DeRPN: Taking a Further Step toward More General Object Detection

Lele Xie, Yuliang Liu, Lianwen Jin,* Zecheng Xie
School of Electronic and Information Engineering, South China University of Technology
xie.lele@mail.scut.edu.cn, liu.yuliang@mail.scut.edu.cn, *eelwjin@scut.edu.cn, zchengxie@gmail.com

Abstract

Most current detection methods have adopted anchor boxes as regression references. However, the detection performance is sensitive to the setting of the anchor boxes. A proper setting of anchor boxes may vary significantly across different datasets, which severely limits the universality of the detectors. To improve the adaptivity of the detectors, in this paper, we present a novel dimension-decomposition region proposal network (DeRPN) that can perfectly displace the traditional Region Proposal Network (RPN). DeRPN utilizes an anchor string mechanism to independently match object widths and heights, which is conducive to treating variant object shapes. In addition, a novel scale-sensitive loss is designed to address the imbalanced loss computations of different scaled objects, which can avoid the small objects being overwhelmed by larger ones. Comprehensive experiments conducted on both general object detection datasets (Pascal VOC 2007, 2012 and MS COCO) and scene text detection datasets (ICDAR 2013 and COCO-Text) all prove that our DeRPN can significantly outperform RPN. It is worth mentioning that the proposed DeRPN can be employed directly on different models, tasks, and datasets without any modifications of hyperparameters or specialized optimization, which further demonstrates its adaptivity. The code has been released at https://github.com/HCIILAB/DeRPN.

Introduction

Recently, general object detection has achieved rapid development, driven by the convolutional neural network (CNN). As a significant task in computer vision, general object detection is expected to detect more object classes (Redmon and Farhadi 2017; Singh et al. 2018) and perform impressively on different datasets (Everingham et al. 2010; Lin et al. 2014). Unfortunately, we notice that most of the general object detection methods are not very general. When employing some state-of-the-art methods (Dai et al. 2017; Lin et al. 2018; Zhang et al. 2018b) on different datasets, it is usually indispensable to redesign the hyperparameters of regression references, termed as anchor boxes in Region Proposal Network (RPN) (Ren et al. 2015). Even for some specific object detection tasks, such as scene text detection (Lyu et al. 2018; Zhang et al. 2018a; Liao, Shi, and Bai 2018), directly applying state-of-the-art approaches of general object detection cannot produce their due effects (Liao et al. 2017; Liu and Jin 2017). This problem stems from presupposed anchor boxes with fixed shapes and scales. Although RPN was verified to be an effective method to generate region proposals in (Ren et al. 2015), the anchor boxes adopted in RPN are very sensitive, which limits the adaptivity to variant objects. Once the anchor boxes excessively deviate from the ground truths in a dataset, the performance is curtailed significantly. This is why many specific methods for scene text detection (Liao et al. 2017; Ma et al. 2017; Zhang et al. 2018a) lay emphasis on the design of the anchor boxes, which are intended to be long, narrow, and consistent with the text characteristics. However, manually setting the anchor boxes is cumbersome and it is difficult to guarantee the best performance. Although (Redmon and Farhadi 2017) proposed an algorithm based on K-means clustering to precompute anchor boxes, the improvement was still limited.

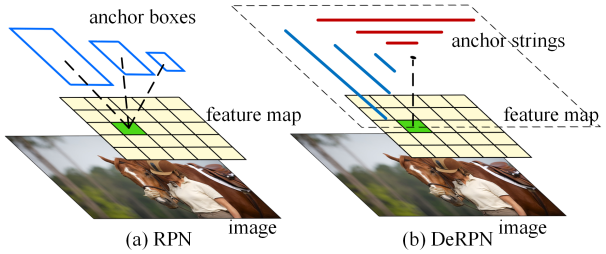

Figure 1: Schemas of RPN and DeRPN: (a) RPN adopts multiple anchor boxes of fixed shapes and scales, (b) DeRPN divides bounding boxes into flexible anchor strings, decoupling width and height.

As illustrated in Fig.1 (a), RPN adopts anchor boxes as regression references. We argue that anchor boxes of fixed scales and shapes have extreme difficulty covering thousands of objects. Therefore, we propose a novel dimension-decomposition region proposal network (DeRPN). In Fig.1 (b), DeRPN divides anchor boxes into several independent segments that we call anchor strings, inheriting the terminology from (Ren et al. 2015). Note that DeRPN decomposes the detection dimension by decoupling the width and height. The main contributions of our work are summarized as follows:

- We propose a novel region proposal network, DeRPN, which has strong adaptivity. Without any modifications for hyperparameters, DeRPN can be directly employed on different models, tasks, or datasets.
- The proposed DeRPN adopts an advanced dimension-decomposition mechanism. Through flexible anchor strings, DeRPN can match objects with optimal regression references, which allows the network to be more smoothly trained.
- We propose a novel scale-sensitive loss function to tackle the imbalance of object scales and prevent small objects from being overwhelmed by lager ones.
- The proposed DeRPN maintains a consistent network and running time with RPN, and thus it can conveniently be transplanted to current two-stage detectors.
- The proposed DeRPN has a higher recall rate and more accurate region proposals than the previous RPN. It outperforms RPN on different tasks and datasets.

Related Works

In the past several years, early object detection methods were confined to traditional methods (Viola and Jones 2001; Felzenszwalb et al. 2010) involving handcrafted features. With the recent significant progress of CNN, more CNN-based methods have been put forward and rapidly dominated the detection task. We introduce some of the works below.

General object detection Object detection methods are categorized into two streams: two-stage approaches and one-stage approaches. The two-stage framework contains two procedures, where the first step is to generate some candidate regions of interest (RoIs) with region proposal methods, e.g., Selective Search (SS) (Uijlings et al. 2013) and RPN (Ren et al. 2015). Then, to determine the accurate regions and clear classes further, the second step resamples features according to the RoIs with a pooling operation (Girshick 2015; Dai et al. 2016). Such two-stage methods (Lin et al. 2017; He et al. 2017a; Li et al. 2018) have achieved high detection accuracy on some challenging datasets. Conversely, the one-stage methods (Redmon and Farhadi 2017; Lin et al. 2018; Zhang et al. 2018b) focus more on the running speed. Without the resampling operation, the procedure of one-stage methods has been simplified manyfold and thus benefits the running speed.

Region proposal methods As an important component in two-stage detectors, region proposal methods have a significant impact on the final detection result. Formerly, correlative image processing methods were used to generate region proposals. Some were based on grouping pixels (Uijlings et al. 2013), and others were based on sliding windows (Zitnick and Dollár 2014). Note that these methods are independent of the detectors. Later, Ren et al. proposed a region proposal network that was integrated in Faster R-CNN to build an end-to-end detector. Notably, as proven by (Ren et al. 2015), RPN outperformed other previous proposal methods, for example SS (Uijlings et al. 2013) and EdgeBoxes (Zitnick and Dollár 2014). To date, the vast majority of two-stage detectors have adopted RPN to generate region proposals.

Scene text detection Different from general objects, scene texts are usually smaller, thinner, and with characteristic texture and rich diversity in their aspect ratios (Tian et al. 2015). In addition, scene text detection can be applied to many scenarios, for example multilingual translation, document digitization, and automatic driving. For these reasons, scene text detection has been regarded as another challenging task in computer vision. Comprehensive survey of scene text detection can be found in (Ye and Doermann 2015). According to the basic detection element, methods of scene text detection are classified into three categories: (1) character based (Hu et al. 2017), (2) word based (Deng et al. 2018), and (3) text line based (Zhang et al. 2016).

Methodology

In this section, we describe the details of the DeRPN. The network and full pipeline are illustrated in Fig. 2.

Modeling

Most current methods regard the object detection as adjustment (reg) and filtration (cls) toward complete bounding boxes. Based on a convolutional neural network, these methods take the features ($\boldsymbol{x}$) from a CNN and input them to a regression layer and classification layer. Usually, the regression layer, implemented by a convolutional or fully connected layer, is a linear operation to predict the parameterized coordinates ($\boldsymbol{t}$). To acquire the predicted bounding boxes, these parameterized coordinates are decoded according to anchor boxes (B_a). In addition, the classification layer applies an activation function (e.g. Sigmoid or Softmax, denoted as σ) on the predicted values to generate the probability (P_B) of bounding boxes. The mathematical descriptions of this procedure are as follows:

$$\boldsymbol{t} = \mathbf{W_r}\boldsymbol{x} + \boldsymbol{b_r} \tag{1}$$

$$B(x, y, w, h) = \psi(\boldsymbol{t}, B_a(x_a, y_a, w_a, h_a)) \tag{2}$$

$$P_B = \sigma(\mathbf{W_c}\boldsymbol{x} + \boldsymbol{b_c}) \tag{3}$$

where $\mathbf{W_r}$ and $\boldsymbol{b_r}$ denote the weights and biases of the regression layer. Similarly, $\mathbf{W_c}$ and $\boldsymbol{b_c}$ are those for the classification layer. x, y, w, and h are coordinates of the bounding boxes. ψ represents an anti-parameterized function that is used to decode coordinates.

However, thousands of objects possess extremely variant shapes, and harshly setting numerous anchor boxes proves to be adverse for training and very time-consuming (Ren et al. 2015; Redmon and Farhadi 2017; Liu and Jin 2017). The underlying problem is that people are difficult to design appropriate anchor boxes to treat diverse objects. The excessive deviations between anchor boxes and ground truths will aggravate the learning burden of detectors and dramatically curtail performance. A novel solution is to decompose the detection dimension by decoupling the width and height to alleviate the impact from variant shapes of objects. With the dimension-decomposition mechanism, we introduce anchor strings ($S_a^w(x_a, w_a)$, $S_a^h(y_a, h_a)$) that serve as independent regression references for the object width and height. Differently, anchor strings predict independent segments ($S_w(x, w)$,$S_h(y, h)$) and corresponding probabilities

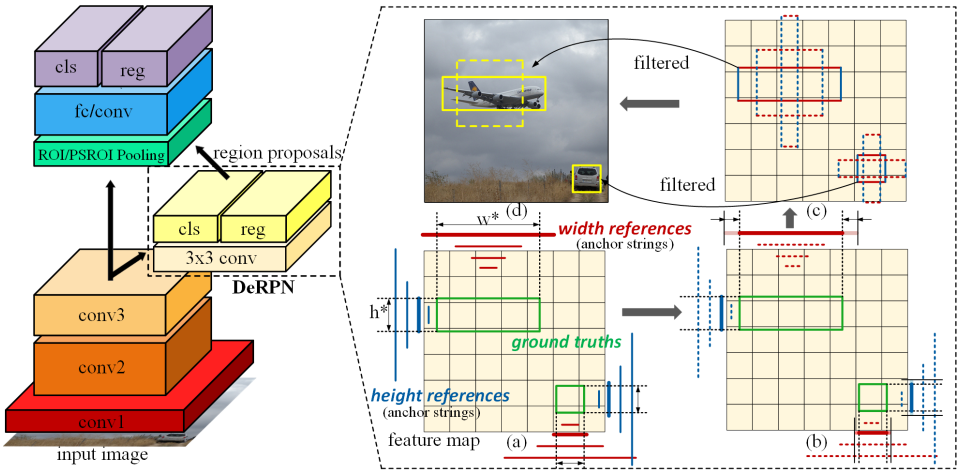

Figure 2: DeRPN network and pipeline. (a) Object widths and heights are independently matched with anchor strings. Bold lines represent well-matched anchor strings. This step is employed during the training phase. (b) We apply classification and regression on anchor strings. Dashed lines are anchor strings of low probability. (c) Predicted width and height segments are combined to compose bounding boxes. (d) We filter bounding boxes by probability and NMS to generate region proposals.

(P_s^w, P_s^h), instead of full bounding boxes. We describe the procedure as follows:

$$\boldsymbol{t}^w = \mathbf{W_r}^w \boldsymbol{x} + \boldsymbol{b_r^w} \qquad S_w(x,w) = \psi(\boldsymbol{t}^w, S_a^w(x_a, w_a)) \quad (4)$$

$$\boldsymbol{t}^h = \mathbf{W_r}^h \boldsymbol{x} + \boldsymbol{b_r^h} \qquad S_h(y,h) = \psi(\boldsymbol{t}^h, S_a^h(y_a, h_a))) \quad (5)$$

$$P_s^w = \sigma(\mathbf{W_c}^w \boldsymbol{x} + \boldsymbol{b_c^w}) \qquad P_s^h = \sigma(\mathbf{W_c}^h \boldsymbol{x} + \boldsymbol{b_c^h}) \quad (6)$$

As the two-dimensional bounding boxes are required for the detection results, we need to reasonably combine the predicted segments to recover the bounding boxes. The combination procedure is given by

$$B(x,y,w,h) = f(S_w(x,w), S_h(y,h)) \quad (7)$$

$$P_B = g(P_s^w, P_s^h) \quad (8)$$

where, f represents a kind of rule or algorithm to combine predicted segments. Meanwhile, g is a function (e.g., arithmetic mean, harmonic mean) which evaluates the probabilities of combined bounding boxes.

Through discretization, we assume that the object width or height in a dataset has n kinds of changes. Fully combining all widths and heights determines approximately n^2 object shapes with which the anchor box needs to be faced. In other words, the matching complexity of the anchor box is $O(n^2)$. Nevertheless, when adopting the dimension-decomposition mechanism, n kinds of widths and heights are independently matched with anchor strings, which produces a lower matching complexity of $O(n)$.

Dimension Decomposition

Anchor strings The previous RPN regarded anchor boxes as its regression references. For the difficulty of matching various objects with fixed bounding boxes, RPN heavily relies on the design of anchor boxes and further loses satisfactory adaptivity. By contrast, DeRPN breaks the two-dimensional boxes into a series of independent one-dimensional segments called anchor strings. Through this dimension decomposition, we separately handle the object width and height for classification and regression, which can alleviate the impact from diverse object shapes.

Anchor strings are expected to cover widths and heights of all objects. We set the anchor strings as a geometric progression (denoted as $\{a_n\}$), i.e., (16, 32, 64, 128, 256, 512, 1024). Theoretically, this geometric progression can apply to widths or heights within a large range of $[8\sqrt{2}, 1024\sqrt{2}]$, which can cover most of the objects in many scenarios. In this paper, both the width and height use the same but independent geometric progression as their anchor strings. As illustrated in Fig. 2 (a), these anchor strings are assigned at each feature map location. Each anchor string is responsible for predicting the width segment or height segment, instead of a full bounding box.

The next concern is how to choose the best-matched anchor strings for an object. In RPN, an anchor box is chosen based on the intersection over union (IoU) between the anchor box and ground truth. If the IoU is over 0.7, or the IoU is the largest one, that anchor box is regarded as positive. Owing to the deviation between the anchor box and object, sometimes the IoU is very small (less than 0.2). Under this situation, the anchor box is probably still chosen for its largest IoU, which produces a significant regression loss in the training state. Conversely, as shown in Fig. 2 (a), DeRPN reasonably matches the objects with anchor strings based on length instead of IoU. The best-matched anchor strings are evaluated by

$$M_j = \{i \mid \arg\min_i |\log e_j - \log a_i|\} \cup \{i, i+1 \mid \left| \frac{e_j}{a_i} - \sqrt{q} \right| \leq \beta\}$$

$$(i = 1, 2, 3, ..., N) \quad (9)$$

where M_j denotes the index set of matched anchor strings for the j-th object. e_j is the object edge (width or height). N and q represent the number of terms, and the common ratio

(in this paper, q is set to 2) of geometric progression $\{a_n\}$, respectively. Term a_i is i-th anchor string in $\{a_n\}$. Note that the first term in this equation represents the closest anchor string to edge e_j. The second term describes a transition interval within $[(\sqrt{q}-\beta)\times a_i, (\sqrt{q}+\beta)\times a_i]$, where β is used to adjust magnitude of interval, which is intended to be 0.1 in our experiments. If e_j locates in the transition interval, both i and $i+1$ are chosen as matched indexes. We employ the transition interval to reduce ambiguities resulting from factors such as image noises and ground truth deviations.

Notably, the anchor string mechanism of DeRPN introduces boundedness that ensures a stable training procedure. By neglecting the transition interval, it is easy to prove that the largest deviation (measured by ratio) between the anchor string and object edge is at most $\sqrt{q}$, which means the regression loss of DeRPN is bounded. Compared to RPN, the unexpected small IoU usually generates a significant regression loss. Empirically, RPN cannot even converge if the anchor boxes deviate excessively from the ground truths.

Label assignment To assign labels, initially, we define aligned anchor strings that locate at the object centers on a feature map. The positive labels are assigned to aligned anchor strings if they are well matched with corresponding objects, as identified with Eq. (9). Except for the aligned ones, we employ an observe-to-distribute strategy to determine other anchor strings. That is, we first observe the regression results for each anchor string. After regression, the predicted segments (widths or heights) of the anchor strings are combined to compose region proposals. If the region proposals have high IoUs (over 0.6) with ground truths, we distribute positive labels to the corresponding anchor strings. The detailed combination procedure is introduced in the next section. Anchor strings that do not satisfy the above conditions will simply be treated as negative.

The previous RPN statically assigns labels merely based on the IoU. Owing to deviations between the anchor boxes and objects, sometimes the IoUs are very small, which introduces a huge regression loss. Compared with RPN, DeRPN has proposed a new kind of secure, dynamic, and reasonable mechanism for label assignment. In most cases, the features at the object centers are representative, and thus such a label assignment method for aligned anchor strings is reasonable. Except for the centers, we cannot identify whether the features at other positions are important. Consequently, we adopt a dynamic observe-to-distribute strategy to determine the labels at other positions conservatively.

Consistent network DeRPN has maintained the consistent network architecture with RPN, and thus it can conveniently be transplanted to current two-stage detectors. Since it adopts the same network architecture, the running time of DeRPN is approximately the same as RPN. In addition, DeRPN also shares convolutional layers with the second-stage detector. To be more specific, as shown in Fig. 2, the network is constituted with a 3×3 convolutional layer, followed by two sibling 1×1 convolutional layers (for classification and regression). The number of terms in geometric progression $\{a_n\}$ is denoted as N. Since the width and height employ independent anchor strings, the classification layer predicts $2 \times 2N$ scores to estimate whether an anchor string is matched or not matched with object edges. Besides, each anchor string predicts a width segment or height segment with the coordinates of (x, w) or (y, h), respectively. Therefore, the regression layer also predicts $2 \times 2N$ values.

Scale-sensitive loss function The distribution of object scale is often imbalanced (Lin et al. 2014), and there are usually more large objects than smaller ones. If we simply mix objects together to evaluate loss, the small objects will be severely overwhelmed by the larger ones. Benefiting from the distinct scales of anchor strings, we propose a novel scale-sensitive loss function to handle objects of different scales fairly. The scale-sensitive loss function is defined as

$$L(\{p_i\},\{t_i\}) = \sum_{j=1}^{N}\sum_{i=1}^{M}\frac{1}{|R_j|}L_{cls}(p_i, p_i^*)\cdot 1\{i \in R_j\} + \lambda\sum_{j=1}^{N}\sum_{i=1}^{M}\frac{1}{|G_j|}L_{reg}(t_i, t_i^*)\cdot 1\{i \in G_j\} \quad (10)$$

in which

$$R_j = \{k | s_k = a_j, k = 1, 2, ..., M\}, \quad (11)$$

$$G_j = \{k | s_k \in A, s_k = a_j, and\ p_i^* = 1, k = 1, 2, ..., M\}. \quad (12)$$

Here, N is the number of terms in geometric progression $\{a_n\}$ and M is the batch size. s denotes the anchor string. p_i represents the predicted probability of the i-th anchor string in a mini-batch. The ground-truth label p_i^* is set to 1 if the anchor string is positive. Otherwise, p_i^* is 0. t_i is a predicted vector representing the parameterized coordinates, and t_i^* is the corresponding ground truth vector. A is the set of aligned anchor strings. R_j denotes an index set containing those anchor strings of the same scale, and j is used to indicate the scale corresponding to term a_j in $\{a_n\}$. Similarly, G_j is an index set containing positive aligned anchor strings of the same scale. The classification loss L_{cls} is a cross-entropy loss, and the regression loss L_{reg} is designed as a smoothed L1 loss. λ is a balancing parameter for L_{cls} and L_{reg}, which is empirically set to 10.

After regression, we decode the predicted coordinates as follows:

$$x = x_a + w_a \times t_x, \quad (13)$$

$$y = y_a + h_a \times t_y, \quad (14)$$

$$w = w_a \times \exp(t_w), \quad (15)$$

$$h = h_a \times \exp(t_h), \quad (16)$$

where (x, w) and (x_a, w_a) are, respectively, the coordinates of the predicted width segment and the anchor string. Analogously, (y, h) and (y_a, h_a) are those relative to height.

For each scale, we randomly sample at most 30 positive and negative anchor strings to form a mini-batch. Note that the anchor strings of the same scale share the same weighted coefficient, which is calculated by their own number. This can effectively prevent small objects being overwhelmed by larger ones. As mentioned before, we set anchor strings as a geometric progression, and thus different scales of anchor strings are explicitly distinguished by the common ratio. Therefore, the scale-sensitive loss function can naturally be applied to DeRPN.

Dimension Recombination

By using flexible anchor strings, DeRPN can predict accurate segments, serving as the edges (widths or heights) of objects. However, the final expected region proposals are two-dimensional bounding boxes. We need to reasonably combine the width and height segments to recover the region proposals. The process is illustrated in Fig. 2 (c).

Pixel-wise combination algorithm In this paper, we propose a pixel-wise combination algorithm. First, the predicted width and height segments are decoded according to Eqs. (13)-(16). We then consider the full set of width segments (denoted as W). We filter the width segments based on probability to pick out the top-N ones (W_N). For each width segment (x, w) in W_N, we choose the top-k height segments $(y^{(k)}, h^{(k)})$ at the corresponding pixels. Then, these pairs of width and height segments determine a series of specific bounding boxes $\{(x, y^{(k)}, w, h^{(k)})\}$, denoted as B_w. The probability of a composed bounding box is given as Eq. (17). Similarly, we can acquire $B_h = \{(x^{(k)}, y, w^{(k)}, h)\}$ by repeating the above steps for the height segments. We then employ non-maximum suppression (NMS) on the union set of B_w and B_h with an IoU threshold of 0.7. Finally, the top-M bounding boxes after NMS are regarded as region proposals. In this way, we can obtain a high recall of object instances. Although this combination method introduces some background bounding boxes, the second-stage detector can suppress them with a classifier. In addition, as shown in Eq. (17), the bounding box probability is set as the harmonic mean of width and height probability. This can dramatically pull down the box probability as long as the height or width has a rather small probability, so as to remove this bounding box. We should also note that the combination algorithms are not unique. Other algorithms can be explored to achieve a high recall rate.

$$p^B = 2 \Big/ \left(\frac{1}{p^W} + \frac{1}{p^H} \right) \tag{17}$$

Experiments

In this section, we regarded RPN as the counterpart of our experiments. To verify the adaptivity, we maintained the same hyperparameters for DeRPN throughout all of our experiments without any modifications. In addition, we used the same training and testing settings for RPN and DeRPN to guarantee a fair comparison. For a comprehensive evaluation, we carried out experiments on two kinds of tasks: general object detection and scene text detection.

General Object Detection

Dataset The experimental results are reported in three public benchmarks: PASCAL VOC 2007, PASCAL VOC 2012 (Everingham et al. 2010), and MS COCO (Lin et al. 2014). PASCAL VOC and MS COCO contain 20 and 80 classes.

Region proposals evaluation In this subsection, we drew a comparison for region proposals between DeRPN and RPN. We selected VGG16 (Simonyan and Zisserman 2015) as our backbone, and appended DeRPN or RPN to its "conv5" layer. We then trained models on the union set of VOC 2007 trainval and VOC 2012 trainval ("07+12", 16551 images). The settings of RPN, training, and testing followed that of (Ren et al. 2015). The number of output region proposals was fixed at 300. We enumerated the entire training set with trained DeRPN and RPN to collect their region proposals. Then, we made statistics of IoU distribution, which are illustrated in Fig. 3. It can be seen that the foreground (IoU$\geq$0.5) ratio of DeRPN is almost twice that of RPN, which promotes balance between the foreground and background samples since the former are usually far fewer than the latter. As demonstrated in previous studies in region scaling (Zhang et al. 2018a), hard example mining (Shrivastava, Gupta, and Girshick 2016), and sampling strategy (Cai et al. 2016), the sample balance has a deep impact on the recall rate, which can significantly influence the training effect. Without bells and whistles, DeRPN inherently has the good property of a higher foreground ratio.

According to Fig. 3, we can also evaluate the mean IoU for all region proposals. The calculated mean IoUs of DeRPN and RPN are 0.34 and 0.22, respectively. This reveals that region proposals of DeRPN are more accurate and enclose objects more tightly than RPN. Furthermore, we evaluated the recall rate for DeRPN and RPN on the VOC 2007 test set (~5k images). Different IoU thresholds within [0.5, 0.9] were adopted to verify the efficacy of our method. From Table 1, we can see that the recall rate of DeRPN surpassed that of RPN by a large margin, especially for higher IoUs within [0.7, 0.9]. To sum up, our proposed DeRPN has better properties than RPN, including a higher foreground ratio, more accurate region proposals, and improved recall rate. The superiority of DeRPN benefits from its advanced dimension-decomposition mechanism, which

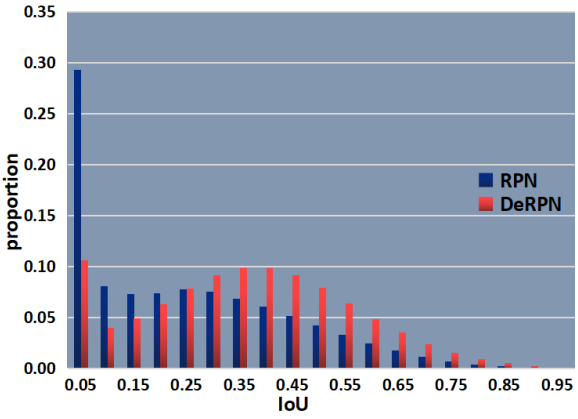

Figure 3: IoU distribution of RPN and DeRPN.

Table 1: Recall rates under different IoUs, evaluated on PASCAL VOC 2007 test set.

IoU	0.5	0.6	0.7	0.75	0.8	0.85	0.9
RPN	96.58	93.19	81.3	64.8	38.5	14.8	3.7
DeRPN	**96.60**	**93.62**	**85.5**	**72.1**	**49.0**	**25.1**	**9.2**

Table 2: Detection results on MS COCO (validation set and test set). Detector is Faster R-CNN (VGG16).

Method	Anchor Type	val						test					
		AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RPN	coco-type	24.3	45.1	23.8	7.7	27.5	38.8	24.4	45.5	23.8	7.7	26.3	37.4
RPN	voc-type	22.9	42.6	22.2	5.6	26.2	37.8	22.8	42.7	22.3	5.6	24.9	36.3
DeRPN	fixed	**25.7**	**46.9**	**25.5**	**9.4**	**28.2**	**39.1**	**25.5**	**47.3**	**25.4**	**9.2**	**26.9**	**38.3**

Table 3: Detection results on PASCAL VOC 2007 and 2012 test sets. Detector is Faster R-CNN.

Method	Backbone	VOC 2007		VOC 2012	
		Data	mAP	Data	mAP
RPN	VGG16	07+12	73.2	07++12	70.4
DeRPN	VGG16	07+12	**76.5**	07++12	**71.9**
RPN	Res101	07+12	76.4	07++12	73.8
DeRPN	Res101	07+12	**78.8**	07++12	**76.5**

considerably reduced the heavy burden from variant object shapes. Through flexible anchor strings, the object width and height independently seek the most appropriate regression references. After combining accurate predicted segments, DeRPN is able to produce high-quality region proposals for diverse objects.

Experiments on PASCAL VOC In order to verify the overall improvement, DeRPN was integrated into a classical framework, Faster R-CNN (FRCN) (Ren et al. 2015). Note that the original FRCN adopts RPN to generate region proposals. We replaced RPN with DeRPN to constitute a new detector, and then conducted comparisons. We still fixed the hyperparameters of DeRPN without any modifications. All other settings including RPN, training, and testing were maintained to be the same as FRCN. The models were implemented based on the famous networks VGG16 and ResNet-101 (He et al. 2016).

First, we trained our models on the set of "07+12" and tested them on the VOC 2007 test set. The experimental results are presented in Table 3. The VGG16 result of DeRPN is 76.5%, which is higher than the 73.2% of RPN. Likewise, the ResNet-101 result of DeRPN surpassed that of RPN by 2.4%. In addition, we used the union set of VOC 2007 trainval+test and V0C 2012 trainval ("07++12", 21503 images) to train our models. The test set is VOC 2012 test (10991 images). From Table 3, we can see DeRPN still retains its superiority, i.e., **71.9**% against 70.4% and **76.5**% against 73.8%.

Experiments on MS COCO In addition, we verified our method on MS COCO 2017, which consists of a training set ($\sim$118k images), test set ($\sim$20k images) and validation set (5k images). We utilized the Faster R-CNN (VGG16) to evaluate the performance of DeRPN and RPN. It is worth noticing that, to handle small objects in MS COCO, Ren et al. redesigned the anchor boxes of RPN, changing the scales from [8,16,32] to [4,8,16,32]. We call the anchor boxes used in MS COCO the coco-type, with voc-type for PASCAL VOC. To investigate the gap between these two types of anchor boxes, we implemented RPN with coco-type and voc-type anchor boxes on MS COCO. In addition, the hyperparameters of DeRPN including the anchor strings were still unchanged, which are denoted as "fixed". The results are listed in Table 2, from which we see that DeRPN still outperforms RPN on both the test and validation sets. The performance of the RPN-based detector was significantly curtailed when we used voc-type anchor boxes on MS COCO. This phenomenon demonstrates that the anchor boxes in RPN are very sensitive. Improper settings of anchor boxes will dramatically decrease the detection accuracy. Nevertheless, there is no such problem for DeRPN.

Scene Text Detection

This section further verifies the adaptivity of DeRPN via scene text detection. Note that scene texts are very challenging for their complex morphology. In general, practical researches revealed that we cannot obtain satisfactory results on scene text datasets by directly using general object detectors (Liao et al. 2017; Liu and Jin 2017). However, DeRPN can be directly employed on any task without any modifications for hyperparameters or specialized optimization.

Dataset We carried out experiments on two benchmarks: ICDAR 2013 (Karatzas et al. 2013) and COCO-text (Veit et al. 2016). ICDAR 2013 contains 229 training images and 233 test images captured from real-world scene. Besides, we append another 4k images that we gathered to enrich the training set. As for the COCO-Text dataset, it is currently the largest dataset for scene text detection. This dataset is based on MS COCO and contains 43,686 training images, 10,000 validation images, and 10,000 test images.

Experiments on ICDAR 2013 We employed FRCN, R-FCN (Dai et al. 2016), and Mask R-CNN (He et al. 2017a) (MRCN) to evaluate our method. As MRCN adopted Feature Pyramid Network (FPN) (Lin et al. 2017), we appended DeRPN on pyramid levels of $\{P_3, P_4, P_5\}$ in FPN. There are three different evaluation protocols for ICDAR 2013: DetEval, ICDAR 2013 evaluation, and IoU evaluation. Under different protocols, the final results are presented as recall, precision and F-measure. As a trade-off, the F-measure is the harmonic mean of recall and precision. From Table 4, it is apparent that DeRPN exhibits superior performance for the three different protocols. The F-measure improvement of DeRPN benefits from the significant increase in the recall rate. In Fig. 4, the presented detection examples also reveal that DeRPN can effectively recall small texts and accurately detect complete long texts of extreme aspect ratios. This further demonstrates that our dimension-decomposition mechanism enables DeRPN to treat variant object shapes reasonably well. We also evaluated the inference time on a single

Table 4: Detection results on ICDAR 2013 test set. Values are expressed in recall/precision/F-measure format.

Method	Proposal	Backbone	FPS	ICDAR2013	DetEval	IoU
FRCN	RPN	VGG16	16.3	70.25 / 84.71 / 76.80	70.90 / 85.16 / 77.38	72.60 / 86.41 / 78.91
	DeRPN	VGG16	15.4	**77.46** / **86.79** / **81.86**	**78.06** / **87.28** / **82.42**	**79.54** / **89.15** / **84.07**
R-FCN	RPN	ResNet-101	13.3	81.77 / **92.67** / 86.88	82.26 / **93.08** / 87.34	80.64 / 91.79 / 85.85
	DeRPN	ResNet-101	12.8	**86.52** / 92.21 / **89.28**	**86.68** / 92.62 / **89.55**	**86.85** / **92.24** / **89.46**
MRCN	RPN	ResNet-101	4.04	84.55 / **94.67** / 89.32	85.02 / **94.69** / 89.60	84.02 / **93.69** / 88.59
	DeRPN	ResNet-101	5.02	**87.80** / 93.71 / **90.66**	**88.60** / 93.94 / **91.19**	**86.12** / 92.18 / **89.05**

Figure 4: Detection examples for ICDAR 2013 test set. Yellow boxes are detected results of DeRPN-based detector, and blue boxes are from RPN-based detector. Green dashed lines denote ground truths.

TITAN XP GPU. In Table 4, the FPS of DeRPN is approximately the same as that of RPN.

Experiments on COCO-Text We conducted experiments on COCO-Text to investigate the effects of different anchor types. We first used coco-type anchor boxes for RPN in the R-FCN baseline. In addition, we attempted to manually design anchor boxes. By imitating the methods of (Zhang et al. 2018a), we changed the aspect ratio from [0.5, 1, 2] to [0.25, 0.5, 1] because the scene texts are usually long and narrow. In addition, by following the method proposed by (Redmon and Farhadi 2017), we enumerated the whole training set of COCO-Text to conduct k-means clustering and then produced 12 anchor boxes for RPN. The hyperparameters of DeRPN still remained unchanged.

The results in Table 5 were evaluated by the COCO-Text protocol. In Table 5, the result is marginally worse than the baseline (coco-type) when employing manually designed anchor boxes, which reveals that it is difficult to guarantee a good result by manually setting anchor boxes for RPN. In addition, the K-means method helps increase performance to some extent. However, this improvement is still limited. When increasing the IoU to 0.75, the average precision of the K-means method is even worse than the baseline. As for DeRPN, without any modifications, it surpassed all other methods by a large margin (more than **10%**).

Furthermore, we compared DeRPN with other specific scene text detectors including (Zhou et al. 2017), (Lyu et al. 2018), (Liao, Shi, and Bai 2018) and so on. As these methods are evaluated by recall, precision, and F-measure, we also followed this criteria to verify our method. All listed results in Table 6 are under the constraints of single-scale testing and single model. From Table 6, we can see that DeRPN achieved a highest F-measure of 57.11% and established a new state-of-the-art result for COCO-Text. Unlike the other listed methods, DeRPN is designed for general object detection, which means DeRPN does not employ specialized optimization for scene text detection. However, owing to its adaptivity, DeRPN still applies to scene text detection and even outperforms specific scene text detectors.

Table 5: Effects of anchor types, evaluated on COCO-Text test set. These methods are based on R-FCN (ResNet-101).

Method	Anchor Type	AP 50	Recall 50	AP 75	Recall 75
RPN	coco-type	40.46	56.76	13.67	23.43
RPN	manually	39.76	56.64	13.49	22.62
RPN	K-means	42.59	62.21	13.19	24.42
DeRPN	fixed	**53.43**	**73.74**	**16.99**	**30.14**

Table 6: Comparison of specific scene text detectors. Results were evaluated on COCO-Text test set. Baseline results are given by (Veit et al. 2016).

Method	Recall	Precision	F-measure
Baseline A (2016)	23.30	83.78	36.48
Baseline B (2016)	10.70	**89.73**	19.14
Baseline C (2016)	4.70	18.56	7.47
Yao et al. (2016)	27.10	43.20	33.30
WordSup (2017)	30.90	45.20	36.80
He et al. (2017)	31.00	46.00	37.00
Lyu et al. (2018)	26.20	69.90	38.10
EAST (2017)	32.40	50.39	39.45
TextBoxes++ (2018)	**56.00**	55.82	55.91
R-FCN+DeRPN (ours)	55.71	58.58	**57.11**

Conclusion

In this paper, we have proposed a novel DeRPN to improve the adaptivity of detectors. Through an advanced dimension-decomposition mechanism, DeRPN can be employed directly on different tasks, datasets, or models without any modifications for hyperparameters or specialized optimization. Comprehensive evaluations demonstrated that the proposed DeRPN can significantly outperform the previous RPN. To our best knowledge, DeRPN is the first method that achieves outstanding performances in both general object detection and scene text detection without any tuning.

In the future, we will try to apply the dimension decomposition mechanism to one-stage detectors and then improve their adaptivity.

Acknowledgement

This research is supported in part by GD-NSF (no.2017A030312006), the National Key Research and Development Program of China (No. 2016YFB1001405), NSFC (Grant No.: 61472144, 61673182, 61771199), and GDSTP (Grant No.:2017A010101027), GZSTP (no. 201607010227).

References

Cai, Z.; Fan, Q.; Feris, R. S.; and Vasconcelos, N. 2016. A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV*, 354–370.

Dai, J.; Li, Y.; He, K.; and Sun, J. 2016. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*, 379–387.

Dai, J.; Qi, H.; Xiong, Y.; Li, Y.; and et. al. 2017. Deformable convolutional networks. In *ICCV*, 764–773.

Deng, D.; Liu, H.; Li, X.; and Cai, D. 2018. Pixellink: Detecting scene text via instance segmentation. In *AAAI*, 6773–6780.

Everingham, M.; Van Gool, L.; Williams, C. K.; and et al. 2010. The pascal visual object classes (voc) challenge. *IJCV* 88(2):303–338.

Felzenszwalb, P. F.; Girshick, R. B.; McAllester, D.; and Ramanan, D. 2010. Object detection with discriminatively trained part-based models. *TPAMI* 32(9):1627–1645.

Girshick, R. 2015. Fast r-cnn. In *ICCV*, 1440–1448.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*, 770–778.

He, K.; Gkioxari, G.; Dollár, P.; and Girshick, R. 2017a. Mask r-cnn. In *ICCV*, 2980–2988.

He, P.; Huang, W.; He, T.; Zhu, Q.; Qiao, Y.; and Li, X. 2017b. Single shot text detector with regional attention. In *ICCV*, 3066–3074.

Hu, H.; Zhang, C.; Luo, Y.; Wang, Y.; Han, J.; and Ding, E. 2017. Wordsup: Exploiting word annotations for character based text detection. In *ICCV*, 4950–4959.

Karatzas, D.; Shafait, F.; Uchida, S.; and et al. 2013. Icdar 2013 robust reading competition. In *ICDAR*, 1484–1493.

Li, Z.; Chen, Y.; Yu, G.; and Deng, Y. 2018. R-fcn++: Towards accurate region-based fully convolutional networks for object detection. In *AAAI*, 7073–7080.

Liao, M.; Shi, B.; Bai, X.; Wang, X.; and Liu, W. 2017. Textboxes: A fast text detector with a single deep neural network. In *AAAI*, 4161–4167.

Liao, M.; Shi, B.; and Bai, X. 2018. Textboxes++: A single-shot oriented scene text detector. *TIP* 27(8):3676–3690.

Lin, T.-Y.; Maire, M.; Belongie, S.; and et. al. 2014. Microsoft coco: Common objects in context. In *ECCV*, 740–755.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature pyramid networks for object detection. In *CVPR*, 2117–2125.

Lin, T.-Y.; Goyal, P.; Girshick, R.; and et. al. 2018. Focal loss for dense object detection. *TPAMI* PP(99):2999–3007.

Liu, Y., and Jin, L. 2017. Deep matching prior network: Toward tighter multi-oriented text detection. In *CVPR*, 3454–3461.

Lyu, P.; Yao, C.; Wu, W.; Yan, S.; and Bai, X. 2018. Multi-oriented scene text detection via corner localization and region segmentation. In *CVPR*, 7553–7563.

Ma, J.; Shao, W.; Ye, H.; Wang, L.; Wang, H.; Zheng, Y.; and Xue, X. 2017. Arbitrary-oriented scene text detection via rotation proposals. *TMM* PP(99):1–1.

Redmon, J., and Farhadi, A. 2017. Yolo9000: Better, faster, stronger. In *CVPR*, 6517–6525.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 91–99.

Shrivastava, A.; Gupta, A.; and Girshick, R. 2016. Training region-based object detectors with online hard example mining. In *CVPR*, 761–769.

Simonyan, K., and Zisserman, A. 2015. Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Singh, B.; Li, H.; Sharma, A.; and Davis, L. S. 2018. R-fcn-3000 at 30fps: Decoupling detection and classification. In *CVPR*, 1081–1090.

Tian, S.; Pan, Y.; Huang, C.; Lu, S.; Yu, K.; and Lim Tan, C. 2015. Text flow: A unified text detection system in natural scene images. In *ICCV*, 4651–4659.

Uijlings, J. R.; Van De Sande, K. E.; Gevers, T.; and Smeulders, A. W. 2013. Selective search for object recognition. *IJCV* 104(2):154–171.

Veit, A.; Matera, T.; Neumann, L.; and et. al. 2016. Cocotext: Dataset and benchmark for text detection and recognition in natural images. *CoRR* abs/1601.07140.

Viola, P., and Jones, M. 2001. Rapid object detection using a boosted cascade of simple features. In *CVPR*, I–I.

Yao, C.; Bai, X.; Sang, N.; Zhou, X.; Zhou, S.; and Cao, Z. 2016. Scene text detection via holistic, multi-channel prediction. *CoRR* abs/1606.09002.

Ye, Q., and Doermann, D. 2015. Text detection and recognition in imagery: A survey. *TPAMI* 37(7):1480–1500.

Zhang, Z.; Zhang, C.; Shen, W.; Yao, C.; Liu, W.; and Bai, X. 2016. Multi-oriented text detection with fully convolutional networks. In *CVPR*, 4159–4167.

Zhang, S.; Liu, Y.; Jin, L.; and Luo, C. 2018a. Feature enhancement network: A refined scene text detector. In *AAAI*, 2612–2619.

Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; and Li, S. Z. 2018b. Single-shot refinement neural network for object detection. In *CVPR*, 4203–4212.

Zhou, X.; Yao, C.; Wen, H.; Wang, Y.; Zhou, S.; He, W.; and Liang, J. 2017. East: An efficient and accurate scene text detector. In *CVPR*, 2642–2651.

Zitnick, C. L., and Dollár, P. 2014. Edge boxes: Locating object proposals from edges. In *ECCV*, 391–405.