Reading Scene Text in Deep Convolutional Sequences

Pan He*1,2, Weilin Huang*1,2, Yu Qiao1, Chen Change Loy2, and Xiaou Tang2,1

1Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
2Multimedia Laboratory, The Chinese University of Hong Kong,
{pan.he;wl.huang;yu.qiao}@siat.ac.cn;{ccloy;xtang}@ie.cuhk.edu.hk

Abstract

We develop a Deep-Text Recurrent Network (DTRN) that regards scene text reading as a sequence labelling problem. We leverage recent advances of deep convolutional neural networks to generate an ordered high-level sequence from a whole word image, avoiding the difficult character segmentation problem. Then a deep recurrent model, building on long short-term memory (LSTM), is developed to robustly recognize the generated CNN sequences, departing from most existing approaches recognising each character independently. Our model has a number of appealing properties in comparison to existing scene text recognition methods: (i) It can recognise highly ambiguous words by leveraging meaningful context information, allowing it to work reliably without either pre- or post-processing; (ii) the deep CNN feature is robust to various image distortions; (iii) it retains the explicit order information in word image, which is essential to discriminate word strings; and (iv) the model does not depend on pre-defined dictionary, and it can process unknown words and arbitrary strings. The proposed model achieves impressive performance on a number of benchmarks, advancing the state-of-the-arts [18, 1] substantially. Codes for the DTRN will be available.

1. Introduction

Text recognition in natural image has received increasing attention in computer vision, due to its numerous practical applications. This problem includes two sub tasks, namely text detection [16, 34, 35, 15, 24] and text-line/word recognition [18, 1, 17, 4, 33]. This work focuses on the latter that aims to retrieve a text string from a cropped word image. Though huge efforts have been devoted to text recognition, reading text in unconstrained environment is still extremely challenging, and remains an open problem, as substantiated in recent literature [17, 1, 33, 18, 1]. The main difficulty arises from the large diversity of text patterns (e.g. low resolution, low contrast, and blurring), and highly complicated background clutters. Consequently, individual character segmentation or separation is extremely challenging.

Most previous studies focus on developing powerful character classifiers, some of which are incorporated with an additional language model, leading to the state-of-the-art performance [18, 4, 33, 20]. These approaches mainly follow the basic pipeline of conventional OCR techniques by first involving a character-level segmentation, then followed by an isolated character classifier and post-processing for
recognition. Several approaches adopt deep neural networks for representation learning, but the recognition is still confined to character-level classification. For example, Bisacco et al. [4] proposed a PhotoOCR system by designing a five-layer neural network for character recognition. In [13], a deep CNN was trained for multiple tasks by sharing features. Note that all these current successful scene text recognition systems are mostly built on isolated character classifier. Their performance is thus severely harmed by the difficulty of character-level segmentation or separation. Importantly, recognizing each character independently discards meaningful context information of the words, significantly reducing its reliability and robustness.

First, we wish to address the issue of context information learning. The main inspiration for approaching this issue comes from the recent success of recurrent neural networks (RNN) for handwriting recognition [12], speech recognition [11], and language translation [28]. We found the strong capability of RNN in learning continuous sequential features particularly well-suited for text recognition task to retain the meaningful interdependencies of the continuous text sequence. We note that RNN has been formulated for recognizing handwritten or documented images [12, 5], nevertheless, the background in these tasks is relatively plain, and the text stroke information can be easily extracted or binarized at pixel level, making it possible to manually design a sequential heuristic feature for the input to RNN. In contrast, the scene text image is much more complicated where pixel-level segmentation is extremely difficult, especially for highly ambiguous images (Figure 1). Thus it is non-trivial to directly apply the sequence labelling models to scene text.

Consequently, the second challenge we need to resolve is the issue of character segmentation. We argue that individual character segmentation is not a ‘must’ in text recognition. The key is to acquire strong representation from the image, with explicit order information. The strong representation ensures robustness to various distortions and background clutters, whilst the explicit order information is crucial to discriminate a meaningful word. The ordered strong feature sequence can then be read by any sequence labeling model for recognition.

To this end, we develop a deep recurrent model that reads word images in deep convolutional sequences. The new model is referred as Deep-Text Recurrent Network (DTRN), of which the pipeline is shown in Fig. 1. It takes both the advantages of the deep CNN for image representation learning and the RNN model for sequence labelling, with the following appealing properties:

1) **Strong and high-level representation without character segmentation** – The DTRN generates a convolutional image sequence, which is explicitly ordered by scanning a sliding window through a word image. The CNN sequence captures meaningful high-level representation of the word image that is robust to various image distortions. The learned representation differs significantly from the manually-designed sequential features used by most prior studies based on sequence labelling [5, 12]. The sequence is generated without any low-level operation or challenging character segmentation.

2) **Exploiting context information** In contrast to existing systems [4, 18, 32] that read each character independently, we formulate the scene text reading as a sequence labelling problem. Specifically, we build our system on the long short-term memory (LSTM) [14], so as to capture the interdependencies inherent in the convolutional sequences. Such consideration allows our system to recognize highly ambiguous words, and work reliably without either pre- or post-processing. In addition, the recurrent property allows it to process sequences of various lengths, going beyond traditional neural networks of fixed-length input and output.

3) **Process unknown words and arbitrary strings** With properly learned deep CNN and RNN, our model does not depend on any pre-defined dictionary, unlike exiting studies [17, 18, 32], and therefore it can process unknown words and arbitrary strings.

We note that CNN and RNN have been widely used in many vision tasks. They have also been independently exploited in the domain of text recognition. Our main contribution in this study is to develop an unified deep recurrent system that leverages both the advantages of CNN and RNN for the difficult scene text recognition problem, which has been solved based on analyzing character independently. This is the first attempt to show the effectiveness of exploiting convolutional sequence with sequence labeling model for challenging scene text recognition. We highlight the considerations required to make this system reliable and discuss the unique advantages offered by it. The proposed DTRN demonstrate promising results on a number of benchmarks, improving recent results of [18, 1] considerably.

2. Related Work

Word image recognition has gained increasing attention over the last several years. Previous work focus on developing a powerful character classifier with a manually-designed image features. Wang et. al. proposed a random ferns model [26] with the HOG feature [6] for character classification [31, 30]. Neumann and Matas employed extremal regions [24], or proposed new oriented strokes [24] for character detection and classification. Their performance is limited by the low-level features. In [20], Lee et. al. developed a mid-level representation of image characters by proposing a discriminative feature pooling method. Similarly, Yao et. al. proposed a mid-level feature, the strokelets, to describe the parts of characters [33].
Recent advances of deep neural networks for image representation encourage the development of more powerful character classifiers with deep features, leading to the state-of-the-art performance on this task. The pioneer work was done by LeCun et. al., who designed a CNN for isolated handwriting digit recognition \[19\]. Wang et. al. proposed a two-layer CNN system for both character detection and classification \[52\]. PhotoOCR system employs a five-layer neural network for character recognition \[4\]. Similarly, Jaderberg et. al. \[18\] proposed novel deep features by employing a Maxout CNN model for learning common features, which were subsequently used for a number of different tasks, such as character classification, location optimization and language model learning.

However, these approaches treat isolated character classification and subsequent word recognition separately. They do not unleash the full potential of word context information in the recognition. They often design complicated optimization algorithm to infer word string by incorporating multiple additional visual cues, or require a number of post-processing steps to refine the results \[18, 4\]. Our model differs significantly to the above studies by exploring the recurrence of deep features, allowing it to leverage the underlying context information directly to recognize the whole word image in a deep sequence, without a language model and any kind of post-processings.

There is another group of studies that recognize text strings from the whole word images. Almazan et. al. \[1\] proposed a subspace regression method to jointly embed both word image and its string into a common subspace, where word is recognised by simply computing the nearest neighbour. Jaderberg et. al. \[17\] developed a powerful deep CNN model to compute a deep feature from a whole word image. Again, our model differs to these studies in the deep recurrent nature. Our deep sequential feature includes explicit spatial order information, which is crucial to discriminate the order-sensitive word string. While the other global representation would lose such strict order information, leading to poorer discrimination power. Furthermore, Jaderberg et. al.’s model \[17\] is strictly constrained by the pre-defined dictionary, making it unable to recognise a novel word. By contrast, our model can process an unknown word and arbitrary strings.

Our approach is partially motivated by the recent success of the RNN models for image captioning \[3, 7, 29\]. These systems apply the CNN model for computing a high-level deep feature from a whole image, followed by a RNN model to decode the deep feature into a sequence of words (a sentence) recurrently. But our task is different from them. The word images include explicit order information of its string, which is a crucial cue to discriminate a word. For the image caption, the image and its target captions do not have such strictly spatial correlation. Global representation may include implicit spatial information of the objects in the image, but applying it to the word image would significantly loss the strict order information. Our goal here is to derive a set of robust sequential features from the word image, and design an new model that bridges the image representation learning and sequence labelling task.

3. Deep-Text Recurrent Networks

The pipeline of Deep-Text Recurrent Network (DTRN) is shown in Fig.1. It starts by encoding a given word image into an ordered sequence with a specially designed CNN model. Then a RNN is employed to decode (recognise) the CNN sequence into a word string. The proposed system is end-to-end, i.e. it takes a word image as input and outputs the corresponding word string, without any pre- and post-processing steps. Both the input word image and the output string can be of varying lengths. This section revisits some important details of CNN and RNN and highlight the considerations that make their combination reliable for scene text recognition.

Formally, we formulate the process of word image recognition as a sequence labeling problem. We maximize the probability of the correct word strings \(S_w\) given an input image \(I\) as follows,

\[
\hat{\theta} = \arg \max_{\theta} \sum_{(I,S_w)} \log P(S_w|I;\theta),
\]

where \(\theta\) are the parameters of the recurrent system. \((I,S_w) \in \Omega\) is a sample pair from a training set, \(\Omega\), where \(S_w = \{S_w^1, S_w^2, ..., S_w^K\}\) is the ground truth word string (containing \(K\) characters) of the image \(I\). Commonly, the chain rule is applied to model the joint probability over \(S_w\),

\[
\log P(S_w|I;\theta) = \sum_{i=1}^{K} \log P(S_w^i|S_w^{i-1};\theta)
\]

Thus we optimize the sum of the log probabilities over all sample pairs in the training set \(\Omega\) to learn the model parameters. We develop a RNN to model the sequential probabilities \(P(S_w^i|S_w^{i-1}, ..., S_w^0;\theta)\), where the variable number of the sequentially conditioned characters can be expressed by an internal state of the RNN in hidden layer, \(h_t\). This internal state is updated when the next sequential input \(x_t\) is presented by computing a non-linear function \(\mathcal{H}\),

\[
h_{t+1} = \mathcal{H}(h_t, x_t)
\]

where the non-linear function \(\mathcal{H}\) defines exact form of the proposed recurrent system. \(X = \{x_1, x_2, x_3, ..., x_T\}\) is the sequential CNN features computed from the word image,

\[
\{x_1, x_2, x_3, ..., x_T\} = \varphi(I)
\]
Designs of the $\phi$ and $\mathcal{H}$ play crucial roles in the proposed system. We develop a CNN model to generate the sequential $x_t$, and define $\mathcal{H}$ with a long short-term memory (LSTM) architecture [14].

3.1. Sequence Generation with Maxout CNN

The main challenge of obtaining high-level sequential representation from the word images arises from the difficulties of correct segmentation at either pixel or character level. We argue that it is not necessary to perform feature extraction at the character level. On the contrary, it is more natural to describe word strings in sequences where their explicit order information is retained. This information is extremely important to discriminate a word string. Furthermore, the variations between continuous examples in a sequence should encode additional information, which could be useful in making more reliable prediction. By considering these factors, we propose to generate an explicitly ordered deep sequence with a CNN model, by sliding a sub-window through the word image.

To this end, we develop a Maxout network [9] for computing the deep feature. It has been shown that the Maxout CNN is powerful for character classification [18, 2]. The basic pipeline is to compute point-wise maximum through a number of grouped feature maps or channels. For our network, the input image is of size $32 \times 32$, corresponding to the size of sliding-window. The network has five convolutional layers, each of which is followed by a two-group or four-group Maxout operation, with different numbers of feature maps, i.e. 48, 64, 128, 128 and 36 respectively. Similar to the CNN model used in [18], our networks does not involve any pooling operation, and the output map of last two convolutional layers are just one pixel. This allows our CNN to convolute the whole word images at once, leading to a significant computational efficiency. For each word image, we resize it into the same height of 32, and keep its original aspect ratio unchanged. We apply the learned filters to the resized image, and get a 128D CNN sequence directly from the output of last second convolutional layer. This operation is similar to computing deep feature independently from the sliding-window by moving it densely through the image, but with much computational efficiency. Our Maxout CNN is trained on 36 classes comprised of case insensitive character sample images.

3.2. Sequence Labeling with RNN

We believe that the interdependencies between the convolutional sequence include meaningful context information which would be greatly helpful to identify an ambitious character. RNN has shown strong capability for learning meaningful structure from an ordered sequence. Another important property of the RNN is that the rate of changes of the internal state can be finely modulated by the recurrent weights, which contributes to its robustness against localised distortions of the input data [12]. Thus we propose the use of RNN in our framework to model the generated CNN sequence $\{x_1, x_2, x_3, \ldots, x_T\}$. The structure of our RNN model is shown in Fig. 3.

The main shortcoming of the standard RNN is the vanishing gradient problem, making it hard to transmit the gradient information consistently over long time [14]. This is a crucial issue in designing a RNN model, and the long short-term memory (LSTM) was proposed specially to address this problem [14]. The LSTM defines a new neuron...
The recurrent model is trained with steepest descent. The parameters are updated per training sequence by using a learning rate of $10^{-4}$ and a momentum of 0.9. Each input sequence is randomly selected from the training set, we perform forward-backward algorithm [12] [10] to jointly optimize the bidirectional LSTM and CTC parameters, where a forward propagation is implemented through whole network, followed by a forward-backward algorithm that aligns the ground truth word strings to the LSTM output maps, $\pi \in B^{-1}(S_w)$, $\pi, p \in \mathbb{R}^{37 \times T}$. The loss function of LSTM output sequence into a word string), but also makes it possible to be trained in an end-to-end fashion by minimizing an overall loss function over $(X, S_w) \in \Omega$. The loss for each sample pair is computed as sum of the negative log likelihood of the true word string,

$$L(X, S_w) = - \sum_{i=1}^{K} \log P(S_w^i | X) \quad (6)$$

Finally, our RNNs model follows a bidirectional LSTM architecture, as shown in Fig. 3. It has two separate LSTM hidden layers that process the input sequence forward and backward, respectively. Both hidden layers are connected to the same output layers, allowing it to access to both past and future information in the sequence. In several sequence labelling tasks, such as handwriting recognition [12] and phoneme recognition [13], the bidirectional RNNs have shown stronger capability than the standard RNNs. Our RNNs model is trained with the Forward-Backward Algorithm that jointly optimizes the bidirectional LSTM and CTC. Details of the learning algorithm can be found in the supplementary material.

### 3.3. Implementation Details

Our CNN model is trained on about $1.8 \times 10^5$ character images cropped from the training sets of a number of benchmarks by Jaderberg et al. [18]. We generate the CNN sequence by applying the trained CNN with a sliding-window (as described in Section 3.1) on the word images, followed by a column-wise normalization. Our recurrent model contains a bidirectional LSTM architecture. Each LSTM layer has 128 LSTM cell memory blocks. The input layer of our RNN model has 128 neurons (corresponding to the dimensions of the CNN sequence, $x_t \in \mathbb{R}^{128}$), which are fully connected to both hidden layers. The outputs of two hidden layers are concatenated, and then fully connected to the output layer of the LSTM with 37 output classes (including an additional non-character class), by using a softmax function. Thus our RNN model has 273445 parameters in total, which are initialized with a Gaussian distribution of mean 0 and standard deviation 0.01 in the training process.

The recurrent model is trained with steepest descent. The parameters are updated per training sequence by using a learning rate of $10^{-4}$ and a momentum of 0.9. Each input sequence is randomly selected from the training set, we perform forward-backward algorithm [12] [10] to jointly optimize the bidirectional LSTM and CTC parameters, where a forward propagation is implemented through whole network, followed by a forward-backward algorithm that aligns the ground truth word strings to the LSTM output maps, $\pi \in B^{-1}(S_w)$, $\pi, p \in \mathbb{R}^{37 \times T}$. The loss function of

or cell structure in the hidden layer with three additional multiplicative gates: the input gate, forget gate and output gate. These new cells are referred as memory cells, which allow the LSTM to learn meaningful long-range interdependencies. The structure of the memory cells is described in Fig. 4. We skip the standard formulation of LSTM and leave it in the supplementary material.

The sequence labelling of varying lengths is processed by recurrently implementing the LSTM memory for each sequential input $x_t$, such that all LSTMs share the same parameters. The output of the LSTM $h_t$ is used to fed to the LSTM at next input $x_{t+1}$. It is also used to compute the current output, which is transformed to the estimated probabilities over all possible characters. It finally generates a sequence of the estimations with the same length as input sequence, $p = \{p_1, p_2, p_3, ..., p_T\}$.

Due to the unsegmented nature of the word image at the character level, the length of the LSTM outputs $(T)$ is not consistent with the length of a target word string, $|S_w| = K$. This makes it difficult to train our recurrent system directly with the target strings. To this end, we follow the recurrent system developed for the handwriting recognition [12] by applying a connectionist temporal classification (CTC) [13] to approximately map the LSTM sequential output $(p)$ into its target string as follow,

$$S_w^* \approx B(\arg\max_{\pi} P(\pi | p)) \quad (5)$$

where the projection $B$ removes the repeated labels and the non-character labels [13]. For example, $B(−gg−o−oo−dd−) = good$. The CTC looks for an approximately optimized path $(\pi)$ with maximum probability through the LSTMs output sequence, which aligns the different lengths of LSTM sequence and the word string.

The CTC is specifically designed for the sequence labelling tasks where it is hard to pre-segment the input sequence to the segments that exactly match a target sequence. In our RNN model, the CTC layer is directly connected to the outputs of LSTMs, and works as the output layer of the whole RNN. It not only allows our model to avoid a number of complicated post-processing (e.g. transforming the sequence to the segments that exactly match a target sequence. where a forward propagation is implemented through whole network, followed by a forward-backward algorithm that aligns the ground truth word strings to the LSTM output maps, $\pi \in B^{-1}(S_w)$, $\pi, p \in \mathbb{R}^{37 \times T}$. The loss function of

![Figure 4. LSTM memory cell.](image-url)
Figure 5. RNNs training process recorded at epoch 0 (row 1), 5 (row 2) and 50 (row 3) with an input CNN sequence from a same word image (row 4). (a) the LSTM output ($p$); (b) the CTC path ($\pi$) mapped from ground truth word string ($B^{-1}(S_w)$); (c) maximum probabilities of the character and segmentation line with $p$ and $\pi$.

E.q.(6) is computed approximately as,

$$L(X, S_w) \approx -\sum_{t=1}^{T} \log P(\pi_t | X)$$

Finally the approximated error is propagated backward to update the parameters. The RNN is trained on about 3000 word images, taken from the training sets of three benchmarks mentioned in the next section. The training process is illustrated in Fig. 5.

4. Experiments and Results

The performance of the proposed DTRN is compared against start-of-the-art methods on three standard benchmarks for cropped word image recognition: the Street View Text [30], ICDAR 2003 [21] and IIIT 5K-word [22]. Specifically, The Street View Text (SVT) [30] has 647 word images collected from Google Street View of road-side scenes. The images are highly challenging with large variations in illumination, character sizes, and fonts. Some images are significantly blurred. It provides a lexicon of 50 words per image for recognition (SVT-50). The ICDAR 2003 (IC03) [21] contains 860 word images cropped from 251 natural images. Lexicons with 50 words per image (IC03-50) and all words of the test set (IC03-FULL) are provided by Wang et. al. [30]. The IIIT 5K-word (IIIT5K) [22] is comprised of 5000 cropped word images from both scene texts and born-digital images. The dataset is split into subsets of 2000 and 3000 images, for test and training, respectively. Each word image is associated with lexicons of 50 words (IIIT5k-50) and 1k words (IIIT5k-1k) for test.

Figure 6. Output confidence maps of the Maxout CNN (middle) and the LSTM output layer of the DTRN (bottom) on a number of ambiguous images (top).

4.1. DTRN vs Maxout CNN

The recurrence property of the proposed DTRN makes it distinct against the current deep CNN models, such as DeepFeatures [18] and Wang et. al.’s system [32]. The advantage is shown clearly in Fig. 6 where the output maps of the LSTM layer and the Maxout CNN of DeepFeatures [18] are compared. As can be observed, the output maps of the DTRN are much clearer than those of the Maxout CNN in a number of highly ambiguous word images. The character probability distribution and segmentation are shown clearly and correctly on the maps, indicating the strong capability of our model for identifying word texts from challenging images. The final word recognition is straightforward by simply applying the $B$ projection (in E.q. 5) on these maps. The DeepFeatures compute word strings directly from these highly ambiguous maps by designing a
A complicated optimization function to incorporate multiple additional cues, such as both case sensitive and insensitive maps, a learned language model, a pre-defined lexicon, breakpoint locations, and a number of heuristic cues. This makes the system difficult to jointly optimize with all these additional information, and also requires to tune a number of model parameters manually. Furthermore, some of these additional cues are even hard to learn correctly and reliably.

4.2. Comparisons with State-of-the-Arts

The evaluation is conducted by following the standard protocol [18, 4, 1, 30, 32], where each word image is associated with a lexicon, and edit distance is computed to find the optimized word. The recognition results of the DTRN on some word image examples are presented in Fig. 7, including both the correct and incorrect results on a number of challenging cases. As can been seen, the DTRN demonstrates excellent capability on recognising extremely ambiguous word images, some of which are even hard to human. This is mainly beneficial from its strong ability to leverage explicit order and meaningful word context information.

The full results of the DTRN on the SVT, ICo03 and IIIT5K datasets are presented in Table 1. They are compared systematically to recent results achieved by mid-level features [33, 20, 27, 25, 22], deep neural networks systems [18, 2, 32] and whole image based representations [1, 8], along with discussions on their advantages and disadvantages.

**Mid-level representation**: Strokelet [33] and Lee et. al.’s method [20] achieved leading performance based on the mid-level features. Though these systems show large improvements over conventional low-level features, their performance are not comparable to our model, with significant reductions in accuracies in all the three datasets.

**Deep neural networks**: As shown in Table 1, deep neural networks methods largely outperform the mid-level approaches, with close to 10% of improvement in all cases. The considerable performance gains mainly come from its ability to learn a deep high-level feature from the word image. DeepFeatures [18] achieved leading results on both the SVT and ICo03 datasets. However, the DeepFeatures are still built on isolate character classifier. By training a similar CNN model with the same training data, the proposed DTRN achieved significant improvements over the DeepFeatures in all three datasets. The results agree with our analysis conducted in Sec. 4.1. On the most widely-used SVT dataset, our model outperforms the DeepFeatures considerably from 86.1% to 92.0%, indicating the superiority of our recurrent model in connecting the isolated deep features in an ordered sequence for recognition. Furthermore, our system does not need to learn the language model and isolated character location information; all of these information are optimized jointly and automatically by our RNN model in an end-to-end fashion.

**Whole image representation**: The DTRN is compared with Almazan et. al.’s approach [1] based on the whole word image representation. Almazan et. al.’s approach achieved 87.0% accuracy on the SVT, slightly over that of DeepFeatures. In the IIIT5k, it yielded 88.6% and 75.6% on small and large lexicons respectively, surpassing previous results with a large margin. Our DTRN strives for a further step by reaching the accuracies of 94.0% and 91.6% on the IIIT5k. The significant improvements may benefit from the explicit order information included in our CNN sequence. It is the key to increase the discriminative power of our model for word representation, which is highly sensitive to the order of characters. The strong discriminative power can be further verified by the consistent high-performance of our system along with the increase of the size of lexicons, where the accuracy of Almazan et. al.’s approach drops significantly.
| Method          | IC03-50 (%) | IC03-FULL (%) | SVT-50 (%) | IIIT5k-50 (%) | IIIT5k-1000 (%) |
|-----------------|-------------|---------------|------------|---------------|----------------|
| ABBYY [30]      | 56.0        | 55.0          | 35.0       | 24.3          | -              |
| Wang [30]       | 76.0        | 62.0          | 57.0       | 64.1          | 57.5           |
| Mishra [22]     | 81.8        | 67.8          | 73.2       | -             | -              |
| Novikova [25]   | 82.8        | -             | 72.9       | -             | -              |
| TSM+CRF [27]    | 87.4        | 79.3          | 73.5       | -             | -              |
| Lee [20]        | 88.0        | 76.0          | 80.0       | -             | -              |
| Strokelets [33] | 88.5        | 80.3          | 75.9       | 80.2          | 69.3           |
| Wang&Wu [32]    | 90.0        | 84.0          | 70.0       | -             | -              |
| Alsharif [2]    | 93.1        | 88.6          | 74.3       | -             | -              |
| DeepFeatures [18]| 96.2        | 91.5          | 86.1       | -             | -              |
| Goel [8]        | 89.7        | -             | 77.3       | -             | -              |
| Almazan [1]     | -           | -             | 87.0       | 88.6          | 75.6           |
| DTRN            | 97.0        | 94.4          | 92.0       | 94.0          | 91.6           |

Training on large additional datasets: We further compare our method with PhotoOCR system [4], which sets a strong baseline on the SVT dataset (90.4%) by using large additional training datasets. It employed about $10^7$ manually labeled examples to learn a powerful deep neural networks for character classification, and also trained a strong language model with a corpus of more than a trillion tokens [4]. Both advantages lead to its high-performance. However, it involves a number of low-level techniques to over-segment characters from the word images, and jointly optimizes the segmentation, character classification and language model with beam search. Furthermore, it includes a number of post-processing steps to further improve the performance, such as punctuation search, secondary language scoring, shape model and junk filter, which make the system highly complicated [4]. The DTRN achieved 1.6% improvement over the PhotoOCR on the SVT dataset. This improvement is also significant by considering only a fraction of the training data (two orders of magnitude less data) we used. While our model works without a language model, and does not need any complicated post-processing step.

Jaderberg et al. [17] proposed another powerful deep CNN model by computing a deep feature from the whole word image. This model achieves the best results in all databases. However, directly comparing our DTRN to it is relatively unfair. Jaderberg et al.’s model casts the word recognition problem as a large-scale classification task by considering the images of a same word as a class. The output classes in the output layer of CNN should include all possible words, which should be at least equal to the total number of words in a regular English dictionary (about 90K). This causes a huge number of model parameters, e.g. $90K \times 4096 \approx 4 \times 10^8$ in just last fully-connected layer (compared to ours $2.7 \times 10^5$), making their model highly difficult to be trained. Their excellent performance was achieved by training about $7 \times 10^7$ word images, while our model was just trained on $3 \times 10^3$ word images. Furthermore, Jaderberg et. al.’s model is not flexible to recognize a new word which is not trained or not included in a pre-defined dictionary. While the scene texts often include many irregular word strings (the number could be unlimited) which are impossible to be known in advanced, such as "AB00d". Therefore their model needs to know all ground-truth words in the test set apriori, and uses them to generate corresponding synthetic word images for training. The DTRN recognises a word image as a set of sequential characters, thus it can process unknown words and arbitrary strings, providing a more principled approach for this task.

5. Conclusion

We have presented a Deep-Text Recurrent Network (DTRN) for scene text recognition. It models the scene text reading as a deep sequence labelling problem that overcomes a number of main limitations on this task. It computes a set of explicitly-ordered deep features from the whole word image, which is not only robust to low-level image distortions, but also highly discriminate to word strings. The recurrence property makes it capable of recognising highly ambiguous images by leveraging meaningful word context information, and also allows it to process unknown words and arbitrary strings, providing a more principled approach for this task. Experimental results show that our model has achieved the state-of-the-art performance on a number of benchmarks.
References

[1] J. Almazán, A. Gordo, A. Fornés, and E. Valveny. Word spotting and recognition with embedded attributes. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 36:2552–2566, 2014.

[2] O. Alsharif and J. Pineau. End-to-end text recognition with hybrid HMM maxout models. *arXiv:1310.1811v1*, 2013.

[3] K. Andrej and F. Li. Deep visual-semantic alignments for generating image descriptions, 2015. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[4] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven. Photoocr: Reading text in uncontrolled conditions, 2013. *IEEE International Conference on Computer Vision (ICCV)*.

[5] T. Breuel, A. Ul-Hasan, M. Azawi, and F. Shafait. High-performance ocr for printed english and fraktur using lstm networks, 2013. *International Conference on Document Analysis and Recognition (ICDAR)*.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection, 2005. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[7] J. Donahue, L. Hendricks, S. Guadarrama, and M. Rohrbach. Long-term recurrent covolutional networks for visual recognition and description. *arXiv:1411.4389v2*, 2014.

[8] V. Goel, A. Mishra, K. Alahari, and C. V. Jawahar. Whole is greater than sum of parts: Recognizing scene text words, 2013. *International Conference on Document Analysis and Recognition (ICDAR)*.

[9] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. *arXiv:1302.4389v4*, 2013.

[10] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks, 2006. *IEEE International Conference on Machine Learning (ICML)*.

[11] A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks, 2014. *IEEE International Conference on Machine Learning (ICML)*.

[12] A. Graves, M. Liwicki, and S. Fernandez. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 31:855–868, 2009.

[13] A. Graves and J. Schmidhuber. Framework phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18:602–610, 2005.

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[15] W. Huang, Z. Lin, J. Yang, and J. Wang. Text localization in natural images using stroke feature transform and text co-variance descriptors, 2013. *IEEE International Conference on Computer Vision (ICCV)*.

[16] W. Huang, Y. Qiao, and X. Tang. Robust scene text detection with convolution neural network induced mser trees, 2014. *European Conference on Computer Vision (ECCV)*.

[17] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Reading text in the wild with convolutional neural networks. *arXiv:1412.1842v2*, 2014.

[18] M. Jaderberg, A. Vedaldi, and A. Zisserman. Deep features for text spotting, 2014. *European Conference on Computer Vision (ECCV)*.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[20] C. Lee, A. Bhardwaj, W. Di, and V. J. andR. Piramuthu. Region-based discriminative feature pooling for scene text recognition, 2014. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[21] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. Icdar 2003 robust reading competitions, 2003. *International Conference on Document Analysis and Recognition (ICDAR)*.

[22] A. Mishra., K. Alahari, and C. Jawahar. Scene text recognition using higher order language priors, 2012. *British Machine Vision Conference (BMVC)*.

[23] L. Neumann and J. Matas. Real-time scene text localization and recognition, 2012. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[24] L. Neumann and J. Matas. Scene text localization and recognition with oriented stroke detection, 2013. *IEEE International Conference on Computer Vision (ICCV)*.

[25] T. Novikova, O. Barinova, P. Kohli, and V. Lempitsky. Large-lexicon attribute-consistent text recognition in natural images, 2012. *European Conference on Computer Vision (ECCV)*.

[26] M. Ozuyosal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code, 2007. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[27] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, and Z. Zhang. Scene text recognition using part-based tree-structured character detection, 2013. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[28] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks, 2014. *Neural Information Processing Systems (NIPS)*.

[29] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: a neural image caption generator. *arXiv:1411.4555v1*, 2014.

[30] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition, 2011. *IEEE International Conference on Computer Vision (ICCV)*.

[31] K. Wang and S. Belongie. Word spotting in the wild, 2010. *European Conference on Computer Vision (ECCV)*.

[32] T. Wang, D. Wu, A. Coates, and A. Y. Ng. End-to-end text recognition with convolutional neural networks, 2012. *IEEE International Conference on Pattern Recognition (ICPR)*.

[33] C. Yao, X. Bai, B. Shi, and W. Liu. Strokelets: A learned multi-scale representation for scene text recognition, 2014. *IEEE Computer Vision and Pattern Recognition (CVPR)*.

[34] X. C. Yin, X. Yin, K. Huang, and H. W. Hao. Robust text detection in natural scene images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 36:970–983, 2014.

[35] Z. Zhang, W. Shen, C. Yao, and X. Bai. Symmetry-based text line detection in natural scenes, 2015. *IEEE Computer Vision and Pattern Recognition (CVPR)*.