Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com

Accurate recognition of words in scenes without character segmentation using recurrent neural network

Bolan Su, Shijian Lu [*]

Institute for Infocomm Research (I2R), #21-01, 1 Fusionopolis Way, Singapore 138632, Singapore

ARTICLE INFO

Article history:
Received 6 June 2016
Received in revised form 20 September 2016
Accepted 15 October 2016
Available online xxx

Keywords:
Scene text recognition
Recurrent neural network

ABSTRACT

Recognition of texts in scenes is one of the most important tasks in many computer vision applications. Though different scene text recognition techniques have been developed, scene text recognition under a generic condition is still a very open and challenging research problem. One major factor that defers the advance in this research area is character touching, where many characters in scene images are heavily touched with each other and cannot be segmented for recognition. In this paper, we proposed a novel scene text recognition technique that performs word level recognition without character segmentation. Our proposed technique has three advantages. First it converts each word image into a sequential signal for the scene text recognition. Second, it adapts the recurrent neural network (RNN) with Long Short Term Memory (LSTM), the technique that has been widely used for handwriting recognition in recent years. Third, by integrating multiple RNNs, an accurate recognition system is developed which is capable of recognizing scene texts including those heavily touched ones without character segmentation. Extensive experiments have been conducted over a number of datasets including several ICDAR Robust Reading datasets and Google Street View dataset. Experiments show that the proposed technique is capable of recognizing texts in scenes accurately.

1. Introduction

Text recognition in scenes is one of the most important research areas in computer vision and it has been studied for many years with different successful applications. Due to the rapid development of mobile sensors and internet technology, a huge amount of digital images are produced every day. Textual regions as one of the most informative regions in scene images need to be interpreted properly and automatically to make these images more accessible and valuable.

The Robust Reading Competitions [1,2] held under the framework of the International Conference on Document Analysis and Recognition(ICDAR) 2011 & 2013 show recent development on this research topic. One of tasks in these competitions is to recognize cropped word images which have little constraints in terms of text fonts, environmental lighting, image background, etc. A number of recognition systems have been reported and evaluated over the benchmarking datasets and the recognition accuracy has been lifted from the initial around 50% to the recent around 80% over the last decades.

Scene text recognition has been investigated in two typical approaches. The first is the traditional OCR (Optical Character Recognition) approach, which first segments text pixels from the image background and then applies some existing OCR engine to recognize the segmented characters. Another is feature based approach, which extracts various visual features such as HOG (histograms of oriented gradients) and SIFT (scale-invariant feature transform) to train a multi-class character classification model.

The traditional OCR techniques have been developed for decades and achieved great success in different commercial systems. On the other hand, most of them are designed for the scanned document texts which are usually well formatted and have a good image quality. They often fail to produce good results when applied for texts in scenes, where characters have little constraints in term of text fonts, environmental lighting, image background, etc. as illustrated in Figs. 1 (a) and (e). Several systems [3–5] have been reported to extract a clean character regions before feeding to OCR engines but they usually suffer from two typical constraints. First, text segmentation in scene images is a non-trivial problem due to uneven illumination, blur, low text background contrast, etc., as illustrated in Figs. 1 (e), and (g). Second, texts in scene images often have perspective distortion and special fonts, which cannot be recognized by traditional OCR engines properly as illustration in Figs. 1 (c) and (h). Different image restoration techniques [6,7] are often required to produce satisfactory recognition results.

The other approach exploits the object recognition techniques that have been extensively studied in recent years. In particular, these techniques can be categorised into two groups, namely character level recognition methods [9–16] and word level recognition methods [17–19]. The character level recognition methods first recognize each character of the word image, and then group all the recognized characters into a word string. Various visual features such as HOG [12,13,20], and part based tree structure [14] have been exploited to represent characters in scenes. The convolutional neural network (CNN) has also been widely used as the character classifier in recent years [9–11,16]. Besides, different clustering strategies have been proposed to group the recognized characters into a word string such as pictorial structure [13], conditional random field [12,14], HMM [9], N-gram model [10,16], etc. On the other hand, segmenting a

[*] Corresponding author.
Email addresses: subl@i2r.a-star.edu.sg (B. Su); slu@i2r.a-star.edu.sg (S. Lu)

http://dx.doi.org/10.1016/j.patcog.2016.10.016
0031-3203/© 2016 Published by Elsevier Ltd.

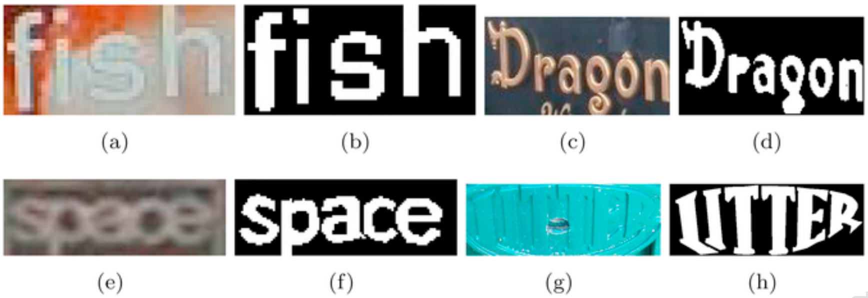

Fig. 1. Four text image examples and their corresponding text segmentation ground truth that are taken from the benchmarking word image dataset [8] (From up to down, the text images become more difficult to recognize). The OCR results obtained using Abbyy Fine Reader 10.0 are (a: r), (b: fish), (c: –), (d: Draoon), (e: –), (f: –), (g: –), (h: –), where '–' denotes no results produced.

word images into character images is often a very challenging task and sometime even impossible as illustrated in Fig. 2 [21].

The word-level recognition treats each word image as a whole and performs recognition without the character segmentation. Different techniques [17–19] have been proposed in recent years and very promising results have been obtained. In particular, the discrete wavelet transform (DWT) method [17] tries to find smallest distance between the word images and the font-renderings words within a lexicon. The attribute embedding method [18] creates a joint embedding space for word images and the word strings within a lexicon and finds a close match. The Whole Word Deep CNN method [19] treats each possible word in the lexicon as an output label of the trained CNN. The common limitation of these methods is that they all require an explicit lexicon which is costly and often inaccessible under many scenarios.

In [23], we proposed a scene text recognition technique that treats a word image as an unsegmented sequence. The major advantage is that it does not require an explicit lexicon (e.g. all the possible words are listed) and can perform the word-level recognition without lexicon or with an implicit lexicon (e.g. some constraints on the output word string) which is much easier to construct. Input images are normalized into the same height and retain the aspect ratio before the feature extraction. The column feature is extracted by using a fixed sized window. The major limitation of [23] is that the column features with a fixed window size cannot capture characteristics of different characters concurrently. The reason is that the aspect ratio of different characters such as ‘i′, ‘I′, ‘W′, and ‘M′ is very different, and so the same character in different fonts.

The new model as presented in this paper addresses the limitation and improves the word recognition accuracy significantly. In particular, we used image patches of different sizes to handle the large character aspect ratio variation and this approach also captures much richer characteristics of texts. Generally speaking, a small image patch can capture the stroke-level features as well as those thin characters such as ‘l′ and ‘i′, whereas a larger image patch is able to capture the character/intra-character level features as well as those wide characters such as ‘M′ and ‘W′. In addition, the new model implemented multiple recurrent neural networks (RNNs) to combine column features from patches of different window sizes. Experiments show that the new model is robust and able to recognize various challenging word images correctly.

The contributions can be summarized as follows:

- First, we design an effective way of converting a word image into a sequential signal so that RNN techniques, which have been successfully used in speech processing and handwriting recognition areas, can be introduced and applied. We adapt RNN for the recognition of texts in scenes, and design a segmentation-free scene word recognition system that obtains superior word recognition accuracy.
- Second, we propose a new ensembling technique that combines outputs from two RNNs for better recognition results. The proposed ensembling technique is generic and can be easily extent to ensemble other models for better performance.
- Third, compared with some systems [10,11] that rely heavily on certain local dataset (which are not available to the public), our system makes use of several publicly available datasets in training stage, hence providing a baseline for easier benchmarking of the ensuing scene text recognition techniques.
- Last but not least, compared with the character based recognition methods, our system only require word level annotation of text for training, which could reduce the effort on generating character level ground truth, as well as character level segmentation greatly.

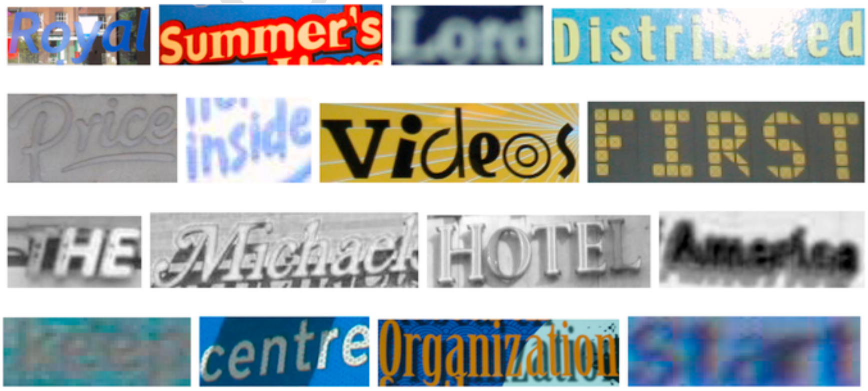

Fig. 2. Word image examples taken from the recent Public Datasets [22,1,2]. All the words in the images are correctly recognized by our proposed method.

2. Proposed method

The proposed technique consists of three key components, including sequential feature generation which converts a word image into a sequential feature, RNN model training where two multi-layer RNNs are trained together with LSTM [24] and connectionist temporal classification (CTC) [25], and an ensembling technique that combines outputs of multiple RNNs to produce improved word recognition accuracy.

The overall system consists of three components. In the first component, a word image is converted into sequences of column feature based on HOG features with different parameter settings. In total, we extracted two feature sets for training. In the second component, two multi layer recurrent neural network (RNN) model with bidirectional Long Short-Term Memory (LSTM) [24] is trained to classify the two sets of sequential data. After that, the score of each word in the lexicon is calculated separately for each RNN model using connectionist temporal classification (CTC) [25] technique. Finally the scores of different networks are combined to generate the final output in the third component.

2.1. Word to sequential feature conversion

The RNN model takes a feature vector as input at each step and it is widely used for sequential data classification. For example, the acoustic feature is extracted at each time frame when RNN is applied to voice data for speech recognition. Similarly, we can extract visual features from each column of a word image and feed them into RNN for word recognition. This idea is straightforward and has been applied for handwriting recognition [25] with promising results.

On the other hand, the RNN cannot be directly applied for scene text recognition because text strokes in scenes often cannot be segmented easily due to the high complexity of the scene image background. With good segmentation, very limited meaningful information can be extracted if we simply take one column of image pixels for processing. Nevertheless, more meaning features can be extracted from a column of image patches instead of a column of image pixels.

Different feature descriptors such as HOG and dense SIFT can be extracted from each image patch for the object recognition task. For the recognition of characters/words in scenes, HOG performs much better than dense SIFT because of the special characteristic of the text images. In particular, text strokes in scene images usually have strong gradient across their boundary. HoG can reliably capture such gradient information and so the shape of text strokes which is critical for the scene text recognition. Dense SIFT instead extracts descriptors in constant steps, where the text stroke information can be easily missed. We compare HOG and dense SIFT through different experiments to be described in Section 3.4.

In particular, the HOG features are first extracted by resizing the input images to the same height to obtain a column feature with the same length. The input images are then convolutionally partitioned into patches with step size 1.

The HOG feature is normalized for each image patch, and those extracted from the same column are then linked together. An average pooling is further applied to incorporate information of neighbouring blocks by averaging the HOG feature vectors within a neighbouring window. A column feature is finally determined by concatenating the averaged HOG feature vectors at the same column, so the vertical positioning information of the text is preserved in the sequential feature. Fig. 3 illustrates the overall feature extraction process using 3×3 image patch. We also tested max pooling in our experiments. Our test shows that average pooling scheme performs slightly better, largely due to the better suppression of the background noise by the average pooling. To ensure that all the column features have the same length, the input word image needs to be normalized to be of the same height M beforehand. Furthermore, the patch size W and neighbouring windows size T can be set empirically, and will be discussed in the experiment section.

2.2. Recurrent neural network modeling

RNN is a special neural network that has been used for handling sequential data. The RNN aims to predict the label of current time stamp with the contextual information of past time stamps. It is a powerful classification model but not widely used in the literature. The major reason is that it often requires a long training process as the error path integral decays exponentially along the sequence [26].

The long short-term memory (LSTM) model [26] was proposed to solve this problem as illustrated in Fig. 4. In LSTM, an internal memory structure is used to replace the nodes in the traditional RNN, where the output activation of the network at time t+1 is determined by the input data of the network at time t+1 and the internal memory stored in the network at time t. The learning procedure under LSTM therefore becomes local and constant. Furthermore, a forget gate is added to determine whether to reset the stored memory [27]. This strategy helps the RNN to remember contextual information and withdraw errors during learning. The memory update and output activation procedure of RNN can be formulated in Eq. (1) as follows:

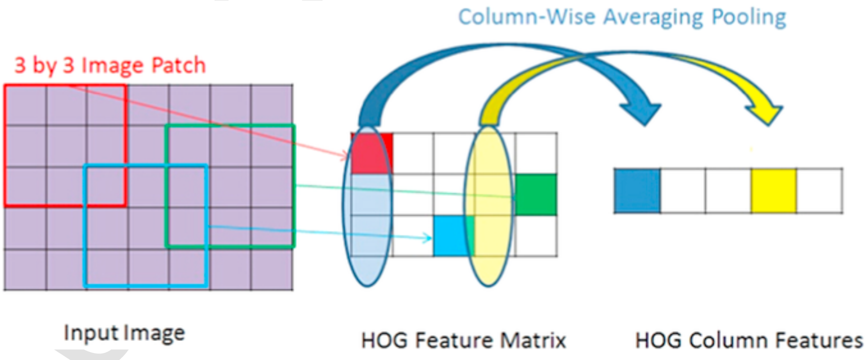

Fig. 3. An illustration of the HOG column feature extraction process: The 5 by 7 rectangle grid is used to represent an input image, where each cell denotes an image pixel. The feature is extracted based on 3 by 3 image patch to form a HOG feature matrix represented by a 3 by 5 grid, each cell of which denotes a HOG feature vector. Finally, the average pooling strategy is applied column by column to construct the HOG column feature.

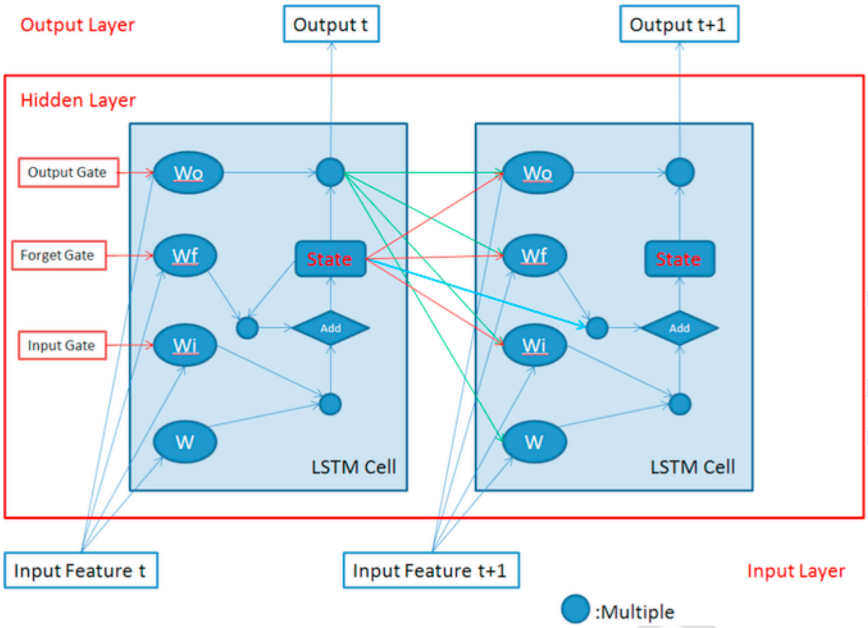

Fig. 4. Illustration of Recurrent Neural Network, where W_o, W_f and W_i stand for the output gate, forget gate and input gate, respectively.

$$S^{t+1} = S^t \cdot f\left(W_f X^{t+1}\right) + f\left(W_i X^{t+1}\right) \cdot f\left(W X^{t+1}\right) \tag{1a}$$

$$Y^{t+1} = f\left(W_o X^{t+1}\right) \cdot g\left(S^{t+1}\right) \tag{1b}$$

where $S^{t+1}, X^{t+1}, Y^{t+1}$ denote the stored memory, input data, and output activation of the network at time t+1, respectively. Functions $g(*)$ and $f(*)$ refer to the sigmoid function that squashes the data. W_* denotes the weight parameters of the network. In addition, the first term $\left(W_{forget} X^{t+1}\right)$ is used to control whether to withdraw previous stored memory S^t.

Bidirectional LSTM is further proposed to predict the current label with past and future contextual information by processing the input sequence in two directions (i.e. from beginning to end and, from end to beginning).

CTC [25] is then applied to the output layer of RNN to label the unsegmented data. In our system, a training sample can be viewed as a pair of input column feature and a target word string $(\mathbf{C}, \mathcal{W})$. The objective function of CTC is then defined as follows:

$$\mathcal{O} = -\sum_{(\mathbf{C},\mathcal{W})\in\mathcal{S}} \ln p(\mathcal{W}|\mathbf{C}) \tag{2}$$

where $\mathcal{S}$ denotes the whole training set and $p(\mathcal{W}|\mathbf{C})$ denotes the conditional probability of word $\mathcal{W}$ given a sequence of column feature $\mathbf{C}$. The target is to minimize $\mathcal{O}$, which is equivalent to maximize the conditional probability $p(\mathcal{W}|\mathbf{C})$.

The output path π of the RNN output activations has the same length of the input sequence $\mathbf{C}$. It is clear that the neighbouring column feature vectors might represent the same character. In addition, some column feature vectors may not represent any labels, an additional 'blank' output label is added into the RNN output layer. The repeating labels and empty labels also need to be removed to map to the target word $\mathcal{W}$. For example, $\left('-','a','a','-','-','b','b','b'\right)$ can be mapped to (a,b), where $'\_'$ denotes the empty label. So the $p(\mathcal{W}|\mathbf{C})$ is defined as follows:

$$p(\mathcal{W}|\mathbf{C}) = \sum_{V(\pi)=\mathcal{W}} p(\pi|\mathbf{C}) \tag{3}$$

where V denotes the operator that translates the output path π to target word $\mathcal{W}$. It is worth to note that the translation process V is not unique. $p(\pi|\mathbf{C})$ refers to the conditional probability of output path π given input sequence $\mathbf{C}$, which is defined as follows:

$$p(\pi|\mathbf{C}) = \prod_{t=1}^{L} p\left(\pi_t|\mathbf{C}\right) = \prod_{t=1}^{L} y^t_{\pi_t} \tag{4}$$

where L denotes the length of the output path and π_t denotes label of output path π at time t. The term y^t denotes the network output of RNN at time t, which can be interpreted as the probability distribution of the output labels at time t. Therefore $y^t_{\pi_t}$ denotes the probability of π_t at time t.

The CTC forward backward algorithm [25] is then applied to calculate $p(\mathcal{W}|\mathbf{C})$. The RNN network is trained by back-propagating the gradient through the output layer based on the objective function as defined in Eq. (2). Once the RNN is trained, it can be used to convert a sequential feature vector into a probability matrix. In particular, the RNN will produce a $L\times G$ probability matrix $\mathbf{Y}$ given an input sequence of column feature vector, where L denotes the length of the sequence, and G denotes the number of possible output labels, where the empty label is not included. Fig. 5 shows an example of probability matrix. Each entry of $\mathbf{Y}$ can be interpreted as the probability of a label at a time step. Hence when the lexicon is unavailable, the recognition result can be derived by combining all the labels with the highest probability of each row.

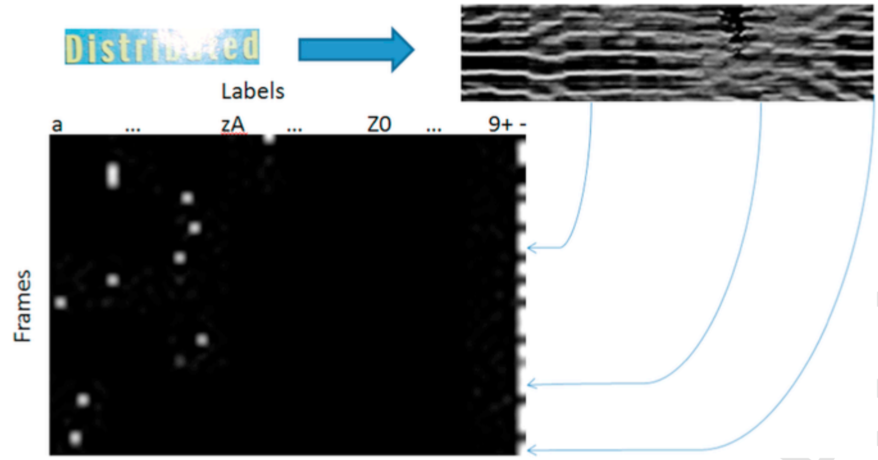

Fig. 5. The output probability matrix of an input word image 'Distributed'. The input word image is first converted into column features. Each frame (column) is fed as the input of the RNNs. The RNNs will generate the output distribution of all the possible labels for each frame. The brightest spot of each row denotes the corresponding label has higher probability.

Compared with the traditional HMM model that generates observations based only on the current hidden state, this RNN approach incorporates the context information including the historical states by using the LSTM structure [25]. We also conduct experiments to compare the performance of HMM and RNN to be discussed in Section 3.4. In addition, the proposed approach does not require explicit labeling of every single column vector of the input sequence. This is very important to the scene text recognition because characters in scenes are often connected, broken, or blurred where the explicit labeling is sometimes nearly an impossible task as illustrated in Fig. 2.

2.3. Ensembling RNNs with lexicon

With a probability matrix $\mathbf{Y}$ and a lexicon $\mathscr{L}$ consisting of a set of possible words, the word recognition can be formulated as searching for the best match word w^* with a highest score. We first calculate a score of each possible word as follows:

$$score_w = p(w|\mathbf{Y}) = \sum_{V(\pi)=w} p(\pi|\mathbf{Y}) \tag{5}$$

where $p(w|\mathbf{Y})$ is the conditional probability of word w given $\mathbf{Y}$. A direct graph can be constructed for the word w so that each node represents a possible label of w. In other words, we need to sum over all the possible paths that can form a word w on the probability matrix $\mathbf{Y}$ to calculate the score of a word w.

A new word w^i can be generated by adding some blank interval into the beginning and ending of w as well as the neighbouring labels of w, where the blank interval denotes the empty label. The length of w^i is $2*|w|+1$, where $|w|$ denotes the length of w. A new $|w^i|\times L$ probability matrix $\mathfrak{P}$ can thus be formed, where $|w^i|$ denotes the length of w^i and L denotes the length of the input sequence. $\mathfrak{P}(m,t)$ denotes the probability of label w^i_m at time t, which can be determined by the probability matrix $\mathbf{Y}$. Each path from $\mathfrak{P}(1,1)$ to $\mathfrak{P}\left(|w^i|,L\right)$ denotes a possible output π of word w, where the probability can be calculated using Eq. (4) as illustrated in Fig. 6.

The problem thus changes to the score accumulation along all the possible paths in $\mathfrak{P}$. It can be solved using dynamic programming. The computational complexity of this algorithm is $O\left(L\cdot|w^i|\right)$.

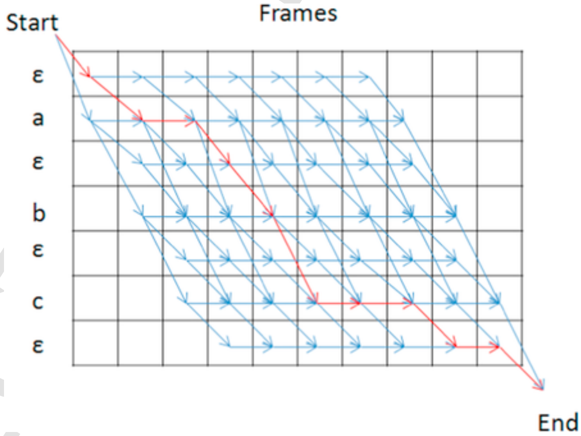

Fig. 6. An illustration of calculation of word score. Take a simple string 'abc' as an example, we first form a m by n matrix, where m denotes the length of string after inserting empty label, n denotes the size of column features. Each cell (i,j) of the matrix denotes the probability of a label i at frame j. The blue arrows show all the possible paths from start to end, while the red arrow denotes a specific path.

If we extract several feature sets with different scales using different parameter settings, there will be more that one trained RNN models. Each RNN model will assign a score to every possible words in the lexicon $\mathscr{L}$. So we can combine the scores given by the two models to obtain the best match word w^* as follows:

$$w^* = \underset{w\in\mathscr{L}}{\operatorname{argmax}} \sum_{i=1}^{n} \left(\alpha^i * score^i_w\right) \tag{6}$$

where $score_w^i$ denotes the assigned score to word w by the ith RNN model as defined in Eq. (5), α^i denotes the weight of each RNN models, which can be determined based its recognition accuracy on the validation dataset as defined in Eq. (7) below.

$$\alpha^i = \frac{acc^i}{\sum_{j=1}^{n} acc^j} \tag{7}$$

where acc^i denotes the classification accuracy of RNN model i on the validating dataset. Basically, the classifier with higher accuracy will be given higher weight.

3. Experiments and discussion

3.1. Experimental protocol

The proposed method has been tested on four datasets, including 1) three ICDAR Robust Reading Competition datasets (ICDAR 2003, ICDAR 2011 and ICDAR 2013) [22,1,2] that consist of scene images captured in different environments, and 2) Google Street View Text dataset (SVT) [13] that mainly consists of images of signboards and shops' names in outdoor environments. Another three datasets are also included for training: ICDAR Born Digital Image Dataset (BDI) [28], Sign Recognition Dataset (SRD) [29] and IIIT5K Dataset [30].

Table 1 shows more details of all these datasets. Since some images appear concurrently in the ICDAR 2003, ICDAR 2011 and ICDAR 2013 datasets, we use only the training images within these three datasets in conjuncted with training images from the other three datasets (i.e. BDI, SRD and IIIT5K) to train a model when testing on the three ICDAR test datasets. For the SVT dataset, we take all the images from other datasets (BDI, SRD, IIIT5K and ICDAR 2003, ICDAR 2011, and ICDAR 2013) together with the SVT train images to train a model and test on the SVT test images. The motivation is to use the publicly available datasets only, so that the ensuing models can be benchmarked in a fair way.

Each image of the dataset contains one word. Our proposed method is evaluated based on the word level recognition accuracy as defined as follows:

$$acc = \frac{\text{Number of Correctly Recognized Word Images}}{\text{Total Number of Word Images}} \tag{8}$$

A word image is considered as correctly recognized only when the recognized word string is the same as the word shown in the image. Characters in a word can be 'A-Z′, 'a-z′, '0–9′ and all other special characters ('+','&', '.', etc).

3.2. Experiment configuration

In the proposed system, we use two sets of parameters as described in Section 2.2:

- Feature Set 32: all cropped word images are normalized to be of the same height, i.e., M=32. The patch size W, the HOG bin number, and the averaging window size T are set to 8, 9, and 5, respectively.
- Feature Set 64: all cropped word images are normalized to be of the same height, i.e., M=64. The patch size W, the HOG bin number, and the averaging window size T are set to 9, 9, and 7, respectively.

We have also tested feature set sizes 16 and 128 when the image size is normalized to 16 and 128 pixel high, respectively. The good feature set size actually depends heavily on the size of images used for training and testing. If the image is small, using a small feature set size could lose textual information and accordingly affects the recognition performance, e.g. the recognition accuracy of feature set 16 on ICDAR03 dataset is only 19%. On the other hand, using a large feature set size, e.g. feature set 128 wont improve the recognition accuracy much but increases the computation costs significantly. We therefore choose feature Set 32 and feature set 64 in our designed system.

Table 1
Information of experimental datasets.

Datasets	# of Training Images	# of Testing Images
ICDAR 2003 [22]	1156	1110
ICDAR 2011 [1]	848	1189
ICDAR 2013 [2]	848	1095
SVT [13]	257	647
IIIT5K	5000	–
BDI	918	–
SRD	215	–

For RNN, the number of input cells is the same as the length of the extracted column feature at 40. The output layer has 64 cells including 62 for characters ([a…z,A…Z,0…9]), one label for special characters ([+,&,$,…]), and one for empty label. The RNN uses 5 hidden layers that have 60, 80, 100, 120 and 140 cells, respectively. We have also tested other hidden layers sizes but the recognition performance is quite similar. On the other hand, the number of hidden layers does affect the recognition performance. In particular, the accuracy drops from 89–82% on the ICDAR03 dataset when the layer number reduces from 5 to 3. Increasing the layer number above 5 instead requires much more computation costs and the trained model also tends to be over-fitted. We therefore choose the current network structure.

3.3. Experimental results on public datasets

We compare our proposed method with several state-of-the-art techniques as shown in Table 2. The compared techniques can be grouped into three categories including 1) **Segmentation based techniques** (markov random field method (MRF) [3], inverse rendering method (IR) [5], nonlinear color enhancement method (NESP) [4]) that segment the text regions from the word images, 2) **Character level recognition techniques** (HMM Maxout model (HMM) [9], HOG based conditional random field method (HOGCRF) [12], CNN model (CNN) [11], Part Based Tree structure method (PBS) [14] [1], Clustering sub-patches of characters method (Strokelets) [15], PhotoOCR [10] and Deep CNN Model (DCNN) [16]) that recognize word images through segmentation and integration of character recognition results and 3) **Word level recognition techniques** (Embedded attributes (AE) [18], Dynamic time warping (DTW) [17], and Whole Word Deep CNN Model(WWDCNN) [19]) that treat each word images as a whole without character segmentation.

To make a fair comparison, we evaluate recognition accuracy on testing data with a lexicon created from all the words in the test set (as denoted by ICDAR03(FULL) and ICDAR11(FULL) in Table 2, as well as with lexicon consisting of 50 random words from the test set (as denoted by ICDAR03(50) and ICDAR11(50) in Table 2 as performed in [17,32,14]. For the SVT dataset, we directly adopt the 50-word lexicon as provided in [13].

Table 2 shows word recognition accuracy of the proposed technique and the compared techniques on ICDAR 2003, ICDAR 2011 and SVT datasets, respectively. Text segmentation methods (MRF [3], IR [5], and NESP [4]) produce lower recognition accuracy than other methods because robust and accurate scene text segmentation is a very challenging task. In addition, the CNN approach performs the best among all the character level recognition methods [9,16,11]. To train a robust CNN character classifier, a large amount of character level training data need to be labeled and synthetic character data is often needed as well.

For the word level recognition methods, our proposed method produces better recognition results compared with the DTW and AE

[1] The accuracy is obtained on 49 classes.

methods. The WWDCNN method performs the best which can be largely attributed to the huge training dataset including 9 million synthetic word images. On the other hand, the deep network model in the WWDCNN method needs to be updated if the lexicon has been changed. Additionally, the WWDCNN method cannot work properly when the lexicon is implicit and the searching space is huge.

Compared with the CNN architecture, our proposed model can perform the word level recognition without requiring the lexicon because it treats each input word image as a sequential signal. Additionally, the character labels are learnt and recognized implicitly during the training and evaluation stages. In addition, to show that the proposed model can perform much better with augmented data, we also train a new text recognition model by adding in a certain part of the newly publicly available data provided by [31] which consists of lots of synthetic text images. Due to the constraints of the computational power, we only incorporate a subset of the dataset with 10,000 randomly selected text images (out of 9 million images available) . As the last row of Table 2 shows, the recognition accuracy is improved with a small subset of the synthetic data compared with that shown in the second last row where the augmented data in [31] is not used.

Based on our study, our proposed model works best when the input word images are more or less horizontal. In fact, one major failure source is due to the severe perspective distortion where words are captured in arbitrary orientations. This limitation could be relieved by perspective/affine rectification which we will investigate in our future work.

In addition, our proposed method obtains a superior word recognition accuracy of 89% for SVT data set as shown in Table 2. The superior performance can be explained by the character-segmentation-free characteristic of our proposed method, because many word images in the SVT dataset are difficult to segment compared with word images in ICDAR datasets as illustrated in Fig. 7. That is why almost all the state-of-the-art techniques perform worse on the SVT dataset as compared with the three ICDAR datasets. At the same time, the superior performance of our proposed technique can also be attributed to the ensembling of the two sets of discriminative visual features, because texts within the SVT dataset often have very different fonts and sizes. The PhotoOCR [10] and WWDCNN [19] also report higher word recognition accuracies (90% and 95% respectively) . As a comparison, our proposed method achieves similar performance and is better in terms of training data size, training time, and computational costs.

Furthermore, we apply our proposed method on the recent ICDAR 2013 Rubust Text Reading Competition dataset [2][2], where 22 algorithms are submitted from 13 research groups. The winning PhotoOCR method [10] makes use of a large multi-layer deep neural network and obtains 83% accuracy on the testing dataset. The WWDCNN [19] also achieves very promising recognition performance (91%) as shown in Table 3. Note that PhotoOCR method does not use lexicon but uses a huge amount of training data including more than 5 million word images. The WWDCNN method also takes advantage of the 9 million synthetic text images. Therefore, a large amount of training data can help to train a better model but the acquisition is often costly and even infeasible under many practical situations. Alternatively, it is often more approachable to leverage on some implicit or explicit lexicon to reduce the searching space and improve the recognition accuracy. As a comparison, our proposed method achieved 90% recognition accuracy when a lexicon with around 1000 words is used as illustrated in the last row of Table 3. The accuracy of our proposed technique without using a lexicon drops to 76%, which is still much higher than other participating methods of the competition as shown in Table 3. The accuracies of our proposed technique on the ICDAR 03, ICDAR 11 and SVT datasets also drop to 72%, 69% and 70%, respectively, when no lexicon is used.

As Tables 2, 3 show, our proposed method does not significantly outperform some state-of-the-art techniques on some dataset. On the other hand, one advantage of our proposed method (beyond the accuracy) is that it is trained on word-level instead of character-level (the annotation is much more time consuming) labeled data. In addition, our method is trained on publicly available datasets which targets to form a baseline for the benchmarking of the ensuing works. As a comparison, many compared methods, e.g. [19] used a much larger amount of training images which are not publicly available.

Last, our system is implemented on Ubuntu 13.10 with 16 GB RAM and Intel 64 bit 3.40 GHz CPU. The training process takes about 1 h on a training set with about 3000 word images. The average time for recognizing a cropped word image is about one second. This speed is comparable with the state-of-the-art techniques, such as PhotoOCR [10], which takes around 1.4 s to recognize a cropped word image. It can be further improved through code optimization and hardware acceleration.

3.4. Discussion

In this subsection, we conduct several experiments to compare the performance of different features and classifiers such as dense SIFT vs HOG and HMM vs RNN. The dense SIFT feature is extracted by using VLFeat[3] with several key parameters set as follows. First, all cropped word images are normalized to the same height 64 pixels. The step size and the bin size (scale) of dense SIFT are set at 2 and 8, respectively. The averaging window size is set at 4. The descriptor length is therefore 128 and the size of column feature is 640.

The HMM is implemented by using Kaldi.[4] The models are HMMs for each of the 63 labels. Each HMM has three emitting states and each state is modeled as a Gaussian Mixure Model (GMM). Experimental results are shown in Table 4. It can be observed that the HOG+RNN approach performs much better than the other two approaches clearly. Compared with the dense SIFT, the HOG feature is more suitable for the scene text recognition task as it captures textual shape information more reliably. The RNN with LSTM also outperforms the traditional HMM method not only for this scene text recognition task but also for the handwriting recognition task [25].

We also investigate the correlation between the lexicon size and the word recognition accuracy. Fig. 8 shows word recognition accuracy of our proposed method over the three ICDAR datasets. As shown in Fig. 8, four lexicon sizes are tested that consist of 5, 10, 20, and 50 words, respectively. The word recognition accuracy keeps improving when the lexicon size becomes smaller.

4. Conclusion

Word recognition under unconstrained conditions is a difficult task and has attracted increasing research interest in recent years. Many methods have been reported to address this problem but there is still a big gap for automatic machine reading of texts in natural scenes. This paper presents a novel scene text recognition system that is based on RNN modeling and ensembling.

[2] http://dag.cvc.uab.es/icdar2013competition

[3] http://www.vlfeat.org/

[4] http://kaldi-asr.org/

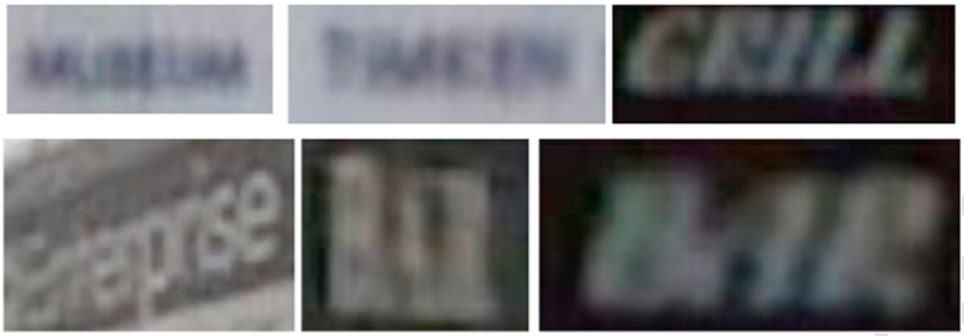

Fig. 7. Some sample images taken from SVT dataset: The text in these images are blurred and heavily touched, where character segmentation is almost an impossible task.

Table 2
Word recognition accuracy on the ICDAR 2003 & 2011 and SVT testing datasets.

Methods	ICDAR 03 (Full)	ICDAR 03 (50)	ICDAR 11 (Full)	ICDAR 11 (50)	SVT
Text Segmentation Techniques					
MRF[3]	0.67	0.69	–	–	–
IR[5]	0.69	0.77	–	–	–
NESP[4]	0.66	–	0.73	–	–
Character Level Recognition Techniques					
PLEX[13]	0.62	0.76	–	–	0.57
HOGCRF[12]	–	0.82	–	–	0.73
PBS (49 classes)[14]	0.79	0.87	0.83	0.87	–
PhotoOCR[10]	–	–	–	–	0.90
CNN[11]	0.84	0.90	–	–	0.70
Strokelets[15]	0.80	0.88	–	–	0.76
HMM[9]	0.89	0.93	–	–	0.74
DCNN[16]	0.92	0.96	–	–	0.86
Word Level Recognition Techniques					
DWT[17]	–	0.90	–	–	0.77
AE[18]	–	–	–	–	0.87
WWDCNN[19]	**0.99**	**0.99**	–	–	**0.95**
Proposed 32	0.84	0.93	0.81	0.90	0.85
Proposed 64	0.85	0.93	0.83	0.90	0.88
Proposed 32+64	0.87	0.94	0.85	0.92	0.89
Proposed 32+64 with partial augmented data by [31]	0.89	0.95	**0.87**	**0.93**	0.91

Table 3
Word recognition accuracy on the ICDAR 13 testing dataset.

Methods	ICDAR 13 (Full)	ICDAR 13 (No Lexicon)
WWDCNN [19]	–	**0.91**
PhotoOCR [10]	–	0.83
NESP [4]	–	0.64
PicRead [2]	–	0.58
Baseline (ABBYY)	–	0.45
Proposed 32+64	0.85	0.70
Proposed 32+64 with partial augmented data by [31]	**0.90**	0.76

Table 4
Recognition Accuracies using different approaches.

Methods	Dense SIFT & RNN	HOG & HMM	HOG & RNN
ICDAR 03(Full)	0.73	0.67	**0.89**
ICDAR 11(Full)	0.72	0.58	**0.87**
ICDAR 13(Full)	0.75	0.68	**0.90**

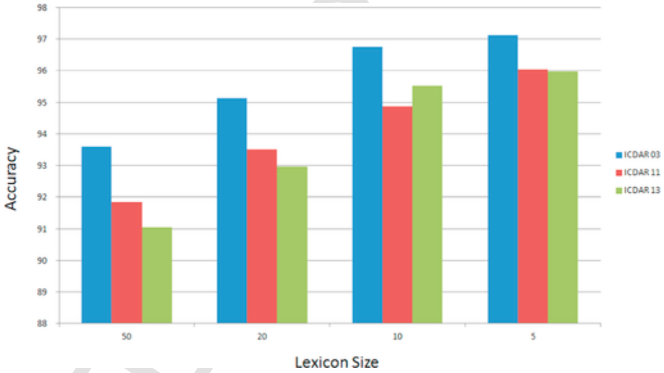

Fig. 8. Word recognition accuracy of our proposed method on ICDAR 03, ICDAR 11 and ICDAR 13 datasets with different lexicon sizes.

Compared with state-of-the-art techniques, our proposed method is able to recognize the whole word images without segmentation. It works by integrating three key novel components. First, it converts a word image into sequential feature vectors and requires no character-level segmentation and recognition. Second, the RNN is introduced and exploited to classify the sequential column feature vectors into word accurately. Third, the proposed model combines two sets of sequential features to produce better results. Experiments on several public datasets show that the proposed technique obtains superior word recognition accuracy. In addition, the proposed technique is trained and tested over several publicly available datasets which could form a good baseline for future benchmarking of other new scene text recognition techniques.

The proposed technique fails typically when texts in scenes are severely curved or suffer from severe perspective distortion as illustrated in Fig. 9. Under such circumstance, each word image as cropped by a perfect rectangle box often includes a large non-text region which introduces a certain amount of noise into the converted sequential feature. The performance of the proposed technique can therefore be improved greatly if a more accurate bounding box can be produced where non-text background can be identified and excluded from the feature extraction. We will look into this issue in our future study.

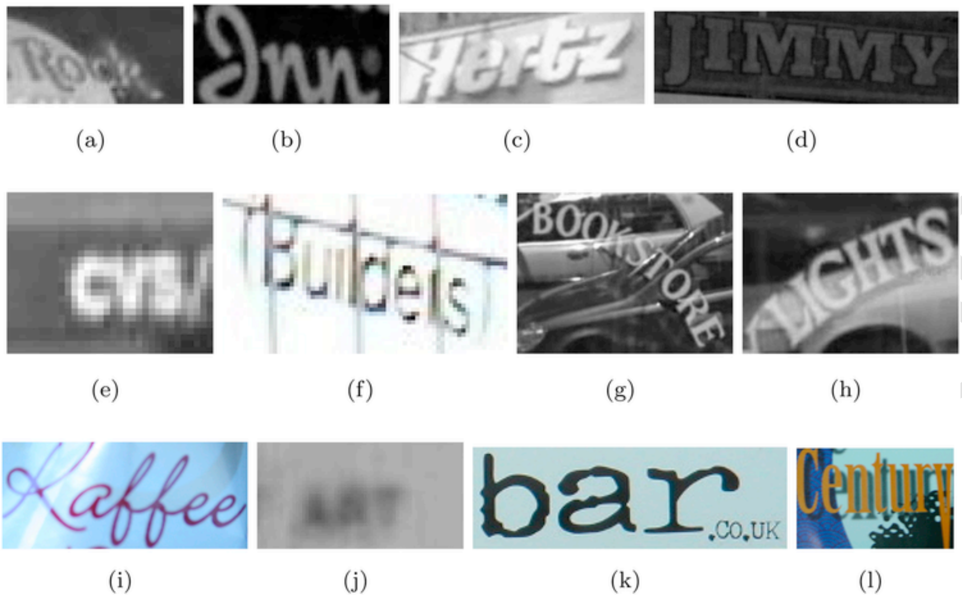

Fig. 9. Examples of word images that fail to be recognized by our method. The recognition results of these images by our proposed method are: a) I, b) SWTW, c) MEAW, d) BMAMAY, e) TWA, f) PUIBEOR, g) MT, h) S, i) Xff9, j) PASN, k) DAr. L, l) Setuy.

References

[1] A. Shahab, F. Shafait, A. Dengel, ICDAR 2011 robust reading competition challenge 2: Reading text in scene images, in: International Conference on Document Analysis and Recognition (ICDAR), 2011, pp. 1491–1496.

[2] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. Gomez i Bigorda, S. Robles Mestre, J. Mas, D. Fernandez Mota, J. Almazan Almazan, L.-P. de las Heras, ICDAR 2013 robust reading competition, in: International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1484–1493.

[3] A. Mishra, K. Alahari, C. V. Jawahar, An MRF model for binarization of natural scene text, in: International Conference on Document Analysis and Recognition (ICDAR), 2011, pp. 11–16.

[4] D. Kumar, M.N. Anil Prasad, A.G. Ramakrishnan, Nesp: Nonlinear enhancement and selection of plane for optimal segmentation and recognition of scene word images, in: SPIE, 2013.

[5] Y. Zhou, J. Feild, E. Learned-Miller, R. Wang, Scene text segmentation via inverse rendering, in: International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 457–461.

[6] G. Myers, R. Bolles, Q.-T. Luong, J. Herson, H. Aradhye, Rectification and recognition of text in 3-d scenes, Int. J. Doc. Anal. Recognit. (IJDAR) 7 (2005) 147–158.

[7] S. Lu, B.M. Chen, C.C. Ko, Perspective rectification of document images using fuzzy set and morphological operations, Image Vis. Comput. 23 (5) (2005) 541–553.

[8] D. Kumar, M.N.A. Prasad, A.G. Ramakrishnan, Benchmarking recognition results on camera captured word image data sets, in: Workshop on Document Analysis and Recognition (DAR), 2012, pp. 100–107.

[9] O. Alsharif, J. Pineau, End-to-end text recognition with hybrid hmm maxout models, International Conference on Learning Representations (ICLR).

[10] A. Bissacco, M. Cummins, Y. Netzer, H. Neven, PhotoOCR: Reading text in uncontrolled conditions, in: International Conference on Computer Vision (ICCV), 2013.

[11] T. Wang, D. Wu, A. Coates, A. Ng, End-to-end text recognition with convolutional neural networks, in: International Conference on Pattern Recognition (ICPR), 2012, pp. 3304–3308.

[12] A. Mishra, K. Alahari, C. Jawahar, Top-down and bottom-up cues for scene text recognition, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 2687–2694.

[13] K. Wang, B. Babenko, S. Belongie, End-to-end scene text recognition, in: International Conference on Computer Vision (ICCV), 2011, pp. 1457–1464.

[14] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, Z. Zhang, Scene text recognition using part-based tree-structured character detection, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2013, pp. 2961–2968.

[15] C. Yao, X. Bai, B. Shi, W. Liu, Strokelets: A learned multi-scale representation for scene text recognition, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 4042–4049.

[16] M. Jaderberg, A. Vedaldi, A. Zisserman, Deep features for text spotting, in: European Conference on Computer Vision (ECCV), 2014, pp. 512–528.

[17] V. Goel, A. Mishra, K. Alahari, C. Jawahar, Whole is greater than sum of parts: Recognizing scene text words, in: International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 398–402.

[18] J. Almazan, A. Gordo, A. Fornes, E. Valveny, Word spotting and recognition with embedded attributes, IEEE Trans. Pattern Anal. Mach. Intell. (2014) 2552–2566.

[19] M. Jaderberg, K. Simonyan, A. Vedaldi, A. Zisserman, Reading text in the wild with convolutional neural networks, Int. J. Comput. Vis. (2015) 1–20.

[20] C. Yao, X. Bai, W. Liu, Y. Ma, Z. Tu, Detecting texts of arbitrary orientations in natural images, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2012, pp. 1083–1090.

[21] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, C. Lim Tan, Text flow: A unified text detection system in natural scene images, 2015, pp. 4651–4659.

[22] S.M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, ICDAR 2003 robust reading competitions, in: International Conference on Document Analysis and Recognition (ICDAR), 2003, pp. 682–687.

[23] B. Su, S. Lu, Accurate scene text recognition based on recurrent neural network, in: Asian Conference on Computer Vision (ACCV), 2014.

[24] A. Graves, J. Schmidhuber., Framewise phoneme classification with bidirectional lstm and other neural network architectures, in: Neural Networks (NN), 2005.

[25] A. Graves, M. Liwicki, S. Fernandez, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, IEEE Trans. Pattern Anal. Mach. Intell. 31 (5) (2009) 855–868.

[26] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.

[27] F.A. Gers, J.A. Schmidhuber, F.A. Cummins, Learning to forget: continual prediction with lstm, Neural Comput. 12 (10) (2000) 2451–2471.

[28] D. Karatzas, S. Mestre, J. Mas, F. Nourbakhsh, P. Roy, ICDAR 2011 robust reading competition - challenge 1: Reading text in born-digital images (web and email), in: International Conference on Document Analysis and Recognition (ICDAR), 2011, pp. 1485–1490.

[29] J. Weinman, E. Learned-Miller, A. Hanson, Scene text recognition using similarity and a lexicon with sparse belief propagation, IEEE Trans. Pattern Anal. Mach. Intell. 31 (10) (2009) 1733–1746.

[30] A. Mishra, K. Alahari, C. V. Jawahar, Scene text recognition using higher order language priors, in: British Machine Vision Conference (BMVC), 2012.

[31] M. Jaderberg, K. Simonyan, A. Vedaldi, A. Zisserman, Synthetic data and artificial neural networks for natural scene text recognition, arXiv preprint arXiv: 1406.2227arXiv:1406.2227.

[32] C.-Y. Lee, A. Bhardwaj, W. Di, V. Jagadeesh, R. Piramuthu, Region-based discriminative feature pooling for scene text recognition, in: Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 4050–4057.

Bolan Su is currently a Research Scientist in the Institute for Infocomm Research, A*STAR, Singapore. He received his B.Sc. degree in computer science in 2008 from Fudan University, Shanghai, China, and his Ph.D. degree in computer science in 2012 from the National University of Singapore, Singapore. His research interests include document image analysis, medical image analysis and computer vision.

Shijian Lu is currently a scientist in Institute for Infocomm Research (I2R), A*STAR Singapore. His current research interests are visual attention, mobile visual analytics, and human machine interaction. Dr Lu is the author/co-author of up to 100 conference/journal papers and over 10 patents. He is currently the head of the Visual Attention Lab in I2R, the co-director of IPAL (a CNRS France-Singapore Joint lab in Singapore), and the Adjunct Assistant Professor in SCSE, NTU. He has won a number of international benchmarking competitions such as DIBCO 2009, 2010, and 2013, Robust Reading Competition 2013, Handwriting Recognition 2014, etc. As the PI, he also got awarded a number grant and industrial projects in the areas of mobile visual analytics, scene text understanding, satellite image analytics, etc.