Deep Learning Based Container Text Recognition

Weishan Zhang
College of computer and Communication Engineering
China University of Petroleum
Qingdao, China
zhangws@upc.edu.cn

Liqian Zhu
College of computer and Communication Engineering
China University of Petroleum
Qingdao, China
zhuliqiancuop@sina.com

Liang Xu
College of computer and Communication Engineering
Beijing University of Science and Technology
Beijing, China

Jiehan Zhou
University of Oulu
Oulu, Finland
jiehan.zhou@oulu.fi

Haoyun Sun
College of computer and Communication Engineering
China University of Petroleum
Qingdao, China
hys upc@163.com

Xin Liu
College of computer and Communication Engineering
China University of Petroleum
Qingdao, China
lx@upc.edu.cn

***Abstract*—Traditional character segmentation has low accuracy for container scene text recognition. Convolutional recurrent neural network (CRNN) and connectionist text proposal network (CTPN) methods cannot extract container text features effectively. This paper proposes a novel Container Text Detection and Recognition Network (CTDRNet) for accurately detecting and recognizing container scene text. The CTDRNet consists of three components: (1) CTDRNet text detection enables to improve detection accuracy for single words; (2) CTDRNet text recognition has faster convergence speed and detection accuracy; (3) CTDRNet post-processing improves detection and recognition accuracy. In the end, the CTDRNet is implemented and evaluated with an accuracy of 96% and processing rate of 2.5 fps.**

Keywords—deep learning, scene text detection, scene text recognition, container

I. INTRODUCTION

Traditional container management systems require manual text entries, which is prone to errors. An automatic text recognition system becomes necessary for modern, accurate and reliable container management.

Automatic container text detection and recognition is a challenging task because text background images usually have complex and illumination conditions, peeling paint and stains, as shown in Fig. 1. Traditional character segmentation is vulnerable to variable environments which lead to low recognition accuracy.

Deep learning has developed rapidly, giving an opportunity to improve scene text detection and recognition. The object detection algorithm promotes the development of text detection. For example, Ren et al. [1] introduced the Faster R-CNN, which proposes an object detection algorithm based on region proposal. Liu et al. [2] introduced the Single Shot Multibox Detector (SSD), which directly predicts the coordinates and classes of bounding boxes without a proposal. Some novel image classification algorithms can extract more efficient image feature. For example, Huang et al. [3] introduced the Densely Connected Convolutional Networks (DenseNet), which connects each layer to every other layer in a feed-forward fashion and requires less computation to achieve high performance, and He et al. [4] introduced the Residual Network (ResNet), which explicitly reformulates the layers as learning residual functions with reference to the layer inputs and can gain accuracy from considerably increased depth. Graves et al. [5] introduced the Connectionist Temporal Classification (CTC), which is applied to text recognition to realize the conversion of image sequences to variable length character sequences. These methods promote the development of scene text detection and recognition.

Fig. 1. Example of container text image.

Deng et al. [6] proposed the PixelLink framework using the segmentation method for text detection. This method only detects a portion of text, and results in inaccurate recognition. Wojna et al. [7] proposed the house number recognition based on the attention mechanism, which only focuses on a portion of text as well and causes missed detection.

We propose CTDRNet for improving container text recognition accuracy and performance. The Container Text Detection Network (CTDNet) for container text detection enables to improve detection accuracy for single words. The Container Text Recognition Network (CTRNet) for container text recognition uses Container Text Recognition Basic Network (CTRBNet) as the basic network for feature extraction, followed by Bi-directional LSTM (Long Short-Term Memory) [8] and CTC [5] layer for text recognition. The CTDRNet could detect and recognize container text with an accuracy of 96% .

The remainder of the paper is organized as follows. Section II discusses the related work. Section III presents the CTDRNet design and implementation. Section IV evaluates the CTDRNet. Section IV draws conclusions.

II. Related Work

A standard container text has 11-digits, consisting of three parts. As shown in Fig. 2, Part 1 consists of four English letters 'TGBU', Part 2 consists of six digits '236004', Part 3 is the check code '3'. The ideal automatic system should be able to separate and detect each part. Container text recognition could be divided into two phases: text detection and recognition. This section reviews related works.

A. Text Detection

The character-based text detection detects character one by one before connecting them as a word, which has poor detection accuracy. There are many word-based detection methods. Bazazian et al. [9] improved the original Text Proposal algorithm by Gomez and Karatzas through combining fully convolutional networks to improve the proposal ranking. Rong et al. [10] proposed a novel recurrent Dense Text Localization Network to sequentially decode the intermediate convolutional representations of a cluttered scene image into a set of distinct text instance detections. Wang et al. [11] proposed a geometry-aware modeling approach tailored for scene text representation with an end-to-end learning scheme. Liu et al. [12] proposed a Markov Clustering Network that can detect text objects with arbitrary size and orientation. Our method has a simpler network structure than the above mentioned approaches and a well performance.

B. Text Recognition

The existing text recognition methods can be divided into CTC-based and attention mechanism-based methods. Wu et al. [13] proposed a scene text recognition method with sliding

Fig. 2. Container text example.

convolutional attention network. Liu et al. [14] proposed a binary convolutional encoder-decoder network together with a bidirectional recurrent neural network, which can provide accurate character recognition. Existing attention-based methods perform poorly on complicated and/or low-quality images. Cheng et al. [15] called this phenomenon "attention drif" and proposed Focusing Attention Network (FAN) method to automatically draw back the drifted attention. To improve the recognition accuracy, we design a specific network module for effectively extracting basic features.

III. CTDRNet Design and Implementation

A. Overview

The CTDRNet consists of three phases as shown in Fig. 3. Phase 1 - CTDRNet text detection precisely localizes a single text. Phase 2 - CTDRNet text recognition recognizes the detected words and converts image sequences into variable-length sequences. Phase 3 - CTDRNet post-processing improves detection and recognition accuracy. The CTDRNet improves the convergence speed and detection accuracy with our designed Container Text Detection Basic Network (CTDBNet) and Container Text Recognition Basic Net (CTRBNet). Each phase is detailed as follows.

B. Container Text Detection Network (CTDNet)

The CTDNet consists of three parts: CTDBNet, Bi-directional LSTM, and text line connection (Fig. 4).

In the phase 1, the CTDNet text detection uses a deep convolutional neural network called CTDBNet for extracting features. The convolution layer uses a 3×3 convolution kernel, consists of five convolution groups and each convolution group consists of six layers. Convolution groups are connected using a

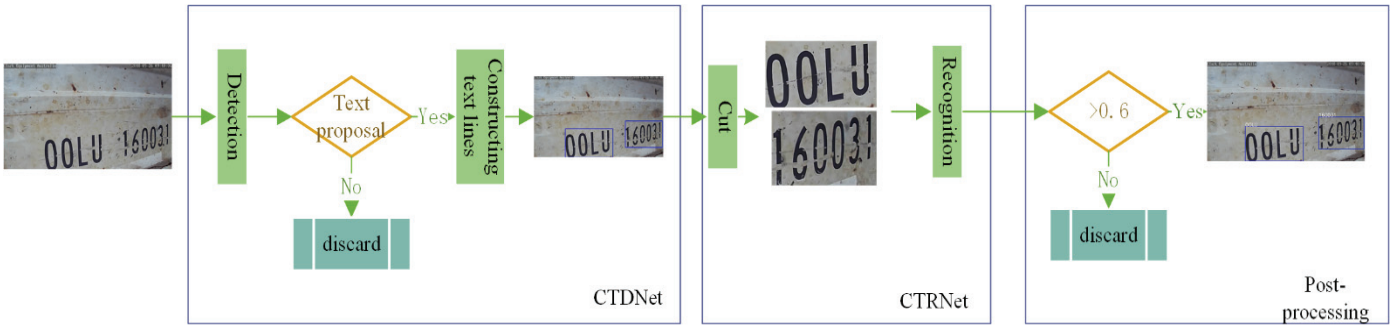

Fig. 3. CTDRNet work flow.

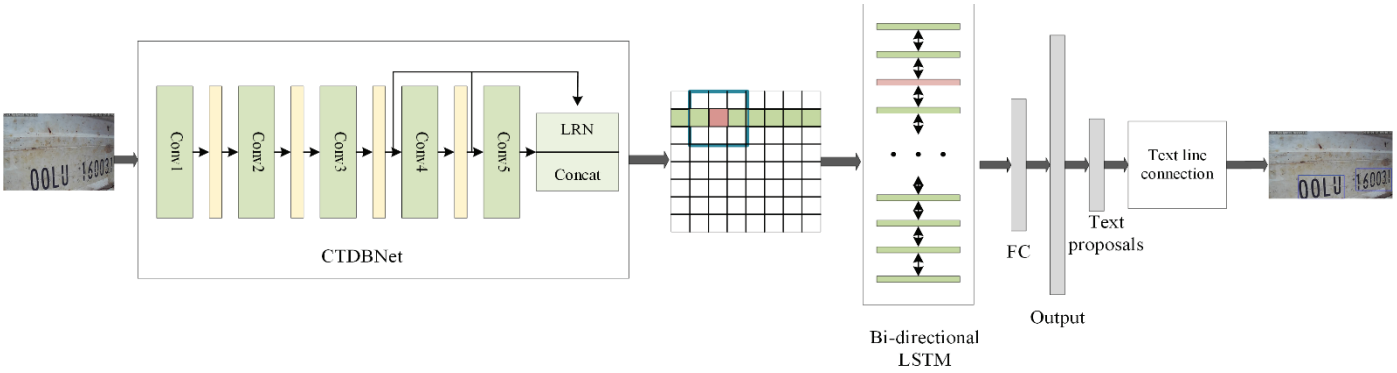

Fig. 4. The CTDNet work flow, where yellow layers indicate connection layers.

connection layer, which consists of a Relu (Rectified Linear Unit) activation function, a 1×1 convolution, an average pooling, and a dropout layer. The CTDNet uses the 1×1 convolution to reduce the output dimensions produced by the convolutional group without changing the feature extraction performance. The dropout effectively reduces the chance of overfitting in deep networks. Although the network composed of multiple convolution groups can extract good semantic information, the increase of depth makes the spatial information rough. In order to enhance the spatial information of the feature map, the feature maps of the last three convolution groups are further processed by local response normalization(LRN) and hyper-connection.

The CTDNet text detection is a fine-grained detection task. In the phase 2, the CTDNet uses the vertical anchor mechanism [16] to obtain accurate text location, in which the width is set to 16 pixels. The CTDNet slides a $3\times3\times C$ window over the last feature map, and then inputs the captured features into the Bi-directional LSTM and output a $W\times256$ matrix. W is the width of the feature map. C is the number of channels. The number of hidden layer neurons in the Bi-directional LSTM is 256. Followed by the 512D full-connected layer. The CTDNet outputs the text/non-text score and vertical location of text proposal.

In the phase 3, all the predicted text proposals can be obtained from the output. Due to the vertical anchor mechanism [16], we need to build them into text line. The construction method is as follows: define a paired neighbor A for a proposal B as $A\rightarrow B$ that meets the following conditions: (1) A is the horizontal distance closest to B, (2) the distance is less than 20 pixels, and (3) their vertical overlap is greater than 0.7. If $A\rightarrow B$ and $B\rightarrow A$, the two proposals are grouped into a pair. A text line is constructed by sequentially joining proposals that satisfy the condition. This method of construction solves the problem of detecting multiple words as one. The test results are shown in Fig. 5.

Loss function is a multi-task loss function, which consists of classification loss and location loss. Equation (1) is the definition of loss function. L_{cls}^{p} is the classification loss which we use logarithmic loss function to distinguish text and non-text. L_{loc}^{l} is the location loss which we use the smooth L_1 function to compute it. In the vertical anchor mechanism[16], each anchor is a training sample. i is the index of an anchor in a mini-batch.

Fig. 5. CTDNet detection results.

p_i is the probability that the text proposal corresponding to the anchor i is predicted as text. p_i^* is the ground true label of the anchor i, $p_i^*=1$ means the ground true label of the anchor i is text, and $p_i^*=0$ means the ground true label of the anchor i is not text. j is the index of valid anchors. The valid anchor means $p_i^*=1$ or overlaps with the ground true text proposal by more than 0.6. l_j is the vertical coordinate of the predicted anchor j, and l_j^* is the ground truth vertical coordinate of the anchor j. N_{cls} and N_{loc} normalize the two sub-items in (1). λ is used to control the relative importance of two sub-items.

$$L(p_i,l_j)=\frac{1}{N_{cls}}\sum_i L_{cls}^{p}(p_i,p_i^*)+\frac{\lambda}{N_{loc}}\sum_j L_{loc}^{l}(l_j,l_j^*) \qquad (1)$$

C. Container Text Recognition Network(CTRNet)

The CTRNet (Fig. 6) consists of CTRBNet, Bi-directional LSTM, and CTC layer, which could be regarded as an encoder decoder structure. The CTRBNet is an encoder that outputs image feature sequences. The Bi-directional LSTM and CTC layer compose a decoder that outputs character sequences.

1) Encoder: The CTRNet sets the input image height to 32 pixels, and the width to $(W\times32)/H$ so that it does not change the aspect ratio of the input image, where W, H are the image width and height. Considering that the character is thin and tall, its height is greater than its width, the CTRNet adopts the stride 2×1 for the pooling layer rather than the stride 2×2. Therefore, in the final feature map, a pixel point corresponding to the receptive field in the original image is thin and tall. The input image is downsampled using two pooling layers with stride 2×2 and three pooling layers with stride 2×1. The final

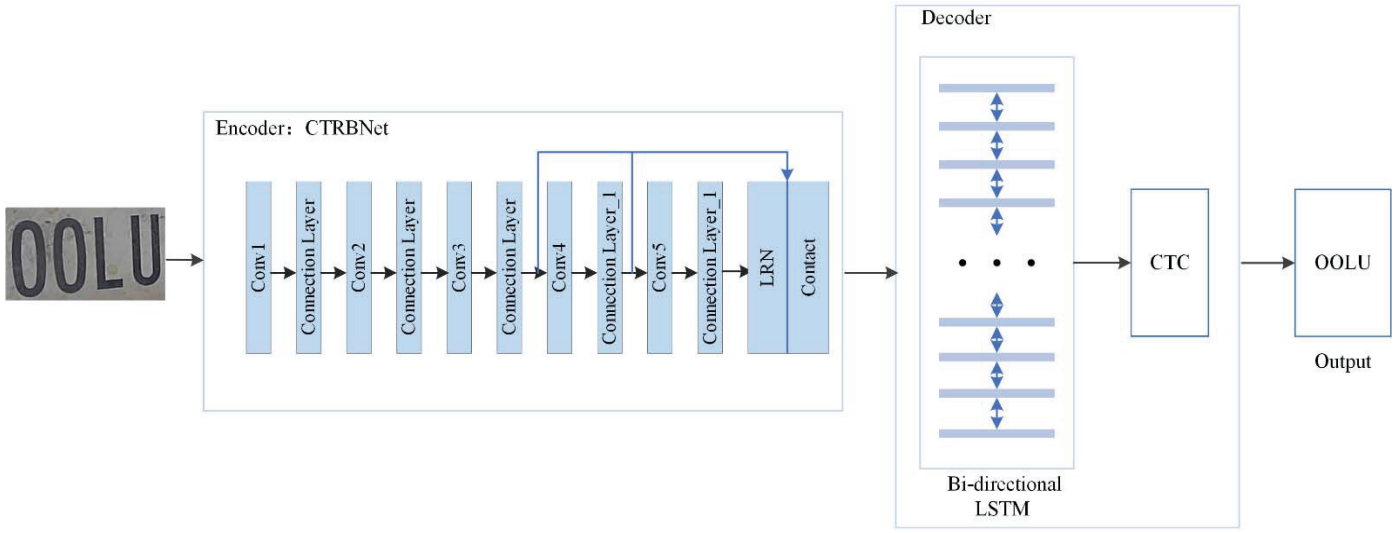

Fig. 6. The CTRNet work flow.

feature map dimension is $b\times1\times[(W\times8)/H]\times C$, b is batch size, 1 is height, $(W\times8)/H$ is width, and C is the number of channels. TABLE I. lists the structure of the CTRBNet for feature extraction.

TABLE I. CTRBNET STRUCTURE

Layers	CTRBNet	Output Size
Conv1	$(3\times3\times conv)\times6$	$32\times[(W\times32)/H]$
Connection Layer	Relu, 1×1 conv,droupout	$32\times[(W\times32)/H]$
	2×2 average pool, Stride 2×2	$16\times[(W\times16)/H]$
Conv2	$(3\times3\times conv)\times6$	$16\times[(W\times16)/H]$
Connection Layer	Relu, 1×1 conv,droupout	$16\times[(W\times16)/H]$
	2×2 average pool, Stride 2×2	$8\times[(W\times8)/H]$
Conv3	$(3\times3\times conv)\times6$	$8\times[(W\times8)/H]$
Connection Layer_1	Relu, 1×1 conv,droupout	$8\times[(W\times8)/H]$
	2×1 average pool, Stride 2×1	$4\times[(W\times8)/H]$
Conv4	$(3\times3\times conv)\times6$	$4\times[(W\times8)/H]$
Connection Layer_1	Relu, 1×1 conv,droupout	$4\times[(W\times8)/H]$
	2×1 average pool, Stride 2×1	$2\times[(W\times8)/H]$
Conv5	$(3\times3\times conv)\times6$	$2\times[(W\times8)/H]$
Connection Layer_1	Relu, 1×1 conv,droupout	$2\times[(W\times8)/H]$
	2×1 average pool, Stride 2×1	$1\times[(W\times8)/H]$

2) Decoder: The CTRNet decoder consists of the Bi-directional LSTM and CTC layer. The input of Bi-directional LSTM is the column vector of a feature map. The output is the probability of characters corresponding to each column vector which composes a probability matrix of $(W\times8)/H\times c$, where c is the number of character labels and set to 63 in the experiment, including 0-9 Arabic numerals, 26 uppercase English letters, 26 lowercase English letters, and a space. $(W\times8)/H$ is the width of the extracted feature map.

The label sequence probability is obtained by passing the output of Bi-directional LSTM to the CTC layer. During training, We adopt the conditional probability defined in the CTC layer as the probability of label sequence. The negative log-likelihood of conditional probability as the loss function to train the network. The CTC layer calculates the sum of the probabilities of all paths that are true label sequences. For example, the path 'o-oo-l-uu-' and the path 'oo-o-ll-u-' (where '-' indicates a space) remove duplicates and spaces to indicate the label sequence 'oolu'. During the test, the character sequence with the highest probability is selected as the recognition result.

3) Post-processing: The CTDRNet post-processing is proposed to improve the detection and recognition accuracy.

The CTC layer calculates all possible character sequences in the input image and the corresponding probabilities. The CTDRNet post-processing chooses the probability greater than 0.6 as the output because the probability of misrecognition as a character sequence is generally less than 0.6. The CTDRNet post-processing discards text boxes with the probability less than 0.6, which effectively reduces the rate of false detections and increases two percentage points on our test set. Fig. 7 shows the recognition results with and without post-processing.

IV. EXPERIMENT AND EVALUATION

A. Experimental Configuration

The configuration for testing the CTDRNet model is as follows: Intel CoreTM i7-5930K CPU @ 3.50GHz × 12, 16G memory, and TITAN X (Pascal) graphics card.

B. Dataset

The Synth80k is a synthetic dataset that randomly horizontally adds words into a series of scene images, which has

(a) without post-processing (b) with post-processing

Fig. 7. Group (a) pictures are the recognition results without the post-processing, and group (b) pictures are the recognition results with the post-processing.

a total of 858,750 images for text detection and recognition. The Synth90k is a synthetic English word dataset used for horizontal text recognition. We use 5000 container pictures as training sets and 500 as test sets. The file for text detection is with the VOC format. The file for text recognition is with the txt format, and each line is composed of image names and character sequences corresponding to the image.

C. Experimental Steps

Training the CTDNet model. First, we convert the synth80 dataset into VOC, and set batch size to 32, with 80,000 iterations. Then we fine-tune the trained model by using the collected container images with 10,000 iterations until it achieves good convergence.

Training the CTRNet model. First, we use the synth90k dataset for training and set batch size to 128, and with 60,000 iterations. Then we fine-tune the trained model by using the real container images with 2000 iterations until it achieves good convergence.

Then we use the well-trained CTDNet and CTRNet models to carry out end-to-end container text detection and recognition, which achieves an accuracy of 96%.

D. Experimental Results and Analysis

CTDRNet consists of two network, including container text detection network (CTDNet) and container text recognition network (CTRNet). The quality of basic feature extraction network determines the accuracy of detection and recognition. In order to verify that the basic feature extraction networks in our two networks have a better performance in feature extraction, we compare them with VGG16. In the TABLE II. , CTDNet(VGG-16) denotes that the VGG16 is the basic feature extraction network for container text detection, and CTDNet uses the basic feature extraction network CTDBNet that we proposed. CTRNet(VGG-16) denotes the basic feature extraction network using VGG16 for text recognition, and CTRNet represents the use of our proposed basic feature extraction network CTRBNet. CTDRNet(without post-processing) represents text detection and recognition without post-processing steps. CTDRNet represents text detection and recognition using post-processing. The experimental results are

Fig. 8. CTDRNet recognition results.

shown in TABLE II. . Our proposed CTDNet has 20 percentage points higher than that of CTPN [16] at detection accuracy with non-noticeable speed reduction. CTDNet using CTDBNet as the basic feature extraction network is 4 percentage points higher than that using VGG16, which shows that our proposed CTDBNet can extract features more effectively. CTRNet using CTRBNet as the basic feature extraction network is 3 percentage points higher than using VGG16, which shows that our proposed CTRBNet can extract features more effectively. The CTDRNet with post-processing has 2 percentage points higher than that of CTDRNet without post-processing at the detection and recognition accuracy with non-noticeable speed reduction. Post-processing steps can effectively reduce the phenomenon of false detection, thus improving the recognition accuracy. The final detection result is shown in Fig. 8.

TABLE II. PERFORMANCE COMPARISON

Net	Detection Accuracy	Recognition Accuracy	Detection and Recognition Accuracy	Speed (fps)
CTPN	0.73	-	-	6
CTDNet	0.96	-	-	5.5
CTDNet (VGG-16)	0.92	-	-	6
CTRNet	-	0.98	-	5
CTRNet (VGG-16)	-	0.95	-	5
CTDRNet	-	-	0.96	2.5
CTDRNet (without post-processing)	-	-	0.94	2.5

In the experiment, we find that the CTRNet reached the accuracy of 98%, while CTRNet (VGG-16) was only 89% under 56,000 iterations. This shows that CTRNet has a faster convergence speed than the CTRNet (VGG-16).

V. CONCLUSIONS

We present the CTDRNet model for container scene text detection. The experimental results demonstrate our CTDRNet method could extract feature effectively and improve recognition accuracy and convergence speed.

In the future, we will further improve the recognition accuracy of the CTDRNet with the vertical container text detection and recognition.

ACKNOWLEDGMENT

The research is supported by the Innovative Method special project of the Ministry of Science and Technology (Grant No. 2015IM010300), Key Research Program of Shandong Province (2017GGX10140), the Fundamental Research Funds for the Central Universities (Grant No. 2015020031).

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards realtime object detection with region proposal networks," in International Conference on Neural Information Processing Systems, 2015, pp. 91–99.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in Computer Vision – ECCV 2016, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.

[3] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks." in CVPR, vol. 1, no. 2, 2017, p. 3.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[5] A. Graves and F. Gomez, "Connectionist temporal classification:labelling unsegmented sequence data with recurrent neural networks," in International Conference on Machine Learning, 2006, pp. 369– 376.

[6] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," 2018.

[7] Z.Wojna,A.N.Gorban,D.S.Lee,K.Murphy,Q.Yu,Y.Li,andJ.Ibarz,"Attention-based extraction of structured information from street view imagery," pp. 844–850, 2017.

[8] Z. Huang, W. Xu, and K. Yu, "Bidirectional lstm-crf models for sequence tagging," arXiv preprint arXiv:1508.01991, 2015.

[9] D. Bazazian, R. Gomez, A. Nicolaou, L. Gomez, D. Karatzas, and A. D. Bagdanov, "Improving text proposals for scene images with fully convolutional networks," 2017.

[10] X. Rong, C. Yi, and Y. Tian, "Unambiguous text localization and retrieval for cluttered scenes," in Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on. IEEE, 2017, pp. 3279–3287.

[11] F. Wang, L. Zhao, X. Li, X. Wang, and D. Tao, "Geometry-aware scene text detection with instance transformation network," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1381–1389.

[12] Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and W. L. Goh, "Learning markov clustering networks for scene text detection," 2018.

[13] Y. C. Wu, F. Yin, X. Y. Zhang, L. Liu, and C. L. Liu, "Scan: Sliding convolutional attention network for scene text recognition," 2018.

[14] Z. Liu, Y. Li, F. Ren, W. L. Goh, and H. Yu, "Squeezedtext: A realtime scene text recognition by binary convolutional encoder-decoder network." in AAAI, 2018.

[15] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou, "Focusing attention: Towards accurate text recognition in natural images," in Computer Vision (ICCV), 2017 IEEE International Conference on. IEEE, 2017, pp. 5086–5094.

[16] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in Computer Vision – ECCV 2016, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 56–72.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," arXiv preprint arXiv:1409.1556, 2014.