

# Robust Text Detection with Vertically-Regressed Proposal Network

Donglai Xiang<sup>1</sup>(✉), Qiang Guo<sup>2</sup>, and Yan Xia<sup>3</sup>

<sup>1</sup> Tsinghua University, Beijing, China  
xd113@mails.tsinghua.edu.cn

<sup>2</sup> National University of Defense Technology, Changsha, China

<sup>3</sup> SenseTime Group Limited, Beijing, China

**Abstract.** Methods for general object detection, such as R-CNN [4] and Fast R-CNN [3], have been successfully applied to text detection, as in [7]. However, there exists difficulty when directly using RPN [10], which is a leading object detection method, for text detection. This is due to the difference between text and general objects. On one hand, text regions have variable lengths, and thus networks must be designed to have large receptive field sizes. On the other hand, positive text regions cannot be measured in the same way as that for general objects at training. In this paper, we introduce a novel vertically-regressed proposal network (VRPN), which allows text regions to be matched by multiple neighboring small anchors. Meanwhile, training regions are selected according to how much they overlap with ground-truth boxes vertically and the location of positive regions is regressed only in the vertical direction. Experiments on dataset provided by ICDAR 2015 Challenge 1 demonstrate the effectiveness of our methods.

**Keywords:** Text detection · Vertical regression · Regional Proposal Network

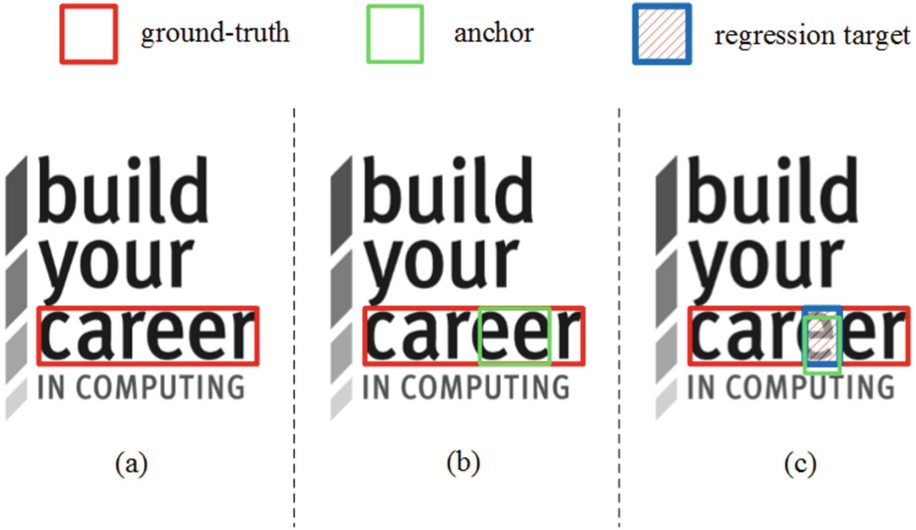
## 1 Introduction

Text detection, or text localization, plays a key role in robust text reading. Due to the special pattern of text regions, most traditional efforts utilize some specially designed features for text detection, such as the Maximally Stable Extremal Region (MSER) method [9] or the Stroke Width Transform (SWT) method [2]. Recently, the success of convolutional neural network in general object detection motivates researchers to bring the experience from general objects to text regions. For example, the method in [7] first generates some text proposals, and then performs classification and regression on them, which is similar to R-CNN [4] and Fast R-CNN [3] in a nutshell.

Text regions, as a special type of object regions, can be effectively detected by utilizing the detection framework of general objects, as revealed by [7]. Then we

---

D.Xiang and Q.Guo—Contributed equally to this work.



**Fig. 1.** (a) An example of text region that occupies almost the whole width of the image is shown in red box. (b) The anchor is shown in green, and it is excluded from positive training samples according to the measurement of RPN, but is included in our method. (c) Our method uses “thin” anchors (in green color) and only regresses the locations of ground-truth in vertical direction (in blue color). (Color figure online)

wonder how the most recent Faster R-CNN method [10] can be adopted for text detection. The core of Faster R-CNN [10] lies in a Region Proposal Network (RPN), in which object proposals are selected based on convolutional feature maps. Although RPN is originally used as a proposal generator, it can serve as a powerful detector itself. This paper will investigate the usage of RPN in the task of text detection.

However, directly applying RPN to text detection may suffer from a number of limitations. First, RPN selects positive training regions based on the intersection-over-union between an anchor and a ground-truth. In this way, anchors which fully overlap with a ground-truth in vertical direction but partly overlap in the horizontal direction are likely to be excluded from positive training samples, although they are text regions indeed (Fig. 1(b)). Second, the length of text regions varies substantially, and there always exist text regions that almost occupy the whole width of image (Fig. 1(a)). In this situation, the networks must be designed to have extremely large receptive field in horizontal direction in order to “see” large text regions thoroughly. As a result, the network is difficult to design: it must either have very deep convolutional layers, or skip many details in early steps by pooling.

To tackle the above problems, in this work we propose a vertically-regressed proposal network (VRPN) for the task of text detection. We select training samples in the view of what are text regions and what are not. In VRPN,

positive training samples are not measured by the intersection-over-union between anchor regions and ground-truth boxes any more. Instead, anchor regions which largely overlap with a ground-truth box in vertical direction while lying inside the ground-truth box in horizontal direction are treated as positives (Fig. 1(b)). Considering that a text region is not a single object but a union of text patterns, we use very “thin” anchors and only regress the vertical coordinates of positive samples (Fig. 1(c)). In this way, training samples are selected based on the nature of text regions and it is not rigid to have a network with an extremely large receptive field size.

The contribution of this work is summarized as follows:

1. We show that the RPN method can be borrowed from general object detection to the task of text detection.
2. We demonstrate that the size of receptive field in a network is critical to the detection of text regions of variable sizes.
3. We propose a novel VRPN method that fully utilizes information in the images and effectively solves the challenge of detecting text regions with variable length.

In experiments, our VRPN method shows substantial improvement compared with the RPN method on the ICDAR 2015 Robust Reading Competition Challenge 1 [8], which demonstrates the effectiveness of the proposed method.

## 2 Related Works

A lot of efforts have been devoted to the task of text detection in literature. The proposed methods can be categorized in two lines: the connected-component methods and the sliding-window methods.

### 2.1 Connected-component Methods

Traditionally, connected-component methods [5, 6, 12, 14, 15] have been more prevalent for the text detection task, because text regions which have special patterns are favorable as detection targets by connected-component methods. Text in images, no matter in what language, consists of a number of disconnected characters with sharp boundaries and obvious patterns against the background. Therefore, connected-component methods rely on some manually designed features, usually pixel-level information, to build fast detectors for character components, including MSER [9], SWT [2] and EdgeBox [16]. False alarms are then filtered out by text/non-text classifiers, such as SVM and CNN, and then character components are connected to get text lines with complex post-processing procedures.

The connected-component approaches have a great advantage in terms of speed, but their drawbacks are obvious as well. First, detectors based on low-level features generate a number of false alarms, leading to difficulties in filtering them out. Second, the process of combining all character components into text

lines is usually quite complicated and inelegant. In addition, using CNNs as merely filters doesn't fully tap their potential, which we can see in the sliding-window methods where CNNs are used as powerful detectors.

## 2.2 Sliding-window Methods

Compared with connected-component methods, sliding-window approaches share more common techniques with general object detections. In sliding-window methods, a window (or windows) scans over the image in different locations and sizes, and features are extracted for the areas inside the window. Then the areas are classified as text or non-text by classifiers [1, 11, 13]. The windows can be boxes with pre-defined sizes, or just be region proposals. Similar to R-CNN [4] and Fast R-CNN [3], the recent text detection method in [7] first generates some text proposals, and then classifies and regresses them.

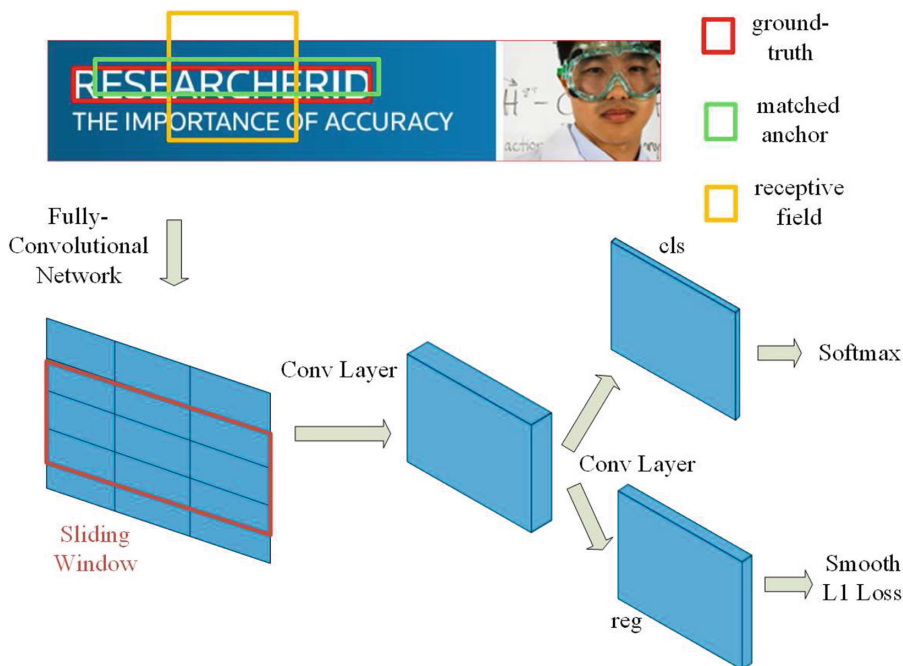
Notably, Ren *et al.* [10] propose a Faster R-CNN framework, which exploits convolutional features for proposing candidate windows. The core of Faster R-CNN is the Regional Proposal Network (RPN). In RPN, anchors are distributed all over the images in a sliding-window fashion, and then regions associated with the anchors are classified and regressed based on convolutional features. Although RPN is originally used as a proposal generator, it can be adopted as a powerful detector itself with great accuracy and high efficiency. However, there are still no works that show how RPN performs when used for text detection.

## 3 Vertically-Regressed Proposal Network for Text Detection

In this section, we introduce our vertically-regressed proposal network (VRPN). We first briefly review the Regional Proposal Network (RPN) proposed by Ren *et al.* [10]. Then we explain its limitation when used for text detection, and introduce our motivation. The proposed VPRN method is presented at last.

### 3.1 Regional Proposal Network

In RPN [10], classification and regression share the same convolutional layers. A number of anchors ( $k$ ) with various sizes and aspect ratios are pre-defined, in order to explore the shape of objects. At training stage, images are fed into a convolutional network to obtain convolutional feature maps. Then a small network (usually a convolutional layer with  $3 \times 3$  kernels followed by ReLU) slides on the feature map. Each point on the feature map corresponds to  $k$  anchors that are centered at the sliding window. Output of the sliding window is then fed into two sibling  $1 \times 1$  convolutional layers: one with  $2k$  outputs for classification (cls), and another with  $4k$  outputs for bounding box regression (reg). For each anchor region, its intersection-over-union (IoU) with all ground-truth boxes is computed. If its IoU with any ground-truth box is higher than a threshold, the region is labeled as positive. The regression target for a positive



**Fig. 2.** Architecture of regional proposal network for the problem of text detection.

sample is its best-matched ground-truth box. At test stage, the network predicts a number of boxes regressed from the anchors, together with their objectiveness scores. Then non-maximal suppression (NMS) is applied on these boxes to remove redundancy. The architecture of RPN is illustrated in Fig. 2.

### 3.2 Vertically-Regressed Proposal Network

*Motivation.* We find that there are a number of limitations when trying to apply RPN to text detection due to the following reasons.

First, the substantially varying sizes of text regions make the network designing challenging. In the original RPN method, networks usually don't have a receptive field that is large enough to cover all long text regions, so the detected location might deviate severely from the ground-truth. This problem might possibly be solved by enlarging the receptive field of the network. This can be achieved by either increasing the stride in early layers, or making the network deeper by adding more convolutional layers. However, on one hand, larger stride in early stage leads to inaccurate representation, which means a lot of details will be missing. This might hinder the accuracy of the prediction. On the other hand, adding more convolutional layers in order to deepen the network usually costs more computation. Neither solution is favorable for a text detection

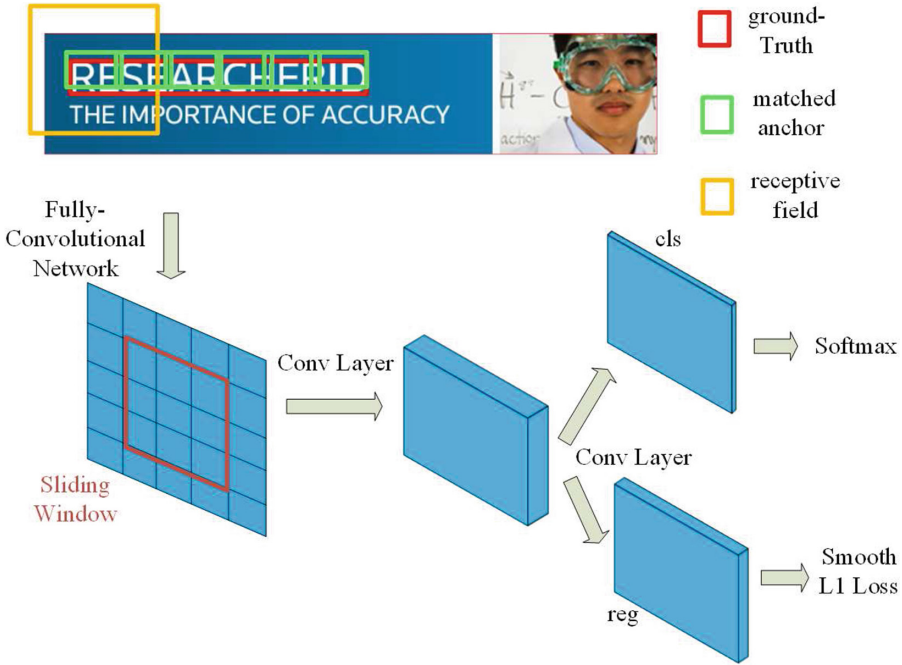


Fig. 3. Architecture of our proposed vertically-regressed proposal network.

system. Therefore, we hope to have a network that is able to deal with large text regions with moderate receptive field size and network size.

Second, as the aspect ratio of text regions are quite different, a large number of anchors with different height and width would be required in order to match the ground-truth boxes. This also increases the computation cost. We expect to have a network that can solve the problem with fewer anchors and less computation effort.

Third, in the training stage of RPN, positive and negative labels are assigned to anchors according to their IoU values with ground-truth. Therefore, some regions that are indeed textual would be excluded from positives. To illustrate, we show an example in Fig. 3. We can see that each of the green anchors region only covers a small part of the whole text region in red, so they will not be selected as positive samples. And even worse, the green regions may be selected as negatives. However, these regions are indeed textual, and the information within these regions might be contributive to training a powerful detector.

*Network Description.* Vertically-regressed proposal network shares a similar network structure to regional proposal network. However, VRPN adopts different approaches in selecting training samples and defining loss.

Our approach defines a new measurement of how well an anchor matches a ground-truth box for text regions. We denote the ground-truth bounding boxes

inside an image by  $\{G_j\} = \{(G_{jx1}, G_{jy1}, G_{jx2}, G_{jy2})\}$ , where  $G_{jx1}$ ,  $G_{jy1}$ ,  $G_{jx2}$  and  $G_{jy2}$  are the x and y coordinates of the top-left and bottom-right vertices of  $G_j$ , respectively. Given an anchor  $A_i = (A_{ix1}, A_{iy1}, A_{ix2}, A_{iy2})$ , we define a candidate target bounding box for  $A_i$  and  $G_j$  as

$$t_{ij} = (\max\{G_{jx1}, A_{ix1}\}, G_{jy1}, \min\{G_{jx2}, A_{ix2}\}, G_{jy2}). \quad (1)$$

The candidate target box is empty if  $\max\{G_{jx1}, A_{ix1}\} > \min\{G_{jx2}, A_{ix2}\}$ .

Then, the matching score between  $A_i$  and  $G_j$  is defined as

$$S_{ij} = \text{IoU}(A_i, t_{ij}). \quad (2)$$

The class label assigned to  $A_i$  is

$$c_i^* = \begin{cases} 1 & \text{if } \max_j \{S_{ij}\} > \text{threshold}, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

If  $A_i$  is assigned as positive, we also define the regression target associated with  $A_i$  as  $T_i = t_{ij}$  where  $j = \text{argmax}_k \{S_{ik}\}$ .

To explain the rule in plain words, we select  $A_i$  as a positive training sample if the matching score between  $A_i$  and any ground-truth box is greater than a threshold. If the anchor is assigned a positive label, it is also trained for regression, and the regression target is just the candidate target bounding box  $t_{ij}$  that has the best matching score with  $A_i$ . This rule to match anchors with ground-truth boxes is illustrated in Fig. 4.

*Loss Definition.* The training loss for VRPN is given as

$$L(\{c_i\}, \{r_i\}) = \frac{1}{N_{\text{cls}}} \sum_i L_{\text{cls}}(c_i, c_i^*) + \frac{\lambda}{N_{\text{reg}}} \sum_i \sum_{j \in \{x, y, w, h\}} c_i^* L_{\text{reg}}(r_i^j, r_i^{j*}). \quad (4)$$

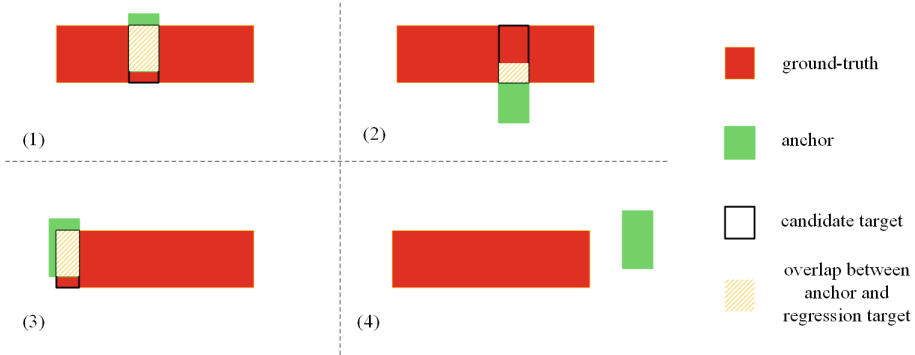
Here  $i$  is the index of anchor regions.  $c_i$  is the class prediction of  $A_i$  and  $c_i^*$  is its label for classification.  $r_i^x$  is a parameterized coordinate of the anchor's x-center, and  $r_i^{x*}$  is the parameterized coordinate of regression target associated with the anchor.  $r_i^y, r_i^{y*}, r_i^w, r_i^{w*}, r_i^h$ , and  $r_i^{h*}$ , are defined likewise. The parameterized coordinates are:

$$\begin{aligned} r^x &= (x - x_a)/w_a, r^y = (y - y_a)/h_a, \\ r^w &= \log(w/w_a), r^h = \log(h/h_a), \end{aligned} \quad (5)$$

and

$$\begin{aligned} r^{x*} &= (x^* - x_a)/w_a, r^{y*} = (y^* - y_a)/h_a, \\ r^{w*} &= \log(w^*/w_a), r^{h*} = \log(h^*/h_a), \end{aligned} \quad (6)$$

where  $[x_a, y_a, w_a, h_a]$  are the x-center coordinate, y-center coordinate, width and height of the anchor  $A_i$ , respectively, while  $[x, y, w, h]$  and  $[x^*, y^*, w^*, h^*]$  are similarly defined for the prediction and the regression target  $T_i$  associated with  $A_i$ .  $L_{\text{cls}}$  is the classification loss, computed by softmax and cross-entropy loss over the text/non-text classes.  $L_{\text{reg}}$  is the regression loss, for which we adopt smooth  $L_1$  loss as defined in [3]. The parameter  $\lambda$  is a weight to balance them.



**Fig. 4.** Four cases for matching an anchor with a ground-truth box. In case (1), the anchor has an IoU above a threshold 0.7 with its candidate target box; in case (2), the anchor deviates too much from the ground-truth in the vertical direction to be matched with it; in case (3), the left end of the anchor is outside the ground-truth so the candidate target has a different left boundary with the anchor; in case (4), the anchor is completely outside the ground-truth and they don’t match each other. Whether an anchor region is assigned as positive or not is based on its IoU with its target.

### 3.3 Implementation Details

Our VRPN method measures how well an anchor matches a ground-truth box mainly according to how much they overlap vertically (if the anchor lies inside the ground-truth box in the horizontal direction). A ground-truth can be matched by several neighbouring anchors, whose x-coordinates differ by a total stride of the network one by one. Therefore, the width of the anchors can be set to be slightly larger than the total stride to ensure that they leave no space in between, and only their height varies to fit different text areas.

At test stage, just like RPN, the output of the network is a bunch of filtered boxes regressed from the anchors. Then non-maximal suppression (NMS) is performed on these boxes at some threshold. Then some of them are merged together if they overlap horizontally in order to match long words.

## 4 Experiments

### 4.1 Dataset and Evaluation Metric

We conduct experiments on the dataset provided for Challenge 1: “Born-Digital Images (Web and Email)” of ICDAR 2015 Robust Reading Competition [8]. The dataset contains 410 images for training and 141 images for testing. We train all networks with the 410 training images in this dataset and conduct testing using the 141 test images.

Considering the fact that NMS and other post-processing steps actually have quite a large impact on the detection results, while what we really want to



compare is the ability of the networks to cover ground-truth boxes with few false alarms, we first evaluate methods by their raw outputs without post-processing. So we use a pixel-level evaluation. We compare the performance of methods by f-measure, which is defined as

$$F_\beta = (1 + \beta^2) \frac{\text{precision} \cdot \text{recall}}{\beta^2 \text{precision} + \text{recall}}. \quad (7)$$

The f-measure is a combination of recall and precision. Recall is defined as the number of pixels in ground-truth boxes that are covered by at least one output box divided by the total number of ground-truth pixels in the dataset. And precision is defined as the number of detected ground-truth pixels divided by all pixels in the output boxes, with repeatedly detected pixels counted only once. We use  $\beta = 1$  in evaluation, *i.e.* the  $F_1$  score, which is the harmonic mean of precision and recall values.

Although post-processing steps are not our focus in this paper, we still provide the performance of all methods after non-maximal suppression (NMS) and basic merging, which will be explained more detailedly in the next paragraph, in order to demonstrate that our conclusion on methods won't be challenged when post-processing steps are added to the pipeline. We still use  $F_1$  score to evaluate performance and the threshold of NMS is set at 0.3 for this experiment setting.

*Merging of Bounding Boxes.* After NMS is performed on the raw output boxes, some of them are merged together in order to match long words and extract text lines. All the boxes in a connected component are merged to one box. The connected components are built by adding two boxes which overlap with each other. Here the ‘‘overlap’’ means one box is completely enclosed by the other or the height of their intersection box divided by the maximum of their heights is above a certain threshold. The four boundaries of the final output box are computed as the leftmost one and the rightmost one, as well as the averaged upper and lower boundaries of boxes inside a connected component.

## 4.2 Experiments on Vertically-Regressed Proposal Network

In this subsection, we verify the motivation of our VRPN method and demonstrate its superiority to RPN for the task of text detection by a number of experiments.

**How Receptive Field Size Affects the Performance of RPN?** In Sect. 3.2 we argue that, in order for the RPN network to regress the box locations accurately, the receptive field of the network has to be large enough, especially in horizontal direction. In order to validate it, we conduct the following experiments in regard to RPN.

We compare two networks with different receptive field sizes, using the RPN method. The structures of the two networks are shown in Table 1. The same 10

**Table 1.** Structure of RPN networks

Layer	Parameters			
Network index	RPN I		RPN II	
Conv	kernel=5, pad=2, stride.w=2, stride_h=1, output=16			
ReLU	-			
Conv	kernel=5, pad=2, stride=1, output=16			
ReLU	-			
Max pooling	kernel=2, stride=2			
Conv	kernel=3, pad=1, stride=1, output=32	kernel=5, pad=2, stride=1, output=32		
ReLU	-			
Conv	kernel=3, pad=1, stride=1, output=32	kernel=5, pad=2, stride=1, output=32		
ReLU	-			
Max pooling	kernel=2, stride=2	kernel=2, stride=2		
Conv	kernel=3, pad=1, stride=1, output=64	kernel=5, pad=2, stride=1, output=64		
ReLU	-			
Conv	kernel=3, pad=1, stride=1, output=64	kernel=5, pad=2, stride=1, output=64		
ReLU	-			
Conv (sliding window) relu	kernel=3, pad=1, stride=1, output=256			
	-			
Conv	kernel=1, pad=0, stride=1, output=20	kernel=1, pad=0, stride=1, output=40	kernel=1, pad=0, stride=1, output=20	kernel=1, pad=0, stride=1, output=40
Loss	softmax loss	smooth $L_1$ loss	softmax loss	smooth $L_1$ loss
Receptive field size	horizontal: 67, vertical: 36		horizontal: 115, vertical: 60	

anchors with different sizes are used, which are obtained by clustering ground-truth boxes in training set. The two networks only differ in the kernel sizes in four convolutional layers, and RPN II has a larger receptive field.

As can be seen from Table 2, the network with a larger receptive field performs better, no matter with or without post-processing. However, having a receptive field size large enough to cover all text regions results in a number of problems, as has been discussed in Sect. 3.2. This motivates us to develop the VRPN method.

**Comparison Between RPN and VRPN.** In this section we conduct experiments to compare the performance of RPN and VRPN. In VRPN, we use the same network structure and anchors as RPN II. So in this experiment, the

**Table 2.** Comparisons between RPN I and RPN II with different receptive field size.

	w/o post-processing			with post-processing		
	Precision	Recall	$F_1$	Precision	Recall	$F_1$
RPN I	0.666	0.513	0.579	0.678	0.654	0.666
RPN II	0.666	0.630	0.648	0.744	0.660	0.699

difference between RPN and VRPN lies in the selection of positive/negative training samples. Results are shown in Table 3. We can see that VRPN achieves a much higher recall and overall  $F_1$  score than RPN. The improvement is because, VRPN selects positive training samples mainly according to how well anchors overlap with ground-truth vertically, which allows much more anchors assigned as positive. At test stage, more anchors are located in text areas where they will be scored higher according to our method of training, and therefore cover more text pixels to achieve a better recall. Also, our precision is close to RPN. This experiment verifies the superiority of VRPN.

**How to Choose Anchors for VRPN?** The VRPN method makes the selection of anchors easy. Because in VRPN, the anchors only need to explore the heights of ground-truth boxes and their width can be fixed to a value slightly larger than the horizontal stride of the network. This makes detection more accurate than using wider anchors as in RPN. We use the following experiments to validate this idea. We compare two VRPN settings with different anchors. For VRPN I, we use the same 10 anchors as before; and for VRPN II, we use 5 anchors whose widths are fixed at 10 (the horizontal stride of the network is 8) and heights range from 8 to 40. The network structure and other hyperparameters remain the same. Experimental results are shown in Table 4.

From the results we can see that VRPN with anchors selected in our designed method achieves better performance than the original manner even with fewer anchors and less computation cost. The improvement is presumably due to finer regression of target bounding boxes, which is a benefit of the smaller anchors. Some examples of detection outputs by VRPN II are shown in Fig. 5.

**Table 3.** Comparisons between RPN and VRPN with the same network structure and anchors.

	w/o post-processing			with post-processing		
	Precision	Recall	$F_1$	Precision	Recall	$F_1$
RPN II	0.666	0.630	0.648	0.744	0.660	0.699
VRPN I	0.645	0.858	0.736	0.730	0.777	0.753

**Table 4.** Comparison between two ways of selecting anchors

	w/o post-processing			with post-processing		
	Precision	Recall	$F_1$	Precision	Recall	$F_1$
VRPN I	0.645	0.858	0.736	0.730	0.777	0.753
VRPN II	0.709	0.822	0.761	0.707	0.826	0.762



**Fig. 5.** Example detections by the VRPN II network. For each column in this figure, the first row shows the original image; the second row shows the raw outputs of the method; and the third row shows the final detection result after merging.

## 5 Conclusion

In this paper, we present a vertically-regressed proposal network. This method originates from RPN and is developed for text detection. In the proposed method, text regions are matched by several consecutive anchor regions, and each anchor region is regressed to predict the upper and lower boundaries of text regions. This allows the network to perform accurate and robust text detection with moderate network structure, fewer anchors and lower computation cost.

Using “thin” anchors in VRPN can be treated as making anchor-level inference on dense feature maps. For future work, we can investigate pixel-level detection which makes inference on a feature map of the same size as the original image.

## References

1. Chen, X., Yuille, A.L.: Detecting and reading text in natural scenes. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2004, vol. 2, pp. II-366. IEEE (2004)
2. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2963–2970. IEEE (2010)
3. Girshick, R.: Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448 (2015)
4. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
5. Huang, W., Lin, Z., Yang, J., Wang, J.: Text localization in natural images using stroke feature transform and text covariance descriptors. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1241–1248 (2013)
6. Huang, W., Qiao, Y., Tang, X.: Robust Scene text detection with convolution neural network induced MSER trees. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part IV. LNCS, vol. 8692, pp. 497–511. Springer, Heidelberg (2014)
7. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. *Int. J. Comput. Vis.* **116**(1), 1–20 (2016)
8. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., et al.: ICDAR 2015 competition on robust reading. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR), pp. 1156–1160. IEEE (2015)
9. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vis. Comput.* **22**(10), 761–767 (2004)
10. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
11. Tian, S., Lu, S., Su, B., Tan, C.L.: Scene text recognition using co-occurrence of histogram of oriented gradients. In: 2013 12th International Conference on Document Analysis and Recognition (ICDAR), pp. 912–916. IEEE (2013)
12. Tian, S., Pan, Y., Huang, C., Lu, S., Yu, K., Lim Tan, C.: Text flow: a unified text detection system in natural scene images. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4651–4659 (2015)
13. Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 1457–1464. IEEE (2011)
14. Ye, Q., Huang, Q., Gao, W., Zhao, D.: Fast and robust text detection in images and video frames. *Image Vis. Comput.* **23**(6), 565–576 (2005)
15. Yin, X.C., Yin, X., Huang, K., Hao, H.W.: Robust text detection in natural scene images. *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(5), 970–983 (2014)
16. Zitnick, C.L., Dollár, P.: Edge boxes: locating object proposals from edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part V. LNCS, vol. 8693, pp. 391–405. Springer, Heidelberg (2014)