Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework

Michal Bušta, Lukáš Neumann and Jiří Matas
Centre for Machine Perception, Department of Cybernetics
Czech Technical University, Prague, Czech Republic
bustam@fel.cvut.cz, neumalu1@cmp.felk.cvut.cz, matas@cmp.felk.cvut.cz

Abstract

A method for scene text localization and recognition is proposed. The novelties include: training of both text detection and recognition in a single end-to-end pass, the structure of the recognition CNN and the geometry of its input layer that preserves the aspect of the text and adapts its resolution to the data.

The proposed method achieves state-of-the-art accuracy in the end-to-end text recognition on two standard datasets – ICDAR 2013 and ICDAR 2015, whilst being an order of magnitude faster than competing methods - the whole pipeline runs at 10 *frames per second on an NVidia K80 GPU.*

1. Introduction

Scene text localization and recognition, a.k.a. text spotting, text-in-the-wild problem or photo OCR, in an open problem with many practical applications, ranging from tools for helping visually impaired or text translation, to use as a part of a larger integrated system, e.g. in robotics, indoor navigation or autonomous driving.

Like many areas of computer vision, the scene text field has greatly benefited from deep learning techniques and accuracy of methods has significantly improved [12, 6]. Most work however focuses either solely on text localization (detection) [18, 26, 6, 15] or on recognition of manually cropped-out words [7, 24]. The problem of scene text recognition has been so far always approached ad-hoc, by connecting the detection module to an existing independent recognition method [6, 15, 8].

In this paper, we propose a novel end-to-end framework which simultaneously detects and recognizes text in scene images. As the first contribution, we present a model which is trained for both text detection and recognition in a single learning framework, and we show that such joint model outperforms the combination of state-of-the-art localization and state-of-the-art recognition methods [6, 4].

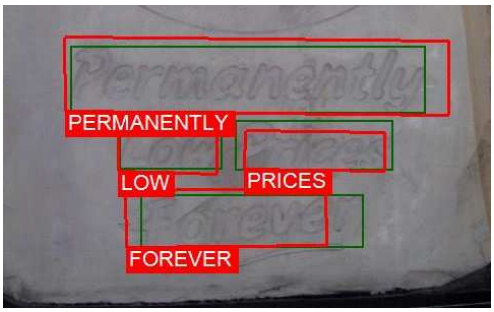

Figure 1. The proposed method detects and recognizes text in scene images at 10fps on an NVidia K80 GPU. Ground truth in green, model output in red. The image is taken from the ICDAR 2013 dataset [13]

As the second contribution, we show how the state-of-the-art object detection methods [22, 23] can be extended for text detection and recognition, taking into account specifics of text such as the exponential number of classes (given an alphabet $\mathcal{A}$, there are up to $\mathcal{A}^L$ possible classes, where L denotes maximum text length) and the sensitivity to hidden parameters such as text aspect and rotation.

The method achieves state-of-the-art results on the standard ICDAR 2013 [13] and ICDAR 2015 [12] datasets and the pipeline runs end-to-end at 10 frames per second on a NVidia K80 GPU, which is more than 10 times faster than the fastest methods.

The rest of the paper is structured as follows. In Section 2, previous work is reviewed. In Section 3, the proposed method is described and in Section 4 evaluated. The paper is concluded in Section 5.

2. Previous Work

2.1. Scene Text Localization

Jaderberg *et al*. [10] train a character-centric CNN [14], which takes a 24×24 image patch and predicts a text/no-text score, a character and a bigram class. The input image

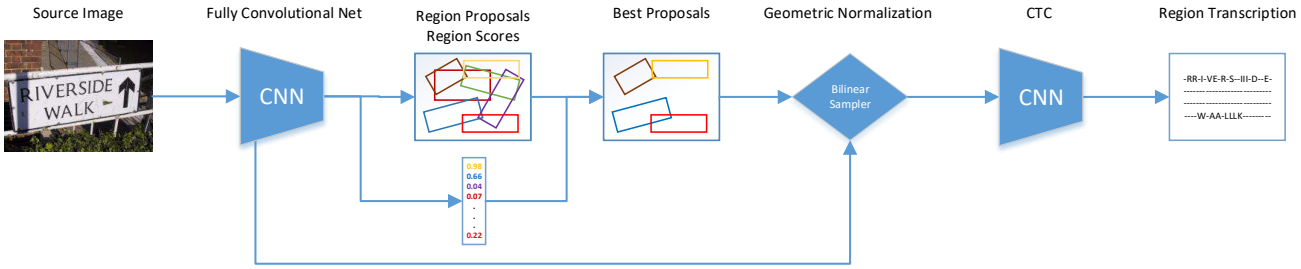

Figure 2. Method overview. Text region proposals are generated by a Region Proposal Network [22]. Each region with a sufficient text confidence is then normalized to a variable-width feature tensor by bilinear sampling. Finally, each region is associated with a sequence of characters or rejected as not text.

is scanned by the trained network in 16 scales and a text saliency map is obtained by taking the text/no-text output of the network. Given the saliency maps, word bounding boxes are then obtained by the run length smoothing algorithm. The method is further improved in [8], where a word-centric approach is introduced. First, horizontal bounding-box proposals are detected by aggregating the output of the standard Edge Boxes [29] and Aggregate Channel Feature [2] detectors. Each proposal is then classified by a Random Forest [1] classifier to reduce the number of false positives and its position and size is further refined by a CNN regressor, to obtain a more suitable cropping of the detected word image.

Gupta *et al*. [6] propose a fully-convolutional regression network, drawing inspiration from the YOLO object detection pipeline [21]. An image is divided into a fixed number of cells (14×14 in the highest resolution), where each cell is associated with 7 values directly predicting the position, rotation and confidence of text. The values are estimated by translation-invariant predictors built on top of the first 9 convolutional layers of the popular VGG-16 architecture [25], trained on synthetic data.

Tian *et al*. [26] adapt the Faster R-CNN architecture [23] by horizontally sliding a 3×3 window on the last convolutional layer of the VGG-16 [25] and applying a Recurrent Neural Network to jointly predict the text/non-text score, the y-axis coordinates and the anchor side-refinement. Similarly, Liao *et al*. [15] adapt the SSD object detector [17] to detect horizontal bounding boxes.

Ma *et al*. [18] adapt the Faster R-CNN architecture and extend it to detect text of different orientations by adding anchor boxes of 6 hand-crafted rotations and 3 aspects. This is in contrast to our work, where the rotation is a continuous parameter and the optimal anchor boxes dimensions are found on the training set.

All the aforementioned methods only localize text, but do not provide text recognition. The end-to-end scene text recognition results, where present, are achieved by simply connecting the particular localization method to one of the cropped-word recognition methods (see Section 2.2).

Last but not least, the methods are significantly slower than the proposed method, the missing recognition stage notwithstanding.

2.2. Scene Text Recognition

Jaderberg *et al*. [8] take a cropped image of a single word, resize it to a fixed size of 32×100 pixels and classify it as one of the words in a dictionary. In their setup, the dictionary contains 90 000 English words and words of the training and testing set. The classifier is trained on a dataset of 9 million synthetic word images uniformly sampled from this dictionary.

Shi *et al*. [24] train a fully-convolutional network with a bidirectional LSTM using the Connectionist Temporal Classification (CTC), which was first introduced by Graves *et al*. [5] for speech recognition to eliminate the need for pre-segmented data. Unlike the proposed method, Shi *et al*. [24] only recognize a single word per image (*i.e*. the output is always just one sequence of characters), they resize the source image to a fixed-sized matrix of 100×32 pixels regardless of how many characters it contains and the method is significantly slower because of the LSTM layer.

2.3. Image Captioning

Johnson *et al*. [11] introduce a Fully Convolutional Localization Network (FCLN) that combines the Faster R-CNN approach of Ren *et al*. [23] based on full VGG-16 [25] with bilinear sampling [9] to generate features for LSTM that produces captions for detected objects. In our method, we use YOLOv2 architecture [22] for its lower complexity, we use the bilinear sampling to produce tensors of variable width to deal with character sequence recognition and we employ a different (and significantly faster) classification stage.

3. Proposed Method

The proposed model localizes text regions in a given scene image and provides text transcription as a sequence of characters for all regions with text (see Figure 2). The

model is jointly optimized for both text localization and recognition in an end-to-end training framework.

3.1. Fully Convolutional Network

We adapt the YOLOv2 architecture [22] for its accuracy and significantly lower complexity than the standard VGG-16 architecture [25, 11], as the full VGG-16 architecture requires 30 billion operations just to process a 224×224 (0.05 Mpx) image [22]. Using YOLOv2 architecture allows us to process images with higher resolution, which is a crucial ability for text recognition - processing at higher resolution is required because a 1Mpx scene image may contain text which is 10 pixels high [12], so scaling down the source image would make the text unreadable.

The proposed method uses the first 18 convolutional and 5 max pool layers from the YOLOv2 architecture, which is based on 3×3 convolutional filters, doubling the number of channels after every pooling step and adding 1×1 filters to compress the representations between the 3×3 filters [22]. We remove the fully-connected layers to make the network fully convolutional, so our model final layer has the dimension of $\frac{W}{32} \times \frac{H}{32} \times 1024$, where W a H denote source image width and height [22].

3.2. Region Proposals

Similarly to Faster R-CNN [23] and YOLOv2 [22], we use a Region Proposal Network (RPN) to generate region proposals, but we add rotation r_θ which is crucial for a successful text recognition. At each position of the last convolutional layer, the model predicts k rotated bounding boxes, where for each bounding box r we predict 6 features - its position r_x, r_y, its dimensions r_w, r_h, its rotation r_θ and its score r_p, which captures the probability that the region contains text.

The bounding box position and dimension is encoded with respect to predefined anchor boxes using the logistic activation function, so the actual bounding box position (x, y) and dimension (w, h) in the source image is given as

$$x = \sigma(r_x) + c_x \quad (1)$$

$$y = \sigma(r_y) + c_y \quad (2)$$

$$w = a_w \exp(r_w) \quad (3)$$

$$h = a_h \exp(r_h) \quad (4)$$

$$\theta = r_\theta \quad (5)$$

where c_x and c_y denote the offset of the cell in the last convolutional layer and a_w and a_h denote the predefined height and width of the anchor box a. The rotation $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$ of the bounding box is predicted directly by r_θ.

We followed the approach of Redmon *et al.* [22] and found suitable anchor box scales and aspects by k-means clustering on the aggregated training set (see Section 3.5). Requiring the anchor boxes to have at least 60% intersection-over-union with the ground truth led to $k = 14$ different anchor boxes dimensions (see Figure 3).

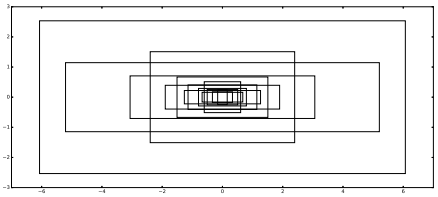

Figure 3. Anchor box widths and heights, or equivalently scales and aspects, were obtained by k-means clustering on the training set. Requiring that each ground truth box had intersection-over-union of at least 60% with one anchor box led to $k = 14$ boxes.

For every image, the RPN produces $\frac{W}{32} \times \frac{H}{32} \times 6k$ boxes, where k is the number of anchor boxes in every location and 6 is the number of predicted parameters (x, y, w, h, θ and the text score).

In the training stage, we use the YOLOv2 approach [22] by taking all positive and negative samples in the source image, where every 20 batches we randomly change the input dimension size into one of $\{352, 416, 480, 544, 608\}$. A positive sample is the region with the highest intersection over union with the ground truth, the other intersecting regions are negatives.

At runtime, we found the best approach is to take all regions with the score r_p above a certain threshold p_{min} and to postpone the non-maxima suppression after the recognition stage, because regions with very similar r_p scores could produce very different transcriptions, and therefore selecting the region with the highest r_p at this stage would not always correspond to the correct transcription (for example, in some cases a region containing letters "TALY" may have slightly higher score r_p than a region containing the full word "ITALY"). We empirically found the value $p_{\text{min}} = 0.1$ to be a reasonable trade-off between accuracy and speed.

3.3. Bilinear Sampling

Each region detected in the previous stage has a different size and rotation and it is therefore necessary to map the features into a tensor of canonical dimensions, which can be used in recognition.

Faster R-CNN [23] uses the RoI pooling approach of Girshick [3], where a $w \times h \times C$ region is mapped onto a fixed-sized $W' \times H' \times C$ grid ($7 \times 7 \times 1024$ in their implementation), where each cell takes the maximum activation of the $\frac{w}{W} \times \frac{h}{H}$ cells in the underlying feature layer.

In our model, we instead use bilinear sampling [9, 11] to map a $w \times h \times C$ region from the source image into a fixed-height $\frac{wH'}{h} \times H' \times C$ tensor ($H' = 32$). This feature representation has a key advantage over the standard RoI approach as it allows the network to normalize rotation and

Type	Channels	Size/Stride	Dim/Act
input	C	-	$\overline{W} \times 32$
conv	32	3×3	leaky ReLU
conv	32	3×3	leaky ReLU
maxpool		$2 \times 2/2$	$\overline{W}/2 \times 16$
conv	64	3×3	leaky ReLU
BatchNorm			
recurrent conv	64	3×3	leaky ReLU
maxpool		$2 \times 2/2$	$\overline{W}/4 \times 8$
conv	128	3×3	leaky ReLU
BatchNorm			
recurrent conv	128	3×3	leaky ReLU
maxpool		$2 \times 2/2 \times 1$	$\overline{W}/4 \times 4$
conv	256	3×3	leaky ReLU
BatchNorm			
recurrent conv	256	3×3	leaky ReLU
maxpool		$2 \times 2/2 \times 1$	$\overline{W}/4 \times 2$
conv	512	3×2	leaky ReLU
conv	512	5×1	leaky ReLU
conv	$\lvert\hat{\mathcal{A}}\rvert$	7×1	$\overline{W}/4 \times 1$
log softmax			

Table 1. Fully-Convolutional Network for Text Recognition

scale, but at the same to persist the aspect and positioning of individual characters, which is crucial for text recognition accuracy (see Section 3.4).

Given the detected region features $\mathbf{U} \in \mathbb{R}^{w \times h \times C}$, they are mapped into a fixed-height tensor $\mathbf{V} \in \mathbb{R}^{\frac{wH'}{h} \times H' \times C}$ as

$$\mathbf{V}^c_{x',y'} = \sum_{x=1}^{w} \sum_{y=1}^{h} \mathbf{U}^c_{x,y} \kappa(x - \mathcal{T}_x(x')) \kappa(y - \mathcal{T}_y(y')) \quad (6)$$

where κ is the bilinear sampling kernel $\kappa(v) = \max(0, 1 - |v|)$ and $\mathcal{T}$ is a point-wise coordinate transformation, which projects co-ordinates x' and y' of the fixed-sized tensor $\mathbf{V}$ to the co-ordinates x and y in the detected region features tensor $\mathbf{U}$.

The transformation allows for shift and scaling in x- and y- axes and rotation and its parameters are taken directly from the region parameters (see Section 3.2).

3.4. Text Recognition

Given the normalized region from the source image, each region is associated with a sequence of characters or rejected as not text in the following process.

The main problem one has to address in this step is the fact, that text regions of different sizes have to be mapped to character sequences of different lengths. Traditionally, the issue is solved by resizing the input to a fixed-sized matrix (typically 100×32 [8, 24]) and the input is then classified by either making every possible character sequence (*i.e.* every word) a separate class of its own [8, 6], thus requiring a list of all possible outputs in the training stage, or by having

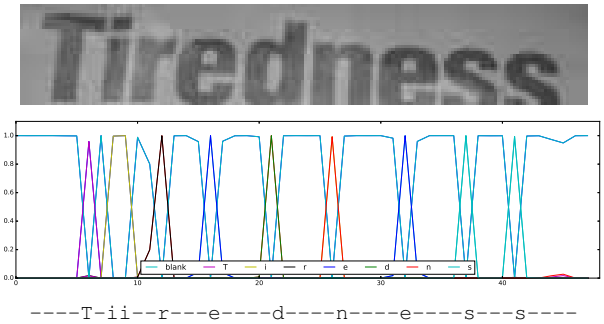

Figure 4. Text recognition using Connectionist Temporal Classification. Input $\overline{W} \times 32$ region (top), CTC output $\frac{\overline{W}}{4} \times |\hat{\mathcal{A}}|$ as the most probable class at given column (middle) and the resulting sequence (bottom)

multiple independent classifiers, where each classifier predicts the character at a predefined position [7].

Our model exploits a novel fully-convolutional network (see Table 1), which takes a variable-width feature tensor $\overline{W} \times H' \times C$ as an input ($\overline{W} = \frac{wH'}{h}$) and outputs a matrix $\frac{\overline{W}}{4} \times |\hat{\mathcal{A}}|$, where $\mathcal{A}$ is the alphabet (*e.g.* all English characters). The matrix height is fixed (it's the number of character classes), but its width grows with the width of the source region and therefore with the length of the expected character sequence.

As a result, a single classifier is used regardless of the position of the character in the word (in contrast to Jaderberg *et al.* [7], where there is an independent classifier for the character "A" as the first character in the word, an independent classifier for the character "A" as the second character in the word, etc). The model also does not require prior knowledge of all words to be detected in the training stage, in contrast to the separate class per character sequence formulation [8].

The model uses Connectionist Temporal Classification (CTC) [5, 24] to transform variable-width feature tensor into a conditional probability distribution over label sequences. The distribution is then used to select the most probable labelling sequence for the text region (see Figure 4).

Let $\mathbf{y} = y_1, y_2, \cdots, y_n$ denote the vector of network outputs of length n from an alphabet $\mathcal{A}$ extended with a blank symbol "–".

The probability of a *path* π is then given as

$$\begin{aligned} p(\pi|\mathbf{y}) &= \prod_{i=1}^{n} y^i_{\pi_i}, \quad \pi \in \hat{\mathcal{A}}^n \quad (7) \\ \hat{\mathcal{A}} &= \mathcal{A} \cup \{-\} \end{aligned}$$

where $y^i_{\pi_i}$ denotes the output probability of the network predicting the label π_i at the position i (*i.e.* the output of the final softmax layer in Table 1).

Let us further define a many-to-one mapping $\mathcal{B} : \hat{\mathcal{A}}^n \mapsto \mathcal{A}^{\leq n}$, where $\hat{\mathcal{A}}^{\leq n}$ is the set of all sequences of shorter or equal in length. The mapping $\mathcal{B}$ removes all blanks and repeated labels, which corresponds to outputting a new label every time the label prediction changes. For example,

$$\mathcal{B}(-\text{ww}-\text{al}-\text{k}) = \mathcal{B}(\text{wwaaa}-\text{l}-\text{k}-) = \text{walk}$$
$$\mathcal{B}(-\text{f}-\text{oo}-\text{o}--\text{d}) = \mathcal{B}(\text{ffoo}-\text{ooo}-\text{d}) = \text{food}$$

The conditional probability of observing the output sequence $\mathbf{w}$ is then given as

$$p(\mathbf{w}|\mathbf{y}) = \sum_{\pi:\, \mathcal{B}(\pi)=\mathbf{w}} p(\pi|\mathbf{y}), \quad \mathbf{w} \in \mathcal{A}^{\leq n} \tag{8}$$

In training, an objective function that maximizes the log likelihood of target labellings $p(\mathbf{w}|\mathbf{y})$ is used [5]. In every training step, the probability $p(\mathbf{w}_{\text{gt}}|\mathbf{y})$ of every text region in the mini-batch is efficiently calculated using a forward-backward algorithm similar to HMMs training [20] and the objective function derivatives are used to update network weights, using the standard back-propagation algorithm ($\mathbf{w}_{\text{gt}}$ denotes the ground truth transcription of the text region).

At test time, the classification output $\mathbf{w}^*$ should be given by the most probable path $p(\mathbf{w}|\mathbf{y})$, which unfortunately is not tractable, and therefore we adapt the approximate approach [5] of taking the most probable labelling

$$\mathbf{w}^* \approx \mathcal{B}\big(\text{argmax}\, p(\pi|\mathbf{y})\big) \tag{9}$$

At the end of this process, each text region in the image has an associated content in the form of a character sequence, or it is rejected as not text when all the labels are blank.

The model typically produces many different boxes for a single text area in the image, we therefore suppress overlapping boxes by a standard non-maxima suppression algorithm based on the text recognition confidence, which is the $p(\mathbf{w}^*|\mathbf{y})$ normalized by the text length.

3.5. Training

We pre-train the detection CNN using the SynthText dataset [6] ($800,000$ synthetic scene images with multiple words per image) for 3 epochs, with weights initialized from ImageNet [22]. The recognition CNN is pre-trained on the Synthetic Word dataset [7] (9 million synthetic cropped word images) for 3 epochs, with weights randomly initialized from the $\mathcal{N}(0, 1)$ distribution.

As the final step, we train both networks simultaneously for 3 epochs on a combined dataset consisting of the SynthText dataset, the Synthetic Word dataset, the ICDAR 2013 Training dataset [13] (229 scene images captured by a professional camera) and the ICDAR 2015 Training dataset [12] (1000 scene images captured by Google Glass). For every image, we randomly crop up to 30% of its width and height. We use standard Stochastic Gradient Descent with momentum 0.9 and learning rate 10^{-3}, divided by 10 after each epoch. One mini-batch takes about 500ms on a NVidia K80 GPU.

Figure 5. End-to-end scene text recognition samples from the ICDAR 2013 dataset. Model output in red, ground truth in green. Note that in some cases (*e.g.* top-right) text is correctly recognized even though the bounding IoU with the ground truth is less than 80%, which would be required by the text localization protocol [13]. Best viewed zoomed in color

	end-to-end			word spotting			speed
	strong	weak	generic	strong	weak	generic	fps
Deep2Text [28]	0.81	0.79	0.77	0.85	0.83	0.79	1.0
TextSpotter [19]	0.77	0.63	0.54	0.85	0.66	0.57	1.0
StradVision [12]	0.81	0.79	0.67	0.84	0.83	0.70	?
Jaderberg *et al*. [8]	0.86	-	-	0.90	0.76	-	*0.3
Gupta *et al*. [6]	-	-	-	-	0.85	-	*0.4
Deep TextSpotter	**0.89**	**0.86**	**0.77**	**0.92**	**0.89**	**0.81**	***10.0**

Table 2. ICDAR 2013 dataset - End-to-end scene text recognition accuracy (f-measure), depending on the lexicon size and whether digits are excluded from the evaluation (denoted as *word spotting*). Methods running on a GPU marked with an asterisk

	end-to-end			word spotting			speed
	strong	weak	generic	strong	weak	generic	fps
TextSpotter [19]	0.35	0.20	0.16	0.37	0.21	0.16	1.0
Stradvision [12]	0.44	-	-	0.46	-	-	?
TextProposals + DictNet [4, 8]	0.53	0.50	0.47	0.56	0.52	0.50	0.2
Deep TextSpotter	**0.54**	**0.51**	**0.47**	**0.58**	**0.53**	**0.51**	***9.0**

Table 3. ICDAR 2015 dataset - End-to-end scene text recognition accuracy (f-measure). Methods running on a GPU marked with an asterisk

4. Experiments

We trained our model once[1] and then evaluated its accuracy on three standard datasets. We evaluate the model in an end-to-end set up, where the objective is to localize and recognize all words in the image in a single step, using the standard evaluation protocol associated with each dataset.

4.1. ICDAR 2013 dataset

In the ICDAR evaluation schema [13, 12], each image in the test set is associated with a list of words (lexicon), which contains the words that the method should localize and recognize, as well as an increasing number of random "distractor" words. There are three sizes of lists provided with each image, depending how heavily contextualized their content is to the specific image:

- *strongly contextualized* - 100 words specific to each image, contains all words in the image and the remaining words are "distractors"
- *weakly contextualized* - all words in the testing set, same list for every image
- *generic* - all words in the testing set plus 90k English words

A word is considered as correctly recognized, when its Intersection-over-Union (IoU) with the ground truth is above 0.5 and the transcription is identical, using case-insensitive comparison [12].

The ICDAR 2013 Dataset [13] is the most-frequently cited dataset for scene text evaluation. It consists of 255 testing images with 716 annotated words, the images were taken by a professional camera so text is typically horizontal and the camera is almost always aimed at it. The dataset is sometimes referred to as the *Focused Scene Text* dataset.

The proposed model achieves state-of-the-art text recognition accuracy (see Table 2) for all 3 lexicon sizes. In the *end-to-end* set up, where all lexicon words plus all digits in an image should be recognized, the maximal f-measure it achieves is 0.89/0.86/0.77 for strongly, weakly and generally contextualized lexicons respectively. Each image is first resized to 544×544 pixels, the average processing time is 100ms per image on a NVidia K80 GPU for the whole pipeline.

While training on the same training data, our model outperforms the combination of the state-of-the-art localization method of Gupta *et al*. [6] with the state-of-the-art recognition method of Jaderberg *et al*. [8] by at least 3 per cent points on every measure, thus demonstrating the advantage of the joint training for the end-to-end task of our model. It is also more than 20 times faster than the method of Gupta *et al*. [6].

Let us further note that our model would not be considered as a state-of-the-art text localization method according to the text localization evaluation protocol, because the standard DetEval tool used for evaluation is based on a series of thresholds which require at least a 80% intersection-over-union with bounding boxes created by human annotators. Our method in contrast does not always achieve the required 80% overlap, but it is still mostly able to recognize the text correctly even when the overlap is lower (see Figure 5).

We argue that evaluating methods purely on text localization accuracy without subsequent recognition is not very informative, because the text localization "accuracy" only aims to fit the way human annotators create bounding boxes around text, but it does not give any estimates on how well a text recognition phase would read text post a successful

[1]Full source code and the trained model are publicly available at `https://github.com/MichalBusta/DeepTextSpotter`

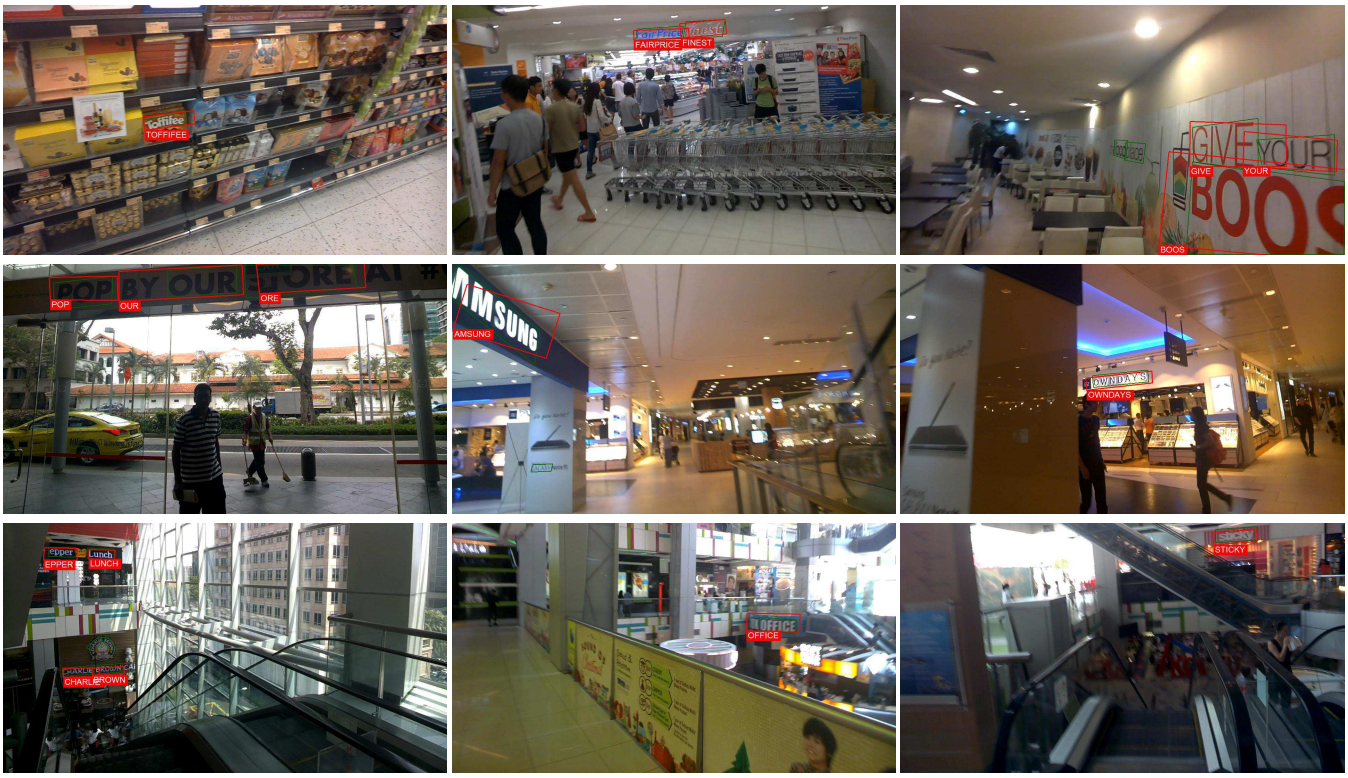

Figure 6. End-to-end scene text recognition samples from the ICDAR 2015 dataset. Model output in red, ground truth in green. Best viewed zoomed in color

Figure 7. *All the images* of the ICDAR 2013 Testing set where the proposed method fails to correctly recognize any text (*i.e.* images with 0% recall)

localization, which should be the prime objective of the text localization metrics.

The main limitation of the proposed model are single characters or short snippets of digits and characters (see Figure 7), which may be partially caused by the fact that such examples are not very frequent in the training set.

4.2. ICDAR 2015 dataset

The *ICDAR 2015 dataset* was introduced in the ICDAR 2015 Robust Reading Competition [12] and it uses the same evaluation protocol as the ICDAR 2013 dataset in the previous section. The dataset consists of 500 test images, which were collected by people wearing Google Glass devices and walking in Singapore. Subsequently, all images with text were selected and annotated. The images in the dataset were taken "not having text in mind", therefore text is much smaller and the images contain a high variability of text fonts and sizes. They also include many realistic effects - *e.g.* occlusion, perspective distortion, blur or noise, so as a result the dataset is significantly more challenging than the ICDAR 2013 dataset (Section 4.1), which contains typically large horizontal text.

The proposed model achieves state-of-the-art end-to-end text recognition accuracy (see Table 3 and Figure 6) for all 3 lexicon sizes. In our experiments, the average processing time was 110ms per image on a NVidia K80 GPU (the image is first resized to 608×608 pixels), which makes the proposed model 45 times faster than currently the best published method of Gomez *et al.* [4]

The main failure mode of the proposed method is blurry

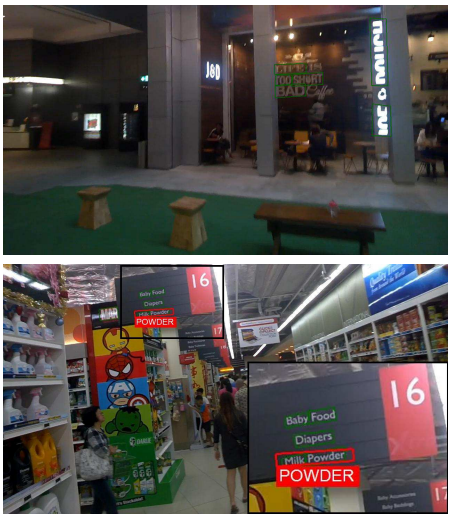

Figure 8. Main failure modes on the ICDAR 2015 dataset. Blurred and noisy text (top), vertical text (top) and small text (bottom). Best viewed zoomed in color

	recall	precision	f-measure
Method A [27]	28.33	68.42	40.07
Method B [27]	9.97	54.46	16.85
Method C [27]	1.66	4.15	2.37
Deep TextSpotter	16.75	31.43	21.85

Table 4. COCO-Text dataset - End to End text recognition

or noisy text (see Figure 8), which are effects not present in the training set (Section 3.5). The method also often fails to detect small text (less than 15 pixels high), which again is due to the lack of such samples in the training stage.

4.3. COCO-Text dataset

The COCO-Text dataset [27] was created by annotating the standard MS COCO dataset [16], which captures images of complex everyday scenes. As a result, the dataset contains 63,686 images with 173,589 labeled text regions, so it is two orders of magnitude larger than any other scene text dataset. Unlike the ICDAR datasets, there is no lexicon used in the evaluation, so methods have to recognize text without any prior knowledge.

The proposed model demonstrates competitive results in the text recognition accuracy (see Table 4 and Figure 9), being only surpassed by Method A[2].

5. Conclusion

A novel framework for scene text localization and recognition was proposed. The model is trained for both text detection and recognition in a single training framework.

The proposed model achieves state-of-the-art accuracy in the end-to-end text recognition on two standard datasets (ICDAR 2013 and ICDAR 2015), whilst being an order of magnitude faster than the previous methods - the whole pipeline runs at 10 frames per second on a NVidia K80 GPU. Our model showed that the state-of-the-art object detection methods [22, 23] can be extended for text detection and recognition, taking into account specifics of text, and still maintaining a low computational complexity.

[2]Method A [27] was authored by Google and neither the training data nor the algorithm is published.

Figure 9. End-to-end scene text recognition samples from the COCO-Text dataset. Model output in red, ground truth in green. Best viewed zoomed in color

We also demonstrated the advantage of the joint training for the end-to-end task, by outperforming the ad-hoc combination of the state-of-the-art localization and state-of-the-art recognition methods [6, 4, 8], while exploiting the same training data.

Last but not least, we showed that optimizing localization accuracy on human-annotated bounding boxes might not improve performance of an end-to-end system, as there is not a clear link between how well a method fits the bounding boxes created by a human annotator and how well a method reads text. Future work includes extending the training set with more realistic effects, single characters and digits.

Acknowledgment

JM was supported by the Czech Science Foundation Project GACR P103/12/G084, LN and MB by Technology Agency of the Czech Republic research program TE01020415 (V3C – Visual Computing Competence Center). Lukas would also like to acknowledge the support of the Google PhD Fellowship and the Google Research Award.

References

[1] A. Bosch, A. Zisserman, and X. Muoz. Image classification using random forests and ferns. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1 –8, oct. 2007. 2

[2] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8):1532–1545, 2014. 2

[3] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 3

[4] L. Gomez-Bigorda and D. Karatzas. Textproposals: A text-specific selective search algorithm for word spotting in the wild. *arXiv preprint arXiv:1604.02619*, 2016. 1, 6, 7, 8

[5] 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, 4, 5

[6] 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*, 2016. 1, 2, 4, 5, 6, 8

[7] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic data and artificial neural networks for natural scene text recognition. In *NIPS Deep Learning Workshop 2014*, 2014. 1, 4, 5

[8] 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, 2, 4, 6, 8

[9] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 2, 3

[10] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting. In *European conference on computer vision*, pages 512–528. Springer, 2014. 1

[11] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4565–4574, 2016. 2, 3

[12] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, B. Andrew, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, F. Shafait, S. Uchida, and E. Valveny. ICDAR 2015 robust reading competition. In *ICDAR 2015*, pages 1156–1160. IEEE, 2013. 1, 3, 5, 6, 7

[13] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, L. P. de las Heras, et al. ICDAR 2013 robust reading competition. In *ICDAR 2013*, pages 1484–1493. IEEE, 2013. 1, 5, 6

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[15] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu. Textboxes: A fast text detector with a single deep neural network. *arXiv preprint arXiv:1611.06779*, 2016. 1, 2

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 8

[17] 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. 2

[18] 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. 1, 2

[19] L. Neumann and J. Matas. Real-time lexicon-free scene text localization and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 38(9):1872–1885, Sept 2016. 6

[20] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. 5

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2

[22] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. *arXiv preprint arXiv:1612.08242*, 2016. 1, 2, 3, 5, 8

[23] 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. 1, 2, 3, 8

[24] 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, 4

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 3

[26] 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

[27] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie. Coco-text: Dataset and benchmark for text detection and recognition in natural images. *arXiv preprint arXiv:1601.07140*, 2016. 8

[28] 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

[29] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014. 2