Pattern Recognition Letters 125 (2019) 228–234

Contents lists available at ScienceDirect

Pattern Recognition Letters

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

Pyrboxes: An efficient multi-scale scene text detector with feature pyramids☆

Fenfen Sheng[a,b], Zhineng Chen[a,*], Wei Zhang[c], Bo Xu[a]

[a] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[b] University of Chinese Academy of Sciences, Beijing 100190, China
[c] JD AI Research, Beijing 100101, China

ARTICLE INFO

Article history:
Received 23 April 2018
Available online 30 April 2019

MSC:
41A05
41A10
65D05
65D17

Keywords:
Scene text detection
Multi-scale text detection
Grouped pyramid module
Efficient and effective

ABSTRACT

Scene text detection has attracted many researches due to its importance to various applications. However, current approaches could not keep a good balance between accuracy and speed, i.e., a high-performance accuracy but with a low processing speed, or vice-versa. In this paper, we propose a novel model, named PyrBoxes, for efficient and effective multi-scale scene text detection. PyrBoxes consists of an SSD-based backbone that utilizes deep layers with strong semantics to detect texts in various sizes, and a proposed grouped pyramid module that leverages basic layers to append detailed locations into detection. Most existing detectors discard features from the basic layers due to the efficiency issue. We argue these layers contain fine-grained information, which is complementary to high-level semantics. Based on this, the grouped pyramid module combines the basic layers recursively into a detection layer via a top-down partition and a bottom-up group. Extensive experiments on both horizontal and oriented benchmarks, including ICDAR2013 Focused Scene Text, ICDAR2015 Incidental Text and COCO-Text, demonstrate that PyrBoxes achieves state-of-the-art or highly competitive performance compared with baselines, while runs significantly faster at inference. Furthermore, by experimenting on another ChiTVText dataset, PyrBoxes shows great generality to Chinese and long text lines. By visualizing some qualitative results, as expected, PyrBoxes provides more accurate locations and reduces the rate of missed detections, especially for small-sized texts.

1. Introduction

Scene text detection, which aims to locate texts in natural images, has drawn increasing interests from both artificial intelligence and computer vision communities. The popularity is mainly due to its essential role in extracting rich semantic information that is highly relevant to scenes or objects. Therefore it has been applied to a wide range of applications, such as geo-location, caption reading and image interpretation.

Although extensive studies have been carried out in the past few years, scene text detection is still challenging due to several difficulties, e.g., low resolution (small texts), low visual quality, complex deformations and cluttered background. Moreover, each natural image tends to include multiple text regions in various scales. Take ICDAR2013 dataset for example, each image contains four text regions in varying sizes on average.

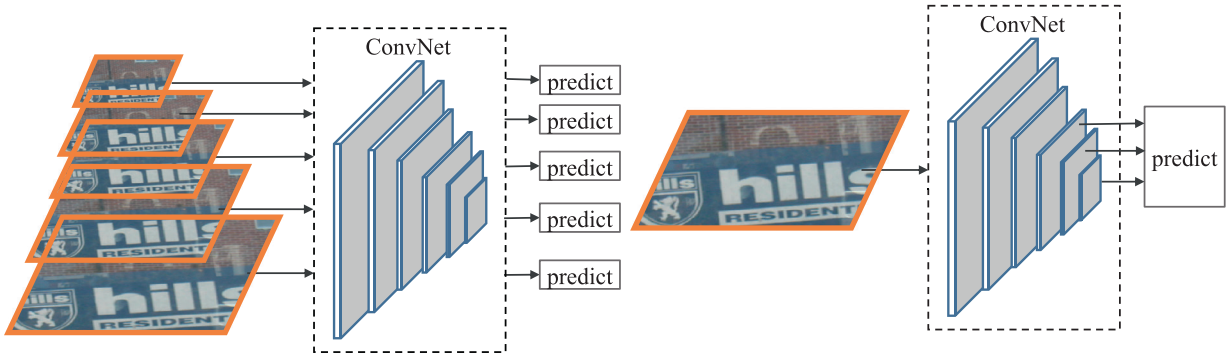
To detect texts in different scales, multi-scale inputs (image pyramid) form the basis of a standard solution. As shown in Fig. 1(a), image pyramid resizes each input image to several different sizes and runs multi-passes of the network for all-scale inputs. Image pyramid could capture text features across a wide range of sizes and accordingly boosts detection performance as shown in [5,21]. However, it increases the processing time considerably, e.g., by 8 times in [10], making it impractical for real applications, especially for mobile platforms with limited computation capability and power. Image pyramid, therefore, is used only at inference rather at training stage, which further leads to an inconsistency between the training and inference phases.

To accelerate speed for multi-scale text detection, researches turn to exploit in-network layers of the deep convolutional network (deep ConvNet). As shown in Fig. 1(b), a deep ConvNet computes a feature hierarchy layer by layer and forms an inherent multi-scale shape by using the subsampling operation. However, the feature pyramid introduces a large semantic gap between different depths. It will harm the feature representation when low-level and high-level features are directly used. To avoid this problem, existing methods like TextBoxes [10] discard reusing the

☆ **Conflict of interest.** ia.ac.cn, jd.com, ucas.ac.cn.

* Corresponding author.
E-mail address: zhineng.chen@ia.ac.cn (Z. Chen).

https://doi.org/10.1016/j.patrec.2019.04.022
0167-8655/



(a) Image pyramid (b) Feature pyramid

Fig. 1. (a) Image pyramid involves multi-passes of the network for multi-scale inputs. It is of high accuracy but low speed. (b) Feature pyramid runs the network only once and uses multiple deep layers to make predictions. It is of high speed but low accuracy.

lower-level features but build a pyramid starting from higher ones in the network, e.g., *conv*4_3 in VGG16 network [18]. Note that *conv*4_3 represents the third convolutional layer (the second number '3') in the fourth convolutional block (the first number '4') of VGG16, the same below. These methods accelerate inference time, but perform poor due to the lack of fine-grained information, especially for small-size text detection.

The goal of this paper is to detect multi-scale scene texts with proper speed and accuracy. To achieve this goal, we propose a novel model, named PyrBoxes, that involves an SSD-based backbone and a grouped pyramid module. Inspired by SSD [13], a generic object detector, PyrBoxes adds several convolutional layers (added layers) with size decreased progressively to the end of VGG16 network. The added layers are leveraged directly for multi-scale text detection due to their strong semantics. Since resolutions of the added layers are too small to contain location details, we propose the grouped pyramid module for more accurate text detection. We argue that layers in the base network (basic layers) undergo only a few times of downsamplings and transformations, e.g., convolution and activation operations, therefore their corresponding receptive fields are big enough to contain sufficient location information. However, among the basic layers, the shallower layers have large feature maps but low-level semantics, while the deeper layers behave oppositely [23]. It is not advisable to apply the basic layers to make predictions directly. To maximize fine-grained information with strong semantics, the grouped pyramid module combines the basic layers recursively into a single layer via a top-down partition and a bottom-up group. The resulted layer companying with the above added layers make the multi-scale text detections with a more robust feature representation and a faster speed.

We conduct extensive experiments on standard benchmarks, including the horizontal (ICDAR2013 Focused Scene Text) and the oriented (ICDAR2015 Incidental Scene Text and COCO-Text) text datasets, and our newly constructed Chinese text dataset ChiTV-Text. Evaluations demonstrate that PyrBoxes achieves the new state-of-the-art or highly competitive performance, while runs at least 7x faster than baselines. Furthermore, by analyzing results on texts with different scales and qualitative detections, we find that PyrBoxes exhibits more accurate locations and a lower rate of missed detections, especially for small-size texts. Our code will be made publicly available soon.

2. Related work

Scene text detection has received significant attention due to the use of deep ConvNets. Quite a few systems have been reported focusing on different aspects, e.g., accuracy and speed.

For higher accuracy, TextFlow [19] utilizes the minimum cost flow network, with character candidates detected by cascade boosting, to solve the error accumulation problem in traditional multi-step approaches. This model achieves good performance but runs at 1.4s per image. Wenhao [5] proposes a direct regression-based method for multi-oriented text detection. By predicting the offsets from a given point rather than a default anchor, it achieves F-measure of 86% on ICDAR2013 benchmark with 0.9s per image.

For faster processing speed, Fast TextBoxes [10] modifies SSD by using irregular convolutional kernels and long anchors. It accelerates inference time with 0.09s per image, but sacrifices a lot in accuracy. To improve detection performance, TextBoxes uses the image pyramid rather than one-scale input. Though achieving a big performance boost, TextBoxes needs 0.73 s to detect one image. FCRN [2], modified from YOLO [16], achieves high accuracy with multi-scale inputs but a slow detection speed at 1.27 s per image. With only a single-scale input, FCRN accelerates inference speed by several times but also with a noticeable performance drop.

Our proposed PyrBoxes leverages in-network pyramid features, but has a significant difference with FPN [11], a state-of-the-art generic object detector. FPN detects objects by first upsampling feature maps one by one then making detections from each of the resulting layers at all levels. Though with performance improvements, FPN brings a heavy burden on both speed (0.25 s each image, carried out on M40 GPU) and memory. To trade off between accuracy and speed, we propose the novel grouped pyramid module to recursively group basic layers into a single layer that contains both fine-grained information and strong semantics. Combining the grouped layer with the added layers, PyrBoxes makes more accurate and efficient predictions with only a single-scale input.

3. Methodology

The architecture of PyrBoxes is depicted in Fig. 2. The whole network consists of an SSD-based convolutional backbone and a grouped pyramid module that includes a top-down partition and a bottom-up group. In the following sections, we will give details of these components.

3.1. SSD-based convolutional backbone

Following SSD, PyrBoxes inherits the popular VGG-16 network by keeping the layers from *conv*1 to *conv*5, converting the last two fully-connected layers into convolutional layers (*conv*6 and *conv*7), truncating the classification layers (*fc*-1000 and softmax), and adding a series of convolutional layers (*conv*8 to *conv*11) to the end with sizes decreased progressively. The added layers (*conv*7

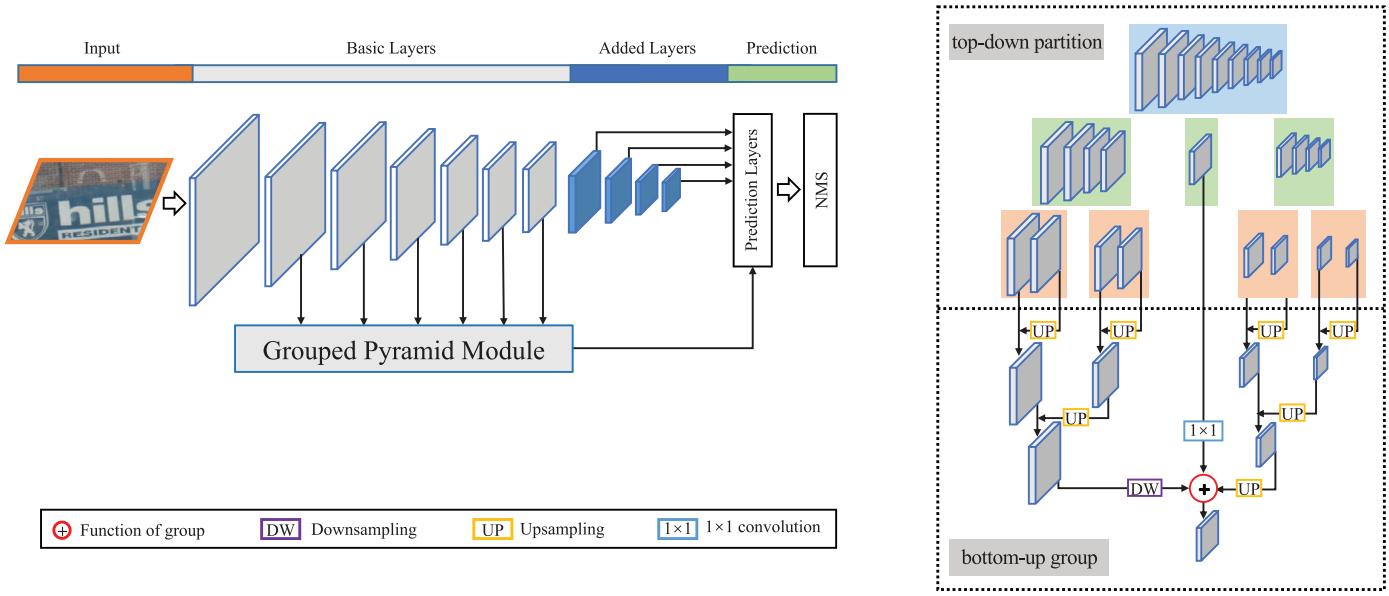


Fig. 2. (left) Network architecture of PyrBoxes. (right) Architecture of the Grouped Pyramid Module, including the top-down partition and the bottom-up group.

to *conv*11) with different receptive fields are leveraged as prediction layers, as the blue part shown in Fig. 2. For each prediction layer, each feature map location simultaneously outputs the text presence scores and bounding boxes. The output boxes include oriented quadrangles and minimum horizontal rectangles containing the corresponding oriented quadrangles. PyrBoxes achieves it by predicting the regression of offsets from a number of default anchors. In this paper, we only use 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 larger 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 all texts, as TextBoxes [10] did.

More precisely, for each horizontal anchor $\mathbf{b}_0 = (x_0, y_0, w_0, h_0)$, where $(x_0,\ y_0)$ means the center point and w_0 and h_0 are the width and height, it can be written as a quadrangle $\mathbf{q_0} = (x^q_{01}, y^q_{01}, x^q_{02}, y^q_{02}, x^q_{03}, y^q_{03}, x^q_{04}, y^q_{04})$. The relationship between $\mathbf{b}_0$ and $\mathbf{q_0}$ is formulated in Eq. (1), where $\mathbf{b}_0$ is the corresponding minimum enclosing the horizontal rectangle of $\mathbf{q_0}$.

$$\begin{aligned} x^q_{01} &= x_0 - w_0/2, \quad y^q_{01} = y_0 - h_0/2, \\ x^q_{02} &= x_0 + w_0/2, \quad y^q_{02} = y_0 - h_0/2, \\ x^q_{03} &= x_0 + w_0/2, \quad y^q_{03} = y_0 + h_0/2, \\ x^q_{04} &= x_0 - w_0/2, \quad y^q_{04} = y_0 + h_0/2. \end{aligned} \tag{1}$$

The predicted horizontal and quadrilateral offsets are formulated as $(\Delta x,\ \Delta y,\ \Delta w,\ \Delta h)$ and $(\Delta x^q_1, \Delta y^q_1, \Delta x^q_2, \Delta y^q_2, \Delta x^q_3, \Delta y^q_3, \Delta x^q_4, \Delta y^q_4)$ respectively. 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, as formatted in Eq. (2).

$$\begin{aligned} x &= x_0 + w_0 \Delta x, \quad w = w_0 \exp(\Delta w), \\ y &= y_0 + h_0 \Delta y, \quad h = h_0 \exp(\Delta h), \\ x^q_n &= x^q_{0n} + w_0 \Delta x^q_n, \quad n = 1, 2, 3, 4 \\ y^q_n &= y^q_{0n} + h_0 \Delta y^q_n, \quad n = 1, 2, 3, 4. \end{aligned} \tag{2}$$

During network training, given all detected boxes in an image, PyrBoxes follows the matching scheme in SSD to match minimum bounding horizontal rectangles to rectangle ground-truths according to their overlaps. The predicted quadrangles are not used for matching due to inefficiency.

3.2. Grouped pyramid module

Apart from the added layers mentioned above, the grouped pyramid module is proposed to further leverage the basic layers of ConvNet for prediction, as the gray part shown in Fig. 2. It consists of a top-down partition and a bottom-up group.

3.2.1. Top-down partition

As shown in Fig. 2, the top-down partition iteratively divides the basic layers into different blocks and finally forms a pyramid with an increasing number of blocks from the top-down direction. Suppose there are n basic layers. At the first level of the pyramid ($l = 0$), all layers are put into one block. At $l = 1$, if n is even, layers will be divided equally into two sub-blocks, each of which contains $n/2$ layers. If n is odd, the middle layer will be put into a single block, and layers on either side are divided into two sub-blocks respectively. Similarly, layers in sub-blocks will be further divided into smaller blocks until the number of layers in the latest block is less than three. After the top-down partition, PyrBoxes obtains a pyramid with $\max(1, \lfloor \log_2 n \rfloor)$ levels.

Specifically, for the VGG16-based network, the basic layers include $\{conv2\_2, conv3\_3, conv4\_3, conv5\_3, conv6\}$ and accordingly form a $2 - level$ pyramid. Considering the large memory footprint of *conv*1, PyrBoxes does not include it in the pyramid.

3.2.2. Bottom-up group

Based on the above pyramid, the bottom-up group combines layers recursively via a bottom-up way, and finally outputs a single layer by integrating all basic layers. After the top-down partition, at the bottom level of the pyramid, each block contains two or three layers with different resolutions. If there are two layers, one with lower resolution will be first upsampled, i.e., 'UP' in Fig. 2, via a deconvolution operation, then combined with the other one via an element-wise summarization. If there are three layers, apart from the one with the lowest resolution upsampled, the one with the highest resolution will be downsampled, i.e., 'DW' in Fig. 2, via a convolution operation. Then these three layers are combined into

one layer. Similarly, for upper levels, layers in each block continue to be combined and finally form a single layer for prediction.

Specifically, for the VGG16-based network, $\{conv2\_2, conv3\_3\}$ and $\{conv5\_3, conv6\}$ are firstly grouped into $group23$ and $group56$, then $\{group23, group56, conv4\_3\}$ are grouped into the final combined layer $group23456$. PyrBoxes uses $group23456$ layer together with the added layers (conv7 to conv11) as the prediction layers.

3.3. Label generation

For each image, we generate both quadrangular and rectangular ground truths if it has only one of them. For a quadrangle $\mathbf{G}_q = (q_1, q_2, q_3, q_4) = (\tilde{x}_1^q, \tilde{y}_1^q, \tilde{x}_2^q, \tilde{y}_2^q, \tilde{x}_3^q, \tilde{y}_3^q, \tilde{x}_4^q, \tilde{y}_4^q)$, where (q_1, q_2, q_3, q_4) are the four vertices in clockwise order with q_1 being the top-left one, its minimum horizontal rectangle enclosing $\mathbf{G}_q$ is formatted as $\mathbf{G}_h = (\tilde{x}_0^h, \tilde{y}_0^h, \tilde{w}_0^h, \tilde{h}_0^h)$, where $(\tilde{x}_0^h, \tilde{y}_0^h)$ is the center and $\tilde{w}_0^h$ and $\tilde{h}_0^h$ are the width and height. Similarly, for each horizontal rectangle $\mathbf{G}_h = (\tilde{x}_0^h, \tilde{y}_0^h, \tilde{w}_0^h, \tilde{h}_0^h)$, its corresponding quadrangle is obtained as $\mathbf{G}_q = (q_1, q_2, q_3, q_4) = (\tilde{x}_1^q, \tilde{y}_1^q, \tilde{x}_2^q, \tilde{y}_2^q, \tilde{x}_3^q, \tilde{y}_3^q, \tilde{x}_4^q, \tilde{y}_4^q)$ by following Eq. (1).

3.4. Multi-task loss function

PyrBoxes simultaneously fulfills the classification task and regression task. Its multi-task loss is represented as:

$$L(x, c, l, g) = \frac{1}{N}\left(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)\right) \tag{3}$$

L_{loc} is the smooth $L1$ loss [1] operated on the matched quadrilateral ground truths (g) and regressed text quadrangles (l). L_{conf} is a standard 2-class softmax loss. N is the number of matched anchors. If $N = 0$, we set the loss to 0. The balance between these two losses is controlled by the parameter α. We set $\alpha = 1$ in our experiments.

4. Experiments

To evaluate PyrBoxes, we conduct extensive experiments on both horizontal and oriented benchmarks widely used in the literature. We will give a detailed description of these datasets for model training and inference, experimental implementation, results with comparisons, and ablation study respectively.

4.1. Datasets

SynthText [2] 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 configurations and a well-set learning algorithm.

ICDAR2013 Focused Scene Text (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.

ICDAR 2015 Incidental Text (IC15) contains 1000 training and 500 test images captured by wearable cameras with relatively low resolutions. Each image includes several oriented texts annotated by four vertices of the quadrangles.

COCO-Text [20] is the largest text detection dataset which comes from the MS COCO dataset. It contains 63686 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.

ChiTVText is our newly constructed dataset for the task of Chinese detection. We firstly collect over one hundred Chinese news videos from 59 TV programs. Then we sample keyframes evenly from these videos and remove duplicate ones. We label the remaining images with axis-aligned rectangles. Texts in ChiTVText are annotated with line-level boxes, which therefore are longer. Finally, we build ChiTVText with 5454 training and 621 test images. As depicted in Fig. 3, ChiTVText mainly contains superimposed captions and complex scene texts.

4.2. Implementation details

4.2.1. Training

PyrBoxes is optimized by SGD with back-propagation [8]. Momentum and weight decay are set to 0.9 and 5×10^{-4} respectively. Learning rate is initialized to 10^{-3} and decayed to 10^{-4} after 40k iterations. Following TextBoxes, all training images are augmented online with random crop and deformation, and lastly resized to 300×300. We firstly pre-train PyrBoxes on SynthText for 120k iterations, then finetune it on each of benchmarks. The number of iterations at finetuning is decided by the sizes of the benchmarks. Note that for ChiTVText, we only train PyrBoxes on ChiTVText training set from scratch for 120k iterations. All implementations are carried out on a PC with one Nvidia Titan X GPU.

4.2.2. Inference

For each test image, PyrBoxes simultaneously outputs text presence scores, quadrilateral and corresponding horizontal boundaries of texts. The outputs then undergo a two-step NMS process 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 the NMS on quadrangles is applied among remaining candidate boxes 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. Experiments on horizontal text benchmarks

We firstly evaluate PyrBoxes on horizontal text datasets, i.e., IC13 datasets and ChiTVText, to demonstrate its effectiveness.

4.3.1. ICDAR 2013 focused scene text (IC13)

Experimental results are depicted in Table 1. Compared to existing methods, PyrBoxes achieves highly competitive performance, while requires merely 0.11 s per image at inference. Specifically, RRPN and FEN obtain higher accuracies but with 3× or 9× slower than ours. SegLink accelerates the inference time but accompanied with performance reduction. Moreover, as closely related works to ours, Fast-TextBoxes obtains faster speed but lower accuracy without leveraging the basic layers, while TextBoxes uses image pyramid to improve accuracy but is much time-consuming (0.73s per image). PyrBoxes achieves a good trade-off between accuracy and speed thanks to the proposed grouped pyramid module. Thus, PyrBoxes is appealing when taken into account both accuracy and speed, e.g., mobile platforms or wearable devices. Qualitative detections including comparisons and failure cases are shown in Fig. 3. As shown, PyrBoxes exhibits more accurate locations and a lower rate of missed detections than TextBoxes++, which is consistent with the above analysis. However, PyrBoxes performs poor on some digits and large-size texts, and outputs some detected boxes covering background regions along with the text regions. We attribute it to the use of fixed anchors, where the scales and positions of anchors could not be adjusted during network training. As scene texts tend to have a large variation in sizes, aspect ratios and orientations, fixed anchors are insufficient to cover diverse text patterns and thus lead to degenerate models.

Fig. 3. Qualitative results of PyrBoxes on IC13 (column 1), ChiTVText (column 2), IC15 (column 3), COCO-Text (column 4) and failure detections (column 5). The last row of column 1 ∼ 4 are detection results from TextBoxes++. For column 1 ∼ 4, the green boxes are detections; For column 5, the red boxes are detections and green boxes are ground truths. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Performance comparison on IC13 dataset. Precision (P), Recall (R), F-measure (F), and inference time are reported.

Methods	IC13 Eval			DetEval			Time/s
	P	R	F	P	R	F	
TextFlow [19]	0.85	0.76	0.80	–	–	–	1.40
FCRNall+multi-filt [2]	–	–	–	0.92	0.76	0.83	1.27
Text-Block FCN [25]	0.88	0.78	0.83	–	–	–	–
SSD [13]	0.80	0.60	0.68	0.80	0.60	0.69	0.10
Text-CNN [4]	0.93	0.73	0.82	–	–	–	–
DDR [5]	0.92	0.81	0.86	–	–	–	0.90
RRPN [15]	**0.95**	0.88	**0.91**	–	–	-	0.30
SegLink [17]	0.87	0.83	0.85	–	–	–	0.05
FEN [24]	0.93	**0.89**	0.91	0.93	**0.89**	**0.91**	0.90
Fast TextBoxes [10]	0.86	0.74	0.80	0.88	0.74	0.81	**0.09**
TextBoxes [10]	0.88	0.83	0.85	0.89	0.83	0.86	0.73
Our Proposed: PyrBoxes	0.91	0.85	0.88	**0.93**	0.86	0.89	0.11

Table 2
Performance comparison on ChiTVText dataset. Precision (P), Recall (R), F-measure (F), and inference time are reported.

Methods	IC13 Eval			DetEval			Time/s
	P	R	F	P	R	F	
Baidu API	0.79	0.59	0.68	0.79	0.64	0.71	–
SSD [13]	0.83	0.82	0.83	0.82	0.82	0.82	0.10
TextBoxes [10]	0.85	0.86	0.85	0.85	0.86	0.85	0.73
Our Proposed: PyrBoxes	**0.89**	**0.89**	**0.89**	**0.90**	**0.89**	**0.89**	0.11

4.3.2. ChiTVText dataset

Different from English scene texts, Chinese texts tend to be longer and more complex. To evaluate the generality of PyrBoxes, we perform experiments on ChiTVText dataset. In order to have a strict quantitative comparison, we also train SSD and TextBoxes with ChiTVText training set, following training strategies provided in their papers. Besides, Baidu provides a text detector API[1] to the public, which could be treated as a commercial baseline. Results listed in Table 2 indicate that PyrBoxes achieves the best performance among these methods, while still runs fast. Some qualitative detection are shown in Fig. 3. Based on the above analysis, PyrBoxes shows excellent generalization to different domains and longer text lines without additional considerations on training strategy.

4.4. Experiments on oriented text benchmarks

We further evaluate PyrBoxes on two oriented text datasets to assess its versatility for arbitrary oriented text detection.

[1] https://cloud.baidu.com/product/ocr/general.

Table 3
Performance comparison on IC15 dataset. Precision (P), Recall (R), F-measure (F), and inference time are reported. * means multi-scale test.

Methods	IC15 Eval			Time/s
	P	R	F	
SegLink [17]	0.768	0.731	0.750	**0.05**
WordSup [6]	0.793	0.770	0.782	0.52
DMPNet [14]	0.732	0.682	0.706	-
SSTD [3]	0.800	0.730	0.770	0.13
DDR [5] *	0.820	**0.800**	0.810	0.90
EAST [26]	0.836	0.735	0.782	0.06
EAST [26] *	0.833	0.783	0.800	0.08
R2CNN [7] *	0.856	0.797	0.825	2.25
RRPN [15]	0.820	0.730	0.770	0.30
RRPN [15] *	0.840	0.770	0.800	0.30
TextBoxes++ [9]	0.872	0.767	0.817	0.09
TextBoxes++ [9] *	**0.878**	0.785	0.829	0.43
Our Proposed: PyrBoxes	0.875	0.794	**0.832**	0.12

Table 4
Performance comparison on COCO-Text. Precision (P), Recall (R), F-measure (F), and inference time are reported. * means multi-scale test.

Methods	COCO-Text Eval			Time/s
	P	R	F	
Baseline A [20]	0.838	0.233	0.365	–
Baseline B [20]	0.897	0.107	0.191	–
Baseline C [20]	0.186	0.047	0.075	–
Yao [22] *	0.432	0.271	0.333	7.20
SSTD [3] *	0.460	0.310	0.370	0.13
EAST [26] *	0.504	0.324	0.395	0.08
TextBoxes++ [9]	0.558	0.560	0.559	0.09
TextBoxes++ [9] *	0.609	0.567	0.587	0.43
Our Proposed: PyrBoxes	0.734	0.508	0.601	0.12

4.4.1. ICDAR 2015 incidental text (IC15)

Quantitative results following the standard evaluation protocol is given in Table 3. PyrBoxes outperforms state-of-the-art results with a comparatively higher speed. Specifically, PyrBoxes achieves a F-measure of 0.832 and surpasses TextBoxes++ by 0.3 percent, while still 3x faster than it. With the single-scale input, TextBoxes++ takes only 0.09s per image but with a lower accuracy. With comparative precisions, PyrBoxes achieves a higher F-measure due to a higher recall. Some qualitative results are shown in Fig. 3.

4.4.2. COCO-Text

Performances of PyrBoxes and other competitive methods are listed in Table 4. Considering that COCO-Text is the largest and most challenging benchmark to date, PyrBoxes achieves the best performance with 0.601 in F-measure. Specifically, PyrBoxes improves multi-scale TextBoxes++ by 1.4 percent. Some qualitative detection results are shown in Fig. 3. As depicted, our model is robust in detecting oriented texts with a large variation in positions and scales.

4.5. Ablation study

To better understand PyrBoxes, we execute controlled experiments on IC13 dataset. All experiments are under the same setting, except for the specified changes in different comparisons.

4.5.1. Performance on different text scales

From Tables 1 to 4, we find that PyrBoxes achieves a higher performance mainly due to a higher recall. To better explain this observation, we investigate PyrBoxes on different text scales. Specifically, we divide IC13 dataset into three parts: small (area $< 32^2$), medium ($32^2 <$ area $< 96^2$) and large (area $> 96^2$) according to the area definition in Microsoft COCO [12]. We list recall values of Fast TextBoxes, TextBoxes and PyrBoxes in Table 5 and present representative visualization in Fig. 3. As expected, PyrBoxes achieves the highest recall on all text scales, and localizes texts with a lower missed rate, especially for small-size texts.

Table 5
Recall at different text scales. Small (S, area $< 32^2$), Medium (M, $32^2 <$ area $< 96^2$) and Large (L, area $> 96^2$) are reported.

Methods	IC13 Eval			DetEval		
	S	M	L	S	M	L
Fast TextBoxes	0.37	0.79	0.81	0.37	0.79	0.81
TextBoxes	0.62	0.85	0.86	0.62	0.86	0.87
Our Proposed: PyrBoxes	**0.66**	**0.87**	**0.89**	**0.66**	**0.87**	**0.89**

Table 6
Performance of different grouped pyramid module on ICDAR2013 dataset. Inference time and F-measure are reported.

Module	IC13 Eval F-measure	DetEval F-measure	Time/s
Group6	0.79	0.80	**0.094**
Group56_down	0.81	0.81	0.097
Group56_up	0.82	0.83	0.098
Group456_up	0.84	0.84	0.109
Group456_up_grad	0.86	0.87	0.116
Group456_middle	0.87	0.88	0.102
Group3456	**0.88**	**0.89**	0.105
Group23456	0.86	0.87	0.138

4.5.2. How to group layers in the same block

For the bottom-up group, apart from the way described in Section 3.2.2, we also test other ways to group layers in the same block. For a block with two layers, taking $\{conv5\_3, conv6\}$ for example, Group56_up first upsamples $conv6$ then sums it with $conv5\_3$. Group56_down downsamples $conv5$ then adds it with $conv6$. As indicated in Table 6, Group56_up achieves better performance than Group56_down with merely a little speed reduction. We attribute it to the fact that the upsampling introduces a larger resolution and more location details than downsampling. For a block with three layers, taking $\{conv4\_3, conv5\_3, conv6\}$ for example, Group456_up first upsamples $conv6$ and $conv5\_3$ to $conv6\_up$ and $conv5\_3\_up$ with different strides, then sums these two layers with $conv4\_3$. Group456_grad_up upsamples $conv6$ to $conv6\_up$ then does addition with $conv5\_3$. The summed layer is upsampled again then made element-wise sum with $conv4\_3$. Group 456_middle downsamples $conv4\_3$ to $conv4\_down$, upsamples $conv6$ to $conv6\_up$, then sums the two layers with $conv5\_3$ together. Table 6 shows that Group 456_middle achieves the best result, which can be interpreted as it obtains a better balance between resolution and semantics. Accordingly, in our final model, we choose the *"up"* strategy in blocks with two layers, and the *"middle"* strategy in blocks with three layers.

4.5.3. Which basic layers are useful

Generally, among the basic layers, the deeper layers have strong semantics but with coarse features, while the shallower layers have large resolutions but with more noises. We add shallow layers gradually into the grouped pyramid module to obtain the best combination. Apart from the Group56_up and Group456_middle mentioned above, we also test Group6, Group3456 and Group23456 that group $\{conv6\}$, $\{conv3\_3, conv4\_3, conv5\_3, conv6\}$ and $\{conv2\_2, conv3\_3, conv4\_3, conv5\_3, conv6\}$ respectively. As shown in Table 6, with more shallow layers added, the performance increases at first then decreases. We guess that $conv2\_2$ layer contains more noises than valuable location information.

Table 7
Performance of different group modes. F-measure on ICDAR2013 dataset are reported.

Group Mode	IC13 Eval F-measure	DetEval F-measure
element-wise sum	**0.88**	**0.89**
element-wise max	0.87	0.88

Besides, incorporating more basic layers also leads to heavier computational cost. In view of both accuracy and speed, we choose Group3456 in our final model.

4.5.4. Which combination modes are the best

In the bottom-up group, we test both the element-wise sum and the element-wise max in the same block, to choose the best group mode. Results in Table 7 show that the element-wise sum achieves a better performance, which indicates its strong ability of combining features among various layers. We choose the element-wise sum in our final model.

5. Conclusion

We have presented PyrBoxes, a novel model for multi-scale scene text detection. Establishing the grouped pyramid module within the SSD-based backbone obtains the state-of-the-art performance, while runs faster at inference due to the use of a single scale input. Extensive experiments conducted on several benchmarks basically validate our proposal. At the moment, PyrBoxes is focused on the bounding box outputs, and one of our future plans is to extend it to output polygon shape and test on Total-Text or CTW1500 datasets.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under grant no. 61772526 and Beijing Science and Technology Program under grant no. Z171100002217015.

References

[1] R. Girshick, Fast r-cnn, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
[2] 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.
[3] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, X. Li, Single shot text detector with regional attention, in: The IEEE International Conference on Computer Vision (ICCV), 6, 2017.
[4] T. He, W. Huang, Y. Qiao, J. Yao, Text-attentional convolutional neural network for scene text detection, IEEE Trans. Image Process. 25 (6) (2016) 2529–2541.
[5] W. He, X.-Y. Zhang, F. Yin, C.-L. Liu, Deep direct regression for multi-oriented scene text detection, (2017). arXiv:1703.08289.
[6] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, E. Ding, Wordsup: Exploiting word annotations for character based text detection, in: Proc. ICCV, 2017.
[7] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, Z. Luo, R2cnn: Rotational region cnn for orientation robust scene text detection, (2017). arXiv:1706.09579.
[8] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, Neural Comput. 1 (4) (1989) 541–551.
[9] M. Liao, B. Shi, X. Bai, Textboxes++: a single-shot oriented scene text detector, IEEE Trans. Image Process. 27 (8) (2018) 3676–3690.
[10] 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.
[11] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie, Feature pyramid networks for object detection,(2016). arXiv:1612.03144.
[12] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft coco: Common objects in context, in: European conference on computer vision, Springer, 2014, pp. 740–755.
[13] 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.
[14] Y. Liu, L. Jin, Deep matching prior network: toward tighter multi-oriented text detection, in: Proc. CVPR, 2017, pp. 3454–3461.
[15] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, X. Xue, Arbitrary-oriented scene text detection via rotation proposals, IEEE Trans. Multimedia (2018).
[16] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: CVPR, 2016, pp. 779–788.
[17] B. Shi, X. Bai, S. Belongie, Detecting oriented text in natural images by linking segments, in: Proc. CVPR, 3, 2017.
[18] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, (2014). arXiv:1409.1556.
[19] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, C. Lim Tan, Text flow: A unified text detection system in natural scene images, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4651–4659.
[20] A. Veit, T. Matera, L. Neumann, J. Matas, S. Belongie, Coco-text: dataset and benchmark for text detection and recognition in natural images, (2016). arXiv:1802.03897.
[21] Y. Wei, W. Shen, D. Zeng, L. Ye, Z. Zhang, Multi-oriented text detection from natural scene images based on a cnn and pruning non-adjacent graph edges, Signal Process. Image Commun. 64 (2018) 89–98.
[22] C. Yao, X. Bai, W. Liu, Y. Ma, Z. Tu, Detecting texts of arbitrary orientations in natural images, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, IEEE, 2012, pp. 1083–1090.
[23] M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in: European Conference on Computer Vision, Springer, 2014, pp. 818–833.
[24] S. Zhang, Y. Liu, L. Jin, C. Luo, Feature enhancement network: a refined scene text detector, (2017). arXiv:1711.04249.
[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] 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.