Pattern Recognition 110 (2021) 107336

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

TextMountain: Accurate scene text detection via instance segmentation

Yixing Zhu, Jun Du*

National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, Anhui, China

ARTICLE INFO

Article history:
Received 28 March 2019
Revised 9 December 2019
Accepted 12 March 2020
Available online 11 May 2020

Keywords:
Scene text detection
Curved text
Multi-oriented text
CNN
Deep learning

ABSTRACT

In this paper, we propose a novel scene text detection method named TextMountain. The key idea of TextMountain is making full use of border-center information. Different from previous works that treat center-border as a binary classification problem, we predict text center-border probability (TCBP) and text center-direction (TCD). The TCBP is just like a mountain whose top is text center and foot is text border. The mountaintop can separate text instances which cannot be easily achieved using semantic segmentation map and its rising direction can plan a road to top for each pixel on mountain foot at the group stage. The TCD helps TCBP learning better. Our label rules will not lead to the ambiguous problem with the transformation of angle, so the proposed method is robust to multi-oriented text and can also handle well curved text. In inference stage, each pixel at the mountain foot needs to search the path to the mountaintop and this process can be efficiently completed in parallel, yielding the efficiency of our method compared with others. The experiments on MLT, ICDAR2015, RCTW-17 and SCUT-CTW1500 datasets demonstrate that the proposed method achieves better or comparable performance in terms of both accuracy and efficiency. It is worth mentioning our method achieves an F-measure of 76.85% on MLT which outperforms the previous methods by a large margin. Code will be made available.

1. Introduction

Recently, scene text detection has become increasingly popular, many researchers pay more attention to it due to its wide applications such as image and video retrieval, automatic driving and scene text translation. Although the academia has been studying for many years, scene text detection is still challenging because of great varieties in shape, size, angle and complex backgrounds.

Most of Traditional scene text detection methods analyze text based on character-based structural features. In recent years, benefitting from deep learning, many methods which adopt deep convolutional neural network (CNN) [1] to extract features and achieve considerable improvement.

In this study, we aim to design a concise method which can detect long text lines without the restriction of receptive field and avoid the ambiguity with the transformation of angle. Besides these we also attempt to accelerate the post-processing step to make our method more efficient. To achieve these goals, our approach follows two principles. First, we do not force the network to treat the text line as a whole object. Second, we will not define the order of vertices and sides for a polygon which may lead to ambiguity. In particular, we utilize a fully convolutional network (FCN) [2] model to generate segmentation maps for predicting maps of text score (TS), text center-border probability (TCBP) and text center-direction (TCD). As shown in Fig. 1, we predict a TCBP where the values for the border/center pixels is 0/1 and the values of other pixels gradually decay from center to border, the rising direction of TCBP can be used to group pixels. This is different from previous methods to treat center-border as a binary classification problem. Besides these, we also predict TCD which is not necessary in inference but can make probability map learn better. Finally, benefitting from the parallel computation of our GPU implementation, the group process consumes very little time. The main contribution of the paper is that we propose a novel pipeline to group instance pixels which can handle well curved and long text. To the best of our knowledge, it is the first time to group text lines by using TCBP's grads, which can be calculated with TCD or TCBP. The proposed TCD yields a better model learning, and TCBP is smoother than TCB binary. In comparison, He et al. [3], He et al. [4] and Wu and Natarajan [5] do not learn extra information to group text lines, He et al. [3] and Wu and Natarajan [5] directly expands outputting quadrangle with ratio of shrinkage. Our experiments show that using TCBP's grads to group text achieves better performance compared with expanding with ratio of shrinkage and expanding cannot handle curved text line. He et al. [4] firstly

* Corresponding author.
E-mail addresses: zyxsa@mail.ustc.edu.cn (Y. Zhu), jundu@ustc.edu.cn (J. Du).

https://doi.org/10.1016/j.patcog.2020.107336
0031-3203/

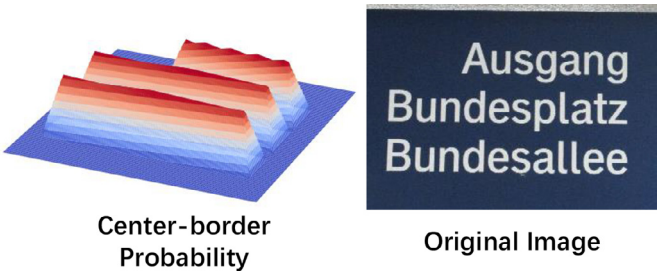


Fig. 1. TextMountain: The mountaintop of TCBP can separate text lines and the mountain rising direction can plan a road to mountaintop for each pixel on mountain foot at the group stage.

predicts text center segmentation map, then uses text center segmentation to predict each instance segmentation map one by one which is inefficient when the number of text lines is large. Deng et al. [6] only sets the outermost layer pixels to border which is not quite reasonable. Because when the outputting of PixelLink shifts one pixel, it is totally wrong on the loss of PixelLink. But our center-border probability is smooth, accordingly outputting shifts one pixel only causes a small increase in the loss function. As the offset increases, IoU will gradually decrease and our loss will gradually increase. This process is continuous and smooth. Loss of TextMountain and IoU are negatively correlated. Compared with other segmentation methods, the proposed method achieves better or comparable performance in terms of both accuracy and efficiency. The contributions of this paper are four-fold:

- We propose a novel method named TextMountain which is composed of TS, TCBP and TCD. Our experiments verify that using TCBP's grad can greatly improve grouping text lines and the process can be efficiently implemented in parallel without Non-Maximum Suppression (NMS) while TCD can help TCBP learning better.
- The proposed method can well and efficiently handle long, multi-oriented and curved scene text.
- The proposed method achieves state-of-the-art or comparable results in both quadrangle and curved polygon labeled datasets.

2. Related work

Traditional text detection methods mainly use extremal region, border information or character's morphological information to locate text such as Stroke Width Transform (SWT) [7] and Maximally Stable Extremal Regions (MSER) [8]. Shivakumara et al. [9] uses HOG features with an SVM classifier for scene text detection. Sun et al. [10] proposes a text detector based on neural networks and color-enhanced contrasting extremal region (CER). With the emergence of deep learning, many methods try using deep neural nets to solve this problem and greatly exceed traditional methods on both performance and robustness. Deep learning based methods can be mainly divided into two categories. One is regression based method which generates text bounding boxes by regressing the coordinates of boxes. The other one is segmentation based method which predicts the segmentation map of text line and the key part is how to split adjacent text lines.

Regression based method: Many text detection methods design their systems based on object detection methods and make corresponding improvements to the particularity of text. Liao et al. [11] improves [12] by adjusting the aspect ratios of anchor and then setting them with vertical offsets, and adopts irregular 1×5 convolutional filters for long text lines. Different from traditional horizontal rectangle labeled object detection method, Ma et al. [13] proposes rotation region proposal networks which generate rotated rectangles in region proposal step, Jiang et al. [14] still proposes horizontal rectangles in region proposal step but regresses a rotated rectangle in R-CNN step. Liao et al. [15] adapts [11] for multi-oriented scene text which regresses four vertices of target box and also uses recognition information to refine detection results. Liao et al. [16] classifies text line and regresses its location with different feature which achieves significant improvement on oriented text line. He et al. [17] and Zhou et al. [18] investigate to generate shrinked text line segmentation map then regress text sides or vertices on text center. Although most of scene text lines are quadrangle, there are also curved texts in a natural scene and quadrangle label may lead to background noises for these texts. Considering this, Liu et al. [19] proposes a dataset named SCUT-CTW1500 whose text line is labeled with polygon and a novel method called CTD which regresses multiple points on text lines, a TLOC component is also proposed to learn the correlation between points. But regressing both x and y coordinates of multiple points will increase redundancy, Zhu and Du [20] slides a line along horizontal box, then only regresses x or y coordinates of intersection between sliding lines and text polygon. In order to make recognition easier, Long et al. [21] proposes TextSnake which can effectively represent text lines in curved form and straighten curved text lines. SegLink [22] and SegLink++ [23] both detect scene text by linking prediction. Zhu et al. [24] proposes Rotated Cascade R-CNN and LocSLPR for shape robust object detection which can handle well curved text and polygon labeled objects.

Segmentation based method: FCN [2] is widely used to generate text segmentation map. The key point of these methods is how to split adjacent text lines. [25] predicts segmentation map of text regions and centroid of each character, then combines these information to group text lines. He et al. [3], 4], Wu and Natarajan [5] use text center or text border map to separate text lines and accelerate post-processing. Deng et al. [6] defines border on pixel level with 8-direction and uses their connected-direction to separate and group text lines. Lyu et al. [26] adopts corner detection to propose boxes and calculates each box's score on position-sensitive segmentation map. Wang et al. [27] proposes PSENet which generates various scales of shrinked text segmentation maps, then gradually expands kernels to generate final instance segmentation map which can handle well curved text lines. Xu et al. [28] proposes TextField which predicts the direction field of text line, then restores text line with these information. Xue et al. [29] detects text lines with multi-scale inputting feature and directlly regresses contour. It can also handle well curved text line. Bai et al. [30] proposes multi-scale spatial partition network (MSP-Net) to classify images which contain text or not. Khare et al. [31] proposes a blind deconvolution model which suppresses blurred pixels to enhance the edge intensity for scene text detection and recognition in video. Pastor [32] designs a robust text TS detector for noisy manuscripts. There are also many methods which adopt box proposal based instance segmentation network such as Mask R-CNN [33] for text detection. Dai et al. [34] fuses multiple-layer feature map for RPN and RoI, finally predicts a segmentation map of text. Besides instance segmentation methods designed for text, there are many general object instance segmentation methods. De Brabandere et al. [35] proposes a discriminative loss function that can push close pixels which belong to the same objects and distant pixels which belong to different objects. Novotny et al. [36] proposes semi-convolutional embedding which can separate object instances cannot be easily separated with traditional convolutional operator and this operator can also improve performance of Mask R-CNN. Bai and Urtasun [37] proposes a Deep watershed transform network for instance segmentation. Zhou et al. [38] exploits class peak responses for instance mask extraction. Different from general object instance segmentation methods, we design our system with rules of text shape which can make our system more efficient and robust because the shape of text line is more regu-

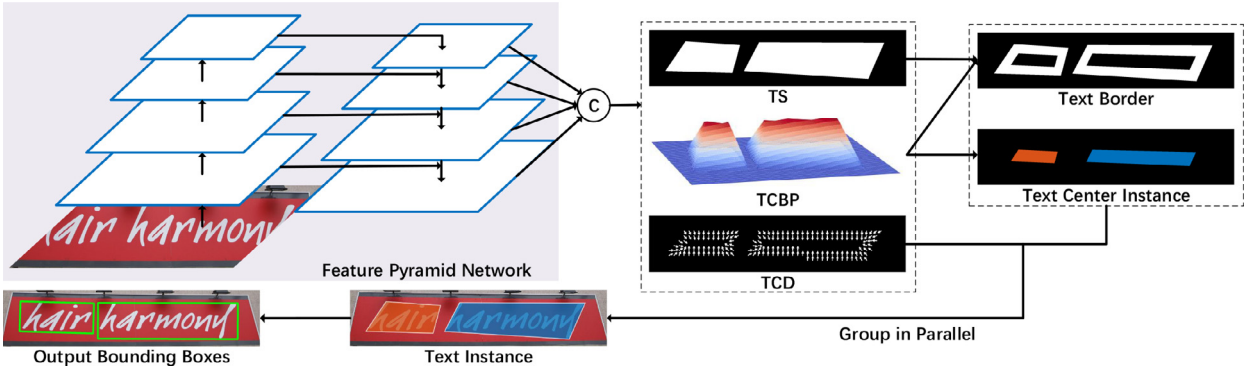


Fig. 2. The pipeline of our system: the Feature Pyramid Network (FPN) with feature fusion network, output (TS, TCBP and TCD) and post-processing.

lar than general objects. It is primarily inspired by De Brabandere et al. [35], pixels on the adjacent side of text lines have totally different features, so we use center-direction map to push close pixels which belong to the same text line and distant pixels which belong to different text lines. We also share a similar idea in Zhou et al. [38], in our design, each text is a mountain on center-border probability map, we make a full use of mountain information in post-processing. Although our definition is based on regular shape of text line, we do not define the order of vertices to avoid the ambiguity with the transformation of angles.

3. Method

3.1. Overview

This section introduces the details of TextMountain. The key idea of TextMountain is training a FCN network [2] adapted for text detection that outputs maps of text score, text center-border probability and text center-direction. As shown in Fig. 2, on the text border probability map, each text line is like a mountain whose peak is text center and foot is text border. Firstly, we set a threshold to generate text center instance map and text border map from TCBP and TS, then we calculate the score of each text center instance with its mean score on TS. The text center can separate text lines that cannot be easily separated in TS. Next, each pixel on text border map searches for its peak by TCBP using the rising direction or directly using the direction of TCD. Each pixel walks toward text center until it arrives at one text center then the pixel belongs to this text center. This step is very simple without complex rules which can be easily completed using parallel computation on GPU and consumes a little time.

3.2. Network design

Inspired by recent works related to semantic segmentation [27,39–41], we design our model based on feature pyramid network (FPN) [42]. FPN is a widely-used module in object detection and it is also used in semantic segmentaion in UPerNet [39]. The skeleton of our network is shown in Fig. 2. The parameter's setting follows FPN, the sizes of the FPN's feature maps are 1/4, 1/8, 1/16, 1/32 proportional to the original size of inputs and the number of channels is 256. Firstly we upsample FPN's feature maps to 1/4 size with bilinear interpolation, then concatenate these feature maps. Now we get a 1/4 size feature maps whose channels are 1024 (256 $\times$ 4), finally 3 $\times$ 3, 1 $\times$ 1 convolution layers are applied. The output feature map has 5 channels for TS, TCBP and TCD, TS has 2 channels for 0/1 classification, TCBP has one channel, and TCD has 2 channels. Then we upsample the output maps by 4 times which is with the same size of input images.

3.3. Loss functions

There are three tasks in this method, they are TS, TCBP and TCD. And the overall loss function can be formulated as:

$$L = L_{\mathrm{TS}} + \lambda_1 L_{\mathrm{TCBP}} + \lambda_2 L_{\mathrm{TCD}} \tag{1}$$

where L_{TS}, L_{TCBP} and L_{TCD} are the loss functions of TS, TCBP and TCD, respectively. λ_1 and λ_2 are the balancing factors of the three tasks. By default, we set λ_1 to 5. Because TCBP is among [0,1] and TCD is among [-1,1], we set λ_2 to 2.5 to balance the two losses. Next, we will introduce the details of each task.

3.4. TS

TS classifies each pixel as text or non-text, we label pixels inside text polygons as positive samples whereas pixels outside text polygons as negative samples. Class-imbalance is a serious problem because most of pixels in one image are negative samples, there are many methods to solve class-imbalance problem such as class-balanced cross-entropy [18,43], proportional selection and hard negative mining [6]. In this paper, we use the hard negative mining which selects the most hard negative samples, the positive and negative sample ratio is set to 1/3. The loss function in this task can be formulated as:

$$L_{\mathrm{TS}} = \frac{1}{|\Omega|} \sum_{\mathbf{x} \in \Omega} L_{cls}(TS_{\mathbf{x}}, TS^*_{\mathbf{x}}) \tag{2}$$

where $\mathbf{x} \in \mathbb{R}^2$ represents the two-dimensional coordinates of pixels, Ω is the set of negative pixels selected by hard negative mining and all positive samples, $|\Omega|$ represents the number of these pixels, L_{cls} is a cross-entropy loss function, $TS^*_{\mathbf{x}}$ is the ground truth and $TS_{\mathbf{x}}$ is the prediction score.

3.5. TCBP

In recent years, most of text detection methods [3,5,6,18,27] define center-border problem as a binary classification problem. In this paper, we treat center-border as a probability map because we think the hard decision of border or center is not always accurate. And there is more information in TCBP, the rising direction of probability growth points to text center which is useful for grouping pixels. In our design, the definition of curved text is the same with quadrangle, both of them is based on four sides of text line (a curved text line also has four sides but two of them may be curved). Firstly, we illustrate our method on quadrangle text line, then extend to curved text line. In fact, there are many reasonable methods to compute center-border map and their results are the same when text line is a rectangle. To facilitate a simpler label rule which is easier to learn for network, we only use the vertical

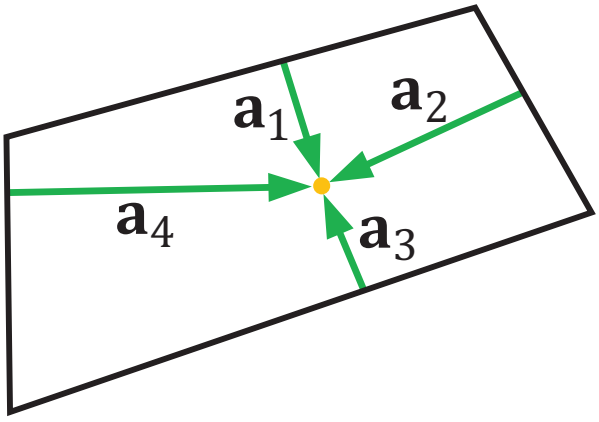


Fig. 3. The vertical lines of four sides (the polygon is a quadrilateral of any shape). In fact, the order of $\{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4\}$ doesn't make difference.

lines of four sides to compute labels, illustrated in Fig. 3. Firstly, we define the height of text line as:

$$h_{\mathbf{x}} = \min(\|\mathbf{a}_1\| + \|\mathbf{a}_3\|, \|\mathbf{a}_2\| + \|\mathbf{a}_4\|) \tag{3}$$

where $\mathbf{a}_i$ is a vector which is the vertical line of i-th side passing point $\mathbf{x}$ and the direction is from the intersection of side and vertical line to point $\mathbf{x}$, $\|\mathbf{a}_i\|$ is the length of $\mathbf{a}_i$. $\min(\|\mathbf{a}_1\| + \|\mathbf{a}_3\|, \|\mathbf{a}_2\| + \|\mathbf{a}_4\|)$ represents the minimum distance from point to two opposite sides which approximates the height text line, then the TCBP of $\mathbf{x}$ can be calculated as:

$$TCBP_{\mathbf{x}} = \frac{2 \times \min(\|\mathbf{a}_1\|, \|\mathbf{a}_2\|, \|\mathbf{a}_3\|, \|\mathbf{a}_4\|)}{h_{\mathbf{x}}} \tag{4}$$

$TCBP_{\mathbf{x}}$ is a continuous function in the range of [0, 1]. The loss of TCBP can be formulated as:

$$L_{\text{TCBP}} = \frac{\sum_{\mathbf{x}} TS^*_{\mathbf{x}} |TCBP_{\mathbf{x}} - TCBP^*_{\mathbf{x}}|}{\sum_{\mathbf{x}} TS^*_{\mathbf{x}}} \tag{5}$$

where $TCBP_{\mathbf{x}}$ and $TCBP^*_{\mathbf{x}}$ represent the prediction and ground truth of the TCBP, the activation layer of $TCBP_{\mathbf{x}}$ is sigmoid because $TCBP_{\mathbf{x}}$ is among [0, 1]. $TS^*_{\mathbf{x}}$ is the ground truth of the TS, we only calculate L_{TCBP} on text region. $|TCBP_{\mathbf{x}} - TCBP^*_{\mathbf{x}}|$ represents the distance between prediction and ground truth. In this paper, we use L1 Loss.

3.6. TCD

Although there are enough information in TCBP to group pixels, we find that network can achieve better performance by predicting TCD. Each pixel on TCD will point to the center that it belongs to, it is like the fastest mountain route. The text center-direction is different from text angle defined in Zhou et al. [18], Long et al. [21]. When height and width of text are equal, it is hard to define text's angle in Long et al. [21] and when text's angle is about 45°, it is hard to define text's angle in Zhou et al. [18]. There is no such problem in our method. In particular, the direction vector can be formulated as:

$$\mathbf{v}_{\mathbf{x}} = \sum_{i=1}^{4} [\frac{h_{\mathbf{x}}}{2} - \|\mathbf{a}_i\|]_+ \times \frac{\mathbf{a}_i}{\|\mathbf{a}_i\|} \tag{6}$$

where $h_{\mathbf{x}}$ is height of text line which is defined in Eq. (3), $[z]_+$ represents $\max(z, 0)$. We think each side has a thrust to push the point to the center and the pushing direction is the vertical line from side to point. The closer point to side, the greater the thrust is. And the thrust will be zero if the distance is longer than half of the height. Pixels on the intersection edges of two adjacent text lines have similar TCBP values but totally different TCD values. So

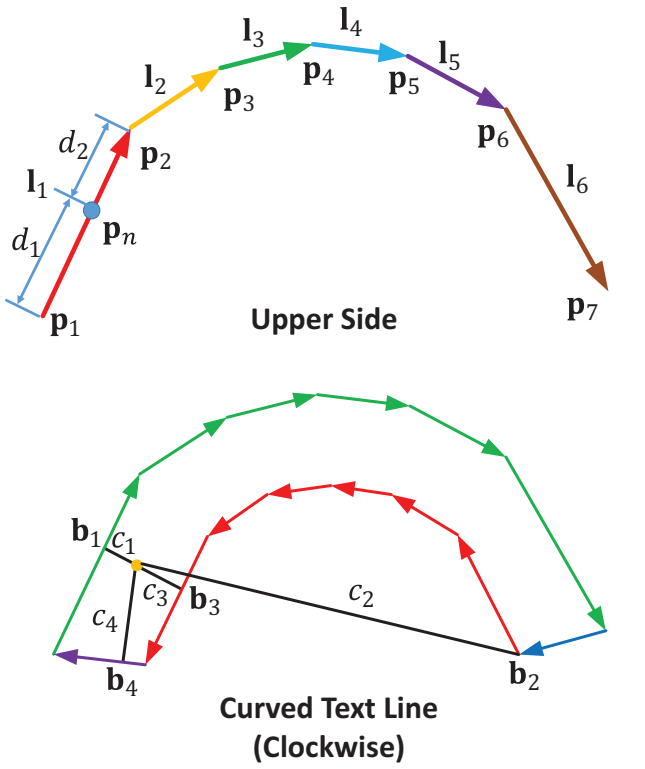


Fig. 4. Bottom: curved text line. Top: the upper side of text line. Each line or side is in a different color for better visualization.

it can help with separating adjacent text lines. Because we only need the direction of vector, the vector is normalized as:

$$\mathbf{u}_{\mathbf{x}} = \frac{\mathbf{v}_{\mathbf{x}}}{\| \mathbf{v}_{\mathbf{x}} \|} \tag{7}$$

L1 Loss is also used in this task and the loss function can be formulated as:

$$L_{\text{TCD}} = \frac{\sum_{\mathbf{x}} TS^*_{\mathbf{x}} (TCBP_{\mathbf{x}} < \gamma) |\mathbf{u}_{\mathbf{x}} - \mathbf{u}^*_{\mathbf{x}}|}{\sum_{\mathbf{x}} TS^*_{\mathbf{x}} (TCBP_{\mathbf{x}} < \gamma)} \tag{8}$$

where $TS^*_{\mathbf{x}}$ is the ground truth of score map, $TCBP_{\mathbf{x}}$ is the prediction of TCBP and γ is a center threshold. L_{TCD} is only valid on border region, because the TCD may increase ambiguity in center region and the TCD is only used by border pixels in inference. $\mathbf{u}_{\mathbf{x}}$ and $\mathbf{u}_{\mathbf{x}}{}^*$ are prediction and ground-truth of TCD respectively. $\mathbf{u}_{\mathbf{x}}$ is a two-dimensional vector with each dimension among $[-1, 1]$, so we activate them with sigmoid then multiply them by 2 and subtract 1.

3.7. Label calculation for curved text line

We illustrate the detail of our label calculation on SCUT-CTW1500 [19] and the label rule can easily be extended to other curved text datasets. SCUT-CTW1500 labels each text with 14 vertices and seven of them form a curved line. As shown in Fig. 4 (bottom), curved text line also has four sides, but two of them may be curved and the direction of lines is clockwise. Actually, the curved line is smooth and angle of line is gradually changing, labeling with finite points will lead to mutation, so we firstly smooth angle of line before calculating label. Taking one curved side as an example, one side is labeled with 7 points and 6 lines in Fig. 4 (top). $\mathbf{l}_i$ is the ith line, $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ are the start point and the end point of $\mathbf{l}_i$. $\mathbf{f}_{\mathbf{l}_i}$ is the ith line's unit vector and $\mathbf{f}_{\mathbf{p}_i}$ is the unit vector on the ith point, we use the mean of point's adjacent lines

as its value which can be formulated as:

$$\mathbf{f}_{\mathbf{p}_i} = \begin{cases} \mathbf{f}_{\mathbf{l}_1} & i = 1 \\ \mathbf{f}_{\mathbf{l}_6} & i = 7 \\ \dfrac{\mathbf{f}_{\mathbf{l}_{i-1}} + \mathbf{f}_{\mathbf{l}_i}}{\|\mathbf{f}_{\mathbf{l}_{i-1}} + \mathbf{f}_{\mathbf{l}_i}\|} & \text{otherwise} \end{cases} \tag{9}$$

and other points' unit vectors are calculated via bilinear interpolation. Taking $\mathbf{p}_n$ as an example, $\mathbf{p}_n$ is between $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$, $\mathbf{f}_{\mathbf{p}_n}$ is calculated as:

$$\mathbf{f}_{\mathbf{p}_n} = \frac{d_{i+1}\mathbf{f}_{\mathbf{p}_i} + d_i\mathbf{f}_{\mathbf{p}_{i+1}}}{\|d_{i+1}\mathbf{f}_{\mathbf{p}_i} + d_i\mathbf{f}_{\mathbf{p}_{i+1}}\|} \tag{10}$$

where d_i is the distance of $\mathbf{p}_n$ to $\mathbf{p}_i$, and the center-direction angle of $\mathbf{p}_n$ is:

$$\theta_{\mathbf{p}_n} = \angle(\mathbf{f}_{\mathbf{p}_n}) + \frac{\pi}{2} \tag{11}$$

where $\angle(\mathbf{f})$ represents the angle of vector $\mathbf{f}$, and we rotate the vector $\frac{\pi}{2}$ clockwise make it point to center.

Now we begin calculating labels. As shown in Fig. 4 (bottom), we calculate the closest point from side to calculated point then mark it as $\mathbf{b}_i$, the TCD is same as before, height of text can be formulated as:

$$h_{\mathbf{x}} = \min(c_1 + c_3, c_2 + c_4) \tag{12}$$

where $\mathbf{x}$ is the calculated point, c_i is the distance of point $\mathbf{b}_i$ to point $\mathbf{x}$, then the TCBP of $\mathbf{x}$ can be calculated as:

$$TCBP_{\mathbf{x}} = \frac{2 \times \min(c_1, c_2, c_3, c_4)}{h_{\mathbf{x}}} \tag{13}$$

As for TCD, firstly we calculate θ of point $\mathbf{b}_i$ with Eqs. (10) and (11) then mark it as $\theta_{\mathbf{b}_i}$, TCD can be formulated as:

$$v_{\mathbf{x}}^1 = \sum_{i=1}^{4} \left[\frac{h_{\mathbf{x}}}{2} - c_i\right]_+ \times \cos(\theta_{\mathbf{b}_i}) \tag{14}$$

$$v_{\mathbf{x}}^2 = \sum_{i=1}^{4} \left[\frac{h_{\mathbf{x}}}{2} - c_i\right]_+ \times \sin(\theta_{\mathbf{b}_i}) \tag{15}$$

$$\mathbf{v}_{\mathbf{x}} = (v_{\mathbf{x}}^1, v_{\mathbf{x}}^2) \tag{16}$$

where $h_{\mathbf{x}}$ is the height of text line, $[z]_+$ represents $\max(z, 0)$, $\mathbf{v}_{\mathbf{x}}$ is center-direction vector, we norm it as before:

$$\mathbf{u}_{\mathbf{x}} = \frac{\mathbf{v}_{\mathbf{x}}}{\|\mathbf{v}_{\mathbf{x}}\|} \tag{17}$$

In order to better illustrate our label rule, we show some labeled examples of MLT [44], ICDAR2015 [45], RCTW-17 [46] and SCUT-CTW1500 [19] in Fig. 5. As shown, the center-border probability map gradually decays from center to border, the center-direction points to center and gradually rotates as pixel moves.

3.8. Group in parallel

After we obtain TS, TCBP and TCD, a threshold γ is set to generate mountain peak map by $TCBP_{\mathbf{x}} > \gamma$ and γ is set to 0.6 by default. As shown in Fig. 2, this is an instance segmentation map of text mountain peak calculated by connected domain and each peak is painted in different colors. Now we need to predict which peak each pixel on mountain foot belongs to. The calculation process is as follows: Firstly, we can generate an oriented graph by TCBP or TCD. For TCBP, we select the largest point in 8-neighbor of each pixel as the next point. For TCD, the next point is calculated as follows:

$$(u_{\mathbf{x}}^1, u_{\mathbf{x}}^2) = \mathbf{u}_{\mathbf{x}} \tag{18}$$

$$next_{\mathbf{x}}^{(1,2)} = \begin{cases} 1 & u_{\mathbf{x}}^{(1,2)} > \cos\left(\frac{3}{8}\pi\right) \\ -1 & u_{\mathbf{x}}^{(1,2)} < -\cos\left(\frac{3}{8}\pi\right) \\ 0 & \text{otherwise} \end{cases} \tag{19}$$

where $\mathbf{u}_{\mathbf{x}}$ is the predicted vector from TCD. We quantify the direction because there are only 8-neighbor directions, $next_{\mathbf{x}}^{(1,2)}$ is the quantified result. When the oriented graph has been generated, every pixel on mountain foot will climb to its mountain peak step by step and paint itself in peak color. Each point's direction is deterministic, so this task can be efficiently solved in parallel, all pixels can climb to the mountain at the same time. It is not necessary for every pixel to walk through all paths, when some pixels on mountain foot climb to the middle of the mountain, the pixels on the middle of mountain have climbed to the peak and they have been colored, we can directly color pixels on mountain foot with the color of pixels on middle of mountain. Actually we increase a number of computation threads, but just additionally yielding a little computation complexity. To accelerate algorithm and avoid loop, we add a block state map which indicates whether this route is block. The procedure is summarized in Algorithm 1.

Algorithm 1 Group in parallel.

Input: C, S, D, B, ps
C - text instance color map
S - TS after setting threshold
D - center direction map
B - block state map, 1 indicate this route is block.
ps - positive pixels on border
N - number of positive pixels
Output: C
C - text instance color map after grouping

```
B[...] ← 0
for p ∈ ps do
    p_next ← D[p]
    i ← 0
    while 1 do
        i ← i + 1
        if B[p_next] == 1 then
            B[p] ← 1; break
        end if
        if (S[p_next] == 0) or (p_next == p) or (i > N) then
            B[p_next] ← 1; B[p] ← 1; break
        end if
        if C[p_next] ≠ 0 then
            C[p] ← C[p_next]; break
        end if
        p_next ← D[p_next]
    end while
end for
return C
```

This process is very fast which only needs 0.0012s for one image at 1280 × 768 resolution of ICDAR2015 dataset. After we obtaining pixel-level segmentation results, we find the contour of each instance segmentation by findContours [47]. For curved text we directly output text's contour, but for quadrangle text, we calculate a rotated rectangle of the minimum area enclosing the contour by minAreaRect [47].

Fig. 5. Quadrangle text line's label (a) and curved text line's label (b). From top to down: original image, TS, TCBP and TCD. We only show TCD on border region for better visualization.

4. Experiments

To validate the performance of the proposed method, we conduct experiments on four public datasets: MLT, ICDAR-2015, RCTW-17 and SCUT-CTW1500.

4.1. Datasets

Synth800k [48] is a large dataset which contains 800K synthetic images. These images are generated by their dense depth map. The word in image has different fonts, sizes and angle. We use this dataset to pre-train our model for ICDAR2015 and SCUT-CTW1500.

MLT [44] is a dataset provided for ICDAR 2017 competition on multi-lingual scene text detection and script identification. This dataset is composed of complete scene images from 9 languages representing 6 different scripts. Some languages are labeled in word-level such as English, Bangla, French and Arabic. Others are labeled in line-level such as Chinese, Japanese and Korean. This dataset provides 7200 images for training, 1800 images for validating and 9000 images for testing. We use both training set and validation set to train our model.

ICDAR2015 [45] is a dataset provided for ICDAR 2015 competition challenge 4. Each text is labeled on word level. Some texts which are unclear or small are labeled as "DO NOT CARE". There are 1000 images for training and 500 images for testing.

RCTW-17 [46] is a competition on reading Chinese Text in images which provides a large-scale dataset that consists of various kinds of images, including street views, posters, menus, indoor scenes, and screenshots. There are 8034 images for training and 4229 images for testing. Text lines in this dataset are labeled in line-level.

SCUT-CTW1500 [19] is a curved text dataset. Different from previous text detection datasets, each text line is labeled with a polygon whose number of vertices is 14. And the evaluation of SCUT-CTW1500 simply follows the PASCAL VOC protocol [49] but calculates IoU between the polygons instead of quadrangles. There are 1000 images for training and 500 images for testing.

Evaluation Metrics For ICDAR2015, MLT and RCTW-17, we use the online evaluation system provided by each dataset. For SCUT-CTW1500, we evaluate the performance by using the evaluation protocol in Liu et al. [19].

4.2. Implementation details

We use ResNet-50 [50] and VGG-16 [51] that are pre-trained on ImageNet [52] dataset as our backbone. The number of channels of FPN is set to 256, all upsample operators are the bilinear interpolation rather than time-consuming deconvolution, and the computation in convolutional layer includes convolution, batch normalization (BN) [53] and ReLU [54], but we remove all batch normalization on VGG-16. We find that BN can simplify training and model can be trained on higher learning rate with BN, so different training strategy is used on ResNet-50 and VGG-16 backbone. Our code is implemented based on [40].[1] For ResNet-50, following [39,55] stochastic gradient descent (SGD) is used, we adopt "poly" learning rate policy. In particular, learning rate is calculated with $base\_lr \times (1 - \frac{iter}{max\_iter})^{power}$ where *iter* and *max_iter* are number of iterations at present and total number of iterations, *base_lr* is initial learning rate which is set to 0.005 and *power* is set to 0.9 in our experiment. For VGG-16, following [21], Adam optimizer [56] is used, the learning rate is set to 0.0001 initially, then set to 0.00001 after $\frac{2}{3}$ of the entire training. And we set weight decay to 0.0001, momentum to 0.9 and batch size to 12 for both of ResNet-50 and VGG-16. Due to limited computing resources, we do not conduct all experiments with both ResNet-50 and VGG-16. In MLT and RCTW-2017, we conduct experiments on ResNet-50 to compare with [57]. The threshold of text center instance score is set to 0.7

[1] https://github.com/CSAILVision/semantic-segmentation-pytorch.

Table 1
Ablation experimental results. " TS": a semantic segmentation result, "TS+TCB": only use center-border binary map in training and inference. "TS+TCB+TCD(train)": use center-border binary map and TCD in training but only use center-border binary map in inference, TCD(train) means TCD is only used in training but not testing. "TS+TCBP": only use TCBP in both training and inference, "TS+TCB+TCD": use center-border binary map and TCD in training and inference, "TS+TCBP+TCD": use both TCBP and TCD in training and inference, "TS+TCBP+TCD(train)": only use TCBP in inference but model is trained with TCD and TCBP, TCD(train) means TCD is only used in training but not testing. (These models' backbones are ResNet-50.).

Method	P	R	F
TS	57.42	52.53	54.86
TS+TCB	78.59	65.51	71.46
TS+TCB+TCD(train)	81.23	66.60	73.19
TS+TCBP	80.92	67.69	73.72
TS+TCB+TCD	82.34	67.33	74.08
TS+TCBP+TCD	82.78	67.92	74.62
TS+TCBP+TCD(train)	**82.82**	**68.06**	**74.72**

Table 2
Results on MLT. MS means multi-scale. (The backbones of our models are ResNet-50.).

Method	P	R	F
linkage-ER-Flow [44]	44.48	25.59	32.49
TH-DL [44]	67.75	34.78	45.97
SARI_FDU_RRPN_v1 [44]	71.17	55.50	62.37
Sensetime OCR [44]	56.93	69.43	62.56
SCUT_DLVClab1 [44]	80.28	54.54	64.96
Lyu et al. [26]	**83.8**	55.6	66.8
PSENet [27]	75.3	69.2	72.2
Ours	82.82	68.06	74.72
Xue et al. MS [57]	73.9	60.6	66.6
Lyu et al. MS[26]	74.3	70.6	72.4
Ours MS	79.33	**74.51**	**76.85**

and the threshold of text border on TS is set to 0.6. By default, the backbone of our model is ResNet-50, we train our model with TS, TCBP and TCD, but only use TS and TCBP in inference. For MLT and RCTW-17, we only use their respective dataset to train our model. Generating datasets for training is valid for small dataset, there are many studies [48,58] which propose novel methods for genearting text detection and recognition datasets. For ICDAR2015 and SCUT-CTW1500, our model is pre-trained on Synth800k [48] then fine-tuned on their respective dataset. Different datasets vary greatly in size, so we use different configurations in testing for different datasets.

To make network more robust, we adopt data augmentation for preventing overfitting, especially for limited datasets. Firstly, we rotate images by a random angle of $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ with a probability of 0.5. After that a random rotation in range $[-10^\circ, 10^\circ]$ is also applied on images. Next, we randomly crop a 512×512 image from rotated images whose labels are quadrangle using the rules in SSD [12]. But for curved polygon labeled dataset we crop images without crossing texts, because cropping may destroy curved shape of polygon. In the training, some text lines which are unclear and fuzzy are labeled as "DO NOT CARE", we ignore them by setting loss weight to zero. Besides these, the text lines whose height is smaller than 10 pixels are also ignored. The whole algorithm is implemented in PyTorch 0.4.0 [59] and we conduct all experiments on a regular workstation whose CPU is Intel(R) Core(TM) i7-7700 K and GPU is GeForce GTX 1080Ti.

4.3. Experiments on MLT

We combine training set and validation set of MLT to train our model with 180,000 iterations. Firstly, we make controlled experiments to examine how each component affects the model performance. We use the same settings except for each tested component in our experiment and test in single scale by resizing the long side of images to 1800. Then we compare our method with other state-of-the-art methods and also evaluate our model with multiple scales. For multi-scale testing, we resize the long side of images to {1000, 1800, 2600} pixels and merge multi-scale results by non-maximum suppression (NMS).

4.3.1. Ablation experiments

Comparisons with TS Firstly we train a semantic segmentation model named " TS" which only includes TS. Table 1 shows the result of " TS" where semantic segmentation map does not have enough information to separate adjacent text lines. However, TCBP can well address this problem and its gradient direction can be used in grouping pixels. So the results of "TS+TCBP" are much better than those of " TS".

How important are TCBP's grads (TCD) Network can directly predict TCD or calculate TCD with TCBP's grad. If there is only center-border binary map in testing, we expand outputting quadrangle with ratio of shrinkage [3,5]. We can see that using TCBP'grads to group text lines can achieve better performance, "TS+TCB+TCD" gains 0.89% improvement compared with "TS+TCB+TCD(train)". And expanding outputting quadrangle with ratio of shrinkage can only handle quadrangle text lines but not curved text lines.

Although there are enough information in TCBP for both separating and grouping text lines, we find that TCD can make model more robust. In order to prove this, we train models with TCBP but with/without TCD. As shown in Table 1, "TS+TCBP+TCD(train)" outperforms "TS+TCBP" 1%, which indicates only using TCD in training can make TCBP learn better. But we find that using TCD in testing ("TS+TCBP+TCD") will slightly reduce performance (0.1%), this indicates that using TCD in inference produces worse results. Actually we use TCBP to generate text center to split text lines, then use TCBP's grad to group text lines. If one of them is wrong, network will output wrong result, so we find that generating text center and TCBP's grad with consistent TCBP can reduce errors caused by wrong TCD. From Eq. (4) and Eq. (6), we can see that TCD and TCBP have a strong correlation, they learn via different expressions of the same feature. TCD can push close pixels belong to the same text line and distant pixels belong to different text lines.

Probability map vs. binary map If there is TCD, we can group text lines without TCBP. The TCBP is only used for separating text which can also be done by center-border binary classification map. To prove TCBP still plays an important role in this case, we train a model named "TS+TCB+TCD" with center-border binary classification map and TCD. The center-border binary classification map classifies pixels as two classes (center or border), and the center threshold used in label is γ which is the same as in TCBP. As shown in Table 1, TCBP ("TS+TCBP+TCD") improves center-border binary classification map ("TS+TCB+TCD") with gains of 0.44% on Precision and 0.59% on Recall. The most important part of this study is how to group text lines with TCBP's grad. If there is TCD, the TCBP's grad can be calculated by TCD, TCPB only works for smoothing, so the improvement is minimal. As illustrated in Fig. 6, TCBP is smoother than center-border binary classification map.

4.3.2. Comparing with other state-of-the-art methods

We compare our method with other state-of-the-art methods in Table 2. To be fairly comparable, here the result from [57] is based on ResNet-50. We can observe that our method significantly outperforms other methods. For the single-scale setting, our method achieves the F-measure of 74.72% with an absolute gain of about 8% over the most competing method in Lyu et al. [26].

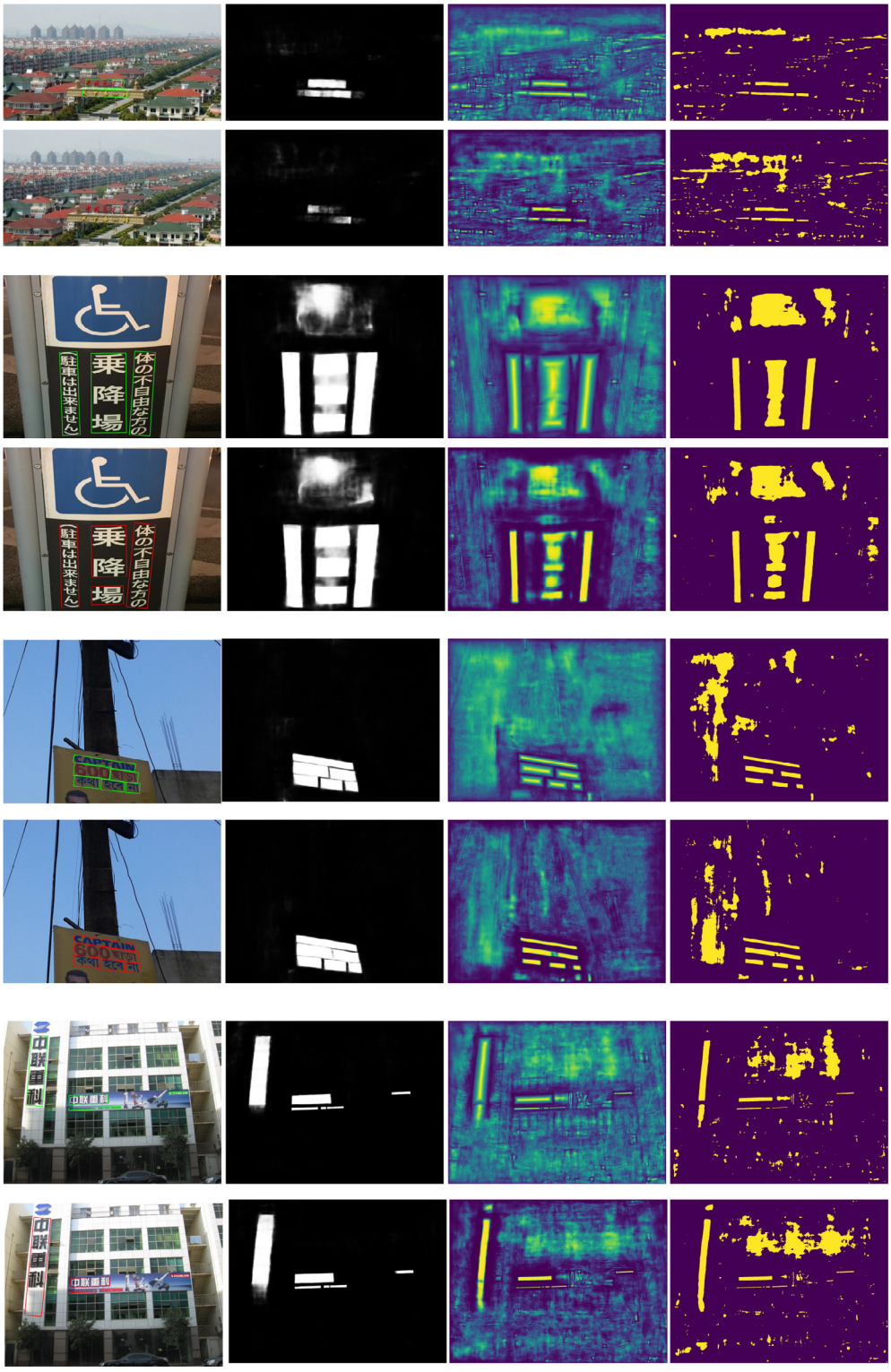


Fig. 6. A comparison of TCBP (Top) and center-border binary classification map (Bottom). From left to right: detection result, TS map, TCBP (or center-border binary classification) map and center-border map after setting threshold.

For the multi-scale setting, TextMountain achieves the F-measure of 76.85% with an absolute gain of about 4% over the most competing method in Lyu et al. [26].

4.4. Experiments on ICDAR2015

We validate the performance of our method on ICDAR2015 dataset to evaluate its ability for oriented text. We fine-tune our model 60,000 iterations on ICDAR2015 training set. Following [6,21,26] we resize images to 1280 × 768 in inference and report the single-scale result. We compare our method with other state-of-the-art methods and show results in Table 3. Many methods' results on ICDAR2015 are based on VGG-16 such as [6,21], to fairly compare with these methods, we both conduct experiments on ResNet-50 and VGG-16, and only pre-train our model on Synth800k. With the same backbone, our method achieves

Fig. 7. Qualitative results by the proposed method. From left to right: MLT, ICDAR2015, RCTW-17 and SCUT-CTW1500.

Table 3
Results on ICDAR2015.

Methods	P	R	F	FPS
Zhang et al. [25]	71	43	54	–
SegLink [22]	73.1	76.8	75.0	–
EAST [18]	83.57	73.47	78.20	**13.2**
EAST [18]	83.27	78.33	80.72	–
He et al. [17]	82	80	81	–
PixelLink [6]	85.5	82.0	83.7	3.0
Lyu et al. [26]	**89.5**	79.7	84.3	1
TextSnake [21]	84.9	80.4	82.6	1.1
PSENet [27]	86.9	**84.5**	85.7	1.6
MSR [29]	86.6	78.4	82.3	4.3
Ours (ResNet-50)	87.31	**84.11**	85.68	10.4
Ours (VGG-16)	**89.52**	83.05	**86.16**	9.7

Table 4
The efficiency experiment on ICDAR2015 and RCTW-17. C is the time (in second) consumed in CPU group, G is the time (in second) consumed in GPU group and other is the time (in second) consumed in other stages.

Dataset	Resolution	C	G	Other
IC15	1280 × 768	0.2086	0.0012	0.0946
RCTW-17	500 (long)	0.1646	0.0003	0.0268
RCTW-17	1500 (long)	1.2910	0.0013	0.1653

better performance (precision: 89.52%, recall: 83.05% and F-measure: 86.16%) compared with other segmentation based methods [6,21,26]. Moreover, by acceleration using parallel grouping, TextMountain can run at a speed of 9.7 FPS, is faster than most methods. To explore the efficiency of parallel grouping, we compare two methods, one is implemented in cython based on the code of PixelLink[2] and the other one is implemented in GPU with Algorithm 1. In Table 4, obviously our grouping using GPU parallel computation is much faster (200 ×) than cython code.

[2] https://github.com/ZJULearning/pixel_link.

Table 5
Results on RCTW-17. (* indicate the result is from [16], the backbones of our models are ResNet-50.).

Methods	P	R	F	FPS
Official baseline [46]	76.03	40.44	52.78	8.9
EAST-ResNet* [18]	59.7	47.8	53.1	7.4
RRD [16]	72.4	45.3	55.7	**10**
Ours	**80.80**	55.24	65.63	6
RRD MS [16]	77.5	59.1	67.0	–
Xue et al. MS [57]	74.2	58.5	65.4	–
Ours MS	76.82	**60.29**	**67.56**	–

4.5. Experiments on RCTW-17

We validate the performance of our method on RCTW-17 dataset to evaluate its ability for long Chinese text. We train our model with 8034 images from RCTW-17 for 180,000 iterations. Results are shown in Table 5. For a fair comparison, the result from [57] is based on ResNet-50. Firstly, we test our model in the single-scale setting, we resize the long side of images to 1500. Compared with regression based methods (EAST, RRD), our method achieves much better performance on high resolution images, because the receptive field of our method does not need to cover the whole text line. In the multi-scale setting, the long side of images is resized to {500, 1000, 1500, 2000, 2500}. The F-measure of TextMountain is 67.56%, yielding comparable results with other methods. In Table 4, we also evaluate the efficiency of grouping. As the size of the image increases, CPU grouping consumes plenty of time. For example, when the long side of image is 1500, CPU grouping consumes 1.2910s for one image, but our GPU grouping only needs 0.0013s. It proves the proposed algorithm is quite efficient for high resolution images.

4.6. Experiments on SCUT-CTW1500

On SCUT-CTW1500, we evaluate the performance of our method for detecting curved text lines. We fine-tune our model 60,000 iterations on SCUT-CTW1500 training set and resize the long side of images to 800 in inference. For fairness, we report our single-scale result. The results are listed in Table 6. The proposed method achieves much better results (81.3%, 82.4%, 81.9% in preci-

Table 6
Results on SCUT-CTW1500. (* indicate the result is from [19]).

Method	P	R	F
Seglink* [22]	42.3	40.0	40.8
SWT* [7]	20.7	9.0	12.5
CTPN* [60]	60.4	53.8	56.9
EAST* [18]	78.7	49.1	60.4
DMPNet* [61]	69.9	56.0	62.2
CTD+TLOC [19]	77.4	69.8	73.4
TextSnake [21]	67.9	**85.3**	75.6
MSR [29]	**85.0**	78.3	81.5
PSENet [27]	84.8	79.7	82.2
Ours (ResNet-50)	83.3	83.6	**83.4**
Ours (VGG-16)	81.3	82.4	81.9

sion, recall and F-measure) based on VGG-16 compared with other segmentation methods, and achieves 83.4% F-measure based on ResNet-50 which proves that TextMountain can handle well curved text lines.

5. Conclusion

In this study, we propose a novel text detection method named TextMountain. Our method uses TCBP to accurately define center and border probability, and uses TCBP'grad to group text lines. The proposed method achieves state-of-the-art or comparable performance on both traditional quadrangle labeled datasets (MLT, ICDAR2015, RCTW-17) and newly-released polygon labeled curved dataset (SCUT-CTW1500). From Fig. 7, we can observe that our method is robust to the variation of shape, angle and size. And benefitting from the parallel grouping, the proposed method is also efficient. We can see that the proposed method achieves much better results over MLT than over RCTW and ICDAR15. This is because the proposed method is a segmentation method. Compared with regression method, segmentation method only needs to cover the short side of text line while regression method must cover both the short and long sides. It needs smaller receptive field compared with regression method. So it can achieve better performance on MLT whose text lines may be long compared with regression method. But ICDAR2015 is an English text dataset, the word is short compared with MLT. Accordingly segmentation method does not have advantage compared with regression method in ICDAR2015. Compared with RCTW-17, MLT has more consistent labeling rules, for example, the English text lines of MLT are always word-level but the text lines of RCTW-17 may be labeled on word-level or line-level. Regression method can propose bounding boxes with various proportions, when the labeling rules are difficult to learn, it can achieve better recall rate compared with segmentation method. So our method achieves better performance on MLT compared with RCTW-17.

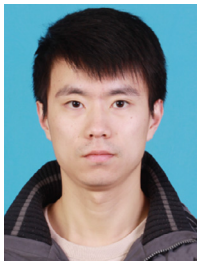
Acknowledgments

This work was supported in part by the National Key R&D Program of China under grant 2017YFB1002202, in part by the National Natural Science Foundation of China under grants 61671422 and U1613211, in part by the Key Science and Technology Project of Anhui Province under grant 17030901005, and in part by the MOE-Microsoft Key Laboratory of University of Science and Technology of China.

References

[1] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[2] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
[3] T. He, W. Huang, Y. Qiao, J. Yao, Accurate text localization in natural image with cascaded convolutional text network, (2016). arXiv:1603.09423
[4] D. He, X. Yang, C. Liang, Z. Zhou, G. Alexander, I. Ororbia, D. Kifer, C.L. Giles, Multi-scale FCN with cascaded instance aware segmentation for arbitrary oriented word spotting in the wild., in: CVPR, 2017, pp. 474–483.
[5] Y. Wu, P. Natarajan, Self-organized text detection with minimal post-processing via border learning, in: Proc. ICCV, 2017.
[6] D. Deng, H. Liu, X. Li, D. Cai, PixelLink: detecting scene text via instance segmentation, in: Thirty-Second AAAI Conference on Artificial Intelligence, 2018.
[7] B. Epshtein, E. Ofek, Y. Wexler, Detecting text in natural scenes with stroke width transform, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, IEEE, 2010, pp. 2963–2970.
[8] L. Neumann, J. Matas, Real-time scene text localization and recognition, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 3538–3545.
[9] P. Shivakumara, R. Raghavendra, L. Qin, K.B. Raja, T. Lu, U. Pal, A new multi-modal approach to bib number/text detection and recognition in marathon images, Pattern Recognit. 61 (2017) 479–491.
[10] L. Sun, Q. Huo, W. Jia, K. Chen, A robust approach for text detection from natural scene images, Pattern Recognit. 48 (9) (2015) 2906–2920.
[11] M. Liao, B. Shi, X. Bai, X. Wang, W. Liu, Textboxes: a fast text detector with a single deep neural network., in: AAAI, 2017, pp. 4161–4167.
[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, SSD: single shot multibox detector, in: European Conference on Computer Vision, Springer, 2016, pp. 21–37.
[13] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, X. Xue, Arbitrary-oriented scene text detection via rotation proposals, IEEE Trans. Multimed. 20 (11) (2018) 3111–3122.
[14] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, Z. Luo, R 2 CNN: rotational region CNN for arbitrarily-oriented scene text detection, in: 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, 2018, pp. 3610–3615.
[15] M. Liao, B. Shi, X. Bai, Textboxes++: a single-shot oriented scene text detector, IEEE Trans. Image Process. 27 (8) (2018) 3676–3690.
[16] M. Liao, Z. Zhu, B. Shi, G.-s. Xia, X. Bai, Rotation-sensitive regression for oriented scene text detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5909–5918.
[17] W. He, X.-Y. Zhang, F. Yin, C.-L. Liu, Deep direct regression for multi-oriented scene text detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 745–753.
[18] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, J. Liang, East: an efficient and accurate scene text detector, in: Proc. CVPR, 2017, pp. 2642–2651.
[19] Y. Liu, L. Jin, S. Zhang, C. Luo, S. Zhang, Curved scene text detection via transverse and longitudinal sequence connection, Pattern Recognit. 90 (2019) 337–345.
[20] Y. Zhu, J. Du, Sliding line point regression for shape robust scene text detection, in: 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, 2018, pp. 3735–3740.
[21] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, C. Yao, TextSnake: a flexible representation for detecting text of arbitrary shapes, in: European Conference on Computer Vision, Springer, 2018, pp. 19–35.
[22] B. Shi, X. Bai, S. Belongie, Detecting oriented text in natural images by linking segments, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 2550–2558.
[23] J. Tang, Z. Yang, Y. Wang, Q. Zheng, Y. Xu, X. Bai, Detecting dense and arbitrary-shaped scene text by instance-aware component grouping, Pattern Recognit. 96 (2019) 106954.
[24] Y. Zhu, C. Ma, J. Du, Rotated cascade r-CNN: a shape robust detector with coordinate regression, Pattern Recognit. 96 (2019) 106964.
[25] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, X. Bai, Multi-oriented text detection with fully convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4159–4167.
[26] P. Lyu, C. Yao, W. Wu, S. Yan, X. Bai, Multi-oriented scene text detection via corner localization and region segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7553–7563.
[27] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, S. Shao, Shape robust text detection with progressive scale expansion network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9336–9345.
[28] Y. Xu, Y. Wang, W. Zhou, Y. Wang, Z. Yang, X. Bai, TextField: learning a deep direction field for irregular scene text detection, IEEE Trans. Image Process. 28 (11) (2019) 5566–5579.
[29] C. Xue, S. Lu, W. Zhang, MSR: Multi-scale shape regression for scene text detection, in: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 989–995, doi:10.24963/ijcai.2019/139.
[30] X. Bai, B. Shi, C. Zhang, X. Cai, L. Qi, Text/non-text image classification in the wild with convolutional neural networks, Pattern Recognit. 66 (2017) 437–446.
[31] V. Khare, P. Shivakumara, P. Raveendran, M. Blumenstein, A blind deconvolution model for scene text detection and recognition in video, Pattern Recognit. 54 (2016) 128–148.
[32] M. Pastor, Text baseline detection, a single page trained system, Pattern Recognit. 94 (2019) 149–161.

[33] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-CNN, in: Computer Vision (ICCV), 2017 IEEE International Conference on, IEEE, 2017, pp. 2980–2988.
[34] Y. Dai, Z. Huang, Y. Gao, Y. Xu, K. Chen, J. Guo, W. Qiu, Fused text segmentation networks for multi-oriented scene text detection, in: 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, 2018, pp. 3604–3609.
[35] B. De Brabandere, D. Neven, L. Van Gool, Semantic Instance Segmentation with a Discriminative Loss Function, (2017). arXiv:1708.02551
[36] D. Novotny, S. Albanie, D. Larlus, A. Vedaldi, Semi-convolutional operators for instance segmentation, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 86–102.
[37] M. Bai, R. Urtasun, Deep watershed transform for instance segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2017, pp. 2858–2866.
[38] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, J. Jiao, Weakly supervised instance segmentation using class peak response, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 3791–3800.
[39] T. Xiao, Y. Liu, B. Zhou, Y. Jiang, J. Sun, Unified perceptual parsing for scene understanding, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 418–434.
[40] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, A. Torralba, Scene parsing through ADE20K dataset, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017.
[41] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, A. Torralba, Semantic understanding of scenes through the ADE20K dataset, Int. J. Comput. Vis. 127 (3) (2019) 302–321.
[42] T.-Y. Lin, P. Dollár, R.B. Girshick, K. He, B. Hariharan, S.J. Belongie, Feature pyramid networks for object detection., in: CVPR, 1, 2017, p. 4.
[43] S. Xie, Z. Tu, Holistically-nested edge detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1395–1403.
[44] Mlt-challenge, (http://rrc.cvc.uab.es/?ch=8&com=introduction).
[45] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V.R. Chandrasekhar, S. Lu, et al., ICDAR 2015 competition on robust reading, in: Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, IEEE, 2015, pp. 1156–1160.
[46] B. Shi, C. Yao, M. Liao, M. Yang, P. Xu, L. Cui, S. Belongie, S. Lu, X. Bai, ICDAR2017 competition on reading chinese text in the wild (RCTW-17), in: Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on, 1, IEEE, 2017, pp. 1429–1434.
[47] Opencv, (https://opencv.org/).
[48] A. Gupta, A. Vedaldi, A. Zisserman, Synthetic data for text localisation in natural images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2315–2324.
[49] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (VOC) challenge, Int. J. Comput. Vis. 88 (2) (2010) 303–338.
[50] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[51] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, (2014), arXiv:1409.1556
[52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, ImageNet: a large-scale hierarchical image database, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, Ieee, 2009, pp. 248–255.
[53] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37, JMLR. org, 2015, pp. 448–456.
[54] V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010, pp. 807–814.
[55] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A.L. Yuille, DeepLab: semantic image segmentation with deep convolutional nets, Atrous convolution, and fully connected CRFs, IEEE Trans. Pattern Anal. Mach. Intell. 40 (4) (2018) 834–848.
[56] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, (2014). arXiv:1412.6980
[57] C. Xue, S. Lu, F. Zhan, Accurate scene text detection through border semantics awareness and bootstrapping, in: European Conference on Computer Vision, Springer, 2018, pp. 370–387.
[58] F. Zhan, S. Lu, C. Xue, Verisimilar image synthesis for accurate detection and recognition of texts in scenes, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 249–266.
[59] Pytorch, (https://pytorch.org/).
[60] Z. Tian, W. Huang, T. He, P. He, Y. Qiao, Detecting text in natural image with connectionist text proposal network, in: European Conference on Computer Vision, Springer, 2016, pp. 56–72.
[61] Y. Liu, L. Jin, Deep matching prior network: Toward tighter multi-oriented text detection, in: Proc. CVPR, 2017, pp. 3454–3461.



Yixing Zhu received a B.Eng. degree from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), in 2017. He is currently a Master's candidate at USTC. His current research area includes deep learning, OCR and object detection in aerial images.

Jun Du received B.Eng. and Ph.D. degrees from the Department of Electronic Engineering and Information Science, University of Science and Technology of China (USTC), in 2004 and 2009, respectively. From 2004 to 2009, he was with the iFlytek Speech Lab of USTC. During the above period, he worked as an Intern twice for 9 months at Microsoft Research Asia (MSRA), Beijing. In 2007, he also worked as a Research Assistant for 6 months in the Department of Computer Science at the University of Hong Kong. From July 2009 to June 2010, he worked at iFlytek Research on speech recognition. From July 2010 to January 2013, he joined MSRA as an Associate Researcher, working on handwriting recognition, OCR and speech recognition. Since February 2013, he has been with the National Engineering Laboratory for Speech and Language Information Processing (NEL-SLIP) of USTC.