A SINGLE-SHOT ORIENTED SCENE TEXT DETECTOR WITH LEARNABLE ANCHORS

Fenfen Sheng[1,2], *Zhineng Chen*[1], *Tao Mei*[3], *Bo Xu*[1]

[1]Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[2]University of Chinese Academy of Sciences, Beijing 100190, China
[3]JD AI Research, Beijing, 100101, China
shengfenfen2015@ia.ac.cn, zhineng.chen@ia.ac.cn, tmei@jd.com, xubo@ia.ac.cn

ABSTRACT

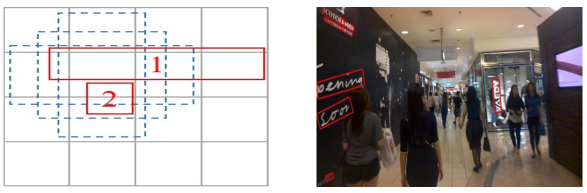
Current regression based text detectors mainly use fixed anchors, where scales and positions can not be changed during network training. As scene texts tend to have large variation in orientations, aspect ratios and sizes, fixed anchors are insufficient to cover all varieties. This paper proposes a novel text detector with learnable anchors, named LATD. LATD contains two prediction branches. One aims to refine scales and locations of anchors according to the characteristics of scene texts. The other one receives refined anchors as defaults and regresses their offsets to text regions. These two branches are optimized jointly without sacrifices much speed. Meanwhile, we explore the class-imbalance issue between texts and backgrounds, and replace softmax loss with focal loss. Extensive experiments on both oriented and horizontal benchmarks demonstrate the effectiveness of LATD with new state-of-the-art performance. By visualizing qualitative results, as expected, LATD provides more accurate locations and lower rate of missed detections.

Index Terms— oriented scene text detection, learnable anchors, class-imbalance issue

1. INTRODUCTION

Scene text detection, which aims to locate texts in natural images, has drawn increasing interests from both artificial intelligence and computer vision. Extensive studies have been carried out in the past few years, but this task is still challenging due to several difficulties, e.g., low visual quality, cluttered background, complex deformation and arbitrary orientations.

Nowadays single-shot regression based detectors are popularized owing to their accuracy and efficiency [1, 2, 3]. These detectors can be divided into two classes by different regression algorithms [3]. One is based on indirect regression [1, 3], which first creates a set of default anchors then locates texts by regressing matched anchors to ground truths. The other one is based on direct regression [2, 4], which outputs locations from a given point directly but often suffers from low accuracy especially for long texts [5]. Indirect regression based methods have achieved remarkable performance



(a) (b)

Fig. 1: (a) An illustration of fixed anchors, where horizontal instead oriented anchors are shown for a better visualization. Blue dashed and red solid lines indicate anchors and ground truths respectively. (b) An illustration of a scene text image with ground truths (red solid lines). Best viewed in color.

but still face two constraints. Firstly, scales of anchors are fixed and limited. Anchor mechanism is deemed as rectangular proposals with different sizes and aspect ratios defined before network training. Once has been created, their scales are unchangeable, let alone adjusted to scene texts. Secondly, positions of anchors are sparse and fixed. Considering speed, anchors are usually generated from higher layers with small resolution, instead from lower layers or even input images. Besides, anchors are placed on images at equally spaced intervals and their positions could not be shifted neither.

As scene texts have large variation in scales, locations and orientations (see Figure 1 (b)), fixed anchors cause inaccurate detections in the form of missing certain text parts or containing undesired backgrounds. As depicted in Figure 1 (a), after regression process, text 1 would miss its right part due to long length, while text 2 is not even detected as no anchor has sufficient overlap with it. Although TextBoxes [6] adds more aspect ratios and vertical offsets to enrich anchors, it is still insufficient to cover all diverse text patterns but costs more running time. Therefore, fixed anchors have become the bottleneck of indirect regression based detectors.

The goal of this paper is to detect scene texts with learnable anchors. To achieve this goal, we propose a novel indirect regression based model, named LATD, that involves an SSD-based [7] backbone and two prediction branches. One

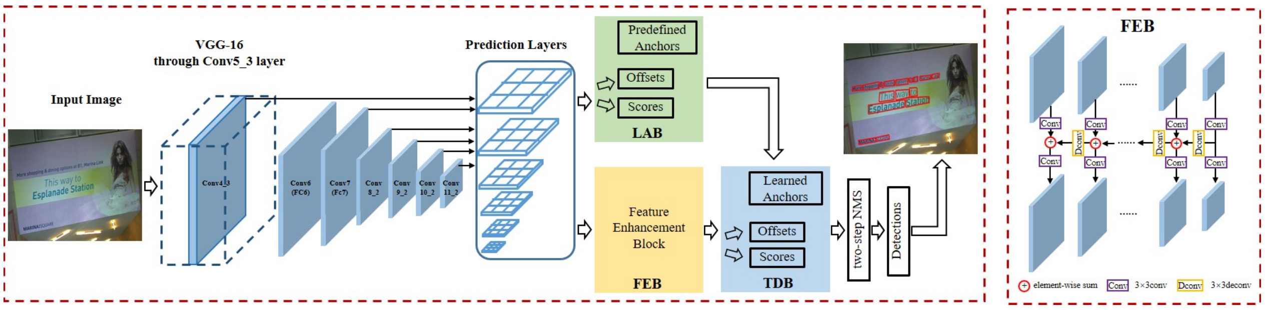


Fig. 2: (left) Network architecture of LATD. (right) Architecture of the feature enhancement block.

branch, named learnable anchor branch (LAB), aims to adjust the positions and scales of predefined anchors according to the characteristic of scene texts. LAB outputs a set of refined anchors with better initialization. The other one, named text detection branch (TDB), aims to predict quadrilateral or horizontal text locations by regressing anchors from LAB rather than fixed ones as previous detectors do. Based on LAB and TDB, LATD shows significant superiority in localizing scene texts with closer bounding boxes. To our knowledge, LATD is the first one to explore learnable anchors for oriented/horizontal text detection.

Additionally, indirect regression based detectors usually generate thousands of anchors per image. After regression process, only a few anchors are matched with text regions and a large proportion are backgrounds. Take ICDAR2013 dataset for example, each image merely contains four texts on average. Therefore, there is an extreme class imbalance between texts and backgrounds. Though some sampling heuristics are adopted, e.g. online hard example mining, easy backgrounds still overwhelm training and lead to degenerate models. In this paper, we modify softmax loss to focal loss [8] by adding two factors to down weight the contribution of easy examples and focusing more on hard examples.

We conduct extensive experiments on both oriented (ICDAR2015 Incidental Scene Text and COCO-Text) and horizontal (ICDAR2013 Focused Scene Text) benchmarks. Evaluations demonstrate that LATD achieves state-of-the-art or highly competitive performance compared to the best model in the literature. By analyzing qualitative results, LATD exhibits more accurate location and reduces the rate of missed detection. Our contributions are summarized as follows:

(I) We analysis the deficiency of fixed anchors used in current detectors, and propose a novel LATD that consists of two prediction branches for anchors refinement and texts detection respectively. LATD could generate more appropriate anchors with refined scales and positions according to the characteristic of scene texts.

(II) We describe the class-imbalance issue between texts and backgrounds, and replace softmax loss with focal loss to relief this problem.

(III) LATD achieves the new state-of-the-art performance and surpasses competitive detectors on both oriented and horizontal text benchmarks.

2. RELATED WORK

One of related works to LATD is RefineDet [9], a recent development in generic object detection. Indeed, LATD is inspired by RefineDet but has distinct differences. First, RefineDet aims to detect general objects but fails on texts with extreme aspect ratios, e.g., too long. LATD relies on specifically designed prediction layers to efficiently solve this problem. Second, RefineDet only generates horizontal rectangles, while LATD could generate arbitrarily oriented quadrangles. Furthermore, RefineDet applies default softmax loss for classification. We argue that it introduces class imbalance issue especially for scene texts with large variation in appearances, and replace it with focal loss to boost performance.

Another related work is TextBoxes++, which is also based on indirect regression but uses fixed anchors. TextBoxes++ applies more aspect ratios and vertical offsets for anchors. Though effective, it is not enough to enumerate arbitrary-oriented texts. LATD explores learnable anchors to refine scales and positions during network training, and accordingly achieves better performance compared to TextBoxes++.

3. METHODOLOGY

The architecture of LATD is depicted in Figure 2. The whole network consists of three main components: the VGG-16 based backbone, two prediction branches (LAB and TDB), and a feature enhancement block (FEB).

Following SSD, LATD inherits VGG-16 network by keeping layers from conv1_1 to conv5_3, converting last two fully-connected layers into convolutional layers (conv6 and conv7), truncating the classification layers (fc-1000 and softmax), and adding a series of convolutional layers (conv8 to conv11) to the end with size decreased progressively. We leverage multiple layers with different receptive fields (conv4_3, conv7 to conv11) as prediction layers. Each of them flows into two prediction branches. LAB predefines anchors and predicts their corresponding offsets, while TDB receives refined anchors from LAB and regresses matched anchors to output text locations. Note that there is a feature enhancement block (FEB) operated on prediction layers before fed into TDB. TDB is leveraged in pursuit of high-level semantics. Finally, detections from different prediction layers are aggregated together and undergo an efficient non-maximum suppression (NMS) to filter out redundant outputs. In the following sections, we will give details of these components.

3.1. Learnable Anchor Branch (LAB)

LAB refines the scales and positions of default anchors to provides better initialization for TDB. LATD only uses horizontal rectangles instead of quadrangles as default anchors for a simpler matching strategy. Anchors tile each feature map with various sizes and aspect ratios. Considering scene texts tend to have a large variation in aspect ratios, e.g., short and long texts, we use 6 aspect ratios including {1,2,3,5,7,10} for each anchor to better cover scene texts, as TextBoxes++ do.

For each horizontal anchor $\mathbf{b}_0 = (x_0, y_0, w_0, h_0)$, where (x_0, y_0) means center point, w_0 and h_0 are width and height, LAB simultaneously predicts its offset $(\Delta x_0, \Delta y_0, \Delta w_0, \Delta h_0)$ and text presence confidence c_0 from background. $\mathbf{b}_0$ is adjusted into the refined anchor $\mathbf{b}_r = (x_r, y_r, w_r, h_r)$, where

$$\begin{aligned} x_r &= x_0 + w_0 \Delta x_0, \quad & w_r &= w_0 \exp\left(\Delta w_0\right), \\ y_r &= y_0 + h_0 \Delta y_0, \quad & h_r &= h_0 \exp\left(\Delta h_0\right). \end{aligned} \tag{1}$$

In training phase, LAB follows rectangle matching scheme as in SSD to match anchors to rectangle ground-truths according to boxes overlaps. Anchor shapes, including positions and scales, are learned by back propagation. These refined anchors are then passed to the corresponding feature maps in TDB.

3.2. Text Detection Branch (TDB)

TDB receives the refined anchors from LAB as default ones, then regresses locations for oriented/horizontal texts. Each refined horizonal anchor could be written as a quadrangle $\mathbf{q_r} = (x^q_{r1}, y^q_{r1}, x^q_{r2}, y^q_{r2}, x^q_{r3}, y^q_{r3}, x^q_{r4}, y^q_{r4})$. The relationship between $\mathbf{b}_r$ and $\mathbf{q_r}$ is formulated in Eq. 2, where $\mathbf{b}_r$ is the corresponding minimum enclosing horizontal rectangle of $\mathbf{q_r}$.

$$\begin{aligned} x^q_{r1} &= x_r - w_r/2, \quad & y^q_{r1} &= y_r - h_r/2, \\ x^q_{r2} &= x_r + w_r/2, \quad & y^q_{r2} &= y_r - h_r/2, \\ x^q_{r3} &= x_r + w_r/2, \quad & y^q_{r3} &= y_r + h_r/2, \\ x^q_{r4} &= x_r - w_r/2, \quad & y^q_{r4} &= y_r + h_r/2. \end{aligned} \tag{2}$$

TDB exploits the same prediction layers used in LAB and outputs text presence confidence c_r and offsets to each associated anchor. The predicted horizontal and quadrilateral offsets are formatted as $(\Delta x_r, \Delta y_r, \Delta w_r, \Delta h_r)$ and $(\Delta x^q_{r1}, \Delta y^q_{r1}, \Delta x^q_{r2}, \Delta y^q_{r2}, \Delta x^q_{r3}, \Delta y^q_{r3}, \Delta x^q_{r4}, \Delta y^q_{r4})$. Thus, a horizontal rectangle $\mathbf{b} = (x, y, w, h)$ and a quadrangle $\mathbf{q} = (x^q_1, y^q_1, x^q_2, y^q_2, x^q_3, y^q_3, x^q_4, y^q_4)$ text boundaries are detected with confidence c_r:

$$\begin{aligned} x &= x_r + w_r \Delta x_r, \quad & w &= w_r \exp\left(\Delta w_r\right), \\ y &= y_r + h_r \Delta y_r, \quad & h &= h_r \exp\left(\Delta h_r\right), \\ x^q_n &= x^q_{rn} + w_r \Delta x^q_{rn}, \quad & n &= 1,2,3,4, \\ y^q_n &= y^q_{rn} + h_r \Delta y^q_{rn}, \quad & n &= 1,2,3,4. \end{aligned} \tag{3}$$

During network training, given all detected boxes, we calculate overlaps between the minimum enclosing horizontal rectangles of quadrangles and that of ground truths, and match them when overlap exceeds given threshold. The predicted quadrangles are not used for matching due to inefficiency.

3.3. Feature Enhancement Block (FEB)

FEB is designed and operated on prediction layers before fed into TDB. As indicated in Figure 2, each prediction layer first undergoes a convolutional layer with same channels, then upsamples its resolution via a deconvolution operation (except the lowest layer), finally is merged with previous layer by element-wise sum. We append a convolutional layer on each merged map to reduce aliasing effect.

3.4. Label Generation

For each image with ground truths (quadrangles or horizontal rectangles), we generate both quadrangular and rectangular ground truths if it has only one. For a quadrangle $\mathbf{G}_q = (q_1, q_2, q_3, q_4) = (\widetilde{x}^q_1, \widetilde{y}^q_1, \widetilde{x}^q_2, \widetilde{y}^q_2, \widetilde{x}^q_3, \widetilde{y}^q_3, \widetilde{x}^q_4, \widetilde{y}^q_4)$, where (q_1, q_2, q_3, q_4) are the four vertices in clockwise order with q_1 the top-left one, its horizontal form, i.e., the minimum horizontal rectangle enclosing $\mathbf{G}_q$, is formatted as $\mathbf{G}_h = \left(\widetilde{x}^h_0, \widetilde{y}^h_0, \widetilde{w}^h_0, \widetilde{h}^h_0\right)$, where $\left(\widetilde{x}^h_0, \widetilde{y}^h_0\right)$ is the center, $\widetilde{w}^h_0$ and $\widetilde{h}^h_0$ are the width and height. Similarly, for each horizontal rectangle $\mathbf{G}_h = \left(\widetilde{x}^h_0, \widetilde{y}^h_0, \widetilde{w}^h_0, \widetilde{h}^h_0\right)$, its corresponding quadrangle is obtained as $\mathbf{G}_q = (q_1, q_2, q_3, q_4) = (\widetilde{x}^q_1, \widetilde{y}^q_1, \widetilde{x}^q_2, \widetilde{y}^q_2, \widetilde{x}^q_3, \widetilde{y}^q_3, \widetilde{x}^q_4, \widetilde{y}^q_4)$ by following Eq. 2.

3.5. Multi-task Loss Function

We define multi-task loss as follows:

$$\begin{aligned} L\left(x, c, l, g\right) = &\frac{1}{N_{\mathbf{LAB}}}\left(L_{conf\_L}\left(x, c_0\right) + \alpha L_{loc\_L}\left(x, l, g\right)\right) \\ &+ \frac{1}{N_{\mathbf{TDB}}}\left(L_{conf\_T}\left(x, c_r\right) + \alpha L_{loc\_T}\left(x, l, g\right)\right) \end{aligned} \tag{4}$$

For LAB, L_{loc_L} is smooth $L1$ loss operated on matched horizontal ground truths and horizontal refined anchors, while L_{conf_L} is 2-class softmax loss. For TDB, L_{loc_T} is smooth

Methods	Recall	Precision	F-measure	Time/s
SegLink [10]	0.731	0.768	0.75	**0.05**
WordSup [11]	0.77	0.793	0.782	0.52
RRPN [12]	0.73	0.82	0.77	0.30
RRD [13]	0.79	0.856	0.822	0.10
Lyu [14]	0.707	**0.941**	0.807	0.28
DMPNet [15]	0.682	0.732	0.706	-
SSTD [3]	0.73	0.80	0.77	0.13
EAST [2]	0.735	0.836	0.782	0.06
TextBoxes++ [1]	0.767	0.872	0.817	0.09
Our Proposed: LATD	**0.781**	0.878	**0.827**	0.10
R2CNN [16] *	0.797	0.856	0.825	2.25
PixelLink+VGG16_2s [17] *	**0.82**	0.855	0.837	0.33
RRPN [12] *	0.77	0.84	0.80	0.30
RRD [13] *	0.8	0.88	0.838	-
Lyu [14] *	0.797	0.895	0.843	1.00
TextSnake [18] *	0.804	0.849	0.826	0.91
Mask TextSpotter [19] *	0.812	0.858	0.834	0.21
DDR [4] *	0.800	0.820	0.810	0.90
EAST [2] *	0.783	0.833	0.807	**0.08**
TextBoxes++ [1] *	0.785	0.878	0.829	0.43
Our Proposed: LATD *	0.804	**0.899**	**0.849**	0.46

Table 1: *Performance comparison on IC15. * means multi-scale test.*

Methods	Recall	Precision	F-measure
Baseline A [20]	0.233	0.838	0.365
Baseline B [20]	0.107	0.897	0.191
Baseline C [20]	0.047	0.186	0.075
RRD [13] *	**0.57**	0.64	0.61
Yao [21] *	0.271	0.432	0.333
SSTD [3] *	0.31	0.46	0.37
EAST [2] *	0.324	0.504	0.395
TextBoxes++ [1]	0.560	0.558	0.559
TextBoxes++ [1] *	0.567	0.609	0.587
Our Proposed: LATD	0.50	0.69	0.58
Our Proposed: LATD *	0.51	**0.74**	**0.61**

Table 2: *Performance comparison on COCO-Text. * means multi-scale test.*

$L1$ loss applied on matched quadrilateral ground truths and regressed quadrangles. L_{conf_T} is the focal loss, where

$$L_{conf\_T}(x, c_r) = -\beta(1 - c_r)^{\gamma} \log(c_r) \tag{5}$$

Focal loss is used to alleviate the side effect where network training is dominated by easily classified backgrounds. Intuitively, β balances the importance of positive/negative examples, and $(1 - c_r)^{\gamma}$ automatically down-weights the contribution of easy examples during training and focuses the model on hard examples. More details please refer to [8]. In our experiments, α is set to 0.2 for quick convergence. β and γ are set to 0.25 and 2.0 respectively according to [8].

4. EXPERIMENTS

To evaluate LATD, we give detailed description of standard datasets, model training and inference, experimental implementation, and results with comparisons respectively.

4.1. Benchmark Datasets

SynthText [22]: SynthText contains 800,000 synthetic images. Each image has multiple texts overlaid on appropriate background regions sampled from natural images. These texts look realistic as the overlaying follows carefully set up configuration and a well-set learning algorithm.

Methods	Recall	Precision	F-measure
SSD [7]	0.74	0.88	0.81
TextBoxes [6]	0.74	0.88	0.81
TextBoxes++ [1]	0.74	0.88	0.81
RefineDet [9]	0.75	0.79	0.77
Our Proposed: LATD	**0.77**	**0.91**	**0.83**
SegLink [10] *	0.83	0.88	0.85
DDR[4] *	0.81	0.92	0.86
SSTD[3] *	0.86	0.89	0.88
WordSup[11] *	**0.88**	0.93	0.90
TextBoxes [6] *	0.83	0.89	0.86
TextBoxes++ [1] *	0.86	0.92	0.89
RefineDet [9] *	0.79	0.87	0.83
Our Proposed: LATD *	0.86	**0.93**	**0.90**

Table 3: *Performance comparison on IC13. * means multi-scale test.*

Component	Proposed Model		
Focal loss	✓		✓
The Learnable Anchor Branch (LAB)	✓	✓	
F-measure	0.827	0.823	0.818

Table 4: *Effects of various components on IC15 dataset.*

ICDAR 2015 Incidental Text (IC15) [23]: IC15 contains $1,000$ training and 500 test images captured by wearable cameras with low resolutions. Each image includes several oriented texts annotated by four vertices of quadrangle.

ICDAR 2013 Focused Scene Text (IC13) [24]: IC13 contains 229 training and 233 test images. Texts in these images are from sign boards, posters and other objects with axis-aligned bounding box annotations.

COCO-Text [20]: COCO-Text is the largest text detection dataset which comes from the MS COCO dataset. It contains $63,686$ images, where $43,686$ images are used for training, $10,000$ for validation, and $10,000$ for test. Although texts in this dataset are in arbitrary orientations, text regions are annotated in the form of axis-aligned bounding boxes.

4.2. Implementation Details

4.2.1. Training

LATD is optimized by Adam. Following TextBoxes++, all training images are augmented online with random crop and deformation. LATD is pre-trained on SynthText and fine-tuned on the standard benchmarks. During pre-training stage, images are resized to 384×384 and trained with learning rate 10^{-3} for 60k iterations. At fine-tune stage, images are firstly resized to 384×384 with learning rate 10^{-3}, then scaled up to 768×768 with learning rate decreased to 10^{-4}. A larger image size is used to achieve better detections for multi-scale texts. The number of iterations at finetune is decided by the sizes of benchmark datasets. All implementations are carried out on a PC with Titan Xp GPUs.

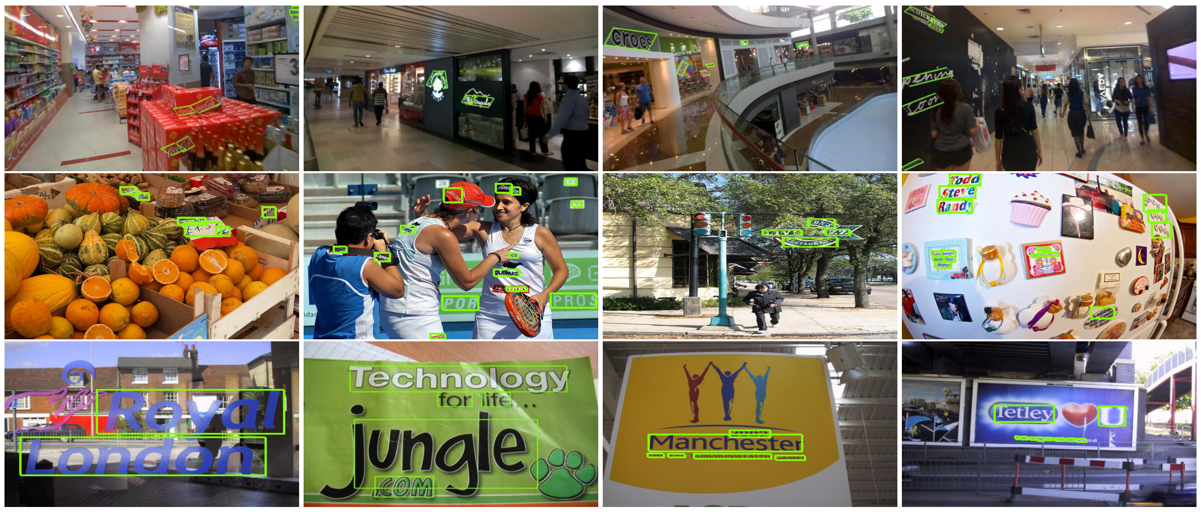


Fig. 3: Qualitative results of LATD on IC15 (row 1), COCO-Text (row 2) and IC13 (row 3) respectively. Best viewed in color.

4.2.2. Inference

In testing, images are resized to 768×768. Following [14, 2, 4], we evaluate our model with multi-scale inputs, i.e., $\{512 \times 512, 768 \times 768, 768 \times 1280, 1280 \times 1280\}$. TDB simultaneously outputs scores, quadrilateral and corresponding horizontal boundaries of texts. The outputs then undergo a two-step NMS to filter out redundant boxes. Since NMS operating on quadrilaterals is more time-consuming than that on horizontal rectangles, we firstly apply NMS on minimum horizontal rectangles with a higher IOU threshold, e.g., 0.5. This step is much less time-consuming and removes many irrelevant boxes. Then NMS on quadrangles is applied with a lower IOU threshold, e.g., 0.2. The two-step NMS is much faster than one-step NMS directly operated on quadrangles. After that, we get the final text detections.

4.3. Results on Oriented Text Benchmarks

We compare LATD with existing methods on two oriented text benchmarks, to verify its effectiveness to oriented texts.

4.3.1. ICDAR 2015 Incidental Text (IC15).

Quantitative results are given in Table 1. LATD outperforms all state-of-the-art results by a large margin in both single (0.827) and multi-scale (0.849) tests. To explore the gain between LATD and other regression based detectors, e.g., direct and indirect regression, we compare results among DDR, EAST and TextBoxes++. From Table 1, it is obvious that indirect regression based models (TextBoxes++ and ours) get better performance than direct regression based ones (DDR and EAST). As TextBoxes++ is the most related work to us, LATD improves TextBoxes++ by 2.0 percent (0.829 vs 0.849) with similar detection speed. The significant improvements demonstrate the effectiveness of the proposed learnable anchors for oriented scene text detection. From qualitative results shown in Figure 3, LATD exhibits more accurate location and lower rate of missed detections, which is consistent with the above analysis.

4.3.2. COCO-Text.

Performances of LATD and other competitive methods are listed in Table 2. Considering that COCO-Text is the largest and most challenging benchmark to date, LATD achieves the best performance with 0.61 in F-measure. Specifically, LATD improves TextBoxes++ by 2.1 percent at single-scale test, and 3 percent are further boosted at multi-scale test. Some qualitative detection results are shown in Figure 3.

4.4. Results on Horizontal Text Benchmarks

We further evaluate LATD on ICDAR 2013 Focused Scene Text (IC13) to assess its versatility for horizontal text detection. Experimental results are depicted in Table 3. LATD achieves highly competitive performance in both precision and recall among existing methods. In particular, we train RefineDet with same training procedure for fair comparison. As reported in Table 3, a straightforward adaption of RefineDet for text detection does not perform well as ours. LATD performs much better owing to the specially designed prediction layers, and the modified classification loss for text detection. Some qualitative detection results are shown in Figure 3.

4.5. Ablation Study

To better understand LATD, we execute controlled experiments on IC15. All experiments are under the same setting, except for specified changes in different comparisons.

4.5.1. *Focal loss.*

To demonstrate the effectiveness of focal loss, we apply standard softmax loss in TDB. Results in Table 4 indicates that focal loss boosts performance by 0.4 percent. We attribute it to the fact that focal loss discounts effect of easy backgrounds and focuses more attention on hard ones, therefore leads to a more robust scene text detector.

4.5.2. *The Refining Anchor Branch (LAB).*

To validate LAB, we redesign LATD by directly using fixed anchors instead of refined ones from LAB. Results in Table 4 show that F-measure is reduced to 0.818. This sharp decline (0.9 percent) demonstrates the deficiency of fixed anchors and the superiority of refined anchors for scene text detection.

5. CONCLUSION AND FUTURE WORK

We have presented LATD for oriented scene text detection. The proposed learnable anchors as well as focal loss obtain state-of-the-art performance, while still runs fast due to its single-shot nature. Extensive experiments basically validate our proposal. Our future work includes extending LATD to curved text detection and an end-to-end text spotting system.

Acknowledgments.

This work is supported by the Natural Science Foundation of China (No. 61772526) and the Strategic Priority Research Program of the Chinese Academy of Sciences (XDBS01070000).

6. REFERENCES

[1] Minghui Liao, "Textboxes++: A single-shot oriented scene text detector," *TIP*, 2018.

[2] Xinyu Zhou, "East: an efficient and accurate scene text detector," in *Proc. CVPR*, 2017, pp. 2642–2651.

[3] Pan He and Huang, "Single shot text detector with regional attention," in *ICCV*, 2017, vol. 6.

[4] Wenhao He, "Deep direct regression for multi-oriented scene text detection," *preprint arXiv:1703.08289*, 2017.

[5] Chuhui Xue, "Accurate scene text detection through border semantics awareness and bootstrapping," *arXiv preprint arXiv:1807.03547*, 2018.

[6] Minghui Liao, "Textboxes: A fast text detector with a single deep neural network.," in *AAAI*, 2017, pp. 4161–4167.

[7] Wei Liu, "Ssd: Single shot multibox detector," in *ECCV*, 2016, pp. 21–37.

[8] Tsung-Yi Lin, "Focal loss for dense object detection," *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[9] Shifeng Zhang, "Single-shot refinement neural network for object detection," *preprint arXiv:1711.06897*, 2018.

[10] Baoguang Shi, "Detecting oriented text in natural images by linking segments," *arXiv preprint arXiv:1703.06520*, 2017.

[11] Han Hu, "Wordsup: Exploiting word annotations for character based text detection," in *Proc. ICCV*, 2017.

[12] Jianqi Ma, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Transactions on Multimedia*, 2018.

[13] Minghui Liao, "Rotation-sensitive regression for oriented scene text detection," in *CVPR*, 2018, pp. 5909–5918.

[14] Pengyuan Lyu, "Multi-oriented scene text detection via corner localization and region segmentation," in *CVPR*, 2018, pp. 7553–7563.

[15] Yuliang Liu and Lianwen Jin, "Deep matching prior network: Toward tighter multi-oriented text detection," in *Proc. CVPR*, 2017, pp. 3454–3461.

[16] Yingying Jiang, "R2cnn: rotational region cnn for orientation robust scene text detection," *arXiv preprint arXiv:1706.09579*, 2017.

[17] Dan Deng, "Pixellink: Detecting scene text via instance segmentation," *arXiv preprint arXiv:1801.01315*, 2018.

[18] Shangbang Long, "Textsnake: A flexible representation for detecting text of arbitrary shapes," *arXiv preprint arXiv:1807.01544*, 2018.

[19] Pengyuan Lyu, "Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," *arXiv preprint arXiv:1807.02242*, 2018.

[20] Andreas Veit, "Coco-text: Dataset and benchmark for text detection and recognition in natural images," *arXiv preprint arXiv:1601.07140*, 2016.

[21] Cong Yao, "Detecting texts of arbitrary orientations in natural images," in *CVPR*. IEEE, 2012, pp. 1083–1090.

[22] Ankush Gupta, "Synthetic data for text localisation in natural images," in *CVPR*, 2016, pp. 2315–2324.

[23] Dimosthenis Karatzas, "Icdar 2015 competition on robust reading," in *ICDAR*. IEEE, 2015, pp. 1156–1160.

[24] Dimosthenis Karatzas, "Icdar 2013 robust reading competition," in *ICDAR*. IEEE, 2013, pp. 1484–1493.