2017 14th IAPR International Conference on Document Analysis and Recognition

Handwritten Chinese Text Recognition Using Separable Multi-Dimensional Recurrent Neural Network

Yi-Chao Wu[1,2], Fei Yin[1], Zhuo Chen[1,2], Cheng-Lin Liu[1,2]
[1]National Laboratory of Pattern Recognition, Institute of Automation of Chinese Academy of Sciences,
95 Zhongguan East Road, Beijing 100190, P.R. China
[2]University of Chinese Academy of Sciences, Beijing, P.R. China
Email: {yichao.wu, fyin, zhuo.chen, liucl}@nlpr.ia.ac.cn

***Abstract*—The Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) has been demonstrated successful in handwritten text recognition of Western and Arabic scripts. It is totally segmentation free and can be trained directly from text line images. However, the application of LSTM-RNNs (including Multi-Dimensional LSTM-RNN (MDLSTM-RNN)) to Chinese text recognition has shown limited success, even when training them with large datasets and using pre-training on datasets of other languages. In this paper, we propose a handwritten Chinese text recognition method by using Separable MDLSTM-RNN (SMDLSTM-RNN) modules, which extract contextual information in various directions, and consume much less computation efforts and resources compared with the traditional MDLSTM-RNN. Experimental results on the ICDAR-2013 competition dataset show that the proposed method performs significantly better than the previous LSTM-based methods, and can compete with the state-of-the-art systems.**

***Keywords*—handwritten Chinese text recognition; separable multidimensional recurrent neural network; bidirectional LSTM-RNN; WFST-based decoding**

I. Introduction

With the development of information technology, handwritten Chinese text recognition (HCTR) has been widely applied in mail address recognition and electronic commercial affairs, etc.. Although it has been intensively studied in the past forty years, HCTR remains a challenging problem because of the diversity of writing styles, the character segmentation difficulty, large character set and unconstrained language domain.

Currently, most HCTR systems achieve high performance with over-segmentation based method [1], [2], which firstly over-segments the textline into consecutive segments, and then gives the recognition results by integrating character classifier, geometric and linguistic context models. However, the over-segmentation HCTR method mainly suffers two problems. Firstly, the over-segmentation algorithm is not always stable, especially in severe overlapped images, where the recognition accuracy could be badly affected. Secondly, since this framework consists of several independently trained modules, it is relatively difficult to achieve the desired output for the whole system.

Apart from the explicit segmentation based strategy, there also exist many works with the implicit segmentation based frameworks. Hidden Markov Model (HMM) based recognition systems achieve great performance in western languages [3], [4]. However, only a few work has been proposed for HCTR [5]–[7], which may be attributed to the large character set and complex appearances of Chinese characters. Recently, the scheme combining Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) and Connectionist Temporal Classification (CTC) [8] has been widely used in textline recognition [9]–[15]. Graves et al. [9] propose this novel RNN-based approach for the first time, and their recognition results outperformed a state-of-the-art HMM-based system on two large unconstrained handwriting databases. [10] introduces a globally trained offline handwriting recogniser that takes raw pixel data as input, using Multi-Dimensional LSTM (MDLSTM), which is more robust to the distortions of offline text images. Using a more efficient GPU based implementation of MDLSTM by processing the input in a diagonal-wise fashion, Paul et al. [13] explore deeper and wider architectures than previously used for handwriting recognition and outperform state of the art results on two databases with a deep multi-dimensional network. By combining RNN with Convolutional Neural Network (CNN), Shi et al. [15] propose an end-to-end trainable neural network for scene text recognition and demonstrate the superiority of the algorithm over the prior arts.

As both an implicit segmentation based and end-to-end framework, the LSTM-based method can overcome the two defects of over-segmentation methods mentioned above, which have been demonstrated successful by several works for HCTR. Ronaldo et al. [16] present initial results on the use of MDLSTM-RNN in recognizing lines of offline handwritten Chinese text without explicit segmentation of the characters, and their results are comparable in performance with the best reported systems. Li et al. [17] investigate a mixture architecture of deep bidirectional (LSTM) layers and feed forward subsampling layers which is used to encode the long contextual history trajectories for online HCTR, and their system achieves significant improvement. By proposing a multi-spatial-context fully convolutional recurrent network (MC-FCRN), Xie et al. [18] report higher accuracy than the best result reported thus far in the literature. To the best of our knowledge, [16] is the only successful work concerning offline HCTR, however, the performance of their system depends heavily on the performance of the pre-trained models in other languages.

In this study, we propose a handwritten Chinese text recognition method by using Separable MDLSTM-RNN (SMDLSTM-RNN) modules. Compared with the traditional MDLSTM-RNN, SMDLSTM-RNN not only extracts contextual information in various directions for better modeling the

2379-2140/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDAR.2017.22

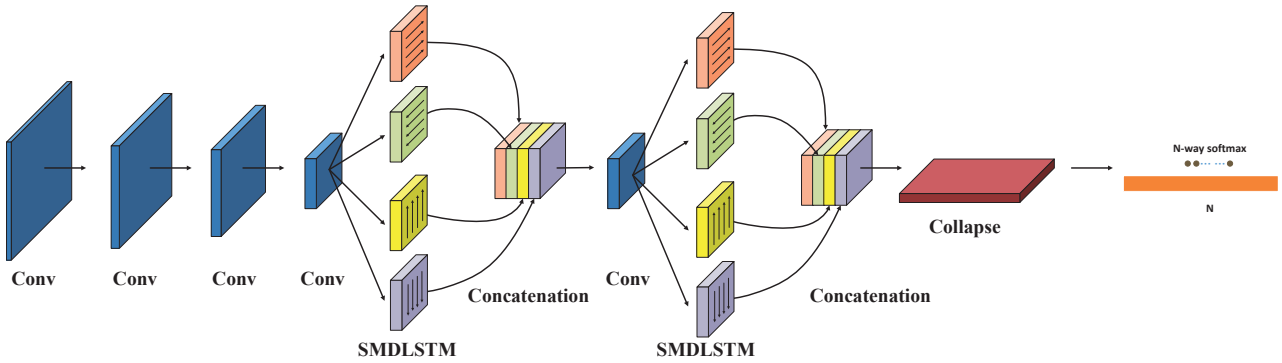


Fig. 1: Diagram of SMDLSTM-RNN based HCTR system.

context, but also consumes much less computation efforts and resources so that we can explore much deeper structures. Experimental results on the ICDAR-2013 competition dataset show that the proposed method performs significantly better than the previous LSTM-based methods, and can compete with the state-of-the-art systems.

The rest of this paper is organized as follows: Section II gives an illustration of the LSTM-based HCTR system, Section III describes the transcription layer and the decoding technique, Section IV presents the experimental results. Finally, the paper is concluded in Section V.

II. Optical Model

A. Proposed Architecture

Similar to the work of [10], [18], we adopt the basic idea of processing the textline image from a small number of simple local features to a large number of complex global features. We build a 7-layer hierarchical structure for modeling sequences of characters, combining the CNN and separable MDLSTM-RNN layers as shown in Fig. 1 (the detailed configuration can been found in Table I).

Firstly, we use convolutional layers to extract the bottom features automatically. The filters of convolutional layers are with a small receptive field 3×3, and all the convolution stride is fixed to one. The number of feature maps is increased from 64 to 256 gradually, while the image is quickly downscaled by spatial pooling to further increase the depth of the network. Then the separable MDLSTM-RNN (SMDLSTM) modules are applied to extract the local context of different directions. After the first SMDLSTM module, one convolution layer is then used to generate a higher-level representation of the local context, and a max-pooling layer is again employed to downscale the input feature maps. After the last MDLSTM layer, the collapse is applied to sum over all the inputs in each vertical line. Finally, the softmax layer is used to perform the classification of the characters (including the 'blank' class). All CNN layers are equipped with the clipped ReLU nonlinearity $\sigma(x) = \min\{\max\{x, 0\}, 20\}$. Batch normalization layers are inserted between different layers to achieve faster convergence and avoid over-fitting. The SMDLSTM module will be introduced in the following.

B. Separable MDLSTM-RNN

Conventional BLSTM was designed to deal with one-dimensional (1D) sequence data. However, for offline HCTR, if transforming the image into 1D sequences, the system would be unable to handle distortions along different dimensions. Therefore, Graves et al. [10] offered a more robust way by using multi-dimensional recurrent neural networks (MDRNNs), which provide recurrent connections along all spatio-temporal dimensions presented in the data. Denote the hidden state for position (u, v) of an MDRNN layer as $h(u, v)$, the previous hidden states along different axes as $h(u-1, v)$ and $h(u, v-1)$, respectively, we obtain the simplified MDRNN feedfoward function in (1):

$$h(u,v) = f(Wx(u,v) + Uh(u-1,v) + Vh(u,v-1) + b), \quad (1)$$

where $x(u, v)$ is the current input, W, U and V are the network wights, b is a bias vector, and $f(\cdot)$ is a nonlinear activation function. Although one such layer is sufficient to give the network access to all context against the scanning direction, it is common practice to use four parallel MDLSTM layers, each of which processes the input in one of the four possible directions, e.g. from the top left to the bottom right, as shown in Fig 2(a). The outputs of the four direction RNNs are then combined, which therefore can receive spatial information

TABLE I: SMDLSTM-based model configuration. The first row is the bottom layer. k, s and p stand for kernel size, stride and padding size, respectively (similarly hereinafter).

Type	Configurations
input	$128(height) \times W$ gray-scale image
Convolution	#maps: 64, k: 3×3, s:1×1, p:1×1
MaxPooling	Window: 2×2, s: 2×2
Convolution	#maps: 128, k: 3×3, s:1×1, p:1×1
BatchNormalization	-
MaxPooling	Window: 2×2, s: 2×2
Convolution	#maps: 256, k: 3×3, s:1×1, p:1×1
MaxPooling	Window: 2×2, s: 2×2
Convolution	#maps: 256, k: 3×3, s:1×1, p:1×1
BatchNormalization	-
MaxPooling	Window: 2×2, s: 1×2
SMDLSTM	#hidden units: 1024
BatchNormalization	-
Convolution	#maps: 512, k: 3×3, s:1×1, p:1×1
BatchNormalization	-
MaxPooling	Window: 2×2, s: 2×1
SMDLSTM	#hidden units: 1024
BatchNormalization	-
Collapse	-
Softmax	#class

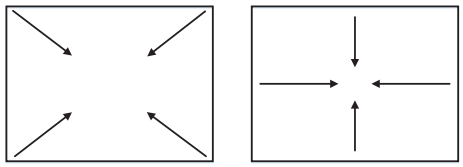
from the whole context.

However, the processing of the original MDLSTM is often of low efficiency, and its capacity has been limited. We adopt the idea that the RNN should capture the context information along all the input dimensions and propose the Separable MDLSTM (SMDLSTM) to replace the origin one. Similar structures have been previously applied to image detection and classification [19], [20] and English words recognition [21], but no work has shown the effectiveness of the this module in HCTR with large character set and complex writing styles.

In SMDLSTM, the conventional LSTM-RNN is used to scan the input data in each direction, i.e., left to right, right to left, bottom to top and top to bottom for offline textline image, which is shown in Fig 2(b). In this way, the MDRNN is actually degraded to four LSTMs as (2).

$$h(u,v) = \begin{cases} f(Wx(u,v) + Uh(u-1,v) + b), & \text{vertical} \\ f(Wx(u,v) + Uh(u,v-1) + b), & \text{horizontal} \end{cases} \tag{2}$$

Each RNN only depends on previous information along horizontal or vertical direction, and all the four RNNs can be processed simultaneously. Moreover, all the rows or columns are considered to be independent and can be computed in parallel for each direction in SMDLSTM, thus can effectively reduce the computation time and resources. The outputs of each RNN are combined (concatenated in this work) to generate a high-level feature expression as well. Compared with the standard MDLSTM, to make the module highly parallel, we do not model the context between different rows or columns in one single SMDLSTM layer, however, when stacking more SMDLSTM or even CNN layers, it is obvious that the whole network can capture more global features including the context from the diagonal directions.



(a) standard (b) separable

Fig. 2: Scanning directions of two MDLSTM.

C. BLSTM-based Model

In addition to the SMDLSTM-based models, we also propose a BLSTM-based model inspired from [17], as shown in Table II. The network architecture consists of two main components, including the convolutional layers and the BLSTM layers. At the bottom of structure, a series of convolutional layers are used to produce a feature sequence. Then three bidirectional LSTM (BLSTM) layers of 512 dimensions are stacked to predict probability distributions for each frame in the feature sequence. Although such kind of BLSTM-based models have achieved great success in scene text recognition [15] and online HCTR [17], as two-dimensional distortions are very common for offline handwritten textline images, it may be insufficient for the BLSTM-based models to capture the 2D information, which will be discussed in experiments.

III. Transcription

Transcription is to convert the output of our optical model into a sequence of character labels. In this work, we adopt the Connectionist Temporal Classification (CTC) [8] layer as our transcription layer.

TABLE II: BLSTM-based model configuration.

Type	Configurations
input	$128(height) \times W$ gray-scale image
Convolution	#maps: 64, k: 3×3, s:1×1, p:1×1
MaxPooling	Window: 2×2, s: 2
Convolution	#maps: 128, k: 3×3, s:1×1, p:1×1
BatchNormalization	-
MaxPooling	Window: 2×2, s: 2×2
Convolution	#maps: 256, k: 3×3, s:1×1, p:1×1
MaxPooling	Window: 2×2, s: 2×2
Convolution	#maps: 256, k: 3×3, s:1×1, p:1×1
BatchNormalization	-
MaxPooling	Window: 2×2, s: 2×2
Convolution	#maps: 512, k: 1×1, s:1×1, p:0×0
Convolution	#maps: 512, k: 4×1, s:4×1, p:0×0
BatchNormalization	-
Convolution	#maps: 1024, k: 2×1, s:2×1, p:0×0
BLSTM	#hidden units: 512
BatchNormalization	-
BLSTM	#hidden units: 512
BatchNormalization	-
BLSTM	#hidden units: 512
BatchNormalization	-
Softmax	#class

CTC maximizes the likelihood of an output sequence by efficiently summing over all possible input-output sequence alignments, and allows the classifier to be trained without any prior alignment between input and target sequences. It uses a softmax output layer to define a separate output distribution $P(k|t)$ at every step t along the input sequence for extended alphabet, including all the transcription labels plus an extra blank symbol which represents an invalid output. A CTC path π is a sequence of length T with blank and label indices. The probability $P(\pi|X)$ is the product of the emission probability at each step:

$$P(\pi|X) = \prod_{t=1}^{T} P(\pi_t|t, X). \tag{3}$$

Since there are many possible ways of separating the labels with blanks, to map from these paths to the transcription, a CTC mapping function B is defined to firstly remove repeated labels and then delete the blank from each output sequence. The conditional probability of an output transcription y can be calculated by summing the probabilities of all the paths mapped onto it by B:

$$P(y|X) = \sum_{\pi \in B^{-1}(y)} P(\pi|X). \tag{4}$$

To avoid direct computation of the above equation, which is computationally expensive, we adopt the forward-backward algorithm [8] to sum over all possible alignments and determine the conditional probability of the target sequence.

A. WFST-based Decoding

Decoding a CTC network means to find the most probable output transcription y for a given input sequence X. It is common sense that we should incorporate language constrains so as to achieve better performance while decoding. In this paper, we adopt the generalized approach based on Weighted Finite-State Transducers (WFSTs) [22], [23] to decode from scratch. A WFST is a finite-state acceptor (FSA) in which each

transition has an input symbol, an output symbol and a weight. A path through the WFST takes a sequence of input symbols and emits a sequence of output symbols. Our decoding method is performed on the Eesen toolkit [24], and we represent the CTC labels, lexicons and language models as the token WFST, lexicon WFST and grammar WFST, respectively.

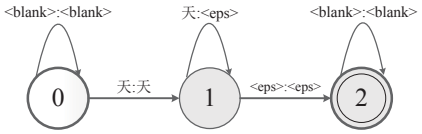


Fig. 3: An example of the token WFST representing the character "天". ⟨blank⟩ stands for the blank token, while ⟨eps⟩ means an empty input or output (similarly hereinafter).

The token WFST (denoted as T) maps a sequence of frame-level CTC labels to a single lexicon unit (character in this study). For a lexicon unit, T is designed to subsume all of its possible label sequences at the frame level. Therefore, this WFST allows occurrences of the blank label, as well as repetitions of any non-blank labels. An example is shown in Fig 3, where all the possible paths are mapped into a singleton lexicon unit "天". The grammar WFST (denoted as G) encodes the permissible word sequences in a language. The WFST symbols are the words (or characters), and the arc weights are the language model probabilities. A toy language model which permits two phrases "天安门" and "天通苑" with G is shown in Fig 4. The lexicon WFST (denoted as L) encodes the mapping from sequences of lexicon units to words. A simple example of L can be shown in Fig. 5.

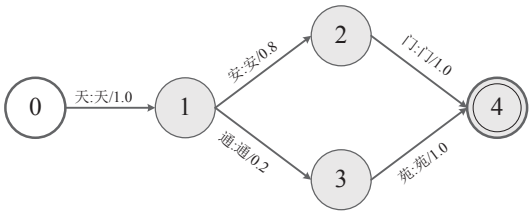


Fig. 4: A toy example of the grammar WFST. The numerical values indicate the transition probabilities of the language model given the previous words.

After compiling the three separate WFSTs, we compose them into a comprehensive search graph. In order to compress the search space and thus speed up decoding procedure, the final WFST TLG is generated by the following order of FST operations:

$$TLG = T \circ min(det(L \circ G)), \tag{5}$$

where $\circ$, det and min denote composition, determinization and minimization respectively. The search graph TLG takes the predictions provided by the optical model as inputs and outputs the recognized sequence of words from a sequence of CTC labels.

IV. Experiments

Our models were implemented on the platform of Torch 7 [25]. We used the CUDA backend and cuDNN v5 accelerated library in our implementation for high performance GPU acceleration. The experiments were performed on a workstation with the Intel(R) Xeon(R) E5-2680v3 2.50GHz CPU, 256GB RAM and four NVIDIA Titan X GPUs.

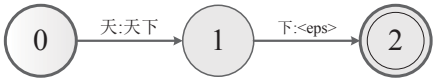


Fig. 5: The lexicon WFST for the word entry "天下".

A. Datasets

We trained our models on the CASIA-HWDB dataset [26], including both unconstrained text lines (HWDB 2.0-2.2) and isolated characters (HWDB 1.0-1.2). We expanded our training set by three means: (1) distort the textline images with techniques such as scaling, shearing and rotating, etc.; (2) randomly shuffle the order of the characters from both string and isolated characters, and form new text lines; (3) synthesize the textline samples with the corpus used to train language models (LMs) [1]. Our models were trained on two datasets of different classes to explore the effects of class number on the recognition accuracies, of which, one has 2,672 classes extracted from the textline samples abbreviated as Train-2672, the other has 7,356 classes abbreviated as Train-7356.

We evaluated the performance of our handwritten Chinese text recognition system on the database from the ICDAR 2013 Chinese Handwriting Recognition Competition [27], abbreviated as ICDAR-2013, which contains 300 test pages of 91,563 characters written by 60 writers who did not contribute to the released CASIA-HWDB database. It should be mentioned that test dataset is a bit smaller than the reported one, we removed the outlier characters not covered by the training data, abbreviated as Test-2672 and Test-7356 respectively. The summary of datasets is listed in Table III. It can be seen that the number of both characters and lines are very close to the standard test set, therefore it is fair to compare with the results of previous works.

TABLE III: Summary of datasets.

Type	#Lines	#Characters
Train-2672	297,106	5,950,411
Train-7356	1,069,678	20,268,321
Test-2672	3,432	89,750
Test-7356	3,432	91,473

B. Implementation Details

For all the textline images, we padded the height of the image to 128 if it is less than that, or proportionally scaled the image to have height 128 if it is larger than that. We carefully designed our system so that the models can be trained on multiple GPUs simultaneously. The image width in same batch were padded to be the same, but could vary between different batches.

The networks were trained with RMSProp [28] with a base learning rate of $5e-4$ and mini-batches of 8 examples. We selected $1/10$ of the training samples for each epoch. During the training, we used the curriculum learning [29] for the first epoch to achieve better convergence, and then transferred to training with random shuffling for the rest epochs. It took about

only 12 and 45 minutes per epoch for 2,672 and 7,356 training sets, respectively. The training can usually be finished after about 160 epochs.

C. Experimental Results

We conducted the experiments with both SMDLSTM-based and BLSTM-based models, each of which were trained on the datasets with 2672 and 7356 classes respectively. Therefore, we abbreviate the optical models in this work as SMDLSTM-2672, SMDLSTM-7256, BLSTM-2672 and BLSTM-7356.

We report the recognition performance using two character-level accuracy metrics following [1], i.e., Correct Rate (CR) and Accurate Rate (AR).

TABLE IV: Recognition Results with the Proposed Models.

Model	AR (%)	CR (%)
SMDLSTM-2672	90.02	90.72
BLSTM-2672	86.77	87.16
SMDLSTM-7356	86.64	87.43
BLSTM-7356	82.97	83.37
[16]	83.5	-

1) Effects of Optical Models: The results of two models with different classes are shown in Table IV. For both 2,672 and 7.356 classes, the SMDLSTM-based models yield better performance than the BLSTM-based models. It is natural that models with fewer classes can achieve better performance, and the accuracy gap between two different structures becomes much more obvious for larger number of classes, which demonstrate the effectiveness of the SMDLSTM structure. Compared with BLSTM-based models which can capture context along only one dimension, the SMDLSTM-based models can process the information from the whole context so that it can better model the textline information. Specially, the SMDLSTM-7356 achieves relatively 21.6% lower than BLSTM-7356 in terms of CER (character error rate, equals $1 - AR$). Furthermore, the model size of SMDLSTM-7356 (18.5M) is smaller than that of the BLSTM-7356 (21.7M). While compared to the standard MDLSTM-based framework [16] which was also trained with more than 7,000 classes, it is obvious that SMDLSTM-based model is significantly better than [16], and moreover, we did not use any other languages to pre-train the model.

2) Effects of Language Models: We trained back-off language models (BLMs) [30] of different orders on a corpus containing about 50 million characters [1], which is the same as those in [1], [2]. The effects of different combinations are shown in Table V (for saving space, we only list the AR).

TABLE V: AR (%) Using Different Language Models.

Model	N-gram order				
	2	3	4	5	8
SMDLSTM-2672	91.87	92.51	92.59	92.59	**92.61**
BLSTM-2672	89.02	89.99	90.07	90.08	90.08
SMDLSTM-7356	89.40	90.26	90.37	90.37	90.38
BLSTM-7356	85.13	86.38	86.49	86.51	86.51
[16]	88.0	89.3	89.4	89.1	-
[1]	-	89.28	-	-	-
[2]	-	-	-	**95.04**	-

Compared with Table IV, it is obvious that the performance of all our systems can be improved by BLMs. When comparing with LMs of different orders, it can be found that the improvement from bigram to trigram is remarkable, the improvement from trigram to higher order LMs (4-, 5- and 8-gram) is only marginal. This can be attributed to the data sparseness problem, which affects higher order LMs more evidently and cancels off the benefit of higher order LMs. The result of SMDLSTM-7356 with the similar BLMs is better than [16], but the accuracy gap becomes smaller, which can be attributed to the corpus-based samples we supplemented. This phenomenon implies that the LSTM-based models may be prone to be overfitting the linguistic context. The SMDLSTM-2672 with 8-gram BLM can achieve the AR of 92.61%, which is a great improvement compared with the previous baseline [1] of ICDAR-2013. However, it is still not as good as the work of [2], which improves HCTR using both neural network language models and convolutional neural network shape models under the over-segmentation framework. This implies that there still could be large room for improvement with LSTM-based models.

3) Results on the Single Character Datasets: To further investigate the intrinsic of LSTM-based model, we conducted the textline recognition experiments on the two isolated character datasets, where there should be no linguistic context. We extracted one from the Test-7356 of 1,381 classes (Test-1381), the other is the standard isolated character set of ICDAR-2013, which contains 224,419 samples of 3,755 classes (abbreviated as Test-3755). Each single character is considered as a textline, and then we can give the AR and CR metrics as well. The recognition results are shown in Table VI. The Test-1381 has exactly the same character as Test-7356, however, both SMDLSTM-7356 and BLSTM-7356 perform worse compared with Table IV. The result on Test-7356 shows significant differences between the two optical models. The recognition accuracy of SMDLSTM-7356 falls to an acceptable range, while we observe a severe deterioration for BLSTM-7356 with only 62.63% of AR. The experiment demonstrates the superiority of SMDLSTM in modeling the character information, however, it also indicates both models may suffer the context overfitting problem as well.

TABLE VI: Results on the Single Character Datasets.

Model	Test-1381		Test-3755	
	AR (%)	CR (%)	AR (%)	CR (%)
SMDLSTM-7356	84.15	86.19	80.09	81.52
BLSTM-7356	81.97	83.26	62.63	64.04

D. Error Analysis

We show some examples of text line recognition by SMDLSTM-7356 in Fig. 6, which reveal several factors causing recognition errors.The severe skew or slant and scribble writing textlines are not easy to give the correct results as shown in Fig. 6(a) and 6(b), although we supplemented some distortion samples. Apart from that, English letters/words and the punctuations are prone to error recognition, which is shown in Fig. 6(c).

V. Conclusion

In this paper, we propose a handwritten Chinese text recognition method by using Separable MDLSTM-RNN (SMDLSTM-RNN) modules, which extract contextual information in various directions, and consume much less computation efforts and resources compared with the traditional

MDLSTM-RNN. Experimental results on the ICDAR-2013 competition dataset show that the proposed method performs significantly better than the previous LSTM-based methods, and can compete with the state-of-the-art systems. It is our future work to solve the context overfitting problem to further improve the performance of our system.

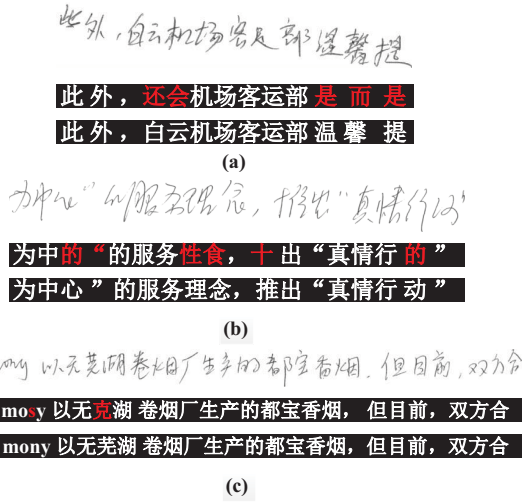


Fig. 6: Error recognition samples. For each example, the first row is the text line image, the second row is the result with SMDLSTM-7356 using 8-gram BLM, third row is the transcript (ground-truth).

ACKNOWLEDGMENTS

We would like to thank Ronaldo Messina for the valuable discussion. This work has been supported by the National Natural Science Foundation of China (NSFC) grants 61573355 and 61633021.

REFERENCES

[1] Q.-F. Wang, F. Yin, and C.-L. Liu, “Handwritten Chinese text recognition by integrating multiple contexts,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 34, no. 8, pp. 1469–1481, 2012.

[2] Y.-C. Wu, F. Yin, and C.-L. Liu, “Improving handwritten Chinese text recognition using neural network language models and convolutional neural network shape models,” *Pattern Recognition*, vol. 65, pp. 251 – 264, 2017.

[3] U.-V. Marti and H. Bunke, “Using a statistical language model to improve the performance of an hmm-based cursive handwriting recognition system,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 15, no. 1, pp. 65–90, 2001.

[4] S. Espana-Boquera, M. J. Castro-Bleda, J. Gorbe-Moya, and F. Zamora-Martinez, “Improving offline handwritten text recognition with hybrid HMM/ANN models,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 33, no. 4, pp. 767–779, 2011.

[5] T.-H. Su, T.-W. Zhang, D.-J. Guan, and H.-J. Huang, “Off-line recognition of realistic Chinese handwriting using segmentation-free strategy,” *Pattern Recognition*, vol. 42, no. 1, pp. 167–182, 2009.

[6] B. Feng, X. Ding, and Y. Wu, “Chinese handwriting recognition using hidden markov models,” in *Proc. 16th ICPR*, vol. 3, pp. 212–215, 2002.

[7] J. Du, Z.-R. Wang, J.-F. Zhai, and J.-S. Hu, “Deep neural network based hidden markov model for offline handwritten Chinese text recognition,” in *Proc. 23rd ICPR*, pp. 3417–3422, 2016.

[8] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *Proc. ICML*, pp. 369–376, 2006.

[9] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, “A novel connectionist system for unconstrained handwriting recognition,” *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 855–868, 2009.

[10] A. Graves and J. Schmidhuber, “Offline handwriting recognition with multidimensional recurrent neural networks,” in *Proc. NIPS*, pp. 545–552, 2009.

[11] A. Ul-Hasan, S. B. Ahmed, F. Rashid, F. Shafait, and T. M. Breuel, “Offline printed urdu nastaleeq script recognition with bidirectional lstm networks,” in *Proc. 12th Int. Conf. on Document Analysis and Recognition*, pp. 1061–1065, 2013.

[12] F. Simistira, A. Ul-Hassan, V. Papavassiliou, B. Gatos, V. Katsouros, and M. Liwicki, “Recognition of historical greek polytonic scripts using lstm networks,” in *Proc. 13th Int. Conf. on Document Analysis and Recognition*, pp. 766–770, 2015.

[13] P. Voigtlaender, P. Doetsch, and H. Ney, “Handwriting recognition with large multidimensional long short-term memory recurrent neural networks,” in *Proc. 15th ICFHR*, pp. 228–233, 2016.

[14] B. Su and S. Lu, “Accurate scene text recognition based on recurrent neural network,” in *Proc. ACCV*, pp. 35–48, 2014.

[15] 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 Trans. Pattern Analysis and Machine Intelligence*, 2016.

[16] R. Messina and J. Louradour, “Segmentation-free handwritten Chinese text recognition with lstm-rnn,” in *Proc. 13th Int. Conf. on Document Analysis and Recognition*, pp. 171–175, 2015.

[17] L. Sun, T. Su, C. Liu, and R. Wang, “Deep lstm networks for online Chinese handwriting recognition,” in *Proc. 15th ICFHR*, pp. 271–276, 2016.

[18] Z. Xie, Z. Sun, L. Jin, H. Ni, and T. Lyons, “Learning spatial-semantic context with fully convolutional recurrent network for online handwritten Chinese text recognition,” *arXiv preprint arXiv:1610.02616*, 2016.

[19] S. Bell, C. Lawrence Zitnick, K. Bala, and R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” in *Proc. CVPR*, pp. 2874–2883, 2016.

[20] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. Courville, and Y. Bengio, “Renet: A recurrent neural network based alternative to convolutional networks,” *arXiv preprint arXiv:1505.00393*, 2015.

[21] Z. Sun, L. Jin, Z. Xie, Z. Feng, and S. Zhang, “Convolutional multi-directional recurrent network for offline handwritten text recognition,” in *Proc. 15th ICFHR*, pp. 240–245, 2016.

[22] M. Mohri, F. Pereira, and M. Riley, “Weighted finite-state transducers in speech recognition,” *Computer Speech & Language*, vol. 16, no. 1, pp. 69–88, 2002.

[23] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, *et al.*, “The kaldi speech recognition toolkit,” in *Proc. ASRU*, pp. 1–4, 2011.

[24] Y. Miao, M. Gowayyed, and F. Metze, “Eesen: End-to-end speech recognition using deep rnn models and wfst-based decoding,” in *Proc. ASRU*, pp. 167–174, 2015.

[25] R. Collobert, K. Kavukcuoglu, and C. Farabet, “Torch7: A MATLAB-like environment for machine learning,” in *Proc. NIPS Workshop of BigLearn*, no. EPFL-CONF-192376, 2011.

[26] C.-L. Liu, F. Yin, D.-H. Wang, and Q.-F. Wang, “CASIA online and offline Chinese handwriting Databases,” in *Proc. 11th Int. Conf. on Document Analysis and Recognition*, pp. 37–41, Sept 2011.

[27] F. Yin, Q.-F. Wang, X.-Y. Zhang, and C.-L. Liu, “ICDAR 2013 Chinese handwriting recognition competition,” in *Proc. 12th Int. Conf. on Document Analysis and Recognition*, pp. 1464–1470, 2013.

[28] T. Tieleman and G. Hinton, “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude,” *COURSERA: Neural Networks for Machine Learning*, vol. 4, no. 2, 2012.

[29] J. Louradour and C. Kermorvant, “Curriculum learning for handwritten text line recognition,” in *Proc. Int. Workshop on DAS*, pp. 56–60, 2014.

[30] A. Stolcke, “Srilm - an extensible language modeling toolkit,” in *Proc. INTERSPEECH*, pp. 901–904, 2002.