Scene Text Detection and Segmentation Based on Cascaded Convolution Neural Networks

Youbao Tang and Xiangqian Wu, *Member, IEEE*

***Abstract*— Scene text detection and segmentation are two important and challenging research problems in the field of computer vision. This paper proposes a novel method for scene text detection and segmentation based on cascaded convolution neural networks (CNNs). In this method, a CNN-based text-aware candidate text region (CTR) extraction model (named detection network, DNet) is designed and trained using both the edges and the whole regions of text, with which coarse CTRs are detected. A CNN-based CTR refinement model (named segmentation network, SNet) is then constructed to precisely segment the coarse CTRs into text to get the refined CTRs. With DNet and SNet, much fewer CTRs are extracted than with traditional approaches while more true text regions are kept. The refined CTRs are finally classified using a CNN-based CTR classification model (named classification network, CNet) to get the final text regions. All of these CNN-based models are modified from VGGNet-16. Extensive experiments on three benchmark data sets demonstrate that the proposed method achieves the state-of-the-art performance and greatly outperforms other scene text detection and segmentation approaches.**

***Index Terms*— Scene Text detection, scene text segmentation, text-aware candidate text region extraction, candidate text region refinement, candidate text region classification**

I. Introduction

SCENE text detection aims to locate the positions of text in different scenes, e.g. guideposts, store marks, and warning signs; it is one of the most important steps for end-to-end scene text recognition. Effective scene text detection can enhance the performance of numerous multimedia applications, e.g. mobile visual searches, content-based image retrieval, and automatic sign translation. A series of international scene text detection competitions [1]–[3] has been successfully organized to drive the research of scene text detection. Due to the unconstrained scene environment, e.g. different text sizes and colors along with complex backgrounds, scene text detection is still a challenging problem in the computer vision community.

The first step of scene text detection is candidate text region (CTR) extraction. Since existing approaches, e.g. sliding window based methods and stroke width transform (SWT) [4] or maximally stable extremal region (MSER) [5] based methods have failed to make full use of the characteristics of text, they extract a large number of non-text candidate regions, which may be much more than the true text regions. This makes the second step, i.e. non-text region filtering, vital for scene text detection. Correctly filtering out these non-text regions is very challenging. So recently, many approaches [6]–[20] have focused on extracting discriminative hand-designed features or CNN based features to classify the candidate regions. The above-mentioned CTR extraction methods are also sensitive to external factors, such as varying illumination, blur in images, and so on. This can result in failed extraction of parts of the true text regions, leading to low recall. For example, the best published recall performances on the ICDAR 2013/2015 text detection dataset are 0.83 reported in the paper [21] and 0.852 reported on the competition website [56].

Fig. 1. Three examples of our scene text detection results on several challenging scenarios.

Many existing scene text detection approaches [18], [19], [22] generate text bounding boxes containing a lot of background, which makes scene text recognition difficult. To deal with this problem, scene text segmentation methods [47]–[51] were proposed to obtain more precise text regions. These methods were hand-crafted and cannot be trained by an end-to-end process.

To solve these problems, this paper proposes a novel method for scene text detection and segmentation based on cascaded convolution neural networks. In this method, a text-aware CTR extraction model and a CTR refinement model are devised to extract CTRs and obtain precise text segmentation results, which can overcome the above problems. The text-aware CTR extraction model detects regions of text (or the coarse CTRs) in the scene images and the CTR refinement model precisely segments the detected regions (or the coarse CTRs) into text in order to get the refined CTRs. Finally, the refined CTRs are fed into a CTR classification model to filter out non-text regions and obtain the final text regions. All of these models are based on a powerful CNN model (i.e. VGGNet-16). Some examples are shown in Fig. 1 to demonstrate the power of

Manuscript received June 25, 2016; revised November 8, 2016 and January 10, 2017; accepted January 11, 2017. Date of publication January 19, 2017; date of current version February 17, 2017. This work was supported by the Natural Science Foundation of China under Grant 61472102 and Grant 61672194. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Xiaochun Cao. *(Corresponding author: Xiangqian Wu.)*

The authors are with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China (e-mail: tangyoubao@hit.edu.cn; xqwu@hit.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2017.2656474

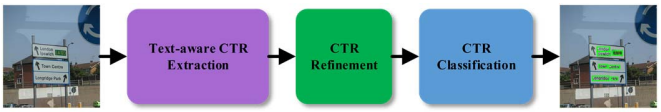


Fig. 2. The framework of the proposed method.

the proposed method. According to this figure, the number of candidate text regions is small and almost equivalent with the number of characters in most cases. Even if the colors in the text regions vary considerably or the text and background have similar colors, text regions can still be successfully detected. The framework of the proposed method is shown in Fig. 2, which consists of three steps, i.e. text-aware CTR extraction, CTR refinement, and CTR classification.

The main contribution of this paper can be summarized as follows: (1) The local and global information (i.e. edges and regions of text) are considered as different supervisory information in different layers to supervise the deep CNN model training for effective text-aware CTR extraction. The extracted coarse CTRs contain more true text regions and much fewer false text regions from scene images than those extracted using existing approaches, so as to obtain a higher recall. (2) The deconvolution network [23] is modified to build the CTR refinement model by integrating the information of shallow layers of the convolution network and deep layers of deconvolution network, which is very effective for refining the coarse CTRs and obtaining precise text region segmentation results. (3) A new image block construction strategy is developed to extract the blocks of CTRs by considering the higher discriminability of multiple characters than individual ones and keeping their aspect ratios, which are further classified into text or non-text using a devised fully convolutional neural network.

II. Related Work

In recent years, a large number of scene text detection approaches [4], [9], [14], [15,] [19], [21], [22], [24]–[42] have been proposed, most of which have been summarized by Ye and Doermann [43]. Generally, the existing approaches can be roughly divided into two groups: sliding window based approaches and connected component based approaches.

The sliding window based approaches [18], [19], [22], [24], [29], [44]–[46] firstly slide a large number of windows with different scales through all possible positions of the image and then extract features to classify them into text and background. One advantage of sliding windows based CTR extraction approaches is finding almost all of the true text regions, but they also result in numerous candidate regions, which need significant effort to be classified. The key factor in determining detection performance is the discriminability of the extracted features. In earlier work, hand-designed low-level features, e.g. HOG and SIFT, are extracted [24], [29], [44]–[46] for classification. To improve classification performance, some researchers [18], [19] have adopted the convolution neural network (CNN) approach to learn deep high-level features in recent years and have achieved state-of-the-art results. The advantage of these approaches lies in a simple and adaptive structure for both training and testing. However, these approaches have two problems. Firstly, to obtain good performance, these approaches have to classify a large number of sliding windows, resulting in a high computation requirement. Secondly, these approaches cannot extract text with pixel-level precision since they directly classify the regions in sliding windows.

Connected component based approaches [4], [6]–[17], [20], [35], [41] firstly cluster the pixels into larger connected components according to the pixels' properties, e.g. intensity, color, and stroke width, and then extract features from the connected components for classification. In these methods, the stroke width transform (SWT) [4] and maximally stable extremal regions (MSER) [5] are two widely used candidate connected component generators with great success in text detection because of their efficiency and stability. However, the performance of the SWT based approaches [4], [7], [9], [10], [17] and MSER based approaches [8], [11]–[16], [20], [35] greatly depends on the results of edge detectors and MSER extractors, which are very sensitive to illumination and noise. Compared with sliding window based methods, one advantage of these approaches is that they can greatly reduce the number of candidate regions, but they also miss some true character regions. Although the number of candidate regions is reduced, this number is still much larger than the number of the true character regions.

For scene text segmentation, Bai *et al.* [47] estimated the seed points of texts and backgrounds according to some low-level features, and then used semi-supervised learning to segment the text. Bosamiya *et al.* [48] used the Otsu algorithm to binarize the scene image for scene text segmentation. Mishra *et al.* [49] formulated the binarization of scene text as a MRF model optimization problem and used an iterative graph cut scheme to find the optimal binarization. Zhou *et al.* [50] designed a scene text segmentation method based on inverse rendering, which iteratively optimizes the rendering parameters. Most of the existing methods miss a lot of text regions when the contrast is low and do not include text-specific information to filter out the image background [51]. To overcome this problem, Tian *et al.* [51] applied multi-level MSER and a text candidate selection algorithm which incorporates text-specific features to improve the segmentation performance. But their extracted text-specific features were hand-designed and the segmentation method cannot be trained by an end-to-end process. This paper proposes a CNN-based end-to-end text segmentation model to handle these problems.

III. Text-Aware CTR Extraction

Compared with traditional candidate text region (CTR) generation techniques (e.g. SWT-based and MSER-based ones), this paper proposes a novel CTR generation method based on text-aware saliency detection, which is able to highlight the text regions so as to greatly reduce the number of CTRs. It has been proven that deeply supervised networks [52] have been successfully used in image classification [52] and edge detection [53]. Therefore, we construct a deeply supervised CNN network (named detection network, DNet) to predict the saliency for each pixel; based on this, the initial locations of text can be detected.

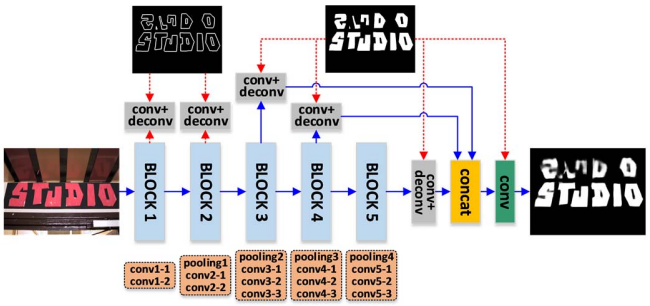


Fig. 3. The architecture of DNet.

To make DNet focus on the text regions, the information which reflects the properties of the text is used as supervisory information to train the CNN model. The shape of the text regions is one of the most important pieces of information for distinguishing between text and backgrounds. The edges and the whole regions of text can represent its shape. As we know, CNN learns local and global features as we move from the shallow to deep layers. For text, the edges can be considered as local information and the regions as global information. Therefore, in this work, the edges and regions of text are used as the supervisory information of the shallow and the deep layers of the CNN for model training, respectively.

To get an accurate saliency prediction, the CNN architecture should be deep and have multi-scale stages with different strides, so that we can learn discriminative and multi-scale features for pixels. Training such a deep network from scratch is hard work when the number training samples available is insufficient. So this work chooses VGGNet-16 [41] trained on the ImageNet dataset as the pre-trained model for fine-tuning as done by [53]. VGGNet-16 has six blocks. The first five blocks contain convolutional layers and pooling layers, and the last block contains a pooling layer and fully connected layers. The last block is removed in this work, since the pooling operation in this block makes the feature maps become too small (about $1/32$ of the input image size) to obtain fine full-size prediction and the fully connected layers are time and memory consuming. Based on the first five blocks of VGGNet-16, DNet is constructed as shown in Fig. 3. There are thirteen convolutional layers and four max-pooling layers in the five blocks. Specifically, the numbers of convolution kernels in the five blocks are 64, 128, 256, 512 and 512. For all convolutional layers, the size of all of the convolution kernels is 3*3, and both the stride and padding are 1 pixel. For all max-pooling layers, the operation window size is 2*2 and the stride is 2 pixels.

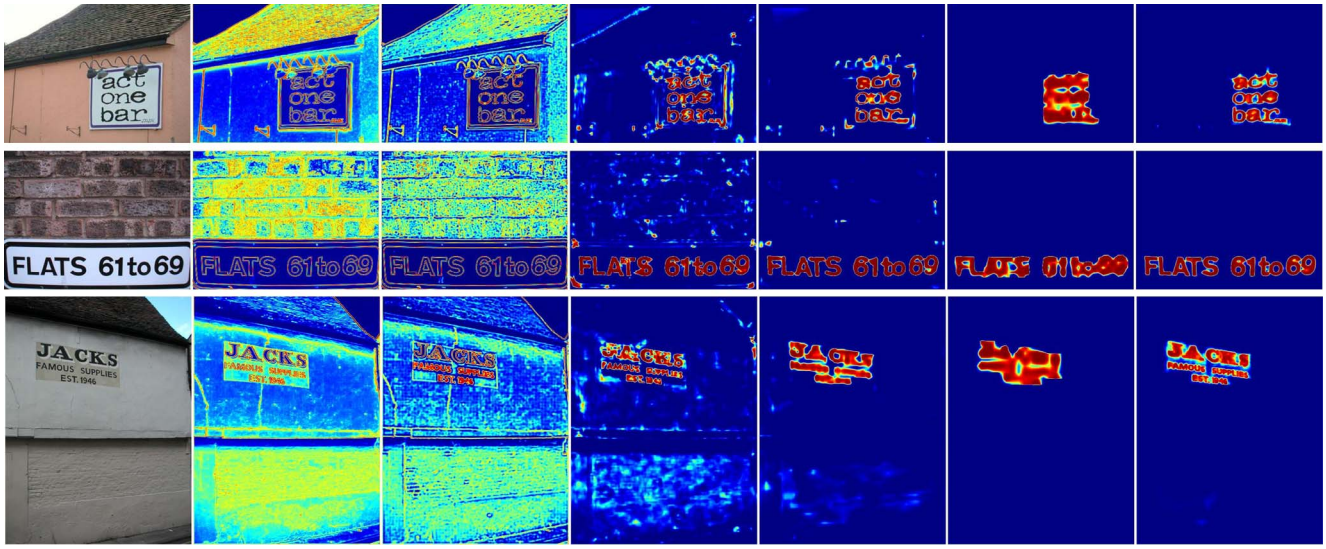
As we know, the shallow CNN layers usually learn local features, especially for edges and there are different kinds of edges including those of text and background in natural scene images. Therefore, only the edges of text are used as supervisory information for the shallow layers to make the CNN model pay more attention to the edges of text during early feature learning in this work. Usually, the deep layers learn the global object features. Therefore, the text regions are used as the supervisory information for the deep layers, so as to learn more discriminative global features to represent the text properties. As we can see, from shallow to deep, the whole DNet always focuses on learning the features of text. We investigate which layers should be supervised by text edges or regions and find that the best performance is obtained when the first two and the last three blocks are supervised by text edges and regions, respectively. He *et al.* [35] also used text regions as supervisory information for model training, but the text region was used as supervisory information in one convolutional layer. In the proposed DNet model (as shown in Fig. 3), the text regions are used as supervisory information in four convolutional layers, and another important information of text, i.e. the edges of text, is also used as supervisory information in the shallow layers, which makes the model pay more attention to the edges of text during early feature learning.

To introduce the supervisory information into the CNN, a side-output is generated, as done in [53], from the last convolutional layer of each block with a convolutional layer and a deconvolutional layer. The additional convolutional layer with a 1*1 convolution kernel converts the feature maps to a probability map, and the additional deconvolutional layer is used to render probability map the same size as the input image. To make the final probability map robust to the size variation of the text, the side-outputs of the last three blocks are fused by concatenating them in the channel direction and using a convolution kernel of size 1*1 to convert the concatenation maps to the final probability map. In fact, convolution with a 1*1 kernel is a weighted fusion process. Here, the side-outputs of the first two blocks are not considered during fusion, since we hope to capture the global information of the text regions during text-aware saliency prediction and we have found through experiments that it does not improve the performance to add the first two side-outputs during fusion. So far, the whole architecture of DNet has been constructed, as shown in Fig. 3. The additional deconvolutional layer and last convolutional layer are followed by a sigmoid activation function.

For DNet training, the errors between all side-outputs and the supervisory ground truth should be computed and backward propagated. Therefore, we need to define a loss function to compute these errors. For most of the scene images, the pixel numbers of text and background are heavy imbalanced. Here, given an image X and its ground truth Y, a cross-entropy loss function defined in [53] is used to balance the loss between text and non-text classes as follows:

$$l_{side}^{m}\left(\mathbf{W},\mathbf{w}^{m}\right) = -\alpha \sum_{i=1}^{|Y_+|} \log P\left(y_i = 1|X;\mathbf{W},\mathbf{w}^{m}\right) - (1-\alpha) \sum_{i=1}^{|Y_-|} \log P\left(y_i = 0|X;\mathbf{W},\mathbf{w}^{m}\right) \quad (1)$$

where $\alpha = |Y_-| / (|Y_+| + |Y_-|)$, $|Y_+|$ and $|Y_-|$mean the number of text pixels and non-text pixels in the ground truth, $\mathbf{W}$ denotes the parameters of all of the network layers in the five blocks, $\mathbf{w}^m$ denotes the weights of the m^{th} side-output layer including a convolutional layer and a deconvolutional layer, and $P\left(y_i = 1|X;\mathbf{W},\mathbf{w}^{m}\right) \in [0, 1]$ is computed using



(a) (b) (c) (d) (e) (f) (g)

Fig. 4. Examples of all side-outputs on test images. (a) The input images. (b)-(f) The side-outputs of all five blocks from shallow to deep. (g) The final fusion text-aware saliency map. For better visualization, the original gray images are shown as pseudo-colors here, more red means larger probability that the pixel belongs to text.

the sigmoid function on the activation value at pixel j. The whole loss of all side-outputs is thus defined as

$$L_{side}\left(\mathbf{W},\mathbf{w}\right)=\sum_{m=1}^{5}l_{side}^{m}\left(\mathbf{W},\mathbf{w}^{m}\right) \tag{2}$$

where $\mathbf{w}=\left(\mathbf{w}^{1},\mathbf{w}^{2},\ldots,\mathbf{w}^{5}\right)$. The loss between the fusion output and ground truth $L_{fuse}\left(\mathbf{W},\mathbf{w},\mathbf{h}\right)$ is also computed using the cross-entropy loss function, where $\mathbf{h}$ denotes the weights of the last fusion convolution kernel. After computing all losses, the standard stochastic gradient descent algorithm is used to minimize the following objective function during training.

$$\left(\mathbf{W},\mathbf{w},\mathbf{h}\right)^{*}=\operatorname{argmin}\left(L_{side}\left(\mathbf{W},\mathbf{w}\right)+L_{fuse}\left(\mathbf{W},\mathbf{w},\mathbf{h}\right)\right) \tag{3}$$

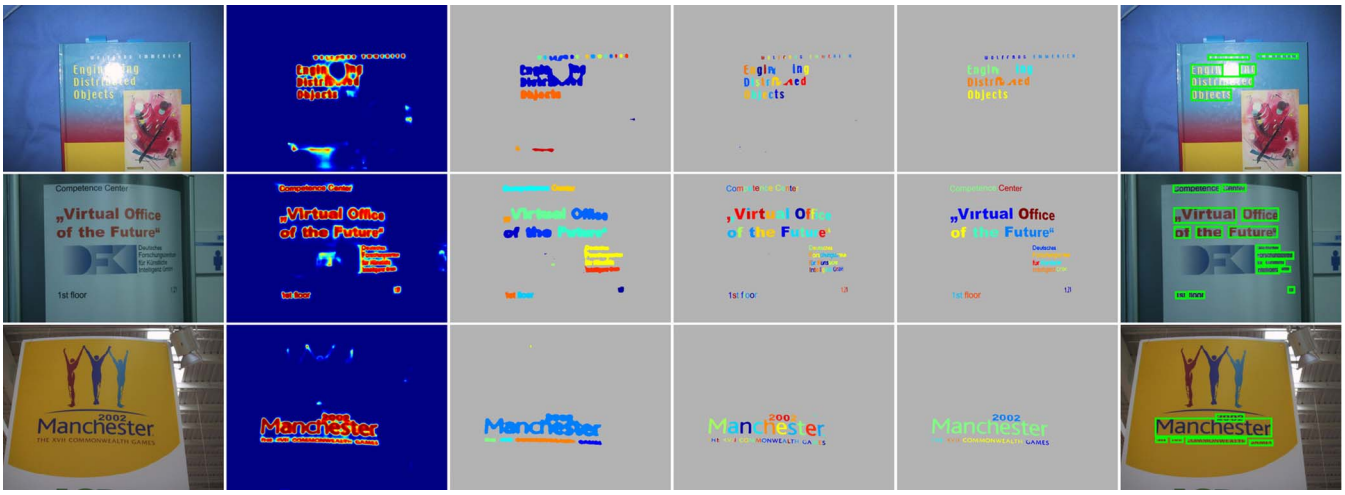
For DNet testing, given an image, we can use the trained CNN model to predict a text-aware salient map. In this work, the fusion probability map is used as the final salient map of the input image, in which the text regions have larger values. And all of the side-outputs also pay more attention to the text edges or regions compared with the background. To better validate this point, three visual examples of all output probability maps are given in Fig. 4. From Fig. 4, we can see that (1) the text edge points have larger values (more red) than most of the background edge points in the side-outputs, which demonstrates the effectiveness of our supervisory scheme. (2) As the network goes deeper, more background regions are suppressed while the text regions keep being highlighted, and the shapes of text regions become blurred, which demonstrates that DNet effectively learns the local and global features as the layers go deeper. (3) The final fusion maps get the best performance of text-aware saliency detection, which means that the fusion strategy sufficiently considers multi-scale information from multiple side-outputs for text-aware saliency detection. It should be noted that the size of the input image can be arbitrary during training and testing, since DNet is a fully convolutional network. But to save the cost of time and memory, we resize the images of width greater than 500 to a fixed width of 500 pixels, retaining their aspect ratios.

After obtaining the text-aware salient map (as shown in Fig. 5(b)), we can easily extract the coarse CTRs from the text-aware salient map. Firstly, we binarize the text-aware salient map with an adaptive threshold computed using Otsu's algorithm to get a binary image (as shown in Fig. 5(c)). Each connected component in the binary image is a coarse CTR. Therefore, the number of coarse CTRs is equal to the number of connected components. Then we crop a sub-image from the original image with the minimum bounding box of each connected component as its coarse CTR image, which will be used as the input of the CTR refinement model.

IV. CTR Refinement

The extracted coarse CTRs usually contain some background regions. Because of the diversity of texts and backgrounds in scene images, it is impossible to consider all cases in the training dataset. Therefore, there will be some errors in the coarse CTRs. And when the texts are close to each other, multiple words or text lines will be considered as one text region in the coarse CTRs. Directly using the coarse CTRs as the text detection result will reduce recall and precision. Moreover, accurate text segmentation can provide helpful information for scene text recognition. Hence, it is necessary to further refine the text region segmentation results based on the coarse CTRs.

Recently, one of the hottest fields in computer vision is semantic image segmentation, which aims to segment different objects from backgrounds with semantic information. The state-of-the-art approaches are based on CNN, e.g. fully convolutional network [54] and deconvolutional network [23].



(a) (b) (c) (d) (e) (f)

Fig. 5. Visual examples to show the process of our text detection method. (a) The input images. (b) The text-aware salient maps. (c) Coarse CTR extraction results. (d) CTR refinement results. (e) CTR classification results. (f) Text detection results. Different colors in (c)-(e) mean different CTRs or words.

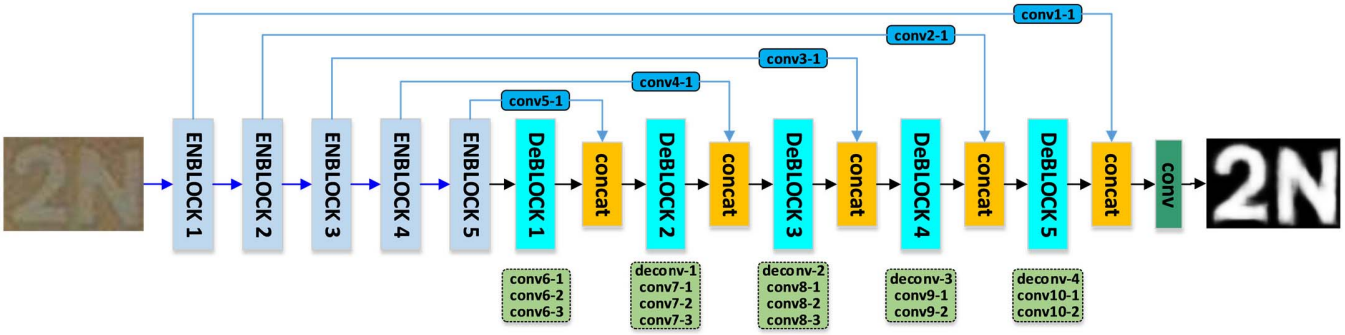


Fig. 6. The architecture of SNet.

Text region segmentation can be considered as a simple two-class problem of semantic image segmentation, i.e. text and non-text. Therefore, we construct a CNN (named segmentation network, SNet) based on the deconvolutional network [23] for CTR refinement and text region segmentation in this work.

Fig. 6 shows the architecture of SNet, which contains ten blocks. The first five blocks (called ENbolck) are the same as those of DNet, which reduce the size of feature maps through feedforwarding. The last five blocks (called DEblock) can be considered as an opposite process of the first five blocks, which enlarge the size of feature maps through the combination of deconvolution and convolution. The layers in each DEblock are shown in Fig. 6. The convolutional layers of the DEblocks have the same configuration as the ones of their symmetric blocks, e.g. conv10-2/conv10-1/conv6-1 has the same configuration as conv1-1/conv1-2/conv5-3. The deconvolutional layers [23] associate a single input activation with multiple outputs, so the output of the deconvolutional layer is an enlarged and dense activation. The filter size and stride of the i^{th} deconvolutional layer (deconv-i) is 2^{i+1} and 2^i, and the filter number is the same as the convolution kernel number of the convolutional layers in the same DEblock. Each convolutional layer or deconvolutional layer is followed by a ReLU activation function. Traditional CNNs learn features in a layer wise manner. However, we hope that the DEblocks of SNet will learn more fine and larger high-level features for text segmentation from the fuzzy and small feature maps. Therefore, in this paper, the high-level object features and low-level features are merged to improve the text segmentation by integrating the information of the shallow layers of the ENblocks into the deep layers of the DEblocks. To be specific, the output of the last convolutional layer in each DEblock is concatenated in the channel direction with the output of the first convolutional layer in its symmetric ENblock, as shown in Fig. 6. The concatenated feature maps are used as the input of the next DEblock except for the last ones. The last concatenated feature maps are the input of a convolutional layer with a 1*1 convolution kernel and the last convolutional layer is followed by a sigmoid activation function. The architecture of SNet has now been introduced, as shown in Fig. 6.

Differing from DNet, SNet takes the coarse CTR images as input instead of the whole original images and only uses the text regions as the supervisory information in the last convolutional layer. For different images, the sizes of the extracted coarse CTR images are different. In this work, all coarse CTR images are resized to a fixed height of 224, retaining

(a) (b) (c) (d) (e)

Fig. 7. Examples of text region segmentation or CTR refinement results of our method (the second row) and the original deconvolutional network (the third row) on test images (the first row).

their aspect ratios. For training, the above-mentioned cross-entropy loss function is used to compute the error between the output of the last convolutional layer and the ground truth. The standard stochastic gradient descent algorithm is used to minimize the loss function. For testing, given an image, a probability map is obtained using the trained CNN model and binarized with an adaptive threshold to get the final text region segmentation results or the refined CTRs. Fig. 7 gives five examples of text region segmentation with the proposed SNet model and the original deconvolutional network model [23] on test images. As we can see, the proposed SNet gets better text segmentation results and is able to successfully segment the text even from the images with complex backgrounds (as shown in Fig. 7(a)-(c)) and blurred text (as shown in Fig. 7(d)-(e)).

The coarse CTR images as the input of SNet are cropped from the original images with the minimum bounding boxes of the coarse CTRs. So the backgrounds of these input images are much simpler than the original images. Hence, we do not need more supervisory information for model training as done by DNet. Through extensive experiments, we demonstrate that the performance changes little but the complexity increases when more supervisory information is added in different layers. We also find that SNet gets better text segmentation performance than DNet when using the same training dataset. Compared with the coarse CTRs, the text can be accurately segmented and refined using SNet, as shown in Fig. 5(d).

V. CTR Classification

In the CTR refinement results (as shown in Fig. 5(d)), there still exist some non-text regions. Therefore, given a CTR image, we need to classify it into text or non-text, which is actually a two-class problem in image classification. In the famous ImageNet challenge, CNN based methods (i.e. VGGNet [41] and GoogleNet [42]) obtained the top performance in the image classification task. So in this work, we modify the architecture of VGGNet-16 to construct a fully convolutional neural network (named CTR classification network, CNet) for CTR classification.

Since CTR classification is a two-class problem and the text is much simpler than the objects of ImageNet, a shallower CNN than the original VGGNet-16 is enough to get good performance for CTR classification. Therefore, the first three blocks of VGGNet-16 are kept and the rest are cut to build CNet. The size of input image is 224*224 in VGGNet-16 which is too large for CTR classification. To save the time and

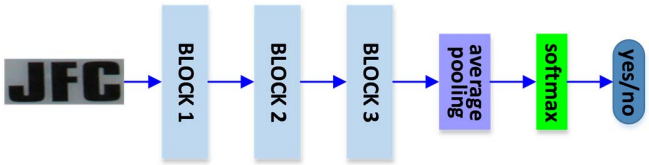


Fig. 8. The architecture of CNet.

memory cost of model training and testing, we fix the height of the input image as 32. The width of the input image is automatically decided by the size of the original CTR image during testing. During training, the width of the input image is fixed at 100 to accelerate the training process. To make the CNN model accept the input images of arbitrary width, we use a global average pooling layer following the third block to generate a feature vector of fixed length instead of a fully connected layer. Since the third block outputs 256 feature maps, the length of the feature vector is 256. Finally, a softmax layer is used to classify the feature vector as text or non-text during testing, and a softmaxloss layer is used to compute the loss during training. The whole architecture of CNet is shown in Fig. 8.

After doing the coarse CTR extraction and CTR refinement, we can obtain a number of refined CTRs, based on which we need to construct CTR images as the inputs of CNet. As we know, the text region images containing multiple characters have higher discriminability than those containing only a single character. In this work, we propose a new image block construction strategy to construct the CTR images. Given a CTR, we search its neighboring CTRs from its left and right directions. To be specific, we assume that the size of the current CTR is $w * h$, the coordinate of its center point is (x, y), and the origin is the top left corner. If its constructed region, whose top left corner's coordinate is $(x-1.5h-0.5w, y-0.5h)$ and bottom right corner's coordinate is $(x+1.5h+0.5w, y+0.5h)$, contains other CTRs whose heights are in the range $[0.5h, 1.5h]$, we merge these CTRs as the constructed CTR of the current CTR. Therefore, the constructed CTR images contain multiple characters in most cases. However, due to the fully connected layers, traditional methods [14], [19], [35] directly resize the current CTR into a fixed size or enlarge the current CTR to form a square whose side length is the height of the current CTR and then resize it to a fixed size. Therefore, their constructed CTR images usually contain only a single character. Compared with traditional methods [14], [19], [35], our image construction strategy has two advantages: (1) our constructed CTR images contain multiple characters which always have more discriminative information than a single character for helping distinguish them from non-text regions, which improves classification performance. (2) because of the fully convolutional neural network, our constructed CTR images are inputted into the classification CNN keeping their aspect ratios and so less information is lost.

After filtering out the non-text regions with CNet, we cluster the text regions into text lines according to their vertical locations and heights. Then the text lines are separated into

words as the final text detection results according to the distances of adjacent text regions in the same text line. The connected components in Fig. 5(e) are the result of CTR classification on Fig. 5(d), and different colors in Fig. 5(e) mean different words. Fig. 5(f) shows the final text detection results indicated by green bounding boxes.

VI. Experiments

A. Implementation

The proposed method needs the supervisory information of text edges and regions to train the DNet and SNet models. However, the publicly available datasets which contain this supervisory information are too small to effectively train these CNN models and it is hard work to manually annotate the supervisory information for a large number of images. In this work, we create a synthetic scene text image generator to emulate real scene text images for DNet training. Following the success of the synthetic word dataset [36], we used the same generation process to construct a larger number of synthetic scene text images, which includes font rendering, border/shadow rendering, base colouring, projective distortion, natural image blending, and noise. For more details of each step, please refer to the [36]. Besides the synthetic scene text image, we also construct a large number of word images and text region images for SNet and CNet training. The word images are generated by cutting the small image in the bounding box of each word (or several overlapped words) from the synthetic scene text images and the text region images are generated by applying the proposed image block construction strategy to the synthetic scene text images. With the above processes, a synthetic training dataset was constructed, which contains about 20,000 scene text images, 200,000 word images, 1,000,000 text region images (the number of characters in the text region images was between 1 and 10), and their corresponding ground truths including edges and text regions. 20,000 scene text images were used to train DNet, 200,000 word images were used to train SNet, and 1,000,000 text region images and another 1,000,000 non-text regions randomly cropped from background images were used to train CNet.

We used Matlab and the popular deep learning tool, i.e. Caffe Library [55], to implement the proposed method. Since all of the CNN networks were based on VGGNet-16, the original VGGNet-16 model trained on the ImageNet dataset was used as the pre-trained model. Then the training datasets were used to fine-tune their corresponding CNN models. For training DNet and SNet, the learning rate, momentum, and batch size were set to 10^{-8} ,0.9, and 1, respectively. They were trained with 3 epochs and the learning rate was reduced by a factor of 10 after every epoch. For CNet training, the learning rate, momentum, and batch size were set to 10^{-3}, 0.9, and 128, respectively. It was trained with 6 epochs and the learning rate was reduced by a factor of 10 after every 2 epochs. The above training process was performed on a PC with an Intel i7-4790k CPU, a TESLA k40c GPU, and 32G RAM. For testing, the average time per image of our scene text detection method was about 1.36 second on the ICDAR2011 test dataset.

B. Datasets and Evaluation Criteria

We test the proposed method on three benchmark datasets, i.e. ICDAR 2011 robust reading competition dataset [1] (denoted as ICDAR2011), ICDAR 2013 robust reading competition dataset [2] (denoted as ICDAR2013), and the street view text dataset (denoted as SVT) [45]. The ICDAR2011 dataset contains 299 training images and 255 test images. The ICDAR2013 dataset contains 229 training images and 233 test images, which is a subset of ICDAR2011 by removing the duplicated images and revising a small part of the ground truth annotations. The scene text detection and segmentation tasks in the latest ICDAR2015 competition [3] use the same datasets as ICDAR2013 [2]. The SVT dataset contains 101 training images and 249 test images which were captured from Google street view and often have low resolution or quality. The background regions of these images contain many similar patterns as text regions and text regions are not fully annotated, which make the SVT dataset extremely challenging.

For evaluation, we follow the criteria used in the ICDAR 2011 [1] and ICDAR 2013 [2] competitions, i.e. precision, recall, and F-measure. Precision measures the ratio between true positives and all detections, while recall measures the ratio of true positives and all of true that should be detected. F-measure, as an overall, single indicator of algorithm performance, is the harmonic mean of precision and recall. It evaluates both quantity and quality of rectangle matches through all images in the database, and considers not only one-to-one matching, but also one-to-many and many-to-one matching. The quality of detection or matching is controlled by two parameters which penalize more on parts matching than larger detection. The precision, recall, and F-measure are computed as follows:

$$Precision = \frac{\sum_i^N \sum_j^{|D^i|} M_D\left(D_j^i, G^i\right)}{\sum_i^N |D^i|} \tag{4}$$

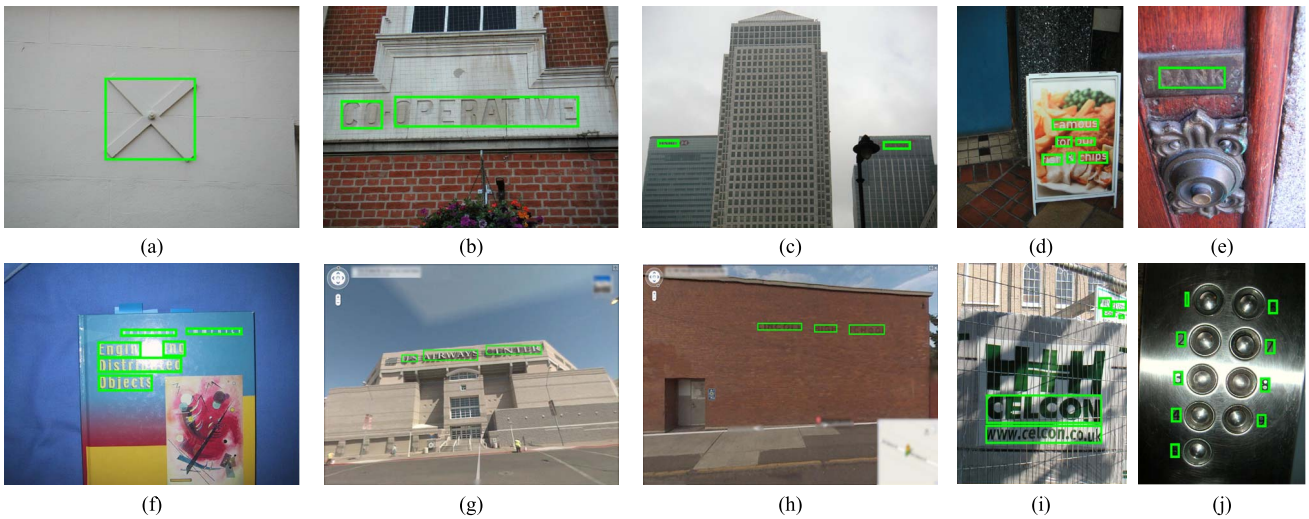
$$Recall = \frac{\sum_i^N \sum_j^{|G^i|} M_G\left(G_j^i, D^i\right)}{\sum_i^N |G^i|} \tag{5}$$

$$F - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6}$$

where N is the total number of images in a dataset, $|D^i|$ and $|G^i|$ are the number of detection and ground true rectangles in the i-th image, $M_D\left(D_j^i, G^i\right)$ and $M_G\left(G_j^i, D^i\right)$ are the matching scores for detection rectangle D^j and ground truth rectangle G^j. Their values are set to 1 for one-to-one matching, 0.8 for one-to-many matching and 0 for no matching. Two rectangles are considered as matched when their overlapping ratio is higher than a defined threshold, which controls the quality of the matching.

C. Performance of Text Detection

Fig. 9 shows several scene text detection examples of the proposed methods on some challenging images, such as single character, complex background, and low contrast, etc., some of which cannot be successfully detected by other state-of-the-art approaches.



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Fig. 9. Some successful results from our scene text detection method.

TABLE I

PERFORMANCE OF DIFFERENT SCENE TEXT DETECTION APPROACHES ON THE ICDAR2011 DATASET

Approach	Year	Precision	Recall	F-measure
Ours	-	0.902	**0.859**	**0.880**
Tian et al [21]	2016	0.89	0.79	0.84
He et al [38]	2016	0.88	0.79	0.84
Gupta et al [42]	2016	**0.915**	0.748	0.823
Qin et al. [20]	2016	0.876	0.772	0.821
He et al. [35]	2016	0.91	0.74	0.82
Jaderberg et al. [36]	2016	-	-	0.81
Tian et al. [22]	2015	0.862	0.762	0.809
Zhang et al. [19]	2015	0.84	0.76	0.80
Huang et al. [14]	2014	0.88	0.71	0.78
Yin et al. [12]	2014	0.863	0.683	0.762
Koo et al. [26]	2013	0.814	0.687	0.745
Yao et al. [17]	2014	0.822	0.657	0.730
Huang et al. [9]	2013	0.82	0.75	0.73
Neumann et al. [10]	2013	0.793	0.664	0.723
Kim et al. [1]	2011	0.830	0.625	0.713

TABLE II

PERFORMANCE OF DIFFERENT SCENE TEXT DETECTION APPROACHES ON THE ICDAR2013/ICDAR2015 DATASET

Approach	Year	Precision	Recall	F-measure
Ours	-	0.919	**0.871**	**0.895**
Tian et al [21]	2016	0.93	0.83	0.88
SenseTime [56]	-	0.888	0.852	0.870
He et al [38]	2016	0.90	0.83	0.86
SCUT-HCII [56]	-	0.872	0.828	0.849
Baidu IDL [56]	-	0.917	0.789	0.848
Qin et al. [20]	2016	0.888	0.787	0.834
Zhu et al. [39]	2016	**0.933**	0.752	0.833
Gupta et al [42]	2016	0.920	0.755	0.830
Zhang et al [40]	2016	0.88	0.78	0.83
Cho et al [41]	2016	0.863	0.785	0.822
He et al. [35]	2016	0.928	0.729	0.817
Tian et al. [22]	2015	0.852	0.759	0.803
Zhang et al. [19]	2015	0.88	0.74	0.80
Lu et al. [37]	2015	0.892	0.696	0.782
iwrr2014 [16]	2014	0.86	0.70	0.77
USTB TexStar [12]	2014	0.88	0.66	0.76
Text Spotter [8]	2012	0.88	0.65	0.74
CASIA_NLPR [2]	2013	0.79	0.68	0.73

Table I and Table II present the full evaluation results of different scene text detection approaches on the ICDAR2011 and ICDAR2013/2015 datasets. From Tables I and II, we can see that the proposed method gets the best recall and F-measure. Compared with the best performance of the other approaches, the recall and F-measure are improved by 8.7% (from 0.79 to 0.859) and 4.8% (from 0.84 to 0.88) on the ICDAR2011 dataset. For the ICDAR2013/2015 dataset, our method improves the recall and F-measure by approximately 1.9% (from 0.852 to 0.871) and 2.5% (from 0.87 to 0.895) compared with the best one reported on the competition website [56], and 4.9% (from 0.83 to 0.871) and 1.7% (from 0.88 to 0.895) compared with the state-of-the-art methods in the literature. For the precision on both datasets, the performance of our method is comparable with the state-of-the-art. That is because our method has the ability to detect text which is not labeled in the ground truth, as shown in Fig. 10. So when computing the precision, these detected text regions are considered as incorrect detection results, which reduces the precision. For the SVT dataset, since it is not fully annotated and the precision measure cannot give a meaningful performance comparison, only the recall measure is used to evaluate the performance of different text detection methods as done by [34]. Table III presents the evaluation results on the SVT dataset. It can be seen that the proposed method gets the best recall of 0.762 increasing by 20.9% (from 0.63 to 0.762). All of these results demonstrate the effectiveness of the proposed method. The good performance of the proposed method may result from the following reasons. Firstly, the proposed DNet and SNet can extract most of the true text regions and exclude most of the background regions. Secondly, CNet can effectively distinguish the text regions and background regions.

Although the proposed method gets the best overall performance on three benchmark datasets and can deal with most of the various challenging scenarios, it fails to work well in

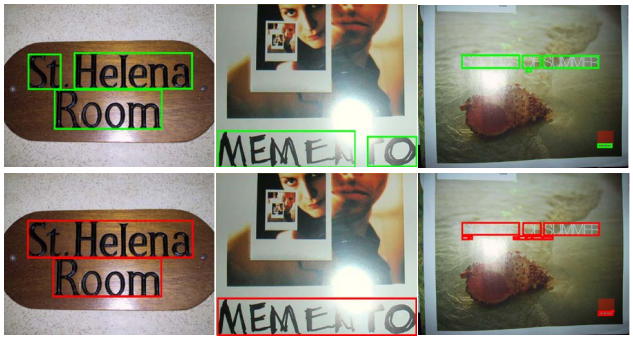
TABLE III

PERFORMANCE OF DIFFERENT SCENE TEXT DETECTION APPROACHES ON THE SVT DATASET

Approach	Year	Recall
Ours	-	**0.762**
Yi et al. [32]	2013	0.63
Xu et al. [33]	2014	0.60
Fraz et al. [34]	2015	0.516
Gupta et al [42]	2016	0.407
Neumann et al. [8]	2012	0.329
Wang et al. [45]	2011	0.29

(a) (b) (c)

Fig. 10. Examples of detection results, in which some text regions are detected by the proposed method (first row) but not labeled in the ground truth (second row).



(a) (b) (c)

Fig. 11. Failure examples of the proposed method. The first row gives the detection results of the proposed method. The second row shows the ground truths.

some cases. For example, when the text regions which belong to the same semantic content have a relatively large distance between them, the word generation process is unable to merge the text regions together, as shown in Fig. 11(a). When the scene images are placed in non-uniform illumination or there is strong light on the text regions, this results in deformation of the text, DNet and SNet cannot fully and correctly segment the deformed text regions, as shown in Fig. 11(b). When the text has very similar color to the background or very small size, the proposed method cannot segment the text regions using DNet, leading to failure to detect the text, as shown in Fig. 11(c). Some post-processing steps may be useful to correct some failed detection results, e.g. the ones shown in Fig. 11(a)-(b).

TABLE IV

PERFORMANCE OF DIFFERENT TEXT SEGMENTATION APPROACHES ON THE ICDAR2013/2015 DATASET IN PIXEL-LEVEL

Approach	Year	Precision	Recall	F-measure
Ours	-	**0.9470**	**0.8560**	**0.8992**
StradVision [56]	-	0.8924	0.7880	0.8370
Lu et al. [37]	2015	0.8210	0.7727	0.7963
I2R_NUS_FAR [2]	2013	0.8170	0.7473	0.7806
BUCT_YST [56]	-	0.8175	0.7456	0.7799
I2R_NUS [2]	2013	0.7904	0.7357	0.7621
USTB_FuStar [2]	2013	0.7445	0.6958	0.7193
Text Detection [2]	2013	0.7620	0.6474	0.7001
NSTextractor [2]	2013	0.7628	0.6071	0.6761
NSTsegmentator [2]	2013	0.6395	0.6841	0.6610
OTCYMIST [2]	2013	0.5853	0.4611	0.5158

D. Performance of Text Segmentation

With the proposed DNet and SNet, the text regions can be accurately segmented from the background at the pixel-level, as shown in Fig. 5(d). Generally, there are still some non-text regions in the segmentation result. With CNet, the non-text regions can be effectively removed and the text regions are retained, as shown in Fig. 5(e). Also, because of only ICDAR2013/2015 providing the pixel-level annotation of text, the ICDAR2013/2015 test dataset is used to evaluate the performance of our segmentation results with the criteria of precision, recall, and F-measure.

Table IV presents the performance of different text segmentation approaches on the ICDAR2013/2015 dataset. The performances of approach StradVision and BUCT_YST are reported on the competition website [56]. From Table IV, it can be seen that our method greatly outperforms all other approaches on all evaluation criteria. High precision of approximately 0.947 means that a large number of foreground pixels in the segmentation results belong to text, and high recall of approximately 0.856 means that we find most of the text pixels with our method. That is because our SNet has powerful ability to accurately segment the text regions at the pixel-level and CNet can well filter out the non-text regions.

E. Contributions of Different Modules

The proposed method consists of three modules, i.e. text-aware CTR extraction (M1), CTR refinement (M2), and CTR classification (M3), which are cascaded to generate the text regions from coarse to fine. To investigate the impact of different modules, several experiments are conducted with the following module combinations, i.e. M1, M1+M3, M1+M2, and M1+M2+M3. For different module combinations, the text detection performance is evaluated and the average numbers of detected regions are counted on the ICDAR2013/2015 dataset. When counting the detected regions, all connected components of the outputs of M1, M2, and M3 are considered as their detected regions. And a detected region is considered as a positive one when 70% of its area overlaps the ground truth. Table V lists the results of the different combinations. Table V shows that: (1) Compared with previous methods [2], [8], [12], the performance of scene text detection

TABLE V

PERFORMANCE OF DIFFERENT MODULE COMBINATIONS ON THE ICDAR2013/2015 DATASET

Approach	Positive No.	Negative No.	Precision	Recall	F-measure
M1	10.52	2.82	0.721	0.807	0.762
M1+M3	10.11	1.05	0.857	0.831	0.844
M1+M2	23.08	6.92	0.614	0.823	0.703
M1+M2+M3	**20.67**	**0.24**	**0.919**	**0.871**	**0.895**

TABLE VI

PERFORMANCE OF TEXT DETECTION WITH DIFFERENT MODULE/CNN COMBINATIONS ON THE ICDAR2013/2015 DATASET

Approach	Precision	Recall	F-measure
M1/DNet+M3	0.857	0.831	0.844
M1/SNet+M3	0.848	0.825	0.836
M1/SNet+M2/SNet+M3	0.893	0.853	0.873
M1/SNet+M2/DNet+M3	0.884	0.846	0.865
M1/DNet+M2/DNet+M3	0.904	0.863	0.883
M1/DNet+M2/SNet+M3	**0.919**	**0.871**	**0.895**

only with the module M1 is promising, which builds a good foundation for the entire model. (2) When the module M3 is integrated, the negative detected regions are remarkably reduced and the text detection performance largely improves (e.g. the F-measures increase 8.2% and 19.2% from M1 to M1+M3 and from M1+M2 to M1+M2+M3, respectively). That is because the module M3 has the powerful ability to filter out the non-text regions. (3) The positive/negative regions detected by M1+M2+M3 are much more/less than those detected by M1+M3, and the text detection performance is much better. The reason is that M2 can accurately segment the detected CTRs, which may contain multiple words, into characters. To summarize, M1, M2 and M3 have different contributions for text detection, and cascading these modules can lead to state-of-the-art performance.

The three modules of the proposed method are accomplished with the cascaded CNNs, i.e. DNet, SNet, and CNet. For the model training of DNet and SNet, the text regions or edges are used as supervisory information. During testing, both DNet and SNet pay attention to text regions. Therefore, they can be used for the tasks of M1 and M2. To evaluate the performance of using different CNNs in different modules, six different module/CNN combinations are tested on the ICDAR2013/2015 dataset. For example, when DNet is used for M1 and SNet is used for M2, this combination is denoted as M1/DNet+M2/SNet+M3. Table VI lists the results of the above experiments. From Table VI, we can see that DNet is better for M1 than SNet while SNet is better for M2 than DNet. That's because for M1, the model should locate the initial positions of text as precisely as possible. Compared with SNet, DNet always focuses on learning discriminative features of text from shallow to deep. For M2, the model should segment the text regions as precisely as possible. Compared with DNet, SNet is able to segment the text regions from coarse to fine due to multiple deconvolutional layers and the integration of high-level features and low-level features. This is the reason that the proposed method (i.e. M1/DNet+M2/SNet+M3) gets the best performance.

TABLE VII

PERFORMANCE OF TEXT DETECTION WITH DIFFERENT SUPERVISORY STRATEGIES ON THE ICDAR2013/2015 DATASET

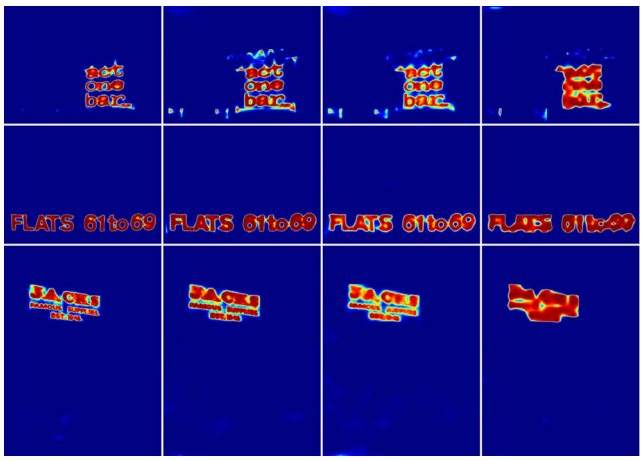
Approach	Precision	Recall	F-measure
Strategy1	**0.919**	**0.871**	**0.895**
Strategy2	0.910	0.867	0.888
Strategy3	0.893	0.854	0.873
Strategy4	0.885	0.849	0.867

F. Performance of Different Supervisory Strategies

To investigate the effect of different supervisory strategies for text-aware CTR extraction, besides the proposed strategy (denoted as Strategy1), we also test three others. The first one uses region supervision in all places (denoted as Strategy2). The second one does not use edge supervision in shallow layers (denoted as Strategy3). The third one only uses region supervision in the last block (denoted as Strategy4). Obviously, from Strategy1 to Strategy4, the supervisory information becomes less and less. The models with different supervisory strategies are trained using the same dataset. Table VII lists the performance of the different strategies for text detection on the ICDAR2013/2015 dataset and Fig. 12 gives three visual examples of text-aware saliency detection with these models. From Table VII and Fig. 12, we can see that (1) from Strategy4 to Strategy1, both the qualities of the detected salient maps and the performance of text detection become better and better, which demonstrates that more supervisory information creates better salient maps and more accurate text detection results. This is because more discriminative and robust features can be learned when more supervisory information is involved. (2) The boundaries of text in the salient maps detected by Strategy1 are sharper than those detected by Strategy2 and the performance of text detection of Strategy1 is also better than that of Strategy2. The reason is that, as supervisory information, the edges of text are more powerful for low-level feature learning than the regions of texts.

G. Performance of the Improvement in Each Module

Based on traditional approaches, some improvements are provided in all modules of the proposed method. For the design of DNet, we use different supervisory information for different layers (i.e. the edges of text used in shallow layers and the regions of text used in deep layers) compared to directly using regions of texts as the supervisory information in all layers, this improvement is denoted as IMP1. For the design of SNet, we integrate the information of shallower layers of the convolution network into the deeper layers of the deconvolution network compared to the original deconvolution network [23], this improvement is denoted as IMP2. For the CTR classification, we propose a new region construction scheme by considering the high discriminability of multiple characters compared to the traditional region resizing scheme,



(a) (b) (c) (d)

Fig. 12. Text-aware saliency detection results of Fig. 4(a) with different supervisory strategies. (a) The proposed supervisory strategy, i.e. Strategy1. (b) Strategy2. (c) Strategy3. (d) Strategy4.

TABLE VIII

PERFORMANCE OF OUR IMPROVEMENTS ON THE ICDAR2013/ICDAR2015 DATASET

Approach		Precision	Recall	F-measure
Text detection	TRA1	0.910	0.867	0.888
	TRA2	0.896	0.857	0.876
	TRA3	0.905	0.859	0.881
	Ours	**0.919**	**0.871**	**0.895**
Text segmentation	TRA2	0.9218	0.8186	0.8671
	Ours	**0.9470**	**0.8560**	**0.8992**

this improvement is denoted as IMP3. In order to demonstrate the usefulness of these improvements, we conducted experiments using one traditional approach to replace its corresponding improvement of our method. For example, when we test the performance of IMP3, we use the traditional region resizing scheme to replace our region construction scheme and fix the rest of our method, then test its performance (denoted as TRA3) on the ICDAR2013/ICDAR2015 dataset. Table VIII lists the performance of our improvements (denoted as TRA1, TRA2, and TRA3 for IMP1, IMP2, and IMP3) on the ICDAR2013/ICDAR2015 dataset. From Table VIII, we can see that each of the improved modules can improve the performance of text detection and the largest gain is obtained by IMP2 of the SNet design. These results demonstrate the effectiveness of our improvements.

Since the original deconvolution network [23] is used for image segmentation, we also test its performance with text segmentation. According to Table VIII, our improvement (i.e. IMP2) based on the original deconvolution network [23] greatly improves the performance of text segmentation. That is because the feature maps of the shallow layers can provide some assisting information to help the feature learning of the deep layers, so as to get more precise segmentation results.

VII. CONCLUSIONS

This paper proposes a novel scene text detection method by using multiple convolutional neural networks. This method consists of three steps including text-aware CTR extraction, CTR refinement, and CTR classification. All steps are accomplished by adopting convolutional neural networks, which makes the proposed method more robust and effective than other approaches. The proposed text-aware CTR extraction model can extract more true text regions and much fewer false text regions than other approaches. The CTR refinement model can effectively remove the background from each CTR and the false text regions. The CNN-based CTR classification model can correctly classify the CTRs and get the true text regions. With all of these processes, the proposed method gets the best recall, F-measure and comparable precision on three benchmark datasets. The proposed method also has the ability to find some text which is not annotated in the ground truth due to the power of all of the CNN networks used.

REFERENCES

[1] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 robust reading competition challenge 2: Reading text in scene images," in *Proc. Int. Conf. Document Anal. Recognit.*, Sep. 2011, pp. 1491–1496.

[2] D. Karatzas *et al.*, "ICDAR 2013 robust reading competition," in *Proc. Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 1484–1493.

[3] D. Karatzas *et al.*, "ICDAR 2015 competition on robust reading," in *Proc. Int. Conf. Document Anal. Recognit.*, Aug. 2015, pp. 1156–1160.

[4] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2963–2970.

[5] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image Vis. Comput.*, vol. 22, no. 10, pp. 761–767, 2004.

[6] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2594–2605, Sep. 2011.

[7] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1083–1090.

[8] L. Neumann and J. Matas, "Real-time scene text localization and recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3538–3545.

[9] W. Huang, Z. Lin, J. Yang, and J. Wang, "Text localization in natural images using stroke feature transform and text covariance descriptors," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1241–1248.

[10] L. Neumann and J. Matas, "Scene text localization and recognition with oriented stroke detection," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2013, pp. 97–104.

[11] C. Shi, C. Wang, B. Xiao, Y. Zhang, and S. Gao, "Scene text detection using graph model built upon maximally stable extremal regions," *Pattern Recognit. Lett.*, vol. 34, no. 2, pp. 107–116, Jan. 2013.

[12] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao, "Robust text detection in natural scene images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 970–983, May 2014.

[13] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Proc. Asian Conf. Comput. Vis.*, 2010, pp. 770–783.

[14] W. Huang, Y. Qiao, and X. Tang, "Robust scene text detection with convolution neural network induced MSER trees," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 497–511.

[15] Y. Li, W. Jia, C. Shen, and A. van den Hengel, "Characterness: An indicator of text in the wild," *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1666–1677, Apr. 2014.

[16] A. Zamberletti, L. Noce, and I. Gallo, "Text localization based on fast feature pyramids and multi-resolution maximally stable extremal regions," in *Proc. Workshop Asian Conf. Comput. Vis.*, 2014, pp. 91–105.

[17] C. Yao, X. Bai, and W. Liu, "A unified framework for multioriented text detection and recognition," *IEEE Trans. Image Process.*, vol. 23, no. 11, pp. 4737–4749, Nov. 2014.

[18] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Deep features for text spotting," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 512–528.

[19] Z. Zhang, W. Shen, C. Yao, and X. Bai, "Symmetry-based text line detection in natural scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2558–2567.

[20] S. Qin and R. Manduchi, "A fast and robust text spotter," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2016, pp. 1–8.
[21] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 56–72.
[22] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, and C. L. Tan, "Text flow: A unified text detection system in natural scene images," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4651–4659.
[23] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1520–1528.
[24] K. Wang and S. Belongie, "Word spotting in the wild," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2010, pp. 591–604.
[25] Y.-F. Pan, X. Hou, and C.-L. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 800–813, Mar. 2011.
[26] H. I. Koo and D. H. Kim, "Scene text detection via connected component clustering and nontext filtering," *IEEE Trans. Image Process.*, vol. 22, no. 6, pp. 2296–2305, Jun. 2013.
[27] C. Yao, X. Bai, B. Shi, and W. Liu, "Strokelets: A learned multi-scale representation for scene text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 4042–4049.
[28] L. Kang, Y. Li, and D. Doermann, "Orientation robust text line detection in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 4034–4041.
[29] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2004, pp. 366–373.
[30] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2013, pp. 785–792.
[31] C. Yi and Y. Tian, "Text detection in natural scene images by stroke Gabor words," in *Proc. Int. Conf. Document Anal. Recognit.*, Sep. 2011, pp. 177–181.
[32] C. Yi and Y. Tian, "Text extraction from scene images by character appearance and structure modeling," *Comput. Vis. Image Understand.*, vol. 117, no. 2, pp. 182–194, Feb. 2013.
[33] H. Xu, L. Xue, and F. Su, "Scene text detection based on robust stroke width transform and deep belief network," in *Proc. Asian Conf. Comput. Vis.*, 2014, pp. 195–209.
[34] M. Fraz, M. S. Sarfraz, and E. A. Edirisinghe, "Exploiting colour information for better scene text detection and recognition," *Int. J. Document Anal. Recognit.*, vol. 18, no. 2, pp. 153–167, Jun. 2015.
[35] T. He, W. Huang, Y. Qiao, and J. Yao, "Text-attentional convolutional neural network for scene text detection," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2529–2541, Jun. 2016.
[36] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *Int. J. Comput. Vis.*, vol. 116, no. 1, pp. 1–20, Jan. 2016.
[37] S. Lu, T. Chen, S. Tian, J.-H. Lim, and C.-L. Tan, "Scene text extraction based on edges and support vector regression," *Int. J. Document Anal. Recognit.*, vol. 18, no. 2, pp. 125–135, Jun. 2015.
[38] T. He, W. Huang, Y. Qiao, and J. Yao. (Mar. 2016). "Accurate text localization in natural image with cascaded convolutional text network." [Online]. Available: https://arxiv.org/abs/1603.09423
[39] S. Zhu and R. Zanibbi, "A text detection system for natural scenes with convolutional feature learning and cascaded classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 625–632.
[40] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4159–4167.
[41] H. Cho, M. Sung, and B. Jun, "Canny text detector: Fast and robust scene text localization algorithm," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3566–3573.
[42] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2315–2324.
[43] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 7, pp. 1480–1500, Jul. 2014.
[44] K. I. Kim, K. Jung, and J. H. Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1639, Dec. 2003.
[45] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1457–1464.
[46] A. Mishra, K. Alahari, and C. V. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2687–2694.
[47] B. Bai, F. Yin, and C. L. Liu, "A seed-based segmentation method for scene text extraction," in *Proc. Int. Workshop Document Anal. Syst.*, Apr. 2014, pp. 262–266.
[48] J. H. Bosamiya, P. Agrawal, P. P. Roy, and R. Balasubramanian, "Script independent scene text segmentation using fast stroke width transform and GrabCut," in *Proc. Asian Conf. Pattern Recognit.*, Nov. 2015, pp. 151–155.
[49] A. Mishra, K. Alahari, and C. V. Jawahar, "An MRF model for binarization of natural scene text," in *Proc. Int. Conf. Document Anal. Recognit.*, Sep. 2011, pp. 11–16.
[50] Y. Zhou, J. Feild, E. Learned-Miller, and R. Wang, "Scene text segmentation via inverse rendering," in *Proc. Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 457–461.
[51] S. Tian, S. Lu, B. Su, and C. L. Tan, "Scene text segmentation with multi-level maximally stable extremal regions," in *Proc. 22nd Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2014, pp. 2703–2708.
[52] C.-Y. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu, "Deeply-supervised nets," in *Proc. AISTATS*, 2015, pp. 562–570.
[53] S. Xie and Z. Tu, "Holistically-nested edge detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Dec. 2015, pp. 1395–1403.
[54] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 3431–3440
[55] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. ACM Multimedia Conf.*, Nov. 2014, pp. 675–678.
[56] *Result of ICDAR2015 Robust Reading Competition*, accessed on Jun. 25, 2016. [Online]. Available: http://rrc.cvc.uab.es/?ch=2&com=evaluation.

Youbao Tang received the B.S. and M.Sc. degrees in computer science from Harbin Institute of Technology (HIT), Harbin, China, in 2009 and 2011 respectively, where he is currently pursuing the Ph.D. degree in computer science. His current research interests include image processing, pattern recognition, computer vision, and biometrics.

Xiangqian Wu (MâŁ™06) received the B.Sc., M.Sc., and Ph.D. degrees from the Harbin Institute of Technology (HIT), Harbin, China, in 1997, 1999, and 2004, respectively, all in computer science. He was a lecturer from 2004 to 2006, an Associate Professor from 2006 to 2009, and a Professor since 2009 with the School of Computer Science and Technology, HIT. He has authored one book and over 100 papers in international journals and conferences. His current research interests include computer vision, pattern recognition, and biometrics and medical image analysis.