Detecting Text in Natural Image with Connectionist Text Proposal Network

Zhi Tian[1], Weilin Huang*[1,2], Tong He[1], Pan He[1], and Yu Qiao[1,3]

[1]Shenzhen Key Lab of Comp. Vis and Pat. Rec.,
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
[2]University of Oxford [3]The Chinese University of Hong Kong
{zhi.tian;wl.huang;tong.he;pan.he;yu.qiao@siat.ac.cn}

Abstract. We propose a novel Connectionist Text Proposal Network (CTPN) that *accurately* localizes text lines in natural image. The CTPN detects a text line in a sequence of fine-scale text proposals directly in convolutional feature maps. We develop a vertical anchor mechanism that jointly predicts location and text/non-text score of each fixed-width proposal, considerably improving localization accuracy. The sequential proposals are naturally connected by a recurrent neural network, which is seamlessly incorporated into the convolutional network, resulting in an end-to-end trainable model. This allows the CTPN to explore rich context information of image, making it powerful to detect extremely ambiguous text. The CTPN works reliably on multi-scale and multi-language text without further post-processing, departing from previous bottom-up methods requiring multi-step post filtering. It achieves 0.88 and 0.61 F-measure on the ICDAR 2013 and 2015 benchmarks, surpassing recent results [8,35] by a large margin. The CTPN is computationally efficient with 0.14s/image, by using the very deep VGG16 model [27]. Online demo is available at: http://textdet.com/.

Keywords: Scene text detection, convolutional network, recurrent neural network, anchor mechanism

1 Introduction

Reading text in natural image has recently attracted increasing attention in computer vision [8,14,15,10,35,11,9,1,28,32]. This is due to its numerous practical applications such as image OCR, multi-language translation, image retrieval, etc. It includes two sub tasks: text detection and recognition. This work focus on the detection task [14,1,28,32], which is more challenging than recognition task carried out on a well-cropped word image [15,9]. Large variance of text patterns and highly cluttered background pose main challenge of *accurate* text localization.

* Corresponding author

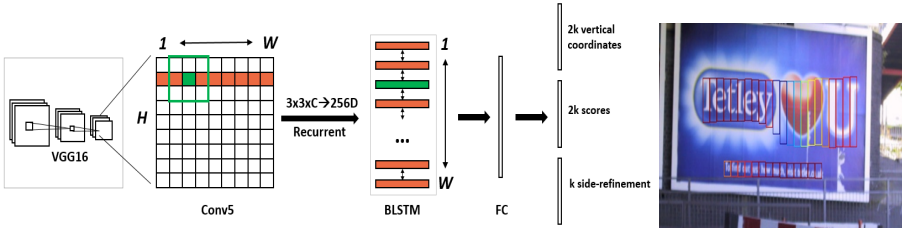

Fig. 1: (a) Architecture of the Connectionist Text Proposal Network (CTPN). We densely slide a 3×3 spatial window through the last convolutional maps (*conv5*) of the VGG16 model [27]. The sequential windows in each row are recurrently connected by a Bi-directional LSTM (BLSTM) [7], where the convolutional feature (3×3×C) of each window is used as input of the 256D BLSTM (including two 128D LSTMs). The RNN layer is connected to a 512D fully-connected layer, followed by the output layer, which jointly predicts text/non-text scores, y-axis coordinates and side-refinement offsets of k anchors. (b) The CTPN outputs sequential fixed-width fine-scale text proposals. Color of each box indicates the text/non-text score. Only the boxes with positive scores are presented.

Current approaches for text detection mostly employ a bottom-up pipeline [28,1,14,32,33]. They commonly start from low-level character or stroke detection, which is typically followed by a number of subsequent steps: non-text component filtering, text line construction and text line verification. These multi-step bottom-up approaches are generally complicated with less robustness and reliability. Their performance heavily rely on the results of character detection, and connected-components methods or sliding-window methods have been proposed. These methods commonly explore low-level features (e.g., based on SWT [3,13], MSER [14,33,23], or HoG [28]) to distinguish text candidates from background. However, they are not robust by identifying individual strokes or characters separately, without context information. For example, it is more confident for people to identify a sequence of characters than an individual one, especially when a character is extremely ambiguous. These limitations often result in a large number of non-text components in character detection, causing main difficulties for handling them in following steps. Furthermore, these false detections are easily accumulated sequentially in bottom-up pipeline, as pointed out in [28]. To address these problems, we exploit strong deep features for detecting text information directly in convolutional maps. We develop text anchor mechanism that *accurately* predicts text locations in fine scale. Then, an in-network recurrent architecture is proposed to connect these fine-scale text proposals in sequences, allowing them to encode rich context information.

Deep Convolutional Neural Networks (CNN) have recently advanced general object detection substantially [25,5,6]. The state-of-the-art method is Faster Region-CNN (R-CNN) system [25] where a Region Proposal Network (RPN) is

proposed to generate high-quality class-agnostic object proposals directly from convolutional feature maps. Then the RPN proposals are fed into a Fast R-CNN [5] model for further classification and refinement, leading to the state-of-the-art performance on generic object detection. *However, it is difficult to apply these general object detection systems directly to scene text detection, which generally requires a higher localization accuracy.* In generic object detection, each object has a well-defined closed boundary [2], while such a well-defined boundary may not exist in text, since a text line or word is composed of a number of separate characters or strokes. For object detection, a typical correct detection is defined loosely, e.g., by an overlap of > 0.5 between the detected bounding box and its ground truth (e.g., the PASCAL standard [4]), since people can recognize an object easily from major part of it. By contrast, reading text comprehensively is a fine-grained recognition task which requires a correct detection that covers a full region of a text line or word. Therefore, text detection generally requires a more *accurate* localization, leading to a different evaluation standard, e.g., the Wolf's standard [30] which is commonly employed by text benchmarks [19,21].

In this work, we fill this gap by extending the RPN architecture [25] to *accurate* text line localization. We present several technical developments that tailor generic object detection model elegantly towards our problem. We strive for a further step by proposing an in-network recurrent mechanism that allows our model to detect text sequence directly in the convolutional maps, avoiding further post-processing by an additional costly CNN detection model.

1.1 Contributions

We propose a novel Connectionist Text Proposal Network (CTPN) that directly localizes text sequences in convolutional layers. This overcomes a number of main limitations raised by previous bottom-up approaches building on character detection. We leverage the advantages of strong deep convolutional features and sharing computation mechanism, and propose the CTPN architecture which is described in Fig. 1. It makes the following major contributions:

First, we cast the problem of text detection into localizing a sequence of fine-scale text proposals. We develop an anchor regression mechanism that jointly predicts vertical location and text/non-text score of each text proposal, resulting in an excellent localization accuracy. This departs from the RPN prediction of a whole object, which is difficult to provide a satisfied localization accuracy.

Second, we propose an in-network recurrence mechanism that elegantly connects sequential text proposals in the convolutional feature maps. This connection allows our detector to explore meaningful context information of text line, making it powerful to detect extremely challenging text reliably.

Third, both methods are integrated seamlessly to meet the nature of text sequence, resulting in a unified end-to-end trainable model. Our method is able to handle multi-scale and multi-lingual text in a single process, avoiding further post filtering or refinement.

Fourth, our method achieves new state-of-the-art results on a number of benchmarks, significantly improving recent results (e.g., 0.88 F-measure over 0.83

in [8] on the ICDAR 2013, and 0.61 F-measure over 0.54 in [35] on the ICDAR 2015). Furthermore, it is computationally efficient, resulting in a $0.14s$/image running time (on the ICDAR 2013) by using the very deep VGG16 model [27].

2 Related Work

Text detection. Past works in scene text detection have been dominated by bottom-up approaches which are generally built on stroke or character detection. They can be roughly grouped into two categories, connected-components (CCs) based approaches and sliding-window based methods. The CCs based approaches discriminate text and non-text pixels by using a fast filter, and then text pixels are greedily grouped into stroke or character candidates, by using low-level properties, e.g., intensity, color, gradient, etc. [33,14,32,13,3]. The sliding-window based methods detect character candidates by densely moving a multi-scale window through an image. The character or non-character window is discriminated by a pre-trained classifier, by using manually-designed features [28,29], or recent CNN features [16]. However, both groups of methods commonly suffer from poor performance of character detection, causing accumulated errors in following component filtering and text line construction steps. Furthermore, robustly filtering out non-character components or confidently verifying detected text lines are even difficult themselves [1,33,14]. Another limitation is that the sliding-window methods are computationally expensive, by running a classifier on a huge number of the sliding windows.

Object detection. Convolutional Neural Networks (CNN) have recently advanced general object detection substantially [25,5,6]. A common strategy is to generate a number of object proposals by employing inexpensive low-level features, and then a strong CNN classifier is applied to further classify and refine the generated proposals. Selective Search (SS) [4] which generates class-agnostic object proposals, is one of the most popular methods applied in recent leading object detection systems, such as Region CNN (R-CNN) [6] and its extensions [5]. Recently, Ren *et al.* [25] proposed a Faster R-CNN system for object detection. They proposed a Region Proposal Network (RPN) that generates high-quality class-agnostic object proposals directly from the convolutional feature maps. The RPN is fast by sharing convolutional computation. However, the RPN proposals are not discriminative, and require a further refinement and classification by an additional costly CNN model, e.g., the Fast R-CNN model [5]. More importantly, text is different significantly from general objects, making it difficult to directly apply general object detection system to this highly domain-specific task.

3 Connectionist Text Proposal Network

This section presents details of the Connectionist Text Proposal Network (CTPN). It includes three key contributions that make it reliable and accurate for text localization: detecting text in fine-scale proposals, recurrent connectionist text proposals, and side-refinement.

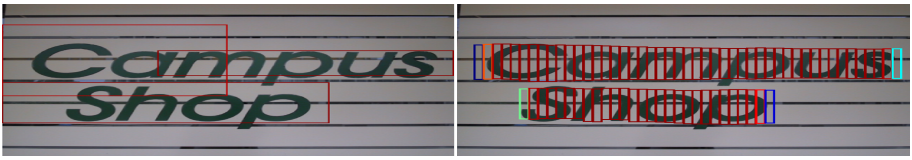

Fig. 2: **Left**: RPN proposals. **Right**: Fine-scale text proposals.

3.1 Detecting Text in Fine-scale Proposals

Similar to Region Proposal Network (RPN) [25], the CTPN is essentially a fully convolutional network that allows an input image of arbitrary size. It detects a text line by densely sliding a small window in the convolutional feature maps, and outputs a sequence of fine-scale (e.g., fixed 16-pixel width) text proposals, as shown in Fig. 1 (b).

We take the very deep 16-layer vggNet (VGG16) [27] as an example to describe our approach, which is readily applicable to other deep models. Architecture of the CTPN is presented in Fig. 1 (a). We use a small spatial window, 3×3, to slide the feature maps of last convolutional layer (e.g., the *conv5* of the VGG16). The size of *conv5* feature maps is determined by the size of input image, while the total stride and receptive field are fixed as 16 and 228 pixels, respectively. Both the total stride and receptive field are fixed by the network architecture. Using a sliding window in the convolutional layer allows it to share convolutional computation, which is the key to reduce computation of the costly sliding-window based methods.

Generally, sliding-window methods adopt multi-scale windows to detect objects of different sizes, where one window scale is fixed to objects of similar size. In [25], Ren *et al.* proposed an efficient anchor regression mechanism that allows the RPN to detect multi-scale objects with a single-scale window. The key insight is that a single window is able to predict objects in a wide range of scales and aspect ratios, by using a number of flexible anchors. We wish to extend this efficient anchor mechanism to our text task. However, text differs from generic objects substantially, which generally have a well-defined enclosed boundary and center, allowing inferring whole object from even a part of it [2]. Text is a sequence which does not have an obvious closed boundary. It may include multi-level components, such as stroke, character, word, text line and text region, which are not distinguished clearly between each other. Text detection is defined in word or text line level, so that it may be easy to make an incorrect detection by defining it as a single object, e.g., detecting part of a word. Therefore, directly predicting the location of a text line or word may be difficult or unreliable, making it hard to get a satisfied accuracy. An example is shown in Fig. 2, where the RPN is directly trained for localizing text lines in an image.

We look for a unique property of text that is able to generalize well to text components in all levels. We observed that word detection by the RPN is difficult to accurately predict the horizontal sides of words, since each character within a word is isolated or separated, making it confused to find the start and

end locations of a word. Obviously, a text line is a sequence which is the main difference between text and generic objects. It is natural to consider a text line as a sequence of fine-scale text proposals, where each proposal generally represents a small part of a text line, e.g., a text piece with 16-pixel width. Each proposal may include a single or multiple strokes, a part of a character, a single or multiple characters, etc. We believe that it would be more accurate to just predict the vertical location of each proposal, by fixing its horizontal location which may be more difficult to predict. This reduces the search space, compared to the RPN which predicts 4 coordinates of an object. We develop a vertical anchor mechanism that simultaneously predicts a text/non-text score and y-axis location of each fine-scale proposal. It is also more reliable to detect a general fixed-width text proposal than identifying an isolate character, which is easily confused with part of a character or multiple characters. Furthermore, detecting a text line in a sequence of fixed-width text proposals also works reliably on text of multiple scales and multiple aspect ratios.

To this end, we design the fine-scale text proposal as follow. Our detector investigates each spatial location in the *conv5* densely. A text proposal is defined to have a fixed width of 16 pixels (in the input image). This is equal to move the detector densely through the *conv5* maps, where the total stride is exactly 16 pixels. Then we design k vertical anchors to predict y-coordinates for each proposal. The k anchors have a same horizontal location with a fixed width of 16 pixels, but their vertical locations are varied in k different heights. In our experiments, we use ten anchors for each proposal, $k = 10$, whose heights are varied from 11 to 273 pixels (by $\div 0.7$ each time) in the input image. The explicit vertical coordinates are measured by the height and y-axis center of a proposal bounding box. We compute relative predicted vertical coordinates ($\mathbf{v}$) with respect to the bounding box location of an anchor as,

$$v_c = (c_y - c_y^a)/h^a, \qquad v_h = \log(h/h^a) \tag{1}$$

$$v_c^* = (c_y^* - c_y^a)/h^a, \qquad v_h^* = \log(h^*/h^a) \tag{2}$$

where $\mathbf{v} = \{v_c, v_h\}$ and $\mathbf{v}^* = \{v_c^*, v_h^*\}$ are the relative predicted coordinates and ground truth coordinates, respectively. c_y^a and h^a are the center (y-axis) and height of the anchor box, which can be pre-computed from an input image. c_y and h are the predicted y-axis coordinates in the input image, while c_y^* and h^* are the ground truth coordinates. Therefore, each predicted text proposal has a bounding box with size of $h \times 16$ (in the input image), as shown in Fig. 1 (b) and Fig. 2 (right). Generally, an text proposal is largely smaller than its effective receptive field which is 228×228.

The detection processing is summarised as follow. Given an input image, we have $W \times H \times C$ *conv5* features maps (by using the VGG16 model), where C is the number of feature maps or channels, and $W \times H$ is the spatial arrangement. When our detector is sliding a 3×3 window densely through the conv5, each sliding-window takes a convolutional feature of $3 \times 3 \times C$ for producing the prediction. For each prediction, the horizontal location (x-coordinates) and k-anchor locations are fixed, which can be pre-computed by mapping the spatial

Fig. 3: **Top**: CTPN without RNN. **Bottom**: CTPN with RNN connection.

window location in the *conv5* onto the input image. Our detector outputs the text/non-text scores and the predicted y-coordinates ($\mathbf{v}$) for k anchors at each window location. The detected text proposals are generated from the anchors having a text/non-text score of > 0.7 (with non-maximum suppression). By the designed vertical anchor and fine-scale detection strategy, our detector is able to handle text lines in a wide range of scales and aspect ratios by using a single-scale image. This further reduces its computation, and at the same time, predicting accurate localizations of the text lines. Compared to the RPN or Faster R-CNN system [25], our fine-scale detection provides more detailed supervised information that naturally leads to a more accurate detection.

3.2 Recurrent Connectionist Text Proposals

To improve localization accuracy, we split a text line into a sequence of fine-scale text proposals, and predict each of them separately. Obviously, it is not robust to regard each isolated proposal independently. This may lead to a number of false detections on non-text objects which have a similar structure as text patterns, such as windows, bricks, leaves, etc. (referred as text-like outliers in [13]). It is also possible to discard some ambiguous patterns which contain weak text information. Several examples are presented in Fig. 3 (top). Text have strong sequential characteristics where the sequential context information is crucial to make a reliable decision. This has been verified by recent work [9] where a recurrent neural network (RNN) is applied to encode this context information for text recognition. Their results have shown that the sequential context information is greatly facilitate the recognition task on cropped word images.

Motivated from this work, we believe that this context information may also be of importance for our detection task. Our detector should be able to explore this important context information to make a more reliable decision, when it works on each individual proposal. Furthermore, we aim to encode this information directly in the convolutional layer, resulting in an elegant and seamless

in-network connection of the fine-scale text proposals. RNN provides a natural choice for encoding this information recurrently using its hidden layers. To this end, we propose to design a RNN layer upon the *conv5*, which takes the convolutional feature of each window as sequential inputs, and updates its internal state recurrently in the hidden layer, H_t,

$$H_t = \varphi(H_{t-1}, X_t), \qquad t = 1, 2, ..., W \tag{3}$$

where $X_t \in R^{3\times3\times C}$ is the input *conv5* feature from t-th sliding-window (3×3). The sliding-window moves densely from left to right, resulting in $t = 1, 2, ..., W$ sequential features for each row. W is the width of the *conv5*. H_t is a recurrent internal state that is computed jointly from both current input (X_t) and previous states encoded in H_{t-1}. The recurrence is computed by using a non-linear function φ, which defines exact form of the recurrent model. We exploit the long short-term memory (LSTM) architecture [12] for our RNN layer. The LSTM was proposed specially to address vanishing gradient problem, by introducing three additional multiplicative gates: the *input gate*, *forget gate* and *output gate*. Details can be found in [12]. Hence the internal state in RNN hidden layer accesses the sequential context information scanned by all previous windows through the recurrent connection. We further extend the RNN layer by using a bi-directional LSTM, which allows it to encode the recurrent context in both directions, so that the connectionist receipt field is able to cover the whole image width, e.g., 228 × width. We use a 128D hidden layer for each LSTM, resulting in a 256D RNN hidden layer, $H_t \in R^{256}$.

The internal state in H_t is mapped to the following FC layer, and output layer for computing the predictions of the t-th proposal. Therefore, our integration with the RNN layer is elegant, resulting in an efficient model that is end-to-end trainable without additional cost. The efficiency of the RNN connection is demonstrated in Fig. 3. Obviously, it reduces false detections considerably, and at the same time, recovers many missed text proposals which contain very weak text information.

3.3 Side-refinement

The fine-scale text proposals are detected accurately and reliably by our CTPN. Text line construction is straightforward by connecting continuous text proposals whose text/non-text score is > 0.7. Text lines are constructed as follow. First, we define a paired neighbour (B_j) for a proposal B_i as $B_j -> B_i$, when (i) B_j is the nearest horizontal distance to B_i, and (ii) this distance is less than 50 pixels, and (iii) their vertical overlap is > 0.7. Second, two proposals are grouped into a pair, if $B_j -> B_i$ and $B_i -> B_j$. Then a text line is constructed by sequentially connecting the pairs having a same proposal.

The fine-scale detection and RNN connection are able to predict accurate localizations in vertical direction. In horizontal direction, the image is divided into a sequence of equal 16-pixel width proposals. This may lead to an inaccurate localization when the text proposals in both horizontal sides are not exactly

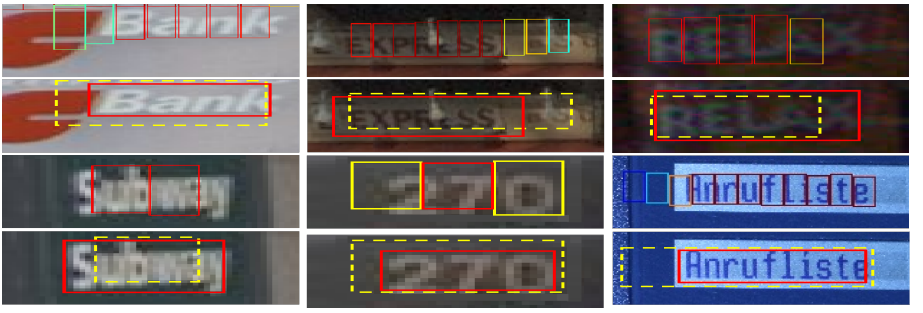

Fig. 4: CTPN detection with (red box) and without (yellow dashed box) the side-refinement. Color of fine-scale proposal box indicate a text/non-text score.

covered by a ground truth text line area, or some side proposals are discarded (e.g., having a low text score), as shown in Fig. 4. This inaccuracy may be not crucial in generic object detection, but should not be ignored in text detection, particularly for those small-scale text lines or words. To address this problem, we propose a side-refinement approach that accurately estimates the offset for each anchor/proposal in both left and right horizontal sides (referred as side-anchor or side-proposal). Similar to the y-coordinate prediction, we compute relative offset as,

$$o = (x_{side} - c_x^a)/w^a, \quad o^* = (x_{side}^* - c_x^a)/w^a \tag{4}$$

where x_{side} is the predicted x-coordinate of the nearest horizontal side (e.g., left or right side) to current anchor. x_{side}^* is the ground truth (GT) side coordinate in x-axis, which is pre-computed from the GT bounding box and anchor location. c_x^a is the center of anchor in x-axis. w^a is the width of anchor, which is fixed, $w^a = 16$. The side-proposals are defined as the start and end proposals when we connect a sequence of detected fine-scale text proposals into a text line. We only use the offsets of the side-proposals to refine the final text line bounding box. Several detection examples improved by side-refinement are presented in Fig. 4. The side-refinement further improves the localization accuracy, leading to about 2% performance improvements on the SWT and Multi-Lingual datasets. Notice that the offset for side-refinement is predicted simultaneously by our model, as shown in Fig. 1. It is not computed from an additional post-processing step.

3.4 Model Outputs and Loss Functions

The proposed CTPN has three outputs which are jointly connected to the last FC layer, as shown in Fig. 1 (a). The three outputs simultaneously predict text/non-text scores ($\mathbf{s}$), vertical coordinates ($\mathbf{v} = \{v_c, v_h\}$ in E.q. (2)) and side-refinement offset ($\mathbf{o}$). We explore k anchors to predict them on each spatial location in the *conv5*, resulting in $2k$, $2k$ and k parameters in the output layer, respectively.

We employ multi-task learning to jointly optimize model parameters. We introduce three loss functions, L_s^{cl}, L_v^{re} and l_o^{re}, which compute errors of text/non-text score, coordinate and side-refinement, respectively. With these considerations, we follow the multi-task loss applied in [5,25], and minimize an overall objective function (L) for an image as,

$$L(\mathbf{s}_i, \mathbf{v}_j, \mathbf{o}_k) = \frac{1}{N_s}\sum_i L_s^{cl}(\mathbf{s}_i, \mathbf{s}_i^*) + \frac{\lambda_1}{N_v}\sum_j L_v^{re}(\mathbf{v}_j, \mathbf{v}_j^*) + \frac{\lambda_2}{N_o}\sum_k L_o^{re}(\mathbf{o}_k, \mathbf{o}_k^*) \quad (5)$$

where each anchor is a training sample, and i is the index of an anchor in a mini-batch. $\mathbf{s}_i$ is the predicted probability of anchor i being a true text. $\mathbf{s}_i^* = \{0, 1\}$ is the ground truth. j is the index of an anchor in the set of valid anchors for y-coordinates regression, which are defined as follow. A valid anchor is a defined positive anchor ($\mathbf{s}_j^* = 1$, described below), or has an Intersection-over-Union (IoU) > 0.5 overlap with a ground truth text proposal. $\mathbf{v}_j$ and $\mathbf{v}_j^*$ are the prediction and ground truth y-coordinates associated with the j-th anchor. k is the index of a side-anchor, which is defined as a set of anchors within a horizontal distance (e.g., 32-pixel) to the left or right side of a ground truth text line bounding box. $\mathbf{o}_k$ and $\mathbf{o}_k^*$ are the predicted and ground truth offsets in x-axis associated to the k-th anchor. L_s^{cl} is the classification loss which we use Softmax loss to distinguish text and non-text. L_v^{re} and L_o^{re} are the regression loss. We follow previous work by using the smooth L_1 function to compute them [5,25]. λ_1 and λ_2 are loss weights to balance different tasks, which are empirically set to 1.0 and 2.0. N_s N_v and N_o are normalization parameters, denoting the total number of anchors used by L_s^{cl}, L_v^{re} and L_o^{re}, respectively.

3.5 Training and Implementation Details

The CTPN can be trained end-to-end by using the standard back-propagation and stochastic gradient descent (SGD). Similar to RPN [25], training samples are the anchors, whose locations can be pre computed in input image, so that the training labels of each anchor can be computed from corresponding GT box.

Training labels. For text/non-text classification, a binary label is assigned to each positive (text) or negative (non-text) anchor. It is defined by computing the IoU overlap with the GT bounding box (divided by anchor location). A positive anchor is defined as : (i) an anchor that has an > 0.7 IoU overlap with any GT box; *or* (ii) the anchor with the highest IoU overlap with a GT box. *By the condition (ii), even a very small text pattern can assign a positive anchor. This is crucial to detect small-scale text patterns, which is one of key advantages of the CTPN.* This is different from generic object detection where the impact of condition (ii) may be not significant. The negative anchors are defined as < 0.5 IoU overlap with all GT boxes. The training labels for the y-coordinate regression ($\mathbf{v}^*$) and offset regression ($\mathbf{o}^*$) are computed as E.q. (2) and (4) respectively.

Training data. In the training process, each mini-batch samples are collected randomly from a single image. The number of anchors for each mini-batch is fixed to $N_s = 128$, with 1:1 ratio for positive and negative samples. A

mini-patch is pad with negative samples if the number of positive ones is fewer than 64. Our model was trained on 3,000 natural images, including 229 images from the ICDAR 2013 training set. We collected the other images ourselves and manually labelled them with text line bounding boxes. All self-collected training images are not overlapped with any test image in all benchmarks. The input image is resized by setting its short side to 600 for training, while keeping its original aspect ratio.

Implementation Details. We follow the standard practice, and explore the very deep VGG16 model [27] pre-trained on the ImageNet data [26]. We initialize the new layers (e.g., the RNN and output layers) by using random weights with Gaussian distribution of 0 mean and 0.01 standard deviation. The model was trained end-to-end by fixing the parameters in the first two convolutional layers. We used 0.9 momentum and 0.0005 weight decay. The learning rate was set to 0.001 in the first 16K iterations, followed by another 4K iterations with 0.0001 learning rate. Our model was implemented in Caffe framework [17].

4 Experimental Results and Discussions

We evaluate the CTPN on five text detection benchmarks, namely the ICDAR 2011 [21], ICDAR 2013 [19], ICDAR 2015 [18], SWT [3], and Multilingual dataset [24]. In our experiments, we first verify the efficiency of each proposed component individually, e.g., the fine-scale text proposal detection or in-network recurrent connection. The ICDAR 2013 is used for this component evaluation.

4.1 Benchmarks and Evaluation Metric

The ICDAR 2011 dataset [21] consists of 229 training images and 255 testing ones, where the images are labelled in word level. *The ICDAR 2013* [19] is similar as the ICDAR 2011, and has in total 462 images, including 229 images and 233 images for training and testing, respectively. *The ICDAR 2015* (Incidental Scene Text - Challenge 4) [18] includes 1,500 images which were collected by using the Google Glass. The training set has 1,000 images, and the remained 500 images are used for test. This dataset is more challenging than previous ones by including arbitrary orientation, very small-scale and low resolution text. *The Multilingual* scene text dataset is collected by [24]. It contains 248 images for training and 239 for testing. The images include multi-languages text, and the ground truth is labelled in text line level. Epshtein *et al.* [3] introduced *the SWT* dataset containing 307 images which include many extremely small-scale text.

We follow previous work by using standard evaluation protocols which are provided by the dataset creators or competition organizers. For the ICDAR 2011 we use the standard protocol proposed by [30], the evaluation on the ICDAR 2013 follows the standard in [19]. For the ICDAR 2015, we used the online evaluation system provided by the organizers as in [18]. The evaluations on the SWT and Multilingual datasets follow the protocols defined in [3] and [24] respectively.

4.2 Fine-Scale Text Proposal Network with Faster R-CNN

We first discuss our fine-scale detection strategy against the RPN and Faster R-CNN system [25]. As can be found in Table 1 (left), the individual RPN is difficult to perform accurate text localization, by generating a large amount of false detections (low precision). By refining the RPN proposals with a Fast R-CNN detection model [5], the Faster R-CNN system improves localization accuracy considerably, with a F-measure of 0.75. One observation is that the Faster R-CNN also increases the recall of original RPN. This may benefit from joint bounding box regression mechanism of the Fast R-CNN, which improves the accuracy of a predicted bounding box. The RPN proposals may roughly localize a major part of a text line or word, but they are not accurate enough by the ICDAR 2013 standard. Obviously, the proposed fine-scale text proposal network (FTPN) improves the Faster R-CNN remarkably in both precision and recall, suggesting that the FTPN is more accurate and reliable, by predicting a sequence of fine-scale text proposals rather than a whole text line.

4.3 Recurrent Connectionist Text Proposals

We discuss impact of recurrent connection on our CTPN. As shown in Fig. 3, the context information is greatly helpful to reduce false detections, such as text-like outliers. It is of great importance for recovering highly ambiguous text (e.g., extremely small-scale ones), which is one of main advantages of our CTPN, as demonstrated in Fig. 6. These appealing properties result in a significant performance boost. As shown in Table 1 (left), with our recurrent connection, the CTPN improves the FTPN substantially from a F-measure of 0.80 to 0.88.

Running time. The implementation time of our CTPN (for whole detection processing) is about $0.14s$ per image with a fixed short side of 600, by using a single GPU. The CTPN without the RNN connection takes about 0.13s/image GPU time. Therefore, the proposed in-network recurrent mechanism increase model computation marginally, with considerable performance gain obtained.

Table 1: Component evaluation on the ICDAR 2013, and State-of-the-art results on the SWT and MULTILINGUAL.

Components on ICDAR 2013				SWT				MULTILINGUAL			
Method	P	R	F	Method	P	R	F	Method	P	R	F
RPN	0.17	0.63	0.27	Epshtein [3]	0.54	0.42	0.47	Pan [24]	0.65	0.66	0.66
Faster R-CNN	0.79	0.71	0.75	Mao [20]	0.58	0.41	0.48	Yin [33]	0.83	0.68	0.75
FTPN (no RNN)	0.83	0.78	0.80	Zhang [34]	**0.68**	0.53	0.60	Tian [28]	**0.85**	0.78	0.81
CTPN	**0.93**	**0.83**	**0.88**	CTPN	**0.68**	**0.65**	**0.66**	CTPN	0.84	**0.80**	**0.82**

4.4 Comparisons with state-of-the-art results

Our detection results on several challenging images are presented in Fig. 5. As can be found, the CTPN works perfectly on these challenging cases, some of

Fig. 5: CTPN detection results several challenging images, including multi-scale and multi-language text lines. Yellow boxes are the ground truth.

which are difficult for many previous methods. It is able to handle multi-scale and multi-language efficiently (e.g., Chinese and Korean).

Table 2: State-of-the-art results on the ICDAR 2011, 2013 and 2015.

ICDAR 2011				ICDAR 2013					ICDAR 2015			
Method	P	R	F	Method	P	R	F	T(s)	Method	P	R	F
Huang [13]	0.82	0.75	0.73	Yin [33]	0.88	0.66	0.76	0.43	CNN Pro.	0.35	0.34	0.35
Yao [31]	0.82	0.66	0.73	Neumann [22]	0.82	0.72	0.77	0.40	Deep2Text	0.50	0.32	0.39
Huang [14]	0.88	0.71	0.78	Neumann [23]	0.82	0.71	0.76	0.40	HUST	0.44	0.38	0.41
Yin [33]	0.86	0.68	0.76	FASText [1]	0.84	0.69	0.77	0.15	AJOU	0.47	0.47	0.47
Zhang [34]	0.84	0.76	0.80	Zhang [34]	0.88	0.74	0.80	60.0	NJU-Text	0.70	0.36	0.47
TextFlow [28]	0.86	0.76	0.81	TextFlow [28]	0.85	0.76	0.80	0.94	StradVision1	0.53	0.46	0.50
Text-CNN [11]	0.91	0.74	0.82	Text-CNN [11]	0.93	0.73	0.82	4.6	StradVision2	0.77	0.37	0.50
Gupta [8]	**0.92**	0.75	0.82	Gupta [8]	0.92	0.76	0.83	**0.07**	Zhang [35]	0.71	0.43	0.54
CTPN	0.89	**0.79**	**0.84**	CTPN	**0.93**	**0.83**	**0.88**	0.14 *	CTPN	**0.74**	**0.52**	**0.61**

The full evaluation was conducted on five benchmarks. Image resolution is varied significantly in different datasets. We set short side of images to 2000 for the SWT and ICDAR 2015, and 600 for the other three. We compare our performance against recently published results in [1,28,34]. As shown in Table 1 and 2, our CTPN achieves the best performance on all five datasets. On the SWT, our improvements are significant on both recall and F-measure, with marginal gain on precision. Our detector performs favourably against the TextFlow on the Multilingual, suggesting that our method generalize well to various languages. On the ICDAR 2013, it outperforms recent TextFlow [28] and FASText [1] remarkably by improving the F-measure from 0.80 to 0.88. The gains are considerable in both precision and recall, with more than +5% and +7% improvements, respectively. In addition, we further compare our method against [8,11,35], which were published after our initial submission. It consistently obtains substantial improvements on F-measure and recall. This may due to strong capability of CTPN for detecting extremely challenging text, e.g., very small-scale ones, some

Fig. 6: CTPN detection results on extremely small-scale cases (in red boxes), where some ground truth boxes are missed. Yellow boxes are the ground truth.

of which are even difficult for human. As shown in Fig. 6, those challenging ones are detected correctly by our detector, but some of them are even missed by the GT labelling, which may reduce our precision in evaluation.

We further investigate running time of various methods, as compared in Table 2. FASText [1] achieves $0.15s$/image CPU time. Our method is slightly faster than it by obtaining $0.14s$/image, but in GPU time. Though it is not fair to compare them directly, the GPU computation has become mainstream with recent great success of deep learning approaches on object detection [25,5,6]. Regardless of running time, our method outperforms the FASText substantially with 11% improvement on F-measure. Our time can be reduced by using a smaller image scale. By using the scale of 450, it is reduced to $0.09s$/image, while obtaining P/R/F of 0.92/0.77/0.84 on the ICDAR 2013, which are compared competitively against Gupta *et al.*'s approach [8] using $0.07s$/image with GPU.

5 Conclusions

We have presented a Connectionist Text Proposal Network (CTPN) - an efficient text detector that is end-to-end trainable. The CTPN detects a text line in a sequence of fine-scale text proposals directly in convolutional maps. We develop vertical anchor mechanism that jointly predicts precise location and text/non-text score for each proposal, which is the key to realize *accurate* localization of text. We propose an in-network RNN layer that connects sequential text proposals elegantly, allowing it to explore meaningful context information. These key technical developments result in a powerful ability to detect highly challenging text, with less false detections. The CTPN is efficient by achieving new state-of-the-art performance on five benchmarks, with $0.14s$/ image running time.

References

1. Busta, M., Neumann, L., Matas, J.: Fastext: Efficient unconstrained scene text detector (2015), in IEEE International Conference on Computer Vision (ICCV)
2. Cheng, M., Zhang, Z., Lin, W., Torr, P.: Bing: Binarized normed gradients for objectness estimation at 300fps (2014), in IEEE Computer Vision and Pattern Recognition (CVPR)
3. Epshtein, B., Ofek, E., Wexler, Y.: Detecting text in natural scenes with stroke width transform (2010), in IEEE Computer Vision and Pattern Recognition (CVPR)
4. Everingham, M., Gool, L.V., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision (IJCV) 88(2), 303–338 (2010)
5. Girshick, R.: Fast r-cnn (2015), in IEEE International Conference on Computer Vision (ICCV)
6. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation (2014), in IEEE Computer Vision and Pattern Recognition (CVPR)
7. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional lstm and other neural network architectures. Neural Networks 18(5), 602–610 (2005)
8. Gupta, A., Vedaldi, A., Zisserman, A.: Synthetic data for text localisation in natural images (2016), in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
9. He, P., Huang, W., Qiao, Y., Loy, C.C., Tang, X.: Reading scene text in deep convolutional sequences (2016), in The 30th AAAI Conference on Artificial Intelligence (AAAI-16)
10. He, T., Huang, W., Qiao, Y., Yao, J.: Accurate text localization in natural image with cascaded convolutional text network (2016), arXiv:1603.09423
11. He, T., Huang, W., Qiao, Y., Yao, J.: Text-attentional convolutional neural networks for scene text detection. IEEE Trans. Image Processing (TIP) 25, 2529–2541 (2016)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Networks 9(8), 1735–1780 (1997)
13. Huang, W., Lin, Z., Yang, J., Wang, J.: Text localization in natural images using stroke feature transform and text covariance descriptors (2013), in IEEE International Conference on Computer Vision (ICCV)
14. Huang, W., Qiao, Y., Tang, X.: Robust scene text detection with convolutional neural networks induced mser trees (2014), in European Conference on Computer Vision (ECCV)
15. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading text in the wild with convolutional neural networks. International Journal of Computer Vision (IJCV) (2016)
16. Jaderberg, M., Vedaldi, A., Zisserman, A.: Deep features for text spotting (2014), in European Conference on Computer Vision (ECCV)
17. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding (2014), in ACM International Conference on Multimedia (ACM MM)
18. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., Shafait, F., Uchida, S.,

Valveny, E.: Icdar 2015 competition on robust reading (2015), in International Conference on Document Analysis and Recognition (ICDAR)
19. Karatzas, D., Shafait, F., Uchida, S., Iwamura, M., i Bigorda, L.G., Mestre, S.R., Mas, J., Mota, D.F., Almazan, J.A., de las Heras., L.P.: Icdar 2013 robust reading competition (2013), in International Conference on Document Analysis and Recognition (ICDAR)
20. Mao, J., Li, H., Zhou, W., Yan, S., Tian, Q.: Scale based region growing for scene text detection (2013), in ACM International Conference on Multimedia (ACM MM)
21. Minetto, R., Thome, N., Cord, M., Fabrizio, J., Marcotegui, B.: Snoopertext: A multiresolution system for text detection in complex visual scenes (2010), in IEEE International Conference on Pattern Recognition (ICIP)
22. Neumann, L., Matas, J.: Efficient scene text localization and recognition with local character refinement (2015), in International Conference on Document Analysis and Recognition (ICDAR)
23. Neumann, L., Matas, J.: Real-time lexicon-free scene text localization and recognition. In IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI) (2015)
24. Pan, Y., Hou, X., Liu, C.: Hybrid approach to detect and localize texts in natural scene images. IEEE Trans. Image Processing (TIP) 20, 800–813 (2011)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks (2015), in Neural Information Processing Systems (NIPS)
26. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Li, F.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision (IJCV) 115(3), 211–252 (2015)
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015), in International Conference on Learning Representation (ICLR)
28. Tian, S., Pan, Y., Huang, C., Lu, S., Yu, K., Tan, C.L.: Text flow: A unified text detection system in natural scene images (2015), in IEEE International Conference on Computer Vision (ICCV)
29. Wang, K., Babenko, B., Belongie, S.: End-to-end scene text recognition (2011), in IEEE International Conference on Computer Vision (ICCV)
30. Wolf, C., Jolion, J.: Object count / area graphs for the evaluation of object detection and segmentation algorithms. International Journal of Document Analysis 8, 280–296 (2006)
31. Yao, C., Bai, X., Liu, W.: A unified framework for multioriented text detection and recognition. IEEE Trans. Image Processing (TIP) 23(11), 4737–4749 (2014)
32. Yin, X.C., Pei, W.Y., Zhang, J., Hao, H.W.: Multi-orientation scene text detection with adaptive clustering. IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI) 37, 1930–1937 (2015)
33. Yin, X.C., Yin, X., Huang, K., Hao, H.W.: Robust text detection in natural scene images. IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI) 36, 970–983 (2014)
34. Zhang, Z., Shen, W., Yao, C., Bai, X.: Symmetry-based text line detection in natural scenes (2015), in IEEE Computer Vision and Pattern Recognition (CVPR)
35. Zhang, Z., Zhang, C., Shen, W., Yao, C., Liu, W., Bai, X.: Multi-oriented text detection with fully convolutional networks (2016), in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)