Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 3871897, 12 pages
https://doi.org/10.1155/2020/3871897

Research Article

Texts as Lines: Text Detection with Weak Supervision

Weijia Wu,[1] Jici Xing,[2] Cheng Yang,[1] Yuxing Wang,[1] and Hong Zhou[1]

[1]*Zhejiang University, Key Laboratory for Biomedical Engineering of Ministry, Hangzhou, China*
[2]*Zhengzhou University, School of Information Engineering Institute, Zhengzhou, China*

Correspondence should be addressed to Yuxing Wang; wangyuxing@zju.edu.cn and Hong Zhou; zhouhong_zju@126.com

Received 11 February 2020; Revised 16 April 2020; Accepted 6 May 2020; Published 10 June 2020

Guest Editor: Qian Zhang

Scene text detection methods based on deep learning have recently shown remarkable improvement. Most text detection methods train deep convolutional neural networks with full masks requiring pixel accuracy for good quality training. Normally, a skilled engineer needs to drag tens of points to create a full mask for the curved text. Therefore, data labelling based on full masks is time consuming and laborious, particularly for curved texts. To reduce the labelling cost, a weakly supervised method is first proposed in this paper. Unlike the other detectors (e.g., PSENet or TextSnake) that use full masks, our method only needs coarse masks for training. More specifically, the coarse mask for one text instance is a line across the text region in our method. Compared with full mask labelling, data labelling using the proposed method could save labelling time while losing much annotation information. In this context, a network pretrained on synthetic data with full masks is used to enhance the coarse masks in a real image. Finally, the enhanced masks are fed back to train our network. Analysis of experiments performed using the model shows that the performance of our method is close to that of the fully supervised methods on ICDAR2015, CTW1500, Total-Text, and MSRA-TD5000.

1. Introduction

At present, natural scene text detection has attracted more attention due to its practical application requirements, such as scene understanding, visual question answering, autonomous driving, text detection [1], and recognition [2, 3]. Text is one of the most fundamental semantics appearing everywhere in daily life, for example, in traffic signs, commodity packages, and advertising posters. These text instances in the real world have varying sizes, random directions, and arbitrary shapes, making them extremely challenging to label and capture accurately. Unlike other general objectives, scene text usually cannot be described accurately by the axis-aligned rectangle, and most detectors using an axis-aligned rectangle only have an F-measure of below 65%, such as TextSnake [4]. Recently, most scene text detectors based on deep learning have tended to detect texts in different shapes with many coordinates for better performance. However, the above detectors require accurate pixel-level labels with expensive costs. The labelling consumes a large amount of manpower and financial resources, especially for texts with arbitrary shapes in complex environments.

The precision of text detection has a close connection with the labelling methods of datasets. For example, several common datasets, ICDAR2013 [5], ICDAR2015 [6], ICDAR2017 [7], Total-Text [8], CTW1500 [9], and MSRA-TD500 [10], have different labelling methods for various texts. ICDAR2013, as one of the common datasets, was introduced during the ICDAR Robust Reading Competition in 2013 and mainly includes horizontal bounding boxes made by two points at the word level. Because of this labelling peculiarity, text detectors [11, 12] using box regression have a great performance on ICDAR2013. ICDAR2015 was released in the ICDAR2015 Robust Reading Competition for multioriented text detection, using quadrilateral boxes as the annotations, as shown in Figure 1(b). EAST [13] and SPCNet [14], as the representatives of detectors, achieved good results on ICDAR2015. ICDAR2017 was a dataset with texts in nine languages for multilingual scene text detection, using quadrilateral boxes as the annotations as well as ICDAR2015. MSRA-TD500 was released in 2012, and the annotation method is the same as that of ICDAR2015. Unlike the above datasets, Total-Text and

Figure 1: Labelling methods. (a) Original image. (b) Bounding box labelling containing a considerable amount of background noise and interference from other text areas. (c) Visualization of segmentation annotation, requiring high labour cost. (d) Proposed annotation methods: coloured regions represent the labelled region, including the grey background and other coloured text lines.

CTW1500 contain many curved texts, which aim to solve the arbitrarily shaped text detection problem. CTW1500 has more than 10k text annotations and at least one curved text per image. Total-Text contains many curved and multi-oriented texts, which require tens of points for accurate labelling. Recently, segmentation-based text detectors [4, 15, 16] have shown promising performance in existing datasets with high-cost labelling. The annotation design becomes more complicated to fit the requirements of text detection in the real world, and the cost also increases.

The bounding box-based labelling method has low labelling costs but cannot fit text instances accurately in the wild, as shown in Figure 1(b). The pixel-based labelling method matches texts with arbitrary shapes in a complex environment but requires high labelling costs, as shown in Figure 1(c). To mitigate this conflict, we explore detecting texts at the pixel level but with a low labelling cost. Precise drawings of the text region are difficult, but using a cross-line to locate text is simple. Therefore, we seek to simplify the complex text labelling as a line named the text line in this work. Compared with the box or full masks, this annotation is extremely simple and contains less pixel information, as shown in Figure 1. Hence, the following two difficulties must be considered:

(i) A weak text line label loses the text edge information and nearly all of the background information, which is rather problematic for supervised training

(ii) The loss function focuses only on the labelled area and is not sensitive to the unlabelled ground truth

To solve the above difficulties, a scene text detector based on weakly supervised learning is proposed in our paper. The model is first pretrained on SynthText to make it sensitive to the text region. Subsequently, in the training process of real data, the pretrained model is used to enhance the text line label. In addition, to enhance the weak label better, a soft label $\in [0, 1]$ containing pixel location (distance) information is used. The contributions of this work are summarized as follows:

(i) We first propose a scene text detector based on weakly supervised learning that significantly simplifies the annotation process without losing much precision.

(ii) A modified crossentropy loss function named *degree crossentropy* is proposed. The loss function can optimize the soft label containing distance information.

2. Related Work

Scene text detection has received significant attention over the past few years, and numerous deep learning-based methods [17–21] have achieved great progress. Increasing

detectors tend to capture texts at the pixel level to detect texts more precisely.

2.1. Bounding Box-Level Text Detection Methods. Bounding box regression-based methods [19, 22] are inspired by general object detection methods such as SSD [23] and Faster R-CNN [24]. TextBoxes++ [25] further regresses to quadrangles instead of horizontal bounding boxes for multioriented text detection. RRD [26] uses rotation-invariant and sensitive features from two separate branches for better long text detection. DSRN [2] maps multiscale convolution features onto a scale invariant space and obtains uniform activation of multisize text instances for detecting texts. Although regression-based methods have achieved state-of-the-art performance, it is still difficult to capture all text information in a bounding box without involving a large proportion of background and even other text instances.

2.2. Pixel-Level Text Detection Methods. Pixel-level text detectors draw inspirations from FCN [23] and Mask R-CNN [27]. Using the mask as the annotation, PixelLink [28] performs text/nontext and links prediction at the pixel level. TextSnake [4] learns to predict local attributes, including the text centre line, text region, radius, and orientation, achieving improvements of up to 20% accuracy on curved benchmarks. CRAFT [15] trains a convolutional neural network producing the character region score and affinity score. PSENet [16] projects the feature map into several branches to produce multiple segmentation maps. TextField [29] detects scene text by predicting a direction field pointing away from the nearest text boundary to each text point. Text mountain [30] predicts text centre-border probability and text centre-direction to detect the scene text. Text detectors based on instance segmentation perform better with higher precision annotation.

2.3. Weak Supervision Semantic Segmentation. Sun et al. [31] leveraged the power of deep semantic segmentation CNNs while avoiding requiring expensive annotations for training. Rtfnet [32] took advantage of thermal images and fused both the RGB and thermal information in a novel deep neural network. Tang et al. [33] proposed a normalized cut loss for semisupervised learning; the loss combines partial cross-entropy on labelled pixels and normalized cut for unlabelled pixels. Wang et al. [1] proposed a self-supervised approach and developed a pipeline to label drivable areas and road anomalies using RGB-D images automatically.

2.4. Weak Supervision Text Detection Methods. WeText [34] trains scene text detection models on a small number of character-level annotated text images, followed by boosting the performance with a much larger number of weakly annotated images at the word/text line level. WordSup [35] trains a character detector by exploiting word annotations in rich large-scale real scene text datasets.

Recently, all detectors have been trained with fully annotated masks, requiring pixel-level accuracy for good quality prediction. Motivated by weakly supervised semantic segmentation [34, 36–38], we propose a weakly supervised scene text detector to alleviate the labelling consumption without losing high precision.

3. Method

In this section, we first introduce the overall pipeline of the proposed network. Second, the label and the procedure for enhancing the text line are described in detail. Furthermore, the designed loss function for weakly supervised learning is introduced. Finally, we list the simple postprocessing mechanism.

3.1. Overview. Figure 2 shows the overall pipeline of the proposed method, which is divided into three steps: (1) the model pretrained on a synthetic dataset [17], (2) label enhanced on a real dataset, and (3) training with the enhanced label. In the first step, the model is pretrained on a synthetic dataset with the full mask to make our model sensitive to the text region. In the second step, the pretrained model outputs an activation map of a real image as a supplement to the weakly annotated label (i.e., text line). In the final step, the enhanced label is fed back to optimize the network parameters. The output of the model in the final step forms the final prediction result through a contour search.

3.2. Labelling and Label Enhancement

3.2.1. Text Line. In this paper, we define the text line as a line across the text region, as shown in Figure 3. All characters within this text region should be connected with a continuous line (e.g., TL-1 to TL-5). There are no width and curvature requirements for these text lines. However, improper annotations such as TL-6 will result in an obvious decline in text detection accuracy. The BG in Figure 3 represents the background annotation, which has no requirements for the geometric parameters (e.g., shape, width, length, and curvature) of the line. As a result, the TL and the BG constitute the original annotation.

3.2.2. Soft Label. The soft label containing the distance (location) information is used in our method. The shortest distance between each text pixel and the background is calculated. Then, we map these distance values to [0, 1] as the soft label. For pixels concentrated in the centre of the text instance, a strong (high) value that tends to 1 should be given. However, for the estimated edge area, a weak (low) value that tends to 0 should be assigned. As shown in Figure 2 (activation map), the distance-mountain-like activation map is predicted from the model pretrained on SynthText. The shape of the soft label is the same as the distance-mountain shape. The value P_i of the label is calculated using the following equation:

$$P_i = \frac{D_i}{D_{\max}}, \tag{1}$$

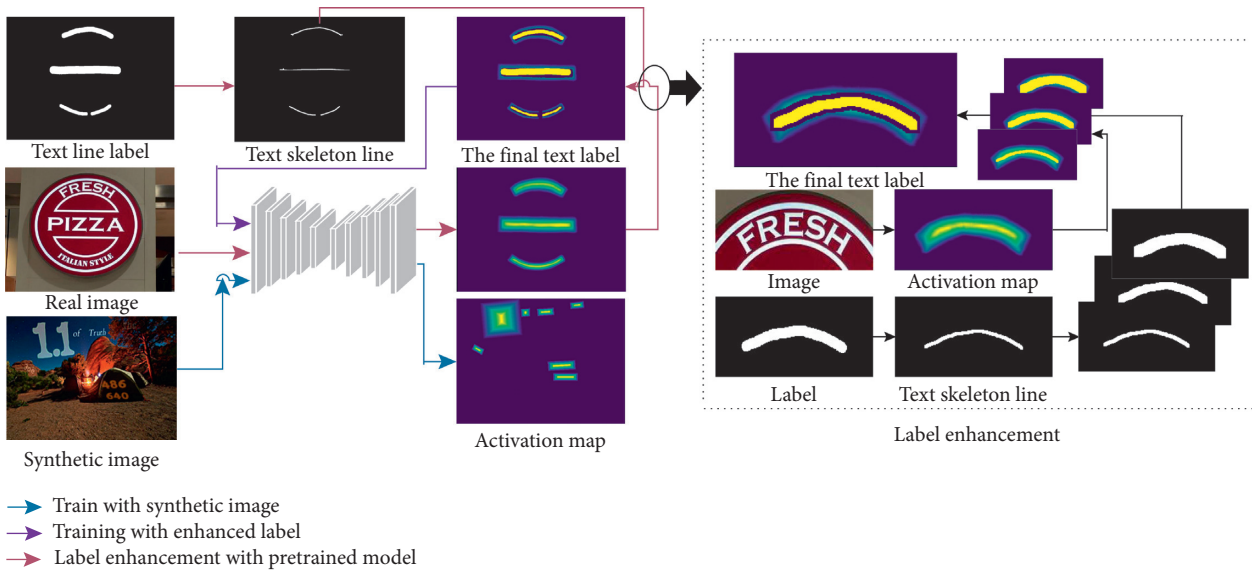

Figure 2: The overall procedure of the proposed method. (1) The network pretraining on SynthText for one epoch with full masks. (2) Enhancing the label (i.e., text line) with the pretrained model. (3) The enhanced annotation information is feedback to train the network in a weakly supervised manner.

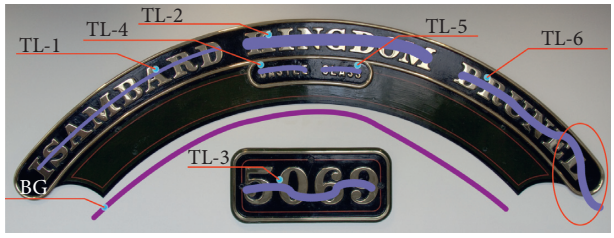

Figure 3: Text line labelling: a sample of text line labelling.

where D_i is the shortest distance between each text pixel (i) and the background pixels. $D_{\max}$ is the maximum value for all $D_i s$ in the same text instance.

3.2.3. Label Enhancement. As shown in Figure 2, label enhancement is an important step in the overall pipeline. The detailed processing of the enhancement is as follows: the network is first pretrained on SynthText for one epoch with full masks, making it sensitive to text areas. The activation maps of real images are generated using the above pretrained model. Then, we extract the text skeleton for the given weakly supervised label. Finally, the intersection of the text activation region and the text skeleton is expanded to obtain more annotation information. The only purpose of label enhancement is to use the text line to locate the correct detection text region in the activation map of the real image and to obtain more supervision information. Enhanced labels only work on the positive part (i.e., text line), while background annotations are excluded.

Figure 2 (right) describes the combination of the text skeleton and activation maps. We first use the text skeleton to locate the corresponding text activation region in the activation map and then attempt to seek the corresponding text edge region through continuous dilation of the intersection of the text activation region and the text skeleton. Detailed seeking refers to considering a pixel as the edge pixel by estimating whether the pixel value approaches 0. Finally, the values of pixels deemed as edge pixels are used as the supplement to enhance the original annotation (i.e., text line). Note that the values in the activation map are not common binary probabilities (i.e., text/nontext prediction) but represent location (distance) values. Therefore, we can use the value of each pixel in the text region to confirm the relative distance from the background.

3.3. Network Design. We chose VGG16 [39] as our feature extractor for a fair comparison with other methods. The images are first downsampled to the multilevel features with five convolution blocks, and five feature maps (*i.e.*,P_1, P_2, P_3, P_4, P_5) are generated in the step. Then, the features are gradually upsampled to the original size and mixed with the corresponding output of the previous convolution block:

$$O = U\Big(P_1 \| U_p\Big(P_2 \| U_p\Big(P_3 \| U_p\Big(P_4 \| U_p(P_5)\Big)\Big)\Big)\Big),$$
$$U_p(p) = \mathrm{Deconv}\Big(\mathrm{Conv}_{(3,3)}\Big(\mathrm{Conv}_{(1,1)}(p)\Big)\Big), \tag{2}$$

where "‖" refers to the feature concatenation and U_p is the upsample function that is used to feed the feature map into the Conv(1, 1)-Conv(3, 3)-Deconv-ReLU layers. The difference in U for U_p is obtained without the ReLU layer and reducing the channel number to 1 as the output. Finally, the output obtained through the sigmoid function is used to calculate the loss of the prediction. In addition to the VGG16, other backbones (i.e., ResNet) were also adopted in a comparative study in Section 4.6 Ablation Study.

3.4. Loss Function. The prediction is a two-dimensional feature map, and we map the value to [0, 1] using the sigmoid function. These values in a text instance are not the confidences of each pixel but represent the degrees of the shortest distance between each pixel and the background. The common binary crossentropy loss function is

$$L_{\mathrm{CE}} = -\sum\left(t \times \ln^{y} + (1-t) \times \ln^{1-y}\right), \tag{3}$$

where t is the ground truth and y is the prediction. The common crossentropy is used to evaluate the confidence of a certain category but cannot calculate the loss value with specific meanings (e.g., our distance values).

In that case, we seek to optimize the loss containing distance values by $L1$ loss: $|f(x) - Y|$ or $L2$ loss: $|f(x) - Y|^2$. However, we find that $L1$ and $L2$ are not sensitive to the distance distribution among [0, 1]. For instance, the $L1$ loss between the ground truth of 0.5 and the prediction result of 0.55 is too small and not conducive to backpropagation.

To solve the above difficulty, the *degree crossentropy* is proposed. The *degree crossentropy* can not only evaluate the confidence of category but also deal with the distance information. Losses for the positive and negative pixels are calculated according to

$$\begin{aligned} L_{\mathrm{negative}} &= L_{\mathrm{CE}}(x, y) \times (\mathrm{GT}(x, y) == 0), \\ L_{\mathrm{positive}} &= L_{\mathrm{DCE}}(x, y) \times U_{c/e} \times (\mathrm{GT}(x, y) \neq 0), \end{aligned} \tag{4}$$

where $L_{\mathrm{CE}}(x, y)$ is the traditional crossentropy loss of pixel (x, y) and $\mathrm{GT}(x, y)$ is the corresponding ground truth of pixel (x, y). Since the enhanced label may not be accurate, we treat the given label and the postenhanced supplements separately. $U_{c/e}$ is a discriminatory mechanism that calculates the losses of the original label and postenhanced part, respectively. $L_{\mathrm{DCE}}(x, y)$ is the *degree crossentropy* loss:

$$L_{\mathrm{DCE}} = -\ln\left[\left(1 - \mathrm{abs}\left(\mathrm{GT} - \mathrm{Pred}_{p}\right)\right)\right], \tag{5}$$

where Pred_{p} is the predicted result after the sigmoid function and GT is the ground truth. The loss of prediction and any goal $\in[0, 1]$ is calculated to help us to deal with distance degree information of the text. The specific implementation of $U_{c/e}$ is described by

$$\begin{aligned} U_{c/e} &= \begin{cases} 1, & P(x, y) \in \mathrm{TL}(x, y), \\ 1, & P(x, y) \in (\mathrm{DP}(x, y) \cap G), \\ 0, & \text{others}, \end{cases} \\ G &= \{\mathrm{abs}(\mathrm{GT}(x, y) - \mathrm{Pred}(x, y)) > \rho\}, \end{aligned} \tag{6}$$

where $P(x, y)$ refers to pixel (x, y) in the entire prediction map. $\mathrm{TL}(x, y)$ and $\mathrm{DP}(x, y)$ represent the annotated pixels (x, y) and postenhanced pixels (x, y), respectively. G is one set of pixels with a difference of more than ρ between the ground truth and prediction. The postenhanced annotation from the pretrained model may not be quite accurate, and noise interference may exist. Several situations are present in label enhancing. For instance, background pixels are viewed as text pixels as positive annotations. The causes are the annotation differences in the datasets and the unreliability of the prediction. To make our network learn from noisy or wrong labels, we propose a discriminatory mechanism called $U_{c/e}$, which calculates the losses of the original label and postenhanced part. In that case, the network performs strong-supervised learning on labelled pixels and distribution supervised learning on postenhanced pixels. More specifically, the predicted pixel values gradually decrease from the text centre to the edge without fitting the value of the label. The difference between the enhanced annotation and predicted results will be considered reasonable if it is smaller than ρ. The value of ρ is set to 0.1 in all the experiments. Therefore, the fault tolerance of $U_{c/e}$ can enhance the robustness of the model and avoid some mistakes from the postenhanced annotation.

3.5. Postprocessing. Most segmentation-based methods with segmentation have a common difficulty in which the separation of text instances that are close to each other is challenging. To solve this problem, we propose the apex-edge expansion algorithm that makes full use of the text-mountain shape. Given the prediction result, each text instance appears as a text mountain, as shown in Figure 4(a), where the text centre line region is the peak and the values of the pixels tend to 1. The text edge pixel areas are similar to the feet of the mountain, and their contents are mostly close to zero. Figure 4 presents a vivid example to illustrate the detailed procedure of the apex-edge expansion algorithm.

The detailed procedure of the apex-edge expansion algorithm is shown in Figures 4(b) and 4(c). The postprocessing mainly includes three parts. (1) The peak of each text mountain is selected to differentiate the different text instances. The pixel block for which the values of each inner pixel approach 1 is the peak. (2) The *dilate* in OpenCV is used to expand the peak region continuously until reaching the mountain foot or meeting other text areas. The expansion process is divided into many steps $S_1, S_2, \ldots, S_n$. S_i $(i \in [1, n])$ represents the entire expansion area in the ith step. $S_i - S_{i-1} (i \in [2, n])$ is called the extended area between two adjacent steps. The criterion of expansion ending is that the average score of the extended area approaches 0 or starts to increase. The average score approaching 0 means that the expansion area is close to the background. The increase in the score means that the expansion area begins to cover other text instances. (3) The contour of the whole text instance is represented by many coordinates as the final prediction result after the expansion. The entire postprocessing is shown in Algorithm 1, where S_n represents the prediction result, and the output D_n is the set of text instances. Dilation (.) is the *dilate* operation in OpenCV. The value and size of the expansion kernel in dilation (.) can be changed to realize different direction expansions and different scale expansions. Mean (.) is used to calculate the average value of a matrix. $\bar{}$ represents complementing the set. $\longrightarrow$ and Δ refer to tending to a number and the value increasing, respectively.

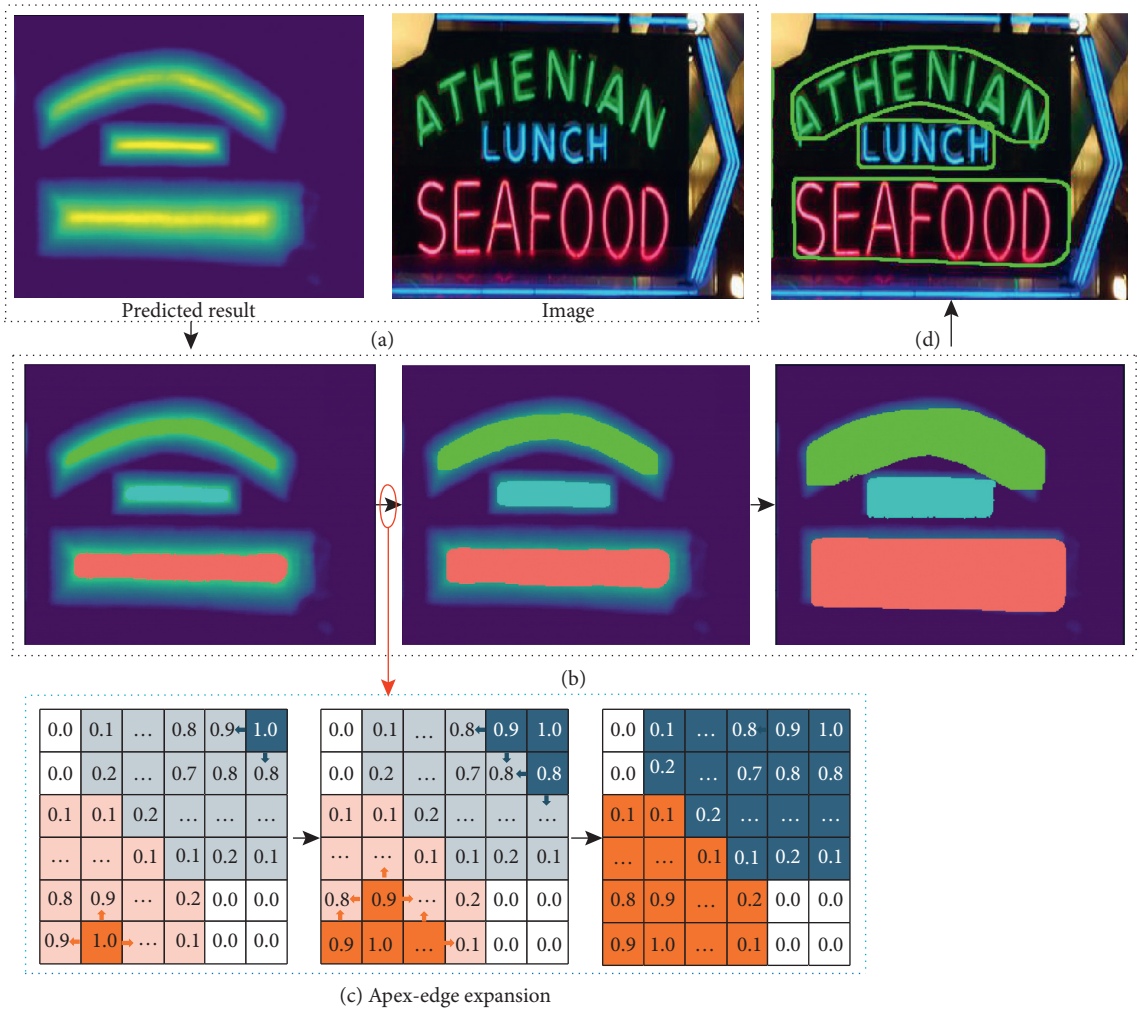

Figure 4: Apex-edge expansion: the diagram shows the whole expansion postprocessing process. (a) The original image and the predicted result. (b) Sketch of the expansion showing how to expand from the apex to the edge of the text instance. (c) A more detailed explanation in the two-dimensional decimal matrix. (d) The prediction result.

4. Experiments

In this section, we evaluate our approach using ICDAR2015, Total-Text, MSRA-TD500, and CTW1500. The experimental results demonstrate that the performance of the proposed method is comparable to those of the other methods.

4.1. Datasets. The datasets used for testing our method are briefly introduced below:

SynthText is a large-scale dataset that contains approximately 800 K synthetic images. These images were created by blending natural images with text rendered with random fonts, sizes, colours, and orientations. We used this dataset to pretrain our model.

ICDAR2015 is a multioriented text detection dataset for English text that includes only 1,000 training images and 500 testing images. The text regions were annotated by four vertices of the quadrilateral.

MSRA-TD500 contains 500 natural images. The indoor images are mainly signs, doorplates, and caution plates, while the outdoor images are mostly guided boards and billboards in complex backgrounds.

Total-Text is a world-level English curved text dataset that is split into training and testing sets with 1,255 and 300 images, respectively. The text in these images includes more than 3 different text orientations: horizontal, multioriented, and curved.

SCUT-CTW1500 contains 1,000 training images and 500 test images, which contain multioriented text, curved text, and irregularly shaped text. Text regions in this dataset are labelled with 14 scene text boundary points at the sentence level.

Data labelling to test our method: we manually marked Total-Text, CTW1500, and TD500. As shown in Figure 5, the annotation method was brief and inexpensive. For ICDAR2015, the official label was used to fit the text line label for the further verification experiment. The detailed fitting method is simple. The text skeleton as a text line is extracted directly from the full label. All annotations will be released.

Input: S_n: Segmentation result
Output: D_n: Text instances
  $D_n \Longleftarrow \phi$
  Apexs $\Longleftarrow [S_n(x, y) \longrightarrow 1]$
  for Apex $\in$ Apexs do
    if Apex > minimum then
      DK (dilated kernel) $\Longleftarrow$ Apex
      while iter < max_iter do
        next_DK $\Longleftarrow$ **Dilation** (DK)//expansion operation
        score_DK $\Longleftarrow$ **Mean** (next_DK$\cap\overline{\text{DK}}\cap S_n$)
        //get the average score of the extended region
        if score_DK $\longrightarrow$ 0 or score_DKΔ**then**
          Enqueue (D_n, next_DK)//push result into D_n
          break
        end if
        DK $\Longleftarrow$ next_DK
      end while
    end if
  end for

Algorithm 1: Apex-edge expansion algorithm.

(a) (b) (c)

Figure 5: The comparison between the text line label and original annotation for three benchmarks. The row on the top is the visualization of the original annotation on (a) CTW1500, (b) Total-Text, and (c) TD500. The middle row is text line annotation and is labelled. The bottom row is the visualization of detection results for the proposed method.

4.2. Implementation Details

4.2.1. Training. The network was pretrained on SynthText for one epoch and fine tuned on other datasets. We adopted the Adam optimizer as our learning rate scheme. During the pretraining phase, the learning rate was fixed to 0.001. During the fine-tuning stage, the learning rate was initially set to 0.0001 and decayed at a rate of 0.94 every 10,000 insertions. All of the experiments were conducted on a regular workstation (CPU: Intel (R) Core (TM) i7-7800X CPU @ 3.50 GHz; GPU: GTX 1080). The model was trained with a batch of 4 on one GPU.

VGG16 was adopted as the backbone network for the contrast experiment in our experiments. All of the experiments use the same training strategy: (1) enhancing the text annotation information with the model pretrained on SynthText and (2) training network on the target dataset. To validate the robustness of the proposed method and keep the

same condition in the comparative experiments, all of the models used in label enhancement were the same model pretrained on SynthText for one epoch.

4.2.2. Data Augmentation. The images were randomly rotated, cropped, and mirrored at a probability of 0.4. Then, colour and lightness were randomly adjusted. Finally, the images were uniformly resized to 512×512.

4.2.3. Postprocessing. We obtained all of the text instances with the apex-edge expansion and then used *findContours* in OpenCV to obtain a set of edge coordinates for each text instance. Finally, the text instances of the regular text datasets (*i.e., MSRA-TD500*) were described by four coordinate points. Methods such as *minAreaRect* in OpenCV were applied to obtain the bounding boxes of text instances. For curved text datasets, we used a set of coordinate points to describe the text instance (Tables 1 and 2).

4.3. Detecting Curve Text. The CTW1500 and Total-Text datasets were used to test the ability of curve text detection. In the experiments, manual text line annotation is used for training. The model pretrained with one epoch on SynthText had two effects: one was to heighten the annotation information, and the other was fine-tuning the pretrained model on other datasets.

The training started with the pretrained model and achieved the best result between 20 and 40 epochs. The F-measure showed a fluctuation of approximately 5%, while the threshold of the peak was in [0.5, 0.8]. For comparative experiments, the threshold of the peak in the apex-edge expansion algorithm was set to 0.6 for CTW1500 and Total-Text for comparative experiments. We continued to expand the peak region until the average score of the extended area approached 0 or met another text instance.

The F-measure of our method with text line was 77.6% on Total-Text, while the F-measure of our method with full masks was 81.1%, as shown in Table 3. The performance with full masks was close to that of the newest method. The difference (3.5%) shows that using the text line can still achieve good results on the challenging poor annotation. The recall (76.7%) was close to the values obtained for the other methods. On CTW1500, our method showed excellent results that were very close to the results obtained by the other strong-supervised methods with an F-measure of 82.3%. The difference (1.9%) between the F-measure of using the text line and that of using the full mask was also acceptable.

4.4. Detecting Long Text. TD500 contains many long text scenes and therefore is an excellent dataset for verifying the robustness of the network in long text cases. In the experiment, text line annotation was enhanced by the model pretrained on SynthText. The pretrained model was also used for fine-tuning on TD500. The threshold of the peak in the apex-edge expansion algorithm was set to 0.6, which is the same value as the experiments on CTW1500 and Total-Text. Table 4 compares the proposed method with state-of-the-art methods on TD500. The proposed method achieved an F-measure of 77.2%, which is competitive with other state-of-the-art detectors trained in a strongly supervised way.

4.5. Detecting Oriented Text. All of the parameter settings and training details for ICDAR2015 were the same as those

Table 1: Experimental results for Total-Text. "PT"refers to the model pretrained with one epoch on SynthText. "Ext." indicates external data. "FM" refers to the model trained with full mask on Total-Text. All listed results were obtained in a strongly supervised manner.

Method	Ext.	Total-Text			
		Precision	Recall	F-measure	FPS
SegLink [40]	—	30.0	23.8	26.7	—
EAST [13]	—	50.0	36.2	42.0	—
Mask TextSpotter [12]	—	69.0	55.0	61.3	—
PSENet [16]	—	81.8	75.1	78.3	3.9
[41]	√	80.9	76.2	78.5	10.0
TextSnake [4]	√	82.7	74.5	78.4	—
PSENet [16]	√	84.0	78.0	80.9	3.9
SPCNet [14]	√	83.0	**82.8**	82.9	—
CRAFT [15]	√	**87.6**	79.9	**83.6**	8.6
FM (ours)	PT	83.1	81.6	82.4	10.4
TAS (ours)	PT	78.5	76.7	77.6	**11.2**

Table 2: Experimental results for CTW1500. "PT"refers to the model pretrained with one epoch on SynthText. "Ext." indicates external data. "FM" refers to the model trained with full mask for strong-supervised learning.

Method	Ext.	CTW1500			
		Precision	Recall	F-measure	FPS
CTPN [42]	—	60.4	53.8	56.9	7.14
SegLink [40]	—	42.3	40.0	40.8	—
EAST [13]	—	78.7	49.1	60.4	—
CTD + TLOC [43]	—	77.4	69.8	73.4	—
PSENet [16]	—	80.6	75.6	78.0	3.9
TextSnake [4]	√	67.9	**85.3**	75.6	—
[41]	√	80.2	80.1	80.1	**10**
TextField [29]	√	79.8	83.0	81.4	
PSENet [16]	√	84.8	79.7	82.2	3.9
CRAFT [15]	√	**88.2**	78.2	**82.9**	8.6
FM (ours)	PT	86.2	80.5	83.2	9.0
TAS (ours)	PT	83.8	80.8	82.3	9.2

Table 3: Effectiveness of label enhancement and the pretrained model on Total-Text.

Method	Pretrain-enhancement	Label	F-measure
Baseline + TL	No	TL	65.0
Baseline + FL	No	Full	81.1
Baseline + TL∗ + PT	SynthText	TL*	77.6
Baseline + TL + PTΔ	CTW1500	TL	79.1
Baseline + TL + PT	SynthText	TL	77.2

FL and TL refer to using full mask and text line label, respectively. TL∗ is the synthetic text line label. PT and PTΔ are the different models pretrained on SynthText and CTW1500, respectively.

Table 4: Experimental results for SCUT-TD500. "PT"refers to the model pretrained on SynthText. "Ext." indicates external data. All compared methods were trained in a strongly supervised way. "FM" refers to the model trained with a full mask for strong-supervised learning.

Method	Ext.	MSRA-TD500			
		Precision	Recall	F-measure	FPS
EAST [13]	—	87.3	67.4	76.1	**13.2**
RRPN [20]	—	82.0	68.0	74.0	—
DeepReg [18]	—	77.0	70.0	74.0	1.1
SegLink [40]	√	86.0	70.0	77.0	8.9
PixelLink [28]	√	83.0	73.2	77.8	3.0
RRD [26]	√	87.0	73.0	79.0	10
TextSnak [4]	√	83.2	73.9	78.3	1.1
[44]	√	87.6	76.2	82.9	—
TextField [29]	√	75.9	87.4	81.3	—
CRAFT [15]	√	**88.2**	78.2	82.9	8.6
MCN [45]	√	79.0	**88.0**	**83.0**	—
FM (ours)	PT	83.2	76.6	79.8	11.6
TAS (ours)	PT	80.6	74.1	77.2	12.0

for the experiments on the curve text datasets. The official label was used to fit the text line label for the further verified experiment on ICDAR2015. Similar to the experiment on TD500, *minAreaRect* in OpenCV was used to obtain the bounding boxes of the text instance, in contrast to several detectors listed in Table 5 that used extra datasets. For instance, the F-measure of PSENet [16] was 80.5% without an extra dataset. The F-measure (79.4%) of our method was already comparatively close to those of the other methods.

4.6. Ablation Study. Three groups of comparative experiments were performed to verify the effectiveness of our method.

4.6.1. Baseline. The baseline was trained with the text line without label enhancement, and the F-measure of the baseline on Total-Text was 65.0%, as shown in Table 3.

4.6.2. Label Enhancement. The results are shown in Table 3, which are further analysed for label enhancement of the model on Total-Text. Training with an unenhanced text line shows an unsatisfactory performance (65.0%), while training with a full mask obtained an F-measure of 81.1%. The large difference (16.1%) indicates that the text line loses important supervision information. After introducing the pretrained model on SynthText to enhance the text line, the performance of the model had an obvious improvement from 65.0% to 77.2%. In addition, using the synthetic text line from the full mask shows better performance (77.6%). The main reason for this is that the manual text line had a larger error in extracting the text skeleton compared to the synthetic text line. In addition, we also compared the performance of the model pretrained on different datasets: synthetic data (i.e., SynthText) and realistic data (i.e., SUCT-CTW1500). The F-measures using SynthText and CTW1500 were 77.2% and 79.1%, respectively. Obviously, the performance of our model pretrained with realistic data shows a few advantages. This also indicates an intrinsic limitation of this method and the dependence on the pretrained model.

Table 5: Experimental results for ICDAR2015. "PT"refers to the model pretrained on SynthText for one epoch. "Ext." indicates using external data.

Method	Ext.	ICDAR15			
		Precision	Recall	F-measure	FPS
CTPN [42]	—	74.2	51.6	60.9	7.1
EAST [13]	—	83.6	73.5	78.2	6.5
RRPN [20]	—	82.0	73.0	77.0	—
PSENet [16]	—	81.4	79.6	80.5	1.6
PixelLink [28]	—	82.9	81.7	82.3	7.3
SegLink [40]	√	73.1	76.8	75.0	—
SSTD [46]	√	80.2	73.9	76.9	7.7
WordSup [35]	√	79.3	77.0	78.2	—
RRD [26]	√	85.6	79.0	82.2	6.5
MCN [45]	√	72.0	80.0	76.0	—
TextField [29]	√	80.5	84.3	82.4	—
TextSnake [4]	√	84.9	80.4	82.6	1.1
PAN [47]	√	84.0	81.9	82.9	**26.1**
PSENet [16]	√	86.9	84.5	85.7	1.6
CRAFT [15]	√	**89.8**	84.3	86.9	8.6
SPCNet [14]	√	88.7	**85.8**	**87.2**	—
TAS (ours)	PT	81.7	77.1	79.4	8.5

4.6.3. Geometric Parameters of the Text Line. As shown in Table 6, the impact of the width and the offset of the text line was evaluated. For the width of the text line, we used different widths of synthetic or manually marked text lines to test our model. For the manually marked text line, we extracted its skeleton of one-pixel width and dilated the skeleton to different widths while the width was less than that of the original text line. For the synthetic text line, the skeleton of one pixel was extracted from the full mask and used to create different widths. While the width of the text line was the same, using the synthetic text line which usually achieved a better performance than using the manual text line, and the average difference was approximately 0.4%. In addition, with increasing width, the F-measure showed a fluctuation of approximately 1%. The offset of the text line was set to 0 in all experiments to evaluate the influence of the text line width.

Apart from the evaluation of the influence of width, the offset between the synthetic text line and centre line of the text instance was also set to test our detection method. The offset in Table 6 refers to the offset error ratio: D_o/D_t. D_o is the distance between the text line and text centre line, and D_t is the width of the text region. In the experiment, we only performed the experiment on the synthetic text line, while the offset between the manual text line and text centre line was difficult to calculate. The text centre line was calculated from the original coordinate annotation, and then we created the text line by setting the corresponding offset ratio. The curvature and width of the created text line were the same as those of the text centre line. All widths of the text

Table 6: Effectiveness of different text line labelling qualities on Total-Text.

Width (pixel)	Method	F-measure	Method	F-measure
1	Baseline + TL	77.2	Baseline + TL∗	77.6
3	Baseline + TL	77.8	Baseline + TL∗	77.8
5	Baseline + TL	77.5	Baseline + TL∗	78.4
7	Baseline + TL	78.1	Baseline + TL∗	78.3
Method	Width (pixel)	Offset (%)	F-measure	—
Baseline + TL*	1	0	77.6	—
Baseline + TL*	1	10	77.4	—
Baseline + TL*	1	20	77.8	—
Baseline + TL*	1	30	76.4	—
Baseline + TL*	1	40	75.2	—

TL* is the synthetic text line with a full mask. TL refers to the manually marked text line.

Table 7: Detection results with different backbones on CTW1500.

Backbone	Method	Precision	Recall	F-measure	FPS
VGG16	Baseline + TL	83.8	80.8	82.3	9.2
VGG11	Baseline + TL	82.5	81.2	81.9	11.2
ResNet-18	Baseline + TL	82.4	79.4	80.9	8.7
ResNet-50	Baseline + TL	84.5	80.8	82.7	6.9
ResNet-101	Baseline + TL	84.3	81.6	82.9	6.6

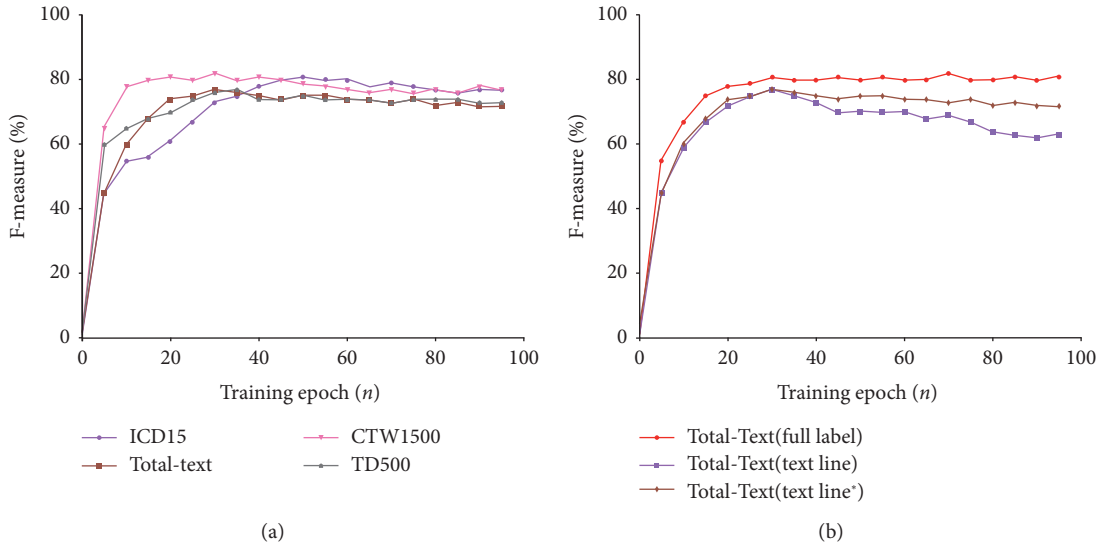

Figure 6: Influence of training epochs: (a) results for different datasets with the text line and (b) results for the same dataset with different annotations. ∗ represents $U_{c/e}$ used in the loss function.

line or text centre line were one pixel in the experiment. While the offset ratio of the text line was below 20%, the F-measure barely fluctuated. While the offset ratio of the text line exceeded 20%, the performance of the model started to be affected slightly, but the fluctuation around 2% was still acceptable.

4.6.4. Backbone. As shown in Table 7, a series of experiments comparing different backbones were performed to evaluate its influence on the proposed method. Similar to VGG16, five feature maps generated from VGG11 were gradually upsampled to the original size. For the ResNet series, four feature maps were used to merge. The F-measure using VGG11 was similar to that of using VGG16, but the latter had a slightly slower inference time. Due to the sophisticated design, the ResNet series had a longer convergence time, but the performance was comparatively accurate and stable.

4.6.5. Loss Function. As shown in Figure 6(a), due to the instability of the enhanced annotation, the F-measure decreased after dozens of epochs on four common datasets, particularly for curved text datasets. As shown in Figure 6(b), training with the text line was unstable relative

to the method with full labelling, and the model with full labelling showed better convergence performance with an increasing number of training epochs. After incorporating $U_{c/e}$ into the loss function, the model with the text line showed improved convergence, with convergence fluctuation of approximately 3%.

5. Conclusion and Future Work

In this paper, we first introduced a novel text detector based on weakly supervised learning. The most prominent feature of the method was proposing a novel labelling named the text line and the full use of the model pretrained on SynthText. The use of a text line can help the detector decrease the cost of labelling, and the pretrained model can improve the performance of the detector. The experiments showed that the text line with low-cost labelling can be used to train an effective text detector and further verify the feasibility of using a synthetic text dataset to enhance weak labels. Efficient low-cost text detectors have potential applications in the field of photo translation. Synthetic data will play an increasingly important role in the field of deep learning in the future. One reason for this is that the high cost of annotation hinders the application of actual scenes for arithmetic. Another reason is that synthetic data are increasingly similar to real-world images, and the development of auxiliary methods promotes the development of synthetic text. In future work, it will be important to train the methods with synthetic data but apply them to the real world.

Data Availability

The data are now made public at https://github.com/xingjici/Texts-as-Lines-Text-Detection-with-Weak-Supervision and the corresponding code is still cleaning up. Data description can be found in Abstraction sector.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Weijia Wu and Jici Xing contributed equally to this work.

Acknowledgments

This work was supported by the National Key Research and Development Project (Grant no. 2019YFC0118202), National Natural Science Foundation of China (Grant nos. 61803332 and 11574269), Natural Science Foundation of Zhejiang Provence (Grant no. LQ18E050001), Fundamental Research Funds for the Central Universities (Grant no. 2019FZJD005), and Scientific Research Fund of Zhejiang Provincial Education Department (Grant no. Y201941642).

References

[1] H. Wang, Y. Sun, and M. Liu, "Self-supervised drivable area and road anomaly segmentation using RGB-D data for robotic wheelchairs," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4386–4393, 2019.

[2] Y. Wang, H. Xie, Z. Fu, and Y. Zhang, "DSRN: a deep scale relationship network for scene text detection," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 947–953, AAAI Press, Macao, China, August 2019.

[3] H. Xie, S. Fang, Z.-J. Zha, Y. Yang, Y. Li, and Y. Zhang, "Convolutional attention networks for scene text recognition," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 15, no. 1, pp. 1–17, 2019.

[4] S. Long, J. Ruan, W. Zhang et al., "TextSnake: a flexible representation for detecting text of arbitrary shapes," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 20–36, Munich, Germany, September 2018.

[5] D. Karatzas, F. Shafait, S. Uchida et al., "ICDAR 2013 robust reading competition," in *Proceedings of the 2013 12th International Conference on Document Analysis and Recognition*, pp. 1484–1493, IEEE, Washington, DC, USA, August 2013.

[6] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou et al., "ICDAR 2015 competition on robust reading," in *Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1156–1160, IEEE, Tunis, Tunisia, August 2015.

[7] N. Nayef, F. Yin, B. Imen et al., "ICDAR 2017 robust reading challenge on multi-lingual scene text detection and script identification-RRC-MLT," in *Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 1454–1459, IEEE, Kyoto, Japan, November 2017.

[8] C. K. Ch'ng and C. S. Chan, "Total-text: a comprehensive dataset for scene text detection and recognition," in *Proceedings of the 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 935–942, IEEE, Kyoto, Japan, November 2017.

[9] Y. Liu, L. Jin, S. Zhang, C. Luo, and S. Zhang, "Curved scene text detection via transverse and longitudinal sequence connection," *Pattern Recognition*, vol. 90, pp. 337–345, 2019.

[10] C. Yao, B. Xiang, W. Liu, Yi Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proceedigs of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1083–1090, IEEE, Providence, RI, USA, June 2012.

[11] He Tong, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, "An end-to-end textspotter with explicit alignment and attention," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5020–5029, Salt Lake City, UT, USA, June 2018.

[12] P. Lyu, M. Liao, C. Yao, W. Wu, and B. Xiang, "Mask textspotter: an end-to-end trainable neural network for spotting text with arbitrary shapes," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 67–83, Munich, Germany, September 2018.

[13] X. Zhou, C. Yao, W. He et al., "East: an efficient and accurate scene text detector," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5551–5560, Honolulu, HI, USA, July 2017.

[14] E. Xie, Y. Zang, S. Shao, G. Yu, C. Yao, and G. Li, "Scene text detection with supervised pyramid context network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 9038–9045, 2019.

[15] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9365–9374, Long Beach, CA, USA, June 2019.

[16] Li Xiang, W. Wang, W. Hou, R.-Ze Liu, T. Lu, and J. Yang, "Shape robust text detection with progressive scale expansion network," 2018, https://arxiv.org/abs/1903.12473.

[17] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2315–2324, Las Vegas, NV, USA, June 2016.

[18] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep direct regression for multi-oriented scene text detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 745–753, Venice, Italy, October 2017.

[19] M. Liao, B. Shi, and X. Bai, "Textboxes++: a single-shot oriented scene text detector," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 3676–3690, 2018.

[20] J. Ma, W. Shao, H. Ye et al., "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 3111–3122, 2018.

[21] S. Zhang, Y. Liu, L. Jin, and C. Luo, "Feature enhancement network: a refined scene text detector," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.

[22] Y. Jiang, X. Zhu, X. Wang et al., "R2CNN: rotational region cnn for orientation robust scene text detection," 2017, https://arxiv.org/abs/1706.09579.

[23] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, Boston, MA, USA, June 2015.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, pp. 91–99, Neural Information Processing Systems Foundation Inc. La Jolla, CA, USA, 2015.

[25] M. Liao, B. Shi, B. Xiang, X. Wang, and W. Liu, "Textboxes: a fast text detector with a single deep neural network," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, CA, USA, February 2017.

[26] M. Liao, Z. Zhu, B. Shi, G.-s. Xia, and B. Xiang, "Rotation-sensitive regression for oriented scene text detection," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5909–5918, Salt Lake City, UT, USA, June 2018.

[27] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2961–2969, Venice, Italy, October 2017.

[28] D. Deng, H. Liu, X. Li, and C. Deng, "Pixellink: detecting scene text via instance segmentation," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.

[29] 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 Transactions on Image Processing*, vol. 28, no. 11, pp. 5566–5579, 2019.

[30] Y. Zhu and J. Du, "Textmountain: accurate scene text detection via instance segmentation," 2018, https://arxiv.org/abs/1811.12786.

[31] T. Sun, Y. Sun, M. Liu, and D.-Y. Yeung, "Movable-object-aware visual slam via weakly supervised semantic segmentation," 2019, https://arxiv.org/abs/1906.03629.

[32] Y. Sun, W. Zuo, and M. Liu, "RTFNet: RGB-thermal fusion network for semantic segmentation of urban scenes," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2576–2583, 2019.

[33] M. Tang, A. Djelouah, F. Perazzi, Y. Boykov, and C. Schroers, "Normalized cut loss for weakly-supervised CNN segmentation," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1818–1827, Salt Lake City, UT, USA, June 2018.

[34] S. Tian, S. Lu, and C. Li, "WeText: scene text detection under weak supervision," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1492–1500, Venice, Italy, October 2017.

[35] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding, "Wordsup: exploiting word annotations for character based text detection," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4940–4949, Venice, Italy, October 2017.

[36] Q. Li, A. Arnab, and P. H. S. Torr, "Weakly-and semi-supervised panoptic segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 102–118, Munich, Germany, September 2018.

[37] Di Lin, J. Dai, J. Jia, K. He, and J. Sun, "ScribbleSup: scribble-supervised convolutional networks for semantic segmentation," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3159–3167, Las Vegas, NV, USA, June 2016.

[38] J. Xu, A. G. Schwing, and R. Urtasun, "Learning to segment under various forms of weak supervision," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3781–3790, Boston, MA, USA, June 2015.

[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[40] B. Shi, B. Xiang, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2550–2558, Honolulu, HI, USA, July 2017.

[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 *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6449–6458, Long Beach, CA, USA, June 2019.

[42] Z. Tian, W. Huang, He Tong, He Pan, and Yu Qiao, "Detecting text in natural image with connectionist text proposal network," in *European Conference on Computer Vision*, pp. 56–72, Springer, Berlin, Germany, 2016.

[43] L. Yuliang, J. Lianwen, Z. Shuaitao, and Z. Sheng, "Detecting curve text in the wild: new dataset and new solution," 2017, https://arxiv.org/abs/1712.02170.

[44] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, "Multi-oriented scene text detection via corner localization and region segmentation," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7553–7563, Salt Lake City, UT, USA, June 2018.

[45] Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and L. G. Wang, "Learning markov clustering networks for scene text detection," 2018, https://arxiv.org/abs/1805.08365.

[46] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, and X. Li, "Single shot text detector with regional attention," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3047–3055, Venice, Italy, October 2017.

[47] W. Wang, E. Xie, X. Song et al., "Efficient and accurate arbitrary-shaped text detection with pixel aggregation network," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8440–8449, Seoul, South Korea, November 2019.