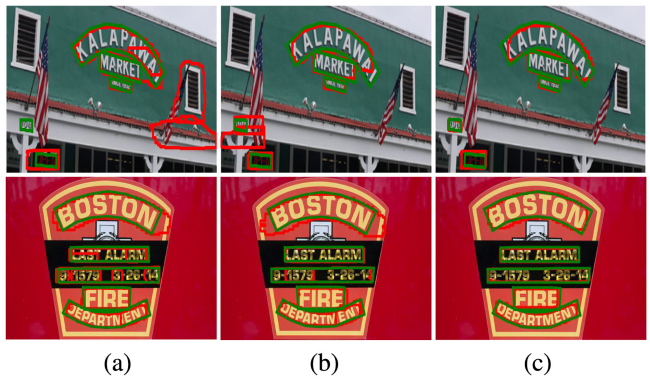

Fig. 8. Examples of comparison. The shown results are detected without TFE (a), with TFE-BU (b) and with TFE-TD (c). Rows denote the different examples. Green bounding boxes denote the ground truth, and red bounding boxes represent the detected results.

2) Exploration of the PRPA Module: According to Table II, the proposed PRPA module has positive effects on boosting the detection performance. When PRPA is integrated into BCTS (row 2), BCTS+PRPA (row 3) has elevated $\mathbf{1.66}\%$ and $\mathbf{1.43}\%$ in

TABLE III
EXPLORATIONS ABOUT THE SIZE OF IRREGULAR FILTERS IN THE TFEU. W/O DENOTES CONVOLUTION WITHOUT IRREGULAR FILTERS

filter size	R(%)	P(%)	F(%)
w/o	81.31	81.84	81.57
1×5, 5×1	81.52	82.24	81.88
1×7, 7×1	**81.81**	**82.51**	**82.16**
1×9, 9×1	81.55	81.92	81.73

(a) (b)

Fig. 9. Qualitative detection results without PRPA (a) and with PRPA (b). Rows denotes the different examples.

terms of *F-measure*, under the evaluation protocol $IoU@0.5$ and $DetEval$. Similarly, when PRPA is employed in BCTS+TFE-TD (row 6), the *F-measure* of BCTS+TFE-TD+RPRA (row 9) has increased by $\mathbf{1.11}\%$ and $\mathbf{0.94}\%$ under the evaluation protocol $IoU@0.5$ and $DetEval$.

As a matter of fact, the PRPA module primarily promotes the $precision$. The $precision$ of BCTS+PRPA has increased by $\mathbf{2.16}\%$ ($IoU@0.5$) and $\mathbf{1.62}\%$ ($DetEval$), compared with BCTS. And BCTS+TFE-TD+PRPA has improvement of $\mathbf{2.30}\%$ ($IoU@0.5$) and $\mathbf{1.99}\%$ ($DetEval$), compared with BCTS+TFE-TD. As shown in Fig. 9, when PRPA is integrated into BCTS+TFE-TD, it is capable of distinguishing some confusing backgrounds, resulting in a decrease of false positives.

Since PRPA pays attention to spatial pyramid pooling features and spatial positions, it may generate more representative features for proposals. To further explore the superiority of PRPA, we conduct experiments to compare PRPA with PRPS and PRPC. Table II has revealed that, under the evaluation protocol $IoU@0.5$, the $precision$ of PRPA has promoted $\mathbf{8.46}\%$ and $\mathbf{3.79}\%$ compared with PRPS and PRPC, respectively. Similarly, under the evaluation protocol $DetEval$, PRPA also outperforms PRPS ($\mathbf{3.97}\%$) and PRPC ($\mathbf{6.46}\%$). Some qualitative detection results are presented in Fig. 10.

3) Exploration of the BCTS Module: As shown in Table II, the proposed BCTS (row 2) outperforms BLTS (row 1) $\mathbf{2.78}\%$ and $\mathbf{2.43}\%$ in terms of *F-measure* under the evaluation protocol $IoU@0.5$ and $DetEval$. After TFE-TD and PRPA are employed, the *F-measure* of BCTS+TFE-TD+PRPA (row 9) has increased by $\mathbf{2.80}\%$ ($IoU@0.5$) and $\mathbf{5.14}\%$ ($DetEval$), compared with BLTS+TFE-TD+PRPA (row 4).

This improvement can be ascribed to the reason that BCTS has encoded the global information of each proposal, which is beneficial to determine the dominant region. BCTS suppresses nondominant regions in each proposal and devotes to making each proposal only contain one text instance, compared with BLTS. As shown in the top row of Fig. 11, when BLTS is integrated into TFE-TD+PRPA, one proposal (cyan) may contain two parts (orange).

(a) (b) (c)

Fig. 10. Examples of qualitative comparison. The shown results are detected by utilizing PRPS (a), PRPC (b) and PRPA (c). Green bounding boxes denote the ground truth. Red bounding boxes represent the detected results.

(a) (b)

Fig. 11. Examples of comparison. The shown results are obtained by adopting BLTS (a) and BCTS (b). The green bounding box is the detected box. The cyan bounding box denotes the proposal. The orange bounding box represents the segmented text parts in one proposal.

Besides, the encoding context of the text region is more discriminative for text/non-text, so that it can decrease the false positives. As shown in the bottom row of Fig. 11, it eliminates the text-like parts when using BCTS instead of BLTS. Table II also illustrates this influence of context, since the $precision$ has a noticeable improvement (e.g., $\mathbf{4.29}\%$ for $IoU@0.5$ and $\mathbf{6.69}\%$ for $DetEval$).

Moreover, the encoding context of the text region also makes BCTS generate more accurate localization. As shown in the bottom row of Fig. 11, when BLTS is adopted, it is obvious that some text regions contain excessive backgrounds, which may lead the detected text instances not to be recalled. Table II also demonstrates BCTS has elevated the $recall$ (e.g. $\mathbf{1.27}\%$ for $IoU@0.5$ and $\mathbf{3.64}\%$ for $DetEval$).

TABLE IV
EXPLORATIONS OF GENERALIZATION ABILITY

Training set	**CTW1500**											
Testing set	$CTW1500$						$totalTEXT$					
Protocol	$IoU@0.5$			$DetEval$			$IoU@0.5$			$DetEval$		
	R(%)	P(%)	F(%)	R′(%)	P′(%)	F′(%)	R(%)	P(%)	F(%)	R′(%)	P′(%)	F′(%)
CTD-TLOC [7]	69.75	77.37	73.36	55.16	62.36	58.54	25.86	42.54	32.17	34.48	44.12	38.71
Ours	81.81	82.51	82.16	69.47	72.35	70.88	31.00	55.48	39.78	47.20	66.59	55.24
Training set	**totalTEXT**											
Testing set	$CTW1500$						$totalTEXT$					
Protocol	$IoU@0.5$			$DetEval$			$IoU@0.5$			$DetEval$		
	R(%)	P(%)	F(%)	R′(%)	P′(%)	F′(%)	R(%)	P(%)	F(%)	R′(%)	P′(%)	F′(%)
CTD-TLOC [7]	50.52	37.42	43.00	42.89	44.09	43.48	64.17	72.76	68.20	46.89	54.06	50.22
Ours	57.79	35.06	43.64	65.47	62.07	63.73	70.76	77.05	73.77	70.68	78.87	74.55

E. Explorations of Generalization Ability

In this subsection, we mainly verify the generalization ability of the proposed method. We train on the training set of *CTW1500*, then test on the testing set of *CTW1500* and *totalTEXT*, respectively. Similarly, our model is trained on the training set of *totalTEXT*, and then the trained model is evaluated on the testing set of *CTW1500* and *totalTEXT*. Note that when we train or test on *totalTEXT*, the training or testing settings are the same with those in *CTW1500*.

As shown in Table IV, when we train the model on the training set of *CTW1500* and then test on the testing set of *totalTEXT*, the performance of our method still excels [2], which increases $\mathbf{7.61}\%$ and $\mathbf{16.53}\%$ in terms of *F-measure* under the evaluation protocol $IoU@0.5$ and $DetEval$. Similarly, when we train on the training set of *totalTEXT* and then test on the testing set of *CTW1500*, our method is also superior to [2]. Notably, under the evaluation protocol $DetEval$, the *F-measure* of our method has an improvement of $\mathbf{20.25}\%$.

F. Comparisons With Related Methods

1) Comparison on CTW1500: When comparing with the related methods on this curved text dataset, we utilize the multi-scale training strategy. The short side of the input image is randomly selected from the set {400, 500, 600, 700, 800, 900}, while the long side is not more than 1,000. When testing, the short side is fixed to 600, and the long side is not more than 800. Our method has achieved $\mathbf{83.3}\%$, $\mathbf{86.1}\%$, $\mathbf{84.6}\%$ in $recall$, $precision$ and *F-measure*, respectively. When we employ the multi-scale testing strategy, fixing the scales to {(400, 600), (600, 800), (1000, 1400)}, our model achieves the best *F-measure*. Note that we do not utilize the synthetic dataset SynthText [64] to pretrain our model. Quantitative evaluating results under the evaluation protocol of $IoU@0.5$ are shown in Table V. Note that the methods [25], [36], [38], [40] are reimplemented for detecting the curved scene text by [2]. Compared with the pioneering work [2], the performance of our method has a significant improvement.

The improvement mainly ascribes to four reasons. First, our method is more discriminative for false positives and false negatives, as shown in Fig. 13(a). Second, our model can generate more accurate segmentation. For example, in Fig. 13(b) and (c), some segmented text regions are over-segmentation that contains lots of backgrounds, while some are under-segmentation that is hard to determine the text boundaries. Third, our method is more robust to various curved shapes. Especially for the irregular curved-shape text, our method could obtain more suitable surroundings, as shown in Fig. 13(d). Fourth, our method is more tolerant to the perspective distortion, which can ensure that the perspective scene text is not segmented to more than one part, as shown in Fig. 13(e). In essence, the last three advantages are all potential to recall the detected text instances.

To further illustrate the excellent performance of our method, we gradually improve the IoU threshold τ of evaluation protocol,

TABLE V
COMPARISONS WITH RELATED METHODS ON THE DATAST $CTW1500$. $SynthText$ DENOTES USING SYNTHETIC DATASET TO PRETRAIN THE MODEL. * STANDS FOR MULTI-SCALE TESTING

Method	*SynthText*	Backbone	R(%)	P(%)	F(%)
Ours*	×	ResNet-50	85.1	85.7	**85.4**
Ours	×	ResNet-50	83.3	86.1	84.6
Huang *et al.* [56]	×	ResNeXt-50	83.2	**86.8**	85.0
Long *et al.* [52]	√	VGG-16	**85.3**	67.9	75.6
Wang *et al.* [51]	√	ResNet-50	84.8	79.7	82.2
Liu *et al.* [7]	×	ResNet-50	69.8	77.4	73.4
Liu *et al.* [36]	×	VGG-16	56.0	69.9	62.2
Zhou *et al.* [40]	×	VGG-16	49.1	78.7	60.4
Shi *et al.* [38]	√	VGG-16	40.0	42.3	40.8
Tian *et al.* [25]	×	VGG-16	53.8	60.4	56.9
Epshtein *et al.* [14]	×	-	9.0	20.7	12.5
Chen *et al.* [63]	×	-	4.4	6.7	5.3

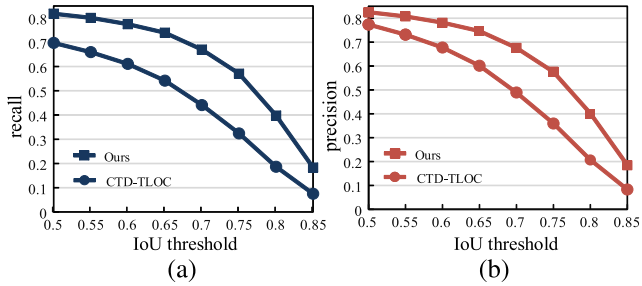

Fig. 12. Comparison of performance between our method and CTD-TLOC [2] under different IoU thresholds (τ) of evaluation protocol.

Fig. 13. Examples of comparison on the dataset CTW1500. The top row comes from CTD-TLOC [2], and the bottom row is ours. Green bounding boxes denote the ground truth, and red bounding boxes represent the detected results.

TABLE VI
COMPARISONS WITH RELATED METHODS ON THE DATASET *totalText*. $SynthText$ DENOTES USING SYNTHETIC DATASET TO PRETRAIN THE MODEL

Method	*SynthText*	Backbone	R′(%)	P′(%)	F′(%)
Ours	√	ResNet-50	**78.6**	84.6	**81.5**
Ours	×	ResNet-50	74.7	83.5	78.9
Lyu *et al.* [57]	√	ResNet-50	55.0	69.0	61.3
Dai *et al.* [55]	√	ResNet-101	78.0	**84.7**	81.3
Wang *et al.* [51]	√	ResNet-50	78.0	84.0	80.9
Long *et al.* [52]	√	VGG-16	74.5	82.7	78.4
Liu *et al.* [7]	×	ResNet-50	46.9	54.1	50.2
Chng *et al.* [8]	×	VGG-16	33.0	40.0	36.0

TABLE VII
COMPARISONS WITH RELATED METHODS ON THE DATASET $ICDAR$-2015. $SynthText$ DENOTES USING SYNTHETIC DATASET TO PRETRAIN THE MODEL. 'CUSTOMIZED' INDICATES THE SELF-DESIGNED NETWORK

Method	*SynthText*	Backbone	R(%)	P(%)	F(%)
Ours	×	ResNet-50	**82.7**	86.2	84.4
Huang *et al.* [56]	×	ResNet-50	81.8	88.2	84.9
Wang *et al.* [51]	√	ResNet-50	84.5	87.0	85.7
Long *et al.* [52]	√	VGG-16	80.4	84.9	82.6
Lyu *et al.* [57]	√	ResNet-50	81.0	91.6	**86.0**
Dai *et al.* [55]	√	ResNet-101	80.0	88.6	84.1
Lyu *et al.* [46]	√	VGG-16	70.7	**94.1**	80.7
Liao *et al.* [5]	√	VGG-16	79.0	85.6	82.2
Ma *et al.* [37]	×	VGG-16	73.2	82.2	77.4
Hu *et al.* [39]	√	VGG-16	77.0	79.3	78.2
He *et al.* [45]	×	VGG-16	73.0	80.0	77.0
He *et al.* [41]	×	Customized	80.0	82.0	81.0
Zhou *et al.* [40]	×	VGG-16	78.3	83.2	80.7
He *et al.* [47]	×	VGG-16	54.0	76.0	63.0
Liu *et al.* [36]	×	VGG-16	68.2	73.2	70.6
Shi *et al.* [38]	√	VGG-16	76.8	73.1	74.9
Zhang *et al.* [44]	×	VGG-16	43.0	71.0	54.0

ranging from 0.5 to 0.85 with an interval of 0.05. The results are presented in Fig. 12, and it is evident that our method is superior to [2] for the $precision$ and $recall$ under different IoU thresholds. Besides, the overall decrease trend of performance, as shown in Fig. 12, has also revealed that our model is more robust to the more rigid evaluation protocol compared with [2]. It indicates that our detections enclose the scene text better.

2) Comparison on totalTEXT: To compare with the related works on this curved text dataset, our model is trained with the multi-scale training strategy, where the short side is randomly from the set $\{600, 700, 800\}$, and the long side is not more than 1,000. When testing, the short side is 600, and the long side is not more than 800. Our method achieves the performances of $\mathbf{74.7}\%$, $\mathbf{83.5}\%$ and $\mathbf{78.9}\%$ in terms of $recall'$, $precision'$ and *F-measure*′, respectively. When we pretrain our model on the synthetic dataset SynthText [64] and then finetune our model on the dataset totalTEXT, it can achieve $\mathbf{78.6}\%$, $\mathbf{84.6}\%$ and $\mathbf{81.5}\%$ in $recall'$, $precision'$ and *F-measure*′, respectively. Table VI has illustrated the quantitative comparisons. Note that the model of Liu *et al.* [2] is trained by us on the same training set with the public code. Compared with [2], our method has significantly promoted the performance. Some examples of qualitative detection results shown in Fig. 14, have revealed that our approach is equipped with a more robust ability to discriminate false positives and false negatives, and produces more precise text boundaries.

3) Comparison on ICDAR-2015: To evaluate the effectiveness of our method on this multi-oriented text dataset, we train our model with the multi-scale training strategy. The short side of the input image is randomly from the set $\{600, 700, 800, 900\}$, and the long side is not more than 1,000. In the inference stage, we only utlize a single scale, where the input image is resized to the short side of 1,000 and the long side of no more than 1,400. As shown in Table VII, our proposed method achieves the results of $\mathbf{82.7}\%$, $\mathbf{86.2}\%$ and $\mathbf{84.4}\%$ in terms of $recall$, $precision$ and *F-measure*, respectively. It is obvious that the $recall$ of our approach is superior to other multi-oriented scene text detectors under challenging scenarios.

4) Comparison on MLT: For evaluating on this multi-lingual scene text dataset, our model is also trained with the multi-scale training strategy. The short side of the training image is randomly selected from the set $\{700, 800, 900, 1000\}$ and the long side is not more than 1,200. When testing, we only use a single scale to resize the input image to the short side of 1,200 and the long

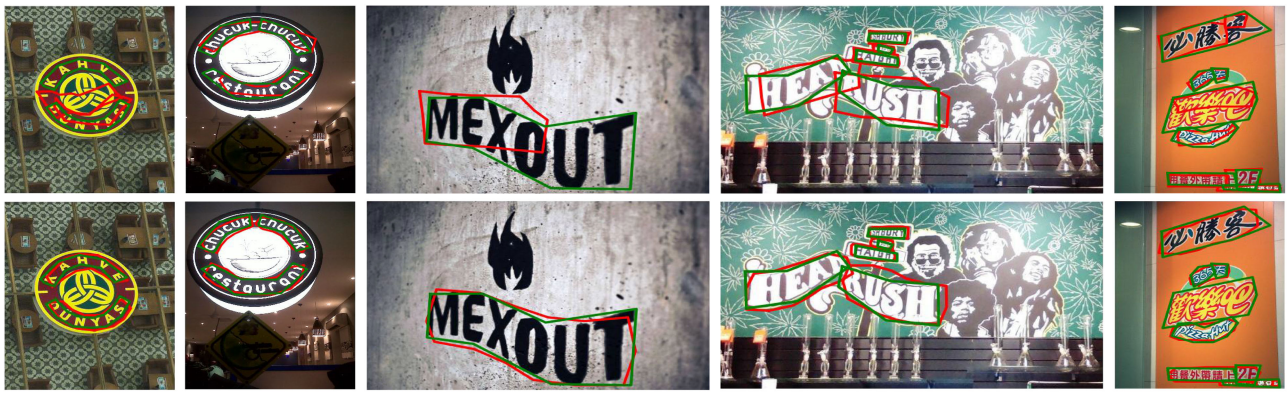

Fig. 14. Examples of comparison on the dataset totalTEXT. The top row comes from CTD-TLOC [2], and the bottom row is our results. Green bounding boxes denote the ground truth, and red bounding boxes represent the detected results.

TABLE VIII
COMPARISONS WITH RELATED METHODS ON THE DATASET *MLT*. *SynthText* DENOTES USING SYNTHETIC DATASET TO PRETRAIN THE MODEL. * STANDS FOR MULTI-SCALE TESTING

Method	*SynthText*	Backbone	R(%)	P(%)	F(%)
Ours	×	ResNet-50	66.8	79.5	72.6
Huang *et al.* [56]	×	ResNet-50	68.6	78.7	**73.3**
Wang *et al.* [51]	√	ResNet-50	69.2	75.3	72.1
Xue *et al.** [48]	×	DenseNet-121	62.1	77.7	69.0
Lyu *et al.** [46]	√	VGG-16	**70.6**	74.3	72.4
Liu *et al.** [43]	×	ResNet-50	62.3	**81.8**	70.7
Ma *et al.* [37]	×	VGG-16	55.5	71.17	62.3

TABLE IX
RUNTIMES OF THE PROPOSED METHOD. THESE RUNTIMES ARE ACQUIRED WITH A SINGLE NVIDIA TITAN X GPU

Dataset	Input scale	Network inference	Post processing
CTW1500	(600, 800)	0.83 s	0.14 s
totalTEXT	(600, 800)	1.12 s	0.18 s
ICDAR-2015	(1000, 1400)	1.81 s	0.27 s
MLT	(1200, 1500)	1.90 s	0.32 s

side of no more than 1,500. Our proposed method can achieve excellent performances of $\mathbf{66.8}\%$, $\mathbf{79.5}\%$ and $\mathbf{72.6}\%$ in $recall$, $precision$ and *F-measure*, respectively. Table VIII has shown the quantitative results. Compared with the state-of-the-art approach [56] that employs the backbone network ResNet-50, our method has a competitive ability to detect the multi-lingual scene text.

G. Runtime Analyses

To analyze the runtime of our proposed model, we first resize the input images into the fixed scales for different datasets *CTW1500*, *totalTEXT*, *ICDAR-2015* and *MLT*, as shown in Table IX. For example, we set the scale of the testing image in *CTW1500* to (600, 800), which indicates that the short side of the input image is 600 and the long side is not more than 800. Concretely, the runtime of our proposed method is composed of two main components: the network inference time and the post-processing time. We report the average time per image in each dataset based on a server with the Intel i7 CPU and one Nvidia Titan X GPU. Based on the hyper-parameter settings in Section IV-C, we set the number of candidate proposals to 600 for all datasets. According to Table IX, we observe that our method can achieve decent runtimes.

Fig. 15. Failure examples. Top examples come from *CTW1500*. Bottom examples are from *totalTEXT*. Green bounding boxes denote the ground truth. Red bounding boxes represent the detected results.

H. Limitations

According to the experimental results, our method can handle most of the challenging and diversified scenarios, but it still fails to detect scene text instances in some difficult cases. For example, our approach cannot eliminate some extremely text-like detected regions, and some low-contrast scene texts are also unsuccessful in being discovered. Another failure situation is that the large character space induces our model to detect individual characters due to the extremely ambiguous text boundaries. Some failure examples are shown in Fig. 15.

V. CONCLUSION

In this paper, we have proposed a segmentation-based framework to detect the curved text in the natural scene images. Our proposed three modules can be integrated into a unified framework to develop an end-to-end learnable task. The

text-related feature enhancement module highlights the text feature representation, which improves the ability to discriminate text/non-text. The pyramid ROI pooling attention model would pay different attention to multi-scale pooling features, which can effectively promote the precision of the model. The box-aware context-based text segmentation module utilizes the global cues of proposals, generating more accurate text boundaries. A mass of experiments are conducted on the public datasets *CTW1500*, *totalTEXT*, *ICDAR-2015* and *MLT*, which illustrate the effectiveness of our model and demonstrate the superiority of our method compared with previous methods.

REFERENCES

[1] T. He *et al.*, "Single shot textspotter with explicit alignment and attention," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 5020–5029.

[2] Y. Liu, L. Jin, S. Zhang, and S. Zhang, "Detecting curve text in the wild: New dataset and new solution," *CoRR*, vol. abs/1712.02170, 2017, [Online]. Available: http://arxiv.org/abs/1712.02170

[3] S. Karaoglu, R. Tao, T. Gevers, and A. W. M. Smeulders, "Words matter: Scene text for image classification and retrieval," *IEEE Trans. Multimedia*, vol. 19, no. 5, pp. 1063–1076, May 2017.

[4] S. Karaoglu, R. Tao, J. C. van Gemert, and T. Gevers, "Con-Text: Text detection for fine-grained object classification," *IEEE Trans. Image Process.*, vol. 26, no. 8, pp. 3965–3980, Aug. 2017.

[5] S. Zhang, Y. Liu, L. Jin, and C. Luo, "Feature enhancement network: A refined scene text detector," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2612–2619.

[6] M. Liao, Z. Zhu, B. Shi, G.-s. Xia, and X. Bai, "Rotation-sensitive regression for oriented scene text detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 5909–5918.

[7] Q. Yang *et al.*, "IncepText: A new inception-text module with deformable PSROI pooling for multi-oriented scene text detection," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 1071–1077.

[8] C. K. Ch'ng and C. S. Chan, "Total-Text: A comprehensive dataset for scene text detection and recognition," in *Proc. Int. Conf. Document Anal. Recognit.*, 2017, pp. 935–942.

[9] Q. Ye and D. S. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 7, pp. 1480–1500, Jul. 2015.

[10] Y. Zhu, C. Yao, and X. Bai, "Scene text detection and recognition: Recent advances and future trends," *Frontiers Comput. Sci.*, vol. 10, no. 1, pp. 19–36, 2016.

[11] S. M. Hanif and L. Prevost, "Text detection and localization in complex scene images using constrained AdaBoost algorithm," in *Proc. Int. Conf. Document Anal. Recognit.*, 2009, pp. 1–5.

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

[13] Y.-F. Pan, X. Hou, and C.-L. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 800–813, Mar. 2011.

[14] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2010, pp. 2963–2970.

[15] L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 3538–3545.

[16] K. L. Bouman, G. Abdollahian, M. Boutin, and E. J. Delp, "A low complexity sign detection and text localization method for mobile applications," *IEEE Trans. Multimedia*, vol. 13, no. 5, pp. 922–934, Oct. 2011.

[17] W. Huang, Z. Lin, J. Yang, and J. Wang, "Text localization in natural images using stroke feature transform and text covariance descriptors," in *Proc. Int. Conf. Comput. Vision*, 2013, pp. 1241–1248.

[18] Y. Li, W. Jia, C. Shen, and A. van den Hengel, "Characterness: An indicator of text in the wild," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1666–1677, Apr. 2014.

[19] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Deep features for text spotting," in *Proc. Eur. Conf. Comput. Vision*, 2014, pp. 512–528.

[20] Z. Zhang, W. Shen, C. Yao, and X. Bai, "Symmetry-based text line detection in natural scenes," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 2558–2567.

[21] T. He, W. Huang, Y. Qiao, and J. Yao, "Text-attentional convolutional neural network for scene text detection," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2529–2541, Jun. 2016.

[22] X. Ren *et al.*, "A convolutional neural network-based chinese text detection algorithm via text structure modeling," *IEEE Trans. Multimedia*, vol. 19, no. 3, pp. 506–518, Mar. 2017.

[23] Y. Tang and X. Wu, "Scene text detection using superpixel based stroke feature transform and deep learning based region classification," *IEEE Trans. Multimedia*, vol. 20, no. 9, pp. 2276–2288, Sep. 2018.

[24] 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 Conf. Artif. Intell.*, 2017, pp. 4161–4167.

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

[26] D. Wu, R. Wang, P. Dai, Y. Zhang, and X. Cao, "Deep strip-based network with cascade learning for scene text localization," in *Proc. Int. Conf. Document Anal. Recognit.*, 2017, pp. 826–831.

[27] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. IEEE Conf. Adv. Neural Inf. Process. Sys.*, 2015, pp. 91–99.

[28] W. Liu *et al.*, "SSD: single shot multibox detector," in *Proc. Eur. Conf. Comput. Vision*, 2016, pp. 21–37.

[29] S. Tian, S. Lu, and C. Li, "WeText: Scene text detection under weak supervision," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 1501–1509.

[30] Y. Tang and X. Wu, "Scene text detection and segmentation based on cascaded convolution neural networks," *IEEE Trans. Image Process.*, vol. 26, no. 3, pp. 1509–1520, Mar. 2017.

[31] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2012, pp. 1083–1090.

[32] L. Kang, Y. Li, and D. Doermann, "Orientation robust text line detection in natural images," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2014, pp. 4034–4041.

[33] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao, "Robust text detection in natural scene images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 970–983, May 2014.

[34] L. Wu, P. Shivakumara, T. Lu, and C. L. Tan, "A new technique for multi-oriented scene text line detection and tracking in video," *IEEE Trans. Multimedia*, vol. 17, no. 8, pp. 1137–1152, Aug. 2015.

[35] W. Huang *et al.*, "Detecting arbitrary oriented text in the wild with a visual attention model," in *Proc. ACM Multimedia Conf.*, 2016, pp. 551–555.

[36] Y. Liu and L. Jin, "Deep matching prior network: Toward tighter multi-oriented text detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3454–3461.

[37] J. Ma *et al.*, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Trans. Multimedia*, vol. 20, no. 11, pp. 3111–3122, Nov. 2018.

[38] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 3482–3490.

[39] H. Hu *et al.*, "WordSup: Exploiting word annotations for character based text detection," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 4950–4959.

[40] X. Zhou *et al.*, "EAST: An efficient and accurate scene text detector," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 2642–2651.

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

[42] M. Bušta, L. Neumann, and J. Matas, "Deep TextSpotter: An end-to-end trainable scene text localization and recognition framework," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 2223–2231.

[43] X. Liu *et al.*, "FOTS: Fast oriented text spotting with a unified network," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 5676–5685.

[44] Z. Zhang *et al.*, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 4159–4167.

[45] P. He *et al.*, "Single shot text detector with regional attention," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 3066–3074.

[46] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, "Multi-oriented scene text detection via corner localization and region segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2018, pp. 7553–7563.

[47] D. He *et al.*, "Multi-scale FCN with cascaded instance aware segmentation for arbitrary oriented word spotting in the wild," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 474–483.

[48] C. Xue, S. Lu, and F. Zhan, "Accurate scene text detection through border semantics awareness and bootstrapping," in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 370–387.

[49] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2015, pp. 3431–3440.
[50] J. Fabrizio, M. Robert-Seidowsky, S. Dubuisson, S. Calarasanu, and R. Boissel, "TextCatcher: A method to detect curved and challenging text in natural scenes," *Int. J. Document Anal. Recognit.*, vol. 19, no. 2, pp. 99–117, 2016.
[51] W. Wang *et al.*, "Shape robust text detection with progressive scale expansion network," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 9336–9345.
[52] S. Long *et al.*, "TextSnake: A flexible representation for detecting text of arbitrary shapes," in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 19–35.
[53] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vision*, 2015, pp. 1520–1528.
[54] T. Lin *et al.*, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 936–944.
[55] Y. Dai *et al.*, "Fused text segmentation networks for multi-oriented scene text detection," in *Proc. IEEE Int. Conf. Pattern Recognit.*, 2018, pp. 3604–3609.
[56] Z. Huang, Z. Zhong, L. Sun, and Q. Huo, "Mask R-CNN with pyramid attention network for scene text detection," in *Proc. IEEE Workshop Appl. Comput. Vision*, 2019, pp. 764–772.
[57] 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. Eur. Conf. Comput. Vision*, 2018, pp. 71–88.
[58] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei, "Fully convolutional instance-aware semantic segmentation," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2017, pp. 4438–4446.
[59] H. Li, P. Xiong, J. An, and L. Wang, "Pyramid attention network for semantic segmentation," in *Proc. Brit. Mach. Vision Conf.*, 2018, p. 285.
[60] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. Int. Conf. Comput. Vision*, 2017, pp. 2980–2988.
[61] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.
[62] C. Wolf and J.-M. Jolion, "Object count/area graphs for the evaluation of object detection and segmentation algorithms," *Int. J. Document Anal. Recognit.*, vol. 8, no. 4, pp. 280–296, 2006.
[63] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2004, pp. 366–373.
[64] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2315–2324.

Pengwen Dai received the B.E. degree from the College of Computer Science, Chongqing University, China, in 2014. He is currently a Ph.D. candidate with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China. His current research interests include scene text detection and recognition.

Hua Zhang received the Ph.D. degrees in computer science from the School of Computer Science and Technology, Tianjin University, Tianjin, China in 2015. He is an Associate Professor with the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include computer vision, multimedia, and machine learning.

Xiaochun Cao received the B.E. and M.E. degrees in computer science from Beihang University, China, and the Ph.D. degree in computer science from the University of Central Florida, Orlando, FL, USA. After graduation, he spent about three years with ObjectVideo, Inc. as a Research Scientist. From 2008 to 2012, he was a Professor with Tianjin University, Tianjin, China. Since 2012, he has been a Professor with the Institute of Information Engineering, Chinese Academy of Sciences. He is also with the Peng Cheng Laboratory, Cyberspace Security Research Center, China, and the School of Cyber Security, University of Chinese Academy of Sciences, China. He is on the Editorial Boards of the IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. In 2004 and 2010, he received the Piero Zamperoni Best Student Paper Award at the International Conference on Pattern Recognition.