FOTS: Fast Oriented Text Spotting with a Unified Network

Xuebo Liu[1], Ding Liang[1], Shi Yan[1], Dagui Chen[1], Yu Qiao[2], and Junjie Yan[1]
[1]SenseTime Group Ltd.
[2]Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
{liuxuebo,liangding,yanshi,chendagui,yanjunjie}@sensetime.com, {yu.qiao}@siat.ac.cn

Abstract

Incidental scene text spotting is considered one of the most difficult and valuable challenges in the document analysis community. Most existing methods treat text detection and recognition as separate tasks. In this work, we propose a unified end-to-end trainable Fast Oriented Text Spotting (FOTS) network for simultaneous detection and recognition, sharing computation and visual information among the two complementary tasks. Specifically, RoIRotate is introduced to share convolutional features between detection and recognition. Benefiting from convolution sharing strategy, our FOTS has little computation overhead compared to baseline text detection network, and the joint training method makes our method perform better than these two-stage methods. Experiments on ICDAR 2015, ICDAR 2017 MLT, and ICDAR 2013 datasets demonstrate that the proposed method outperforms state-of-the-art methods significantly, which further allows us to develop the first real-time oriented text spotting system which surpasses all previous state-of-the-art results by more than 5% on ICDAR 2015 text spotting task while keeping 22.6 fps.

1. Introduction

Reading text in natural images has attracted increasing attention in the computer vision community [49, 43, 53, 44, 14, 15, 34], due to its numerous practical applications in document analysis, scene understanding, robot navigation, and image retrieval. Although previous works have made significant progress in both text detection and text recognition, it is still challenging due to the large variance of text patterns and highly complicated background.

The most common way in scene text reading is to divide it into text detection and text recognition, which are handled as two separate tasks [20, 34]. Deep learning based approaches become dominate in both parts. In text detection, usually a convolutional neural network is used to extract feature maps from a scene image, and then different decoders are used to decode the regions [49, 43, 53]. While in text recognition, a network for sequential prediction is conducted on top of text regions, one by one [44, 14]. It leads to heavy time cost especially for images with a number of text regions. Another problem is that it ignores the correlation in visual cues shared in detection and recognition. A single detection network cannot be supervised by labels from text recognition, and vice versa.

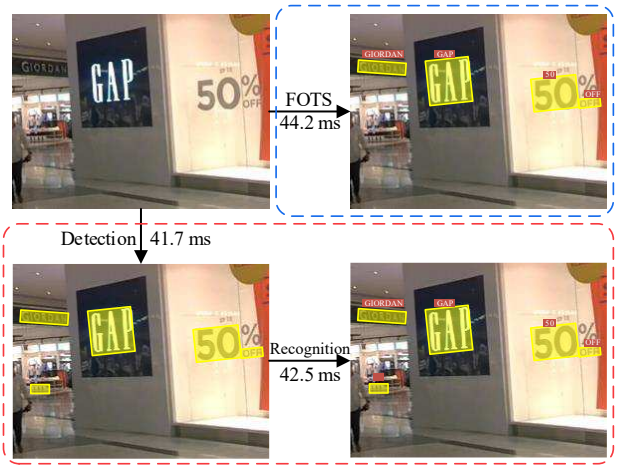


Figure 1: Different to previous two-stage methods, FOTS solves oriented text spotting problem straightforward and efficiently. FOTS can detect and recognize text simultaneously with little computation cost compared to a single text detection network (44.2ms vs. 41.7ms) and almost twice as fast as the two-stage method (44.2ms vs. 84.2ms). This is detailed in Sec. 4.4.

In this paper, we propose to simultaneously consider text detection and recognition. It leads to the fast oriented text spotting system (FOTS) which can be trained end-to-end. In contrast to previous two-stage text spotting, our method learns better features through convolutional neural network, which are shared between text detection and text recognition. Since feature extraction usually takes most of the time, it shrinks the computation to a single detection network, shown in Fig. 1. The key to connect detection and recognition is the *ROIRotate*, which gets proper features from

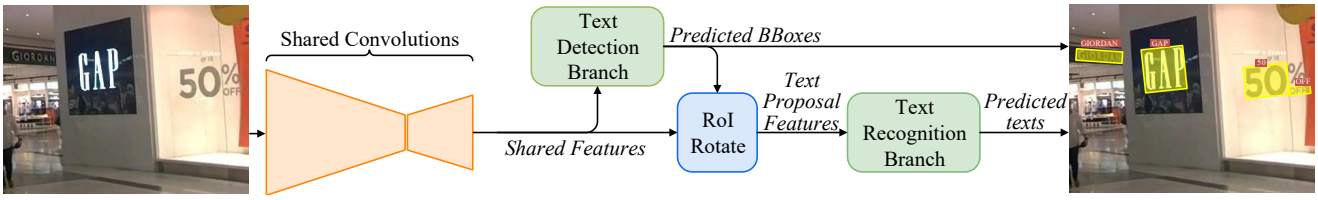


Figure 2: Overall architecture. The network predicts both text regions and text labels in a single forward pass.

feature maps according to the oriented detection bounding boxes.

The architecture is presented in Fig. 2. Feature maps are firstly extracted with shared convolutions. The fully convolutional network based oriented text detection branch is built on top of the feature map to predict the detection bounding boxes. The RoIRotate operator extracts text proposal features corresponding to the detection results from the feature map. The text proposal features are then fed into Recurrent Neural Network (RNN) encoder and Connectionist Temporal Classification (CTC) decoder [9] for text recognition. Since all the modules in the network are differentiable, the whole system can be trained end-to-end. To the best of our knoweldge, this is the first end-to-end trainable framework for oriented text detection and recognition. We find that the network can be easily trained without complicated post-processing and hyper-parameter tuning.

The contributions are summarized as follows.

- We propose an end-to-end trainable framework for fast oriented text spotting. By sharing convolutional features, the network can detect and recognize text simultaneously with little computation overhead, which leads to *real-time* speed.
- We introduce the *RoIRotate*, a new differentiable operator to extract the oriented text regions from convolutional feature maps. This operation unifies text detection and recognition into an end-to-end pipeline.
- FOTS significantly surpasses state-of-the-art methods on a number of text detection and text spotting benchmarks, including ICDAR 2015 [26], ICDAR 2017 MLT [1] and ICDAR 2013 [27].

2. Related Work

Text spotting is an active topic in computer vision and document analysis. In this section, we present a brief introduction to related works including text detection, text recognition and text spotting methods that combine both.

2.1. Text Detection

Most conventional methods of text detection consider text as a composition of characters. These character based methods first localize characters in an image and then group them into words or text lines. Sliding-window-based methods [22, 28, 3, 54] and connected-components based methods [18, 40, 2] are two representative categories in conventional methods.

Recently, many deep learning based methods are proposed to directly detect words in images. Tian *et al*. [49] employ a vertical anchor mechanism to predict the fixed-width sequential proposals and then connect them. Ma *et al*. [39] introduce a novel rotation-based framework for arbitrarily oriented text by proposing Rotation RPN and Rotation RoI pooling. Shi *et al*. [43] first predict text segments and then link them into complete instances using the linkage prediction. With dense predictions and one step post processing, Zhou *et al*. [53] and He *et al*. [15] propose deep direct regression methods for multi-oriented scene text detection.

2.2. Text Recognition

Generally, scene text recognition aims to decode a sequence of label from regularly cropped but variable-length text images. Most previous methods [8, 30] capture individual characters and refine misclassified characters later. Apart from character level approaches, recent text region recognition approaches can be classified into three categories: word classification based, sequence-to-label decode based and sequence-to-sequence model based methods.

Jaderberg *et al*. [19] pose the word recognition problem as a conventional multi-class classification task with a large number of class labels. Su *et al*. [48] frame text recognition as a sequence labelling problem, where RNN is built upon HOG features and adopt CTC as decoder. Shi *et al*. [44] and He *et al*. [14] propose deep recurrent models to encode the max-out CNN features and adopt CTC to decode the encoded sequence. Fujii *et al*. [5] propose an encoder and summarizer network to perform line-level script identification. Lee *et al*. [31] use an attention-based sequence-to-sequence structure to automatically focus on certain extracted CNN features and implicitly learn a character level language model embodied in RNN. To handle irregular input images, Shi *et al*. [45] and Liu *et al*. [37] introduce spatial attention mechanism to transform a distorted text region into a canonical pose suitable for recognition.

2.3. Text Spotting

Most previous text spotting methods first generate text proposals using a text detection model and then recognize

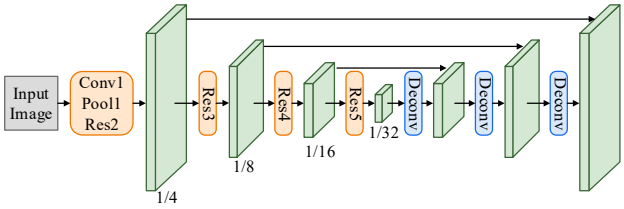


Figure 3: Architecture of shared convolutions. Conv1-Res5 are operations from ResNet-50, and Deconv consists of one convolution to reduce feature channels and one bilinear upsampling operation.

them with a separate text recognition model. Jaderberg *et al*. [20] first generate holistic text proposals with a high recall using an ensemble model, and then use a word classifier for word recognition. Gupta *et al*. [10] train a Fully-Convolutional Regression Network for text detection and adopt the word classifier in [19] for text recognition. Liao *et al*. [34] use an SSD [36] based method for text detection and CRNN [44] for text recognition.

Recently Li *et al*. [33] propose an end-to-end text spotting method, which uses a text proposal network inspired by RPN [41] for text detection and LSTM with attention mechanism [38, 45, 3] for text recognition. Our method has two mainly advantages compared to them: (1) We introduce RoIRotate and use totally different text detection algorithm to solve more complicated and difficult situations, while their method is only suitable for horizontal text. (2) Our method is much better than theirs in terms of speed and performance, and in particular, nearly cost-free text recognition step enables our text spotting system to run at real-time speed, while their method takes approximately 900ms to process an input image of 600×800 pixels.

3. Methodology

FOTS is an end-to-end trainable framework that detects and recognizes all words in a natural scene image simultaneously. It consists of four parts: shared convolutions, the text detection branch, RoIRotate operation and the text recognition branch.

3.1. Overall Architecture

An overview of our framework is illustrated in Fig. 2. The text detection branch and recognition branch share convolutional features, and the architecture of the shared network is shown in Fig. 3. The backbone of the shared network is ResNet-50 [12]. Inspired by Feature Pyramid Network [35], we concatenate low-level feature maps and high-level semantic feature maps. The resolution of feature maps produced by shared convolutions is 1/4 of the input image. The text detection branch outputs dense per-pixel prediction of text using features produced by shared convolutions. With oriented text region proposals produced by detection branch, the proposed RoIRotate converts corresponding shared features into fixed-height representations while keeping the original region aspect ratio. Finally, the text recognition branch recognizes words in region proposals. CNN and LSTM are adopted to encode text sequence information, followed by a CTC decoder. The structure of our text recognition branch is shown in Tab. 1.

Type	Kernel [size, stride]	Out Channels
conv_bn_relu	[3, 1]	64
conv_bn_relu	[3, 1]	64
height-max-pool	[(2, 1), (2, 1)]	64
conv_bn_relu	[3, 1]	128
conv_bn_relu	[3, 1]	128
height-max-pool	[(2, 1), (2, 1)]	128
conv_bn_relu	[3, 1]	256
conv_bn_relu	[3, 1]	256
height-max-pool	[(2, 1), (2, 1)]	256
bi-directional_lstm		256
fully-connected		$\|S\|$

Table 1: The detailed structure of the text recognition branch. All convolutions are followed by batch normalization and ReLU activation. Note that height-max-pool aims to reduce feature dimension along height axis only.

3.2. Text Detection Branch

Inspired by [53, 15], we adopt a fully convolutional network as the text detector. As there are a lot of small text boxes in natural scene images, we upscale the feature maps from 1/32 to 1/4 size of the original input image in shared convolutions. After extracting shared features, one convolution is applied to output dense per-pixel predictions of text to be present. The first channel computes the probability of each pixel being a positive sample. Similar to [53], pixels in shrunk version of the original text regions are considered positive. For each positive sample, the following 4 channels predict its distances to top, bottom, left, right sides of the bounding box that contains this pixel, and the last channel predicts the orientation of the related bounding box. Final detection results are produced by applying thresholding and NMS to these positive samples.

In our experiments, we observe that many patterns similar to text strokes are hard to classify, such as fences, lattices, etc. We adopt online hard example mining (OHEM) [46] to better distinguish these patterns, which also solves the class imbalance problem. This provides a F-measure improvement of about 2% on ICDAR 2015 dataset.

The detection branch loss function is composed of two sterms: text classification term and bounding box regression term. The text classification term can be seen as pixel-wise classification loss for a down-sampled score map. Only shrunk version of the original text region is considered as the positive area, while the area between the bounding box

Figure 4: Illustration of RoIRotate. Here we use the input image to illustrate text locations, but it is actually operated on feature maps in the network. Best view in color.

and the shrunk version is considered as "NOT CARE", and does not contribute to the loss for the classification. Denote the set of selected positive elements by OHEM in the score map as Ω, the loss function for classification can be formulated as:

$$\begin{aligned} L_{\text{cls}} &= \frac{1}{|\Omega|}\sum_{x\in\Omega}\text{H}(p_x, p_x^*) \\ &= \frac{1}{|\Omega|}\sum_{x\in\Omega}(-p_x^*\log p_x - (1-p_x^*)\log(1-p_x)) \end{aligned} \tag{1}$$

where $|\cdot|$ is the number of elements in a set, and $\text{H}(p_x, p_x^*)$ represents the cross entropy loss between p_x, the prediction of the score map, and p_x^*, the binary label that indicates text or non-text.

As for the regression loss, we adopt the IoU loss in [52] and the rotation angle loss in [53], since they are robust to variation in object shape, scale and orientation:

$$L_{\text{reg}} = \frac{1}{|\Omega|}\sum_{x\in\Omega}\text{IoU}(\mathbf{R}_x, \mathbf{R}_x^*) + \lambda_\theta(1-\cos(\theta_x, \theta_x^*)) \tag{2}$$

Here, $\text{IoU}(\mathbf{R}_x, \mathbf{R}_x^*)$ is the IoU loss between the predicted bounding box $\mathbf{R}_x$, and the ground truth $\mathbf{R}_x^*$. The second term is rotation angle loss, where θ_x and θ_x^* represent predicted orientation and the ground truth orientation respectively. We set the hyper-parameter λ_θ to 10 in experiments.

Therefore the full detection loss can be written as:

$$L_{\text{detect}} = L_{\text{cls}} + \lambda_{\text{reg}} L_{\text{reg}} \tag{3}$$

where a hyper-parameter λ_{reg} balances two losses, which is set to 1 in our experiments.

3.3. RoIRotate

RoIRotate applies transformation on oriented feature regions to obtain axis-aligned feature maps, as shown in Fig. 4. In this work, we fix the output height and keep the aspect ratio unchanged to deal with the variation in text length. Compared to RoI pooling [6] and RoIAlign [11], RoIRotate provides a more general operation for extracting features for regions of interest. We also compare to RRoI pooling proposed in RRPN [39]. RRoI pooling transforms the rotated region to a fixed size region through max-pooling, while we use bilinear interpolation to compute the values of the output. This operation avoids misalignments between the RoI and the extracted features, and additionally it makes the lengths of the output features variable, which is more suitable for text recognition.

This process can be divided into two steps. First, affine transformation parameters are computed via predicted or ground truth coordinates of text proposals. Then, affine transformations are applied to shared feature maps for each region respectively, and canonical horizontal feature maps of text regions are obtained. The first step can be formulated as:

$$t_x = l * \cos\theta - t * \sin\theta - x \tag{4}$$

$$t_y = t * \cos\theta + l * \sin\theta - y \tag{5}$$

$$s = \frac{h_t}{t+b} \tag{6}$$

$$w_t = s * (l + r) \tag{7}$$

$$\begin{aligned} \mathbf{M} &= \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \\ &= s\begin{bmatrix} \cos\theta & -\sin\theta & t_x\cos\theta - t_y\sin\theta \\ \sin\theta & \cos\theta & t_x\sin\theta + t_y\cos\theta \\ 0 & 0 & \frac{1}{s} \end{bmatrix} \end{aligned} \tag{8}$$

where $\mathbf{M}$ is the affine transformation matrix. h_t, w_t represent height (equals 8 in our setting) and width of feature maps after affine transformation. (x, y) represents the coordinates of a point in shared feature maps and (t, b, l, r) stands for distance to top, bottom, left, right sides of the text proposal respectively, and θ for the orientation. (t, b, l, r) and θ can be given by ground truth or the detection branch.

With the transformation parameters, it is easy to produce the final RoI feature using the affine transformation:

$$\begin{pmatrix} x_i^s \\ y_i^s \\ 1 \end{pmatrix} = \mathbf{M}^{-1}\begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \tag{9}$$

and for $\forall i \in [1 \dots h_t], \forall j \in [1 \dots w_t], \forall c \in [1 \dots C]$,

$$V_{ij}^c = \sum_n^{h_s}\sum_m^{w_s} U_{nm}^c k(x_{ij}^s - m; \Phi_x) k(y_{ij}^s - n; \Phi_y) \tag{10}$$

where V_{ij}^c is the output value at location (i, j) in channel c and U_{nm}^c is the input value at location (n, m) in channel c. h_s, w_s represent the height and width of the input, and Φ_x, Φ_y are the parameters of a generic sampling kernel $k()$,

which defines the interpolation method, specifically bilinear interpolation in this work. As the width of text proposals may vary, in practice, we pad the feature maps to the longest width and ignore the padding parts in recognition loss function.

Spatial transformer network [21] uses affine transformation in a similar way, but gets transformation parameters via a different method and is mainly used in the image domain, i.e. transforming images themselves. RoIRotate takes feature maps produced by shared convolutions as input, and generates the feature maps of all text proposals, with fixed height and unchanged aspect ratio.

Different from object classification, text recognition is very sensitive to detection noise. A small error in predicted text region could cut off several characters, which is harmful to network training, so we use ground truth text regions instead of predicted text regions during training. When testing, thresholding and NMS are applied to filter predicted text regions. After RoIRotate, transformed feature maps are fed to the text recognition branch.

3.4. Text Recognition Branch

The text recognition branch aims to predict text labels using the region features extracted by shared convolutions and transformed by RoIRotate. Considering the length of the label sequence in text regions, input features to LSTM are reduced only twice (to 1/4 as described in Sec. 3.2) along width axis through shared convolutions from the original image. Otherwise discriminable features in compact text regions, especially those of narrow shaped characters, will be eliminated. Our text recognition branch consists of VGG-like [47] sequential convolutions, poolings with reduction along height axis only, one bi-directional LSTM [42, 16], one fully-connection and the final CTC decoder [9].

First, spatial features are fed into several sequential convolutions and poolings along height axis with dimension reduction to extract higher-level features. For simplicity, all reported results here are based on VGG-like sequential layers as shown in Tab. 1.

Next, the extracted higher-level feature maps $\mathbf{L} \in \mathbb{R}^{C\times H\times W}$ are permuted to time major form as a sequence $\boldsymbol{l}_1, ..., \boldsymbol{l}_W \in \mathbb{R}^{C\times H}$ and fed into RNN for encoding. Here we use a bi-directional LSTM, with $D = 256$ output channels per direction, to capture range dependencies of the input sequential features. Then, hidden states $\boldsymbol{h}_1, ..., \boldsymbol{h}_W \in \mathbb{R}^D$ calculated at each time step in both directions are summed up and fed into a fully-connection, which gives each state its distribution $\boldsymbol{x}_t \in \mathbb{R}^{|S|}$ over the character classes S. To avoid overfitting on small training datasets like ICDAR 2015, we add dropout before fully-connection. Finally, CTC is used to transform frame-wise classification scores to label sequence. Given probability distribution $\boldsymbol{x}_t$ over S of each $\boldsymbol{h}_t$, and ground truth label sequence $\boldsymbol{y}^* = \{y_1, ..., y_T\}, T \leqslant W$, the conditional probability of the label $\boldsymbol{y}^*$ is the sum of probabilities of all paths π agreeing with [9]:

$$p(\boldsymbol{y}^*|\boldsymbol{x}) = \sum_{\pi\in\mathcal{B}^{-1}(\boldsymbol{y}^*)} p(\pi|\boldsymbol{x}) \tag{11}$$

where $\mathcal{B}$ defines a many-to-one map from the set of possible labellings with blanks and repeated labels to $\boldsymbol{y}^*$. The training process attempts to maximize the log likelihood of summation of Eq. (11) over the whole training set. Following [9], the recognition loss can be formulated as:

$$L_{\text{recog}} = -\frac{1}{N}\sum_{n=1}^{N} \log p(\boldsymbol{y}_n^*|\boldsymbol{x}) \tag{12}$$

where N is the number of text regions in an input image, and $\boldsymbol{y}_n^*$ is the recognition label.

Combined with detection loss L_{detect} in Eq. (3), the full multi-task loss function is:

$$L = L_{\text{detect}} + \lambda_{\text{recog}} L_{\text{recog}} \tag{13}$$

where a hyper-parameter λ_{recog} controls the trade-off between two losses. λ_{recog} is set to 1 in our experiments.

3.5. Implementation Details

We use model trained on ImageNet dataset [29] as our pre-trained model. The training process includes two steps: first we use Synth800k dataset [10] to train the network for 10 epochs, and then real data is adopted to fine-tune the model until convergence. Different training datasets are adopted for different tasks, which will be discussed in Sec. 4. Some blurred text regions in ICDAR 2015 and ICDAR 2017 MLT datasets are labeled as "DO NOT CARE", and we ignore them in training.

Data augmentation is important for robustness of deep neural networks, especially when the number of real data is limited, as in our case. First, longer sides of images are resized from 640 pixels to 2560 pixels. Next, images are rotated in range $[-10°, 10°]$ randomly. Then, the heights of images are rescaled with ratio from 0.8 to 1.2 while their widths keep unchanged. Finally, 640×640 random samples are cropped from the transformed images.

As described in Sec. 3.2, we adopt OHEM for better performance. For each image, 512 hard negative samples, 512 random negative samples and all positive samples are selected for classification. As a result, positive-to-negative ratio is increased from 1:60 to 1:3. And for bounding box regression, we select 128 hard positive samples and 128 random positive samples from each image for training.

At test time, after getting predicted text regions from the text detection branch, the proposed RoIRotate applies thresholding and NMS to these text regions and feeds selected text features to the text recognition branch to get final

Method	Detection			Method	End-to-End			Word Spotting		
	P	R	F		S	W	G	S	W	G
SegLink [43]	74.74	76.50	75.61	Baseline OpenCV3.0+Tesseract [26]	13.84	12.01	8.01	14.65	12.63	8.43
SSTD [13]	80.23	73.86	76.91	Deep2Text-MO [51, 50, 20]	16.77	16.77	16.77	17.58	17.58	17.58
WordSup [17]	79.33	77.03	78.16	Beam search CUNI+S [26]	22.14	19.80	17.46	23.37	21.07	18.38
RRPN [39]	83.52	77.13	80.20	NJU Text (Version3) [26]	32.63	-	-	34.10	-	-
EAST [53]	83.27	78.33	80.72	StradVision_v1 [26]	33.21	-	-	34.65	-	-
NLPR-CASIA [15]	82	80	81	Stradvision-2 [26]	43.70	-	-	45.87	-	-
R^2CNN [25]	85.62	79.68	82.54	TextProposals+DictNet [7, 19]	53.30	49.61	47.18	56.00	52.26	49.73
CCFLAB_FTSN [4]	88.65	80.07	84.14	HUST_MCLAB [43, 44]	67.86	-	-	70.57	-	-
Our Detection	88.84	82.04	85.31	Our Two-Stage	77.11	74.54	58.36	80.38	77.66	58.19
FOTS	91.0	85.17	87.99	FOTS	81.09	75.90	60.80	84.68	79.32	63.29
FOTS RT	85.95	79.83	82.78	FOTS RT	73.45	66.31	51.40	76.74	69.23	53.50
FOTS MS	**91.85**	**87.92**	**89.84**	FOTS MS	**83.55**	**79.11**	**65.33**	**87.01**	**82.39**	**67.97**

Table 2: Comparison with other results on ICDAR 2015 with percentage scores. "FOTS MS" represents multi-scale testing and "FOTS RT" represents our real-time version, which will be discussed in Sec. 4.4. "End-to-End" and "Word Spotting" are two types of evaluation protocols for text spotting. "P", "R", "F" represent "Precision", "Recall", "F-measure" respectively and "S", "W", "G" represent F-measure using "Strong", "Weak", "Generic" lexicon respectively.

Method	Precision	Recall	F-measure
linkage-ER-Flow [1]	44.48	25.59	32.49
TH-DL [1]	67.75	34.78	45.97
TDN_SJTU2017 [1]	64.27	47.13	54.38
SARI_FDU_RRPN_v1 [39]	71.17	55.50	62.37
SCUT_DLVClab1 [1]	80.28	54.54	64.96
Our Detection	79.48	57.45	66.69
FOTS	80.95	57.51	67.25
FOTS MS	**81.86**	**62.30**	**70.75**

Table 3: Comparison with other results on ICDAR 2017 MLT scene text detection task.

recognition result. For multi-scale testing, results from all scales are combined and fed to NMS again to get the final results.

4. Experiments

We evaluate the proposed method on three recent challenging public benchmarks: ICDAR 2015 [26], ICDAR 2017 MLT [1] and ICDAR 2013 [27], and surpasses state-of-the-art methods in both text localization and text spotting tasks. All the training data we use is publicly available.

4.1. Benchmark Datasets

ICDAR 2015 is the Challenge 4 of ICDAR 2015 Robust Reading Competition, which is commonly used for oriented scene text detection and spotting. This dataset includes 1000 training images and 500 testing images. These images are captured by Google glasses without taking care of position, so text in the scene can be in arbitrary orientations. For text spotting task, it provides 3 specific lists of words as lexicons for reference in the test phase, named as "Strong", "Weak" and "Generic". "Strong" lexicon provides 100 words per-image including all words that appear in the image. "Weak" lexicon includes all words that appear in the entire test set. And "Generic" lexicon is a 90k word vocabulary. In training, we first train our model using 9000 images from ICDAR 2017 MLT training and validation datasets, then we use 1000 ICDAR 2015 training images and 229 ICDAR 2013 training images to fine-tune our model.

ICDAR 2017 MLT is a large scale multi-lingual text dataset, which includes 7200 training images, 1800 validation images and 9000 testing images. The dataset is composed of complete scene images which come from 9 languages, and text regions in this dataset can be in arbitrary orientations, so it is more diverse and challenging. This dataset does not have text spotting task so we only report our text detection result. We use both training set and validation set to train our model.

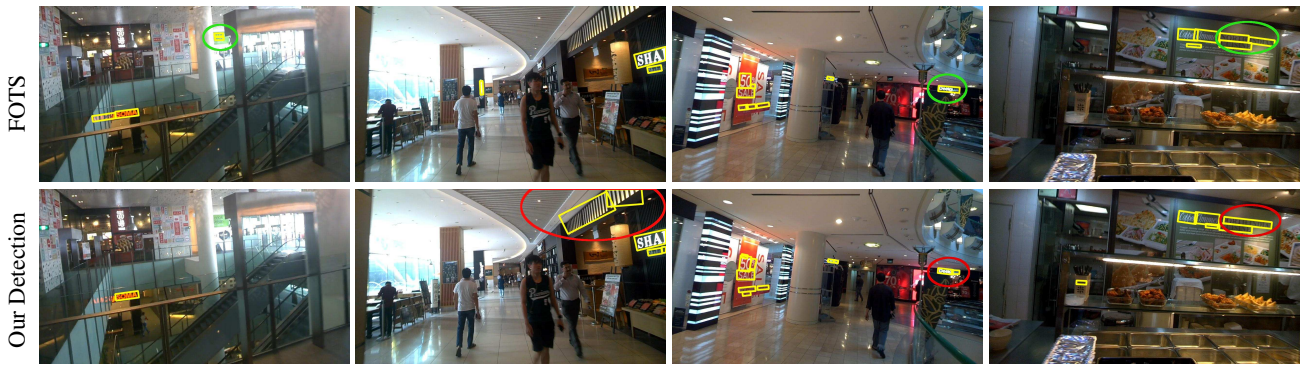
ICDAR 2013 consists of 229 training images and 233 testing images, and similar to ICDAR 2015, it also provides "Strong", "Weak" and "Generic" lexicons for text spotting task. Different to above datasets, it contains only horizontal text. Though our method is designed for oriented text, results in this dataset indicate the proposed method is also suitable for horizontal text. Due to there are too few training images, we first use 9000 images from ICDAR 2017 MLT training and validation datasets to train a pre-trained model and then use 229 ICDAR 2013 training images to fine-tune.

4.2. Comparison with Two-Stage Method

Different from previous works which divide text detection and recognition into two unrelated tasks, our method train these two tasks jointly, and both text detection and recognition can benefit from each other. To verify this, we build a two-stage system, in which text detection and recognition models are trained separately. The detection network is built by removing recognition branch in our proposed net-

Method	Detection		Method	End-to-End			Word Spotting		
	IC13	DetEval		S	W	G	S	W	G
TextBoxes [34]	85	86	NJU Text (Version3) [27]	74.42	-	-	77.89	-	-
CTPN [49]	82.15	87.69	StradVision-1 [27]	81.28	78.51	67.15	85.82	82.84	70.19
R^2CNN [25]	79.68	87.73	Deep2Text II+ [51, 20]	81.81	79.47	76.99	84.84	83.43	78.90
NLPR-CASIA [15]	86	-	VGGMaxBBNet(055) [20, 19]	86.35	-	-	90.49	-	76
SSTD [13]	87	88	FCRNall+multi-filt [10]	-	-	-	-	-	84.7
WordSup [17]	-	90.34	Adelaide_ConvLSTMs [32]	87.19	86.39	80.12	91.39	90.16	82.91
RRPN [39]	-	91	TextBoxes [34]	91.57	89.65	83.89	93.90	91.95	85.92
Jiang *et al.* [24]	89.54	91.85	Li *et al.* [33]	91.08	89.81	84.59	94.16	92.42	**88.20**
Our Detection	86.96	87.32	Our Two-Stage	87.84	86.96	80.79	91.70	90.68	82.97
FOTS	88.23	88.30	FOTS	88.81	87.11	80.81	92.73	90.72	83.51
FOTS MS	**92.50**	**92.82**	FOTS MS	**91.99**	**90.11**	**84.77**	**95.94**	**93.90**	87.76

Table 4: Comparison with other results on ICDAR 2013. "IC13" and "DetEval" represent F-measure under ICDAR 2013 evaluation and DetEval evaluation respectively.



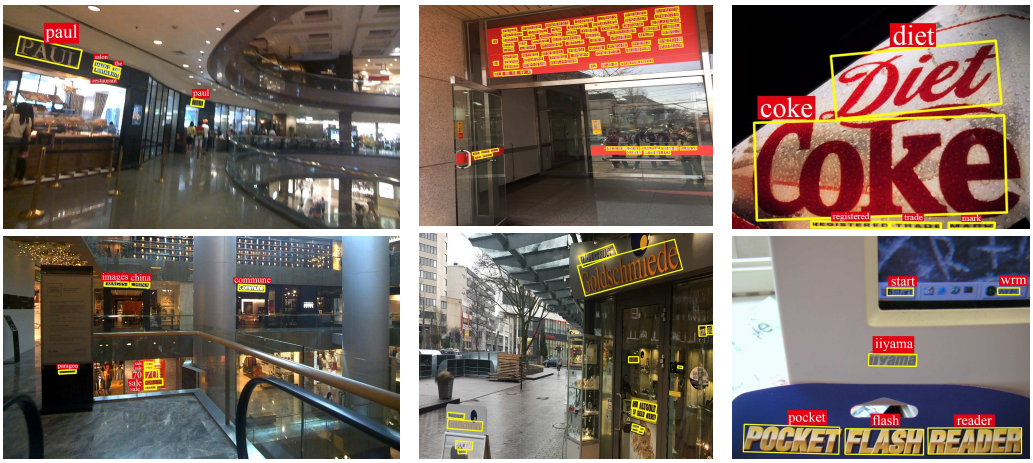
(a) Miss (b) False (c) Split (d) Merge

Figure 5: FOTS reduces Miss, False, Split and Merge errors in detection. Bounding boxes in green ellipses represent correct text regions detected by FOTS, and those in red ellipses represent wrong text regions detected by "Our Detection" method. Best view in color.

work, and similarly, detection branch is removed from origin network to get the recognition network. For recognition network, text line regions cropped from source images are used as training data, similar to previous text recognition methods [44, 14, 37].

As shown in Tab. 2,3,4, our proposed FOTS significantly outperforms the two-stage method "Our Detection" in text localization task and "Our Two-Stage" in text spotting task. Results show that our joint training strategy pushes model parameters to a better converged state.

FOTS performs better in detection because text recognition supervision helps the network to learn detailed character level features. To analyze in detail, we summarize four common issues for text detection, **Miss**: missing some text regions, **False**: regarding some non-text regions as text regions wrongly, **Split**: wrongly spliting a whole text region to several individual parts, **Merge**: wrongly merging several independent text regions together. As shown in Fig. 5, FOTS greatly reduces all of these four types of errors compared to "Our Detection" method. Specifically, "Our Detection" method focuses on the whole text region feature rather than character level feature, so this method does not work well when there is a large variance inside a text region or a text region has similar patterns with its background, etc. As the text recognition supervision forces the model to consider fine details of characters, FOTS learns the semantic information among different characters in one word that have different patterns. It also enhances the difference among characters and background that have similar patterns. As shown in Fig. 5, for the **Miss** case, "Our Detection" method misses the text regions because their color is similar to their background. For the **False** case, "Our Detection" method wrongly recognizes a background region as text because it has "text-like" patterns (*e.g.*, repetitive structured stripes with high contrast), while FOTS avoids this mistake after training with recognition loss which considers details of characters in the proposed region. For the **Split** case, "Our Detection" method splits a text region to two because the left and right sides of this text region have different colors, while FOTS predicts this region as a whole because patterns of characters in this text region are continuous and similar. For the **Merge** case, "Our Detection" method wrongly merges two neighboring text bounding boxes together because they are too close and have similar patterns, while



(a) ICDAR 2015 (b) ICDAR 2017 MLT (c) ICDAR 2013

Figure 6: Results of the proposed method. Note: we only show text detection results of ICDAR 2017 MLT due to the absence of text spotting task.

Dataset	Method	Speed		Params
		Detection	End-to-End	
IC15	Our Two-Stage	7.8 fps	3.7 fps	63.90 M
	FOTS	7.8 fps	7.5 fps	34.98 M
	FOTS RT	24.0 fps	22.6 fps	28.79 M
IC13	Our Two-Stage	23.9 fps	11.2 fps	63.90 M
	FOTS	23.9 fps	22.0 fps	34.98 M

Table 5: Speed and model size compared on different methods. "Our Two-Stage" consists of a detection model with 28.67M parameters and a recognition model with 35.23M parameters.

FOTS utilizes the character level information given by text recognition and captures the space between two words.

4.3. Comparisons with State-of-the-Art Results

In this section, we compare FOTS to state-of-the-art methods. As shown in Tab. 2, 3, 4, our method outperforms all others by a large margin in all datasets. Since ICDAR 2017 MLT does not have text spotting task, we only report our text detection result. All text regions in ICDAR 2013 are labeled by horizontal bounding box while many of them are slightly tilted. As our model is pre-trained using ICDAR 2017 MLT data, it also can predict orientations of text regions. Our final text spotting results keep predicted orientations for better performance, and due to the limitation of the evaluation protocol, our detection results are the minimum horizontal circumscribed rectangles of network predictions. It is worth mentioning that in ICDAR 2015 text spotting task, our method outperforms previous best method [43, 44] by more than 15% in terms of F-measure.

For single-scale testing, FOTS resizes longer side of input images to 2240, 1280, 920 respectively for ICDAR 2015, ICDAR 2017 MLT and ICDAR 2013 to achieve the best results, and we apply 3-5 scales for multi-scale testing.

4.4. Speed and Model Size

As shown in Tab. 5, benefiting from our convolution sharing strategy, FOTS can detect and recognize text jointly with little computation and storage increment compared to a single text detection network (7.5 fps vs. 7.8 fps, 22.0 fps vs. 23.9 fps), and it is almost twice as fast as "Our Two-Stage" method (7.5 fps vs. 3.7 fps, 22.0 fps vs. 11.2 fps). As a consequence, our method achieves state-of-the-art performance while keeping real-time speed.

All of these methods are tested on ICDAR 2015 and ICDAR 2013 test sets. These datasets have 68 text recognition labels, and we evaluate all test images and calculate the average speed. For ICDAR 2015, FOTS uses 2240×1260 size images as inputs, "Our Two-Stage" method uses 2240×1260 images for detection and 32 pixels height cropped text region patches for recognition. As for ICDAR 2013, we resize longer size of input images to 920 and also use 32 pixels height image patches for recognition. To achieve real-time speed, "FOTS RT" replaces ResNet-50 with ResNet-34 and uses 1280×720 images as inputs. All results in Tab. 5 are tested on a modified version Caffe [23] using a TITAN-Xp GPU.

5. Conclusion

In this work, we presented FOTS, an end-to-end trainable framework for oriented scene text spotting. A novel RoIRotate operation is proposed to unify detection and recognition into an end-to-end pipeline. By sharing convolutional features, the text recognition step is nearly cost-free, which enables our system to run at real-time speed. Experiments on standard benchmarks show that our method significantly outperforms previous methods in terms of efficiency and performance.

References

[1] Icdar 2017 robust reading competitions. http://rrc.cvc.uab.es/. Online; accessed 2017-11-1. 2, 6

[2] M. Busta, L. Neumann, and J. Matas. Fastext: Efficient unconstrained scene text detector. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1206–1214, 2015. 2

[3] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 366–373, 2004. 2, 3

[4] Y. Dai, Z. Huang, Y. Gao, and K. Chen. Fused text segmentation networks for multi-oriented scene text detection. *arXiv preprint arXiv:1709.03272*, 2017. 6

[5] Y. Fujii, K. Driesen, J. Baccash, A. Hurst, and A. C. Popat. Sequence-to-label script identification for multilingual ocr. *arXiv preprint arXiv:1708.04671*, 2017. 2

[6] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 4

[7] L. Gómez and D. Karatzas. Textproposals: a text-specific selective search algorithm for word spotting in the wild. *Pattern Recognition*, 70:60–74, 2017. 6

[8] I. J. Goodfellow, Y. Bulatov, J. Ibarz, S. Arnoud, and V. Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv preprint arXiv:1312.6082*, 2013. 2

[9] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376. ACM, 2006. 2, 5

[10] A. Gupta, A. Vedaldi, and A. Zisserman. Synthetic data for text localisation in natural images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2315–2324, 2016. 3, 5, 7

[11] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. *arXiv preprint arXiv:1703.06870*, 2017. 4

[12] 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*, pages 770–778, 2016. 3

[13] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, and X. Li. Single shot text detector with regional attention. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017. 6, 7

[14] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang. Reading scene text in deep convolutional sequences. In *AAAI*, pages 3501–3508, 2016. 1, 2, 7

[15] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu. Deep direct regression for multi-oriented scene text detection. In *The IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 2, 3, 6, 7

[16] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 5

[17] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding. Wordsup: Exploiting word annotations for character based text detection. *arXiv preprint arXiv:1708.06720*, 2017. 6, 7

[18] W. Huang, Z. Lin, J. Yang, and J. Wang. Text localization in natural images using stroke feature transform and text covariance descriptors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1241–1248, 2013. 2

[19] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. *arXiv preprint arXiv:1406.2227*, 2014. 2, 3, 6, 7

[20] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. *International Journal of Computer Vision*, 116(1):1–20, 2016. 1, 3, 6, 7

[21] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 5

[22] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting. In *European conference on computer vision*, pages 512–528. Springer, 2014. 2

[23] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 8

[24] F. Jiang, Z. Hao, and X. Liu. Deep scene text detection with connected component proposals. *arXiv preprint arXiv:1708.05133*, 2017. 7

[25] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo. R2cnn: Rotational region cnn for orientation robust scene text detection. *arXiv preprint arXiv:1706.09579*, 2017. 6, 7

[26] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, et al. Icdar 2015 competition on robust reading. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 1156–1160. IEEE, 2015. 2, 6

[27] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. de las Heras. Icdar 2013 robust reading competition. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 1484–1493. IEEE, 2013. 2, 6, 7

[28] 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 Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1631–1639, Dec 2003. 2

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 5

[30] C.-Y. Lee, A. Bhardwaj, W. Di, V. Jagadeesh, and R. Piramuthu. Region-based discriminative feature pooling for scene text recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4050–4057, 2014. 2

[31] C.-Y. Lee and S. Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2231–2239, 2016. 2

[32] H. Li and C. Shen. Reading car license plates using deep convolutional neural networks and lstms. *arXiv preprint arXiv:1601.05610*, 2016. 7

[33] H. Li, P. Wang, and C. Shen. Towards end-to-end text spotting with convolutional recurrent neural networks. *arXiv preprint arXiv:1707.03985*, 2017. 3, 7

[34] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu. Textboxes: A fast text detector with a single deep neural network. In *AAAI*, pages 4161–4167, 2017. 1, 3, 7

[35] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. *arXiv preprint arXiv:1612.03144*, 2016. 3

[36] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 3

[37] W. Liu, C. Chen, K.-Y. K. Wong, Z. Su, and J. Han. Star-net: A spatial attention residue network for scene text recognition. In *BMVC*, 2016. 2, 7

[38] M.-T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015. 3

[39] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue. Arbitrary-oriented scene text detection via rotation proposals. *arXiv preprint arXiv:1703.01086*, 2017. 2, 4, 6, 7

[40] L. Neumann and J. Matas. Scene text localization and recognition with oriented stroke detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 97–104, 2013. 2

[41] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 3

[42] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. 5

[43] B. Shi, X. Bai, and S. Belongie. Detecting oriented text in natural images by linking segments. *arXiv preprint arXiv:1703.06520*, 2017. 1, 2, 6, 8

[44] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2016. 1, 2, 3, 6, 7, 8

[45] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai. Robust scene text recognition with automatic rectification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4168–4176, 2016. 2, 3

[46] A. Shrivastava, A. Gupta, and R. Girshick. Training region-based object detectors with online hard example mining. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 761–769, 2016. 3

[47] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[48] B. Su and S. Lu. Accurate scene text recognition based on recurrent neural network. In *Asian Conference on Computer Vision*, pages 35–48. Springer, 2014. 2

[49] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao. Detecting text in natural image with connectionist text proposal network. In *European Conference on Computer Vision*, pages 56–72. Springer, 2016. 1, 2, 7

[50] X.-C. Yin, W.-Y. Pei, J. Zhang, and H.-W. Hao. Multi-orientation scene text detection with adaptive clustering. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1930–1937, 2015. 6

[51] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao. Robust text detection in natural scene images. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):970–983, 2014. 6, 7

[52] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang. Unitbox: An advanced object detection network. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 516–520. ACM, 2016. 4

[53] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. East: An efficient and accurate scene text detector. *arXiv preprint arXiv:1704.03155*, 2017. 1, 2, 3, 4, 6

[54] S. Zhu and R. Zanibbi. A text detection system for natural scenes with convolutional feature learning and cascaded classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 625–632, 2016. 2