A New Approach for Integrated Recognition and Correction of Texts from Images

Jian Wei[2], Kai Chen[1,2], Jianhua He[3], Zheng Huang[1,2], Yunrui Lian[4], Yi Zhou[2]
[1]Shanghai Jiaotong University, China.
[2]Onlyou Artificial Intelligence Institute, China.
[3]Aston University, UK.
[4]Xiamen First High School, China.
Corresponding Author: Kai Chen, kaichen@onlyou.com.

***Abstract*—Automatic recognition and error correction of texts from images are critical for many commercial applications such as receipt recognition, which have very high accuracy requirements. In this paper we propose an integrated image based text recognition and correction approach to improve accuracy. There are two levels of text recognition and correction integration in the proposed approach. Firstly, a beam search strategy is designed to generate a set of text candidates, based on the probability distribution of text prediction outcomes from a deep learning recognition model. Then a word-level lexicon check is applied to select only one from the candidate text sentences, which has the highest prediction probability among those with all words present in the lexicon. Jointly the beam search and lexicon check can effectively correct some recognition errors. Secondly, an encoder-decoder language model based corrector is developed to correct potential recognition errors in the selected output texts that fail the lexicon check. Training samples for the corrector are created from the recognition outcomes and can be expanded by associating multiple text candidates with one image label. We conduct experiments on ICDAR'13 and CH10K datasets to evaluate the proposed approach and the impact of these two levels of integration on accuracy. Experiment results show that the proposed approach outperforms the existing one with higher recall and much higher recognition accuracy through effective exploitation of joint recognition and correction design.**

I. Introduction

Recognition of texts from images (e.g. scanned documents and photos) plays a critical role for many applications, such as driving navigation, streamlining document-intensive processes and office automation. With performance greatly boosted by recent breakthroughs in deep learning technologies, text recognition is becoming mature for many practical tasks (such as name card recognition, license plate recognition and hand-written text recognition). However, for many commercial applications, they may have very higher accuracy requirements than the general text recognition tasks. For example, the required accuracy for text recognition in scanned receipts is larger than 99.9%. The complexities of real scenarios such as poor image quality, variable text fonts, background noises make the tasks even more challenging. Improving text recognition accuracy performance is particularly important and challenging for these applications with high accuracy requirements.

Two general approaches can be used to improve text recognition system accuracy, which are development of text recognizer with higher accuracy and application of post-recognition error correction. The main task of a recognizer is to extract text sentences from image segments, which may be cropped from a whole image after text detection and localization processes. Popular text recognizers are based on convolutional neural networks (CNN) deep learning models, including the Convolutional Recurrent Neural Network (CRNN) model [2] and the attentional sequence-to-sequence model [5] [6]. In this paper, we will focus on the CRNN model, which has CNN and recurrent neural network (RNN) modules. The main task of a text corrector is to correct the potential recognition errors in the recognition process. It is noted that, in the existing systems the text recognition and correction processes are separated without interaction between them. For example, with the CRNN model the recognizer produces a matrix of probabilities for text prediction, with a column of probabilities corresponding to the candidate characters (including a blank label) for each character location. The recognized text for the image segment could be simply determined by selecting the characters with the highest prediction probability at each character location and removing the blank labels and consecutively repeating characters. Then the recognized text is presented to the corrector to remove some potential recognition errors. It is noted that the recognizer and the corrector are sequentially connected but isolated, which may lead to performance loss and limit the achievable overall system accuracy.

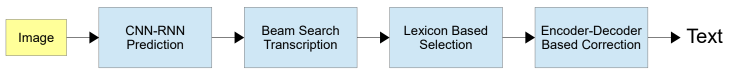


Fig. 1. Operations in the proposed approach.

In view of the potential drawback of the existing text recognition systems, we propose an integrated text recognition and correction approach, in which the recognition and correction processes are integrated to improve accuracy. The idea comes from an observation that in many cases the highest ranked texts from the recognizers in terms of prediction probability are not the correct texts. Therefore more candidate texts from the recognition process should be presented for better correction to improve accuracy. With the integrated approach there are four major operation stages as shown in Fig. 1, namely, prediction stage, transcription stage, selection stage and correction stage.

- In the prediction stage the CNN-RNN model is used for feature extraction and sequence modeling. The RNN layer captures the contextual semantic information and predicts

the character probabilities for each time-step of the text sequences.
- Beam search is used in the transcription stage to generate text candidates which are sorted by recognition scores [1]. In each time-step, beam search iteratively extends and merges beams (i.e. text candidates). Every beam is scored by incorporating RNN output and ranked in a descending order.
- In the selection stage, each candidate is checked separately on if all the words in the text are presented in a lexicon. One text with the highest prediction probability among those passing the word-level lexicon check is selected, which can help correct certain errors produced by the baseline recognizer.
- Finally, an encoder-decoder RNN language model based corrector is adopted to correct the possibly erroneous texts. We generate training samples for the corrector from the recognized candidate texts of real images instead of using manual edited texts.

Experiments are conducted on ICDAR 13 and CH10K benchmarks to evaluate the proposed integrated framework. Compared with the baseline approach for text recognition and correction, the proposed method achieved higher recall and much higher recognition accuracy with effective exploitation of joint recognition and correction design.

II. Related Work

In recent years, there has been much work on text sequence recognition. [2] presented a novel neural network architecture CRNN, which combines feature extraction, sequence modeling with transcription. To train the network, [2] adopted the negative log-likelihood of the conditional probability defined in CTC [3] as the objective. With CTC only the character sequences in images need to be labeled, without need to specify the exact position of the individual characters. The state-of-the-art OCR system [4] also applied CTC in the decoder to generate final output label. An end-to-end neural network model was proposed in [5], which consists of a Spatial Transformer Network (STN) and a Sequence Recognition Network (SRN). STN focuses on text irregularities and rectifies images via a flexible Thin-Plate Spline transformation. SRN is an attentional sequence-to-sequence model that predicts a character sequence directly from the rectified image. [6] extends [5] with revised network architecture and achieves a large performance improvement. In [7] a focusing network is added to adjust attention in the recognition network.

Recent research on text correction were mainly focused on language model in the field of Natural Language Processing (NLP). Broadly speaking there are two kinds of language models in the literature, statistical language model (SLM) and recurrent neural network language model (RNNLM). SLM is based on the n-gram frequencies derived from a large corpus and is used to estimate the probability of a sequence of text units (characters, words, phrases, and so on) at a specific position. [8] applied SLM to decide whether a correction should be made in context. SLM was integrated in [9]–[11] to decode neural network output in the context of text recognition. With great success in speech recognition [12] and machine translation [13], RNNLM was also applied to text correction. [14] investigated the effect of RNNLM, with the aim of improving handwritten Chinese text recognition. Two novel RNNLM based approaches were introduced in [15] for Post-OCR Error Correction.

Compared with the above existing works, our proposed approach has some unique and desirable properties: 1) closely integrated recognition and correction processes; 2) wider search space with multiple candidates to ensure high recall of recognized output; 3) Incorporating lexical constraints and prior knowledge about the language in the selection stage, which is not limited to the pre-defined lexicon and could be generalized to new words; 4) The approach is general and can be applied to different languages and domains.

III. The Integrated Recognition and Correction Network Architecture

As introduced in Section I, the proposed approach has four sequential stages of operations as shown in Fig. 1: 1) CNN-RNN based prediction, 2) Beam Search based transcription, 3) Lexicon based selection and 4) Encoder-Decoder language model based correction. Briefly speaking, the prediction stage applies CNN-RNN model to predict probability distributions for each frame of the text sequence in images. In the transcription stage, beam search is used to generate top N best label candidates which are sorted by recognition scores, where N is a configurable number. Then the selection stage selects the most probable candidate based on prior lexical knowledge. The correction stage corrects potential errors in the selected text by using a trained encoder-decoder language model. It is noted that the selection stage has certain error correction power and is closely integrated to the first two stages. Next more details on the operation of these four stages will be presented.

A. *CNN-RNN Model based Prediction Stage*

In the CNN-RNN recognition model, CNN is the base network acting as feature extractor. In this paper, we use ResNet-34 as the CNN base network. After raw images being fed into the CNN network, feature maps are produced. The highest level of feature maps are reshaped to feature sequences, which is used as the input to the RNN with Map-to-Sequence operation [2]. Specifically, Map-to-Sequence is to flatten each column of the feature maps into a sequence from left to right.

On top of the CNN layer, the extracted feature sequences are forwarded to the RNN layer for sequence modeling. In order to capture contextual information of text from both forward and backward, bidirectional lstm (BLSTM) is used as the RNN unit. Following a two-layer BLSTM module with 256 memory blocks, a fully-connected layer with a softmax classifier is adopted to predict character distributions for each time-step of the output sequence. Consequently, the output of the RNN is a matrix holding the values of per-frame character prediction probabilities. Here we denote the prediction probability matrix as P_{pred} of size $N_C \times T$, where N_C is the size of the corresponding alphabet C and T is the sequence length. Noted that the alphabet contains N_C-1 characters and one $blank$ label. CTC loss [3] is then used to train this CNN-RNN model.

B. Beam Search Transcription Stage

The task of transcription stage is to generate the final label sequence given the probability matrix P_{pred} by RNN. A straightforward and fast way is to pick the most likely character π_t with the highest probability $y^t_{\pi_t}$ at each time-step t and concatenate them into a path $\pi = (\pi_1, \pi_2, ..., \pi_T)$, then apply a collapsing function β defined in [3] to remove duplicate characters and all blanks. This transcription method is called best path search. The whole procedure can be denoted as:

$$b^* = \beta(\text{argmax}_\pi p(\pi|\text{y})), \tag{1}$$

where b^* is the selected label sequence, $\text{y} = (y^1_{\pi_1}, y^2_{\pi_2}, ..., y^T_{\pi_T})$ and $p(\pi|\text{y}) = \prod_{t=1}^{T} y^t_{\pi_t}$ is the product of the character probabilities on the path π.

Obviously, best path search does not necessarily guarantee that a selected path π^* still has the highest probability after collapsing, as there may be multiple paths that could be collapsed into the same label sequence. Mathematically, the probability of a label sequence b is the sum of probabilities of all corresponding paths.

$$p(b|\text{y}) = \sum_{\beta(\pi)=b} p(\pi|\text{y}). \tag{2}$$

Therefore we can revise the optimization objective as:

$$b^* = \text{argmax}_b(p(b|\text{y})). \tag{3}$$

To find a solution to the above optimization problem, we choose beam search instead for transcription. Beam search is an approximation of a breadth-first search that iteratively extends text candidates (called $beams$) and scores them through the tree of possible paths over all time-steps.

The pseudo-code for the beam search method is shown in Algorithm 1 below. Firstly, the algorithm starts with an empty set of beams and a corresponding score. At each time-step, beam search ranks the beams according to their scores from the previous time-step and maintains the top N_{bw} ones, where N_{bw} is a configurable number, representing the beam width. For each of these beams, the score at the current time-step is calculated. Then each beam is extended by all characters in the alphabet and a score is re-calculated.

At the end of the last iteration, we have a set of ranked beams. The ranked beams can be forwarded to the selection stage for further processing. If the selection stage is not used, then the top one is returned as the final result for the recognition process.

At time-step t, the score $p(b, t)$ of a general beam b is defined as the sum of probability of all the corresponding paths ended with $blank$ and non-$blank$ characters:

$$p(b, t) = p_b(b, t) + p_{nb}(b, t). \tag{4}$$

When a beam is extended by a possible character, two cases are considered in the calculation of the score. If the extended character c is a $blank$ or a repetition of the last character in the beam before extension whose paths ended with non-$blank$, the extended beam remains unchanged following the collapsing strategy. Under this case, we have the following respective formula for score updating:

$$p_b(b, t) = p_b(b, t) + p(b, t-1) \cdot P_{\text{pred}}(blank, t), \tag{5}$$

$$p_{nb}(b, t) = p_{nb}(b, t) + p_{nb}(b, t-1) \cdot P_{\text{pred}}(b[-1], t). \tag{6}$$

On the other hand, if the extended character c is a non-repeating non-$blank$ character from $b[-1]$, or c is the same with $b[-1]$ and the corresponding paths are ended with $blank$, the extended beam will be changed with the following respective updated score:

$$p_{nb}(b + c, t) = p_{nb}(b + c, t) + p(b, t-1) \cdot P_{\text{pred}}(c, t), \tag{7}$$

$$p_{nb}(b + c, t) = p_{nb}(b + c, t) + p_b(b, t-1) \cdot P_{\text{pred}}(c, t). \tag{8}$$

More details of beam search can be found in [1].

Algorithm 1 Beam Search.

```
Input: $P_{\text{pred}}$, $N_{\text{bw}}$, $C$
Output: $b^*$
 1: $B = \{\emptyset\}$;
 2: $p(\emptyset, 0) = 1$;
 3: for $t = 1, 2, ..., T$ do
 4:     $BS = sort(B)$;
 5:     $B = \{\}$;
 6:     for $i = 1, 2, ..., N_{\text{bw}}$ do
 7:         $b = BS[i]$;
 8:         compute $p(b, t)$;
 9:         $b \rightarrow B$;
10:         for $c \in C$ do
11:             $b' = b + c$;
12:             compute $p(b', t)$;
13:             $b' \rightarrow B$;
14:         end
15:     end
16: end
17: $b^* = \text{argmax}_b(p(b, T))$;
18: Return $b^*$
```

A potential issue of beam search is the computation complexity, which is $\mathcal{O}(T \cdot N_{\text{bw}} \cdot N_C \cdot log(N_{\text{bw}} \cdot N_C))$ [1]. N_C could be very large for some morphologically-rich languages such as Chinese. Therefore we decrease the search space by extending the beams within the top M characters with the highest probabilities instead of all characters from the alphabet in each time-step. With this processing there will be a slight accuracy loss, but considerable amount of computation will be saved. According to experiments the highest ranked beam usually equals the ground truth label, but this is not always true. In some challenging situations the scores of the top beams are close to one another, indicating the recognition model not able to distinguish them confidently. While the recognition process can stop here with the highest ranked beam as the recognition output, we keep the top N beams as recognized text candidates and send them to the next selection stage to improve the recall and accuracy performance.

C. Lexicon Based Selection Stage

In this stage, we proposed a lexicon-based check method to choose the right text output b^* from the given set of candidates

($cands$) produced by the transcription stage. Algorithm 2 shows the pseudo-code of the selection process.

Before the selection, a large corpus publicly available for the considered language needs to be collected. Specifying the corpus belongs to the similar thematic domain with our recognition task gives more relevant semantic context information and will improve the selection performance. A lexicon of unique words is built by word-level segmentation of the corpus. For example, a standard space delimiter tokenizer can be used in English corpus.

During the selection process, all candidates are checked separately in their score-ranked order. If a candidate is in sentence format, the same tokenizer is used to segment the candidate into words (W). For each word produced from the candidate, we look up the lexicon and check if the word is present in the lexicon. If a processed word is not presented in the lexicon, it is called a mismatched word. If all the words in one candidate are present in the lexicon, this candidate is returned as the result and the selection stage stops.

It must be noted that all the candidates in the set may fail to pass the lexicon check, which means there exists at least one mismatched word in each candidate. This can happen as the texts to be recognized in the wild are not necessarily lexicon-constrained. Therefore a correct candidate could also be blocked in our check due to the unseen words that do not appear in the lexicon. In this case, we choose the highest ranked candidate in the given candidate set as the output of the selection stage.

Algorithm 2 Lexicon Selection.

```
Input: $cands$, $lexicon$
Output: $b^*$
1: for $cand \in cands$ do
2:     $W = segment(cand)$;
3:     if $W \subseteq lexicon$ then
4:         $b^* = cand$;
5:         Return $b^*$;
6:     end
7: end
8: $b^* = cands[1]$;
9: Return $b^*$;
```

D. Encoder-Decoder Based Correction Stage

As discussed on the selection process, the candidate with the highest score is chosen as the selection stage output when all the candidates from the given set have at least one mismatched word. However, the selected text in this case (corresponding to line 9 in Algorithm 2) is more likely inaccurate compared with other texts that pass the lexicon check (line 5 in Algorithm 2). We thus propose an additional stage to correct potential errors of the selected text b^* in this section, aiming to further improve the system accuracy.

The corrector used in this stage is based on a character-level encoder-decoder RNN language model, which considers the selected text from the selection stage as the source sequence and the corrected text as the target sequence. The specific architecture of the corrector is shown in Fig. 2.

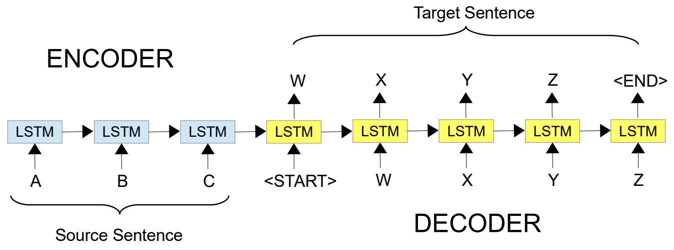


Fig. 2. The encoder-decoder language model architecture.

It is worth noting that the corrector has strong connection to the recognizer. In one hand, to train this model, a training set is created with the outputs from the selection stage of our recognizer and the training images. We will refer to a recognizer output and the ground truth as a pair of a training sample. On the other hand, the training set can be expanded by applying beam search to obtain multiple candidates for one training image. For one image, each candidate paired with the same ground truth label can form a new training sample. The approach of generating the training dataset is different from that used in the general correctors, where training samples are generated manually without any connection to the recognizer. We believe our approach is better than the manual approach as it directly correlates with the recognizer and could allow the corrector to learn the hidden bias in recognition.

After the corrector is trained, the selected text (b^*) from selection stage can be input to the corrector and a corrected text (denoted by b^{**}) is output. One potential issue for a corrector is false positive, which means an incorrect text b^{**} is output for a correct text b^*. Such issue will have large impact on the reliability of our corrector. To avoid false positives, we proposed a candidates-based supervision scheme to be used following the correction process. As we know b^{**} may introduce some new characters with regard to b^*, we use the scheme to check if each new character appears in any other candidates. b^* is still output as the final recognition result after the correction stage if one of the new characters that does not appear in any other candidate. We will replace b^* with b^{**} only under the condition that all the new characters in the text b^{**} appear at least once in other candidates.

IV. Experiments

A. Datasets

Experiments are run to evaluate the effectiveness of the proposed integrated approach and measure the impact of the different processing stages used in the approach. We use three datasets in our experiments, including ICDAR 2013 (IC13) dataset, Synthetic Word (Synth90K) dataset and CH10K dataset.

IC13 [16]: This dataset contains 848 images in the training set and 1095 images in the test set. All the images are extracted at the word-level from real scene images. We exclude the images that contain non-alphanumeric characters from the training and test sets.

Synth90K [17]: This dataset has 9 million images, covering 90 thousand English words. It is synthetically generated and used for our recognizer's pretraining and lexicon creation in IC13.

CH10K: We collect 10 thousand images including real Chinese receipts, invoices, tickets from various domains (tax, transportation, finance, accounting, healthcare, etc). We select 90% images randomly for training and the remaining are used for testing. For each training image, the regions enclosing single sentence text are cropped and ground truth data is added for the regions. For each test image, a commercial text detection model we trained is applied to localize the text regions. Finally 108900 text images were created for the training set and 8661 text images were created for the test set. Each text image is associated with one text sentence.

B. Metrics

In our experiments recognition accuracy, running time and character error rate (CER) are chosen as the metrics of interests for performance evaluation of the proposed method and various combination of the processing stages. Recognition accuracy is computed as the ratio of the number of correctly recognized texts to the number of tested samples. Running time is computed as the average time for recognizing text from a single input image. CER measures the difference between a text and its ground truth in terms of Levenshtein edit distance with the following formula:

$$CER = \frac{N_d}{N_{gt}}, \tag{9}$$

where N_d denotes the minimal number of operations including substitutions, insertions and deletions required to transform the text to its ground truth and N_{gt} is the total number of characters in the ground truth.

C. Implementation Details

During test process, each text image is input to the recognizer first and the top N candidates are obtained with beam search. The configureable parameters N_{bw}, M and N are all set to 10 by default. Then a lexicon-based selection is applied to select the best text prediction. Specifically, the lexicon with the IC13 dataset is created from the 90 thousand English words in the Synth90K dataset and the tokenizer is a standard space splitter. The lexicon with the CH10K dataset is built from the text annotations of CH10K training images and the tokenizer is jieba, which is a well-performing Python Chinese word segmentation tool. The final step is to correct the selected candidate with our trained encoder-decoder language model.

The experiments are run on MXNet [18] deep learning platform with an Intel Xeon E5-2665 2.40GHz server, which has 8 CPU cores, 64 GB memory and a Nvidia GeForce GTX 1080Ti GPU.

D. Experimental Result

We evaluate the proposed approach and the combination of processing stages over the IC13 and CH10K datasets. Table. I reports the accuracy performance. All the evaluated combinations of the processing stages include the prediction stage with the CNN-RNN model. The result of correction stage in IC13 is not presented because our corrector is applied to sentence-level samples. The proposed approach is compared to the baseline approach which uses the best path search strategy following the prediction stage.

TABLE I
OVERALL RECOGNITION ACCURACIES (%) ON IC13 AND CH10K TEST SETS.

Prediction	Best path search	Beam search	Selection	Correction	IC13	CH10K
✓	✓				82.8	60.0
✓		✓			82.9	60.0
✓		✓	✓		**86.6**	71.1
✓		✓	✓	✓	-	**72.9**

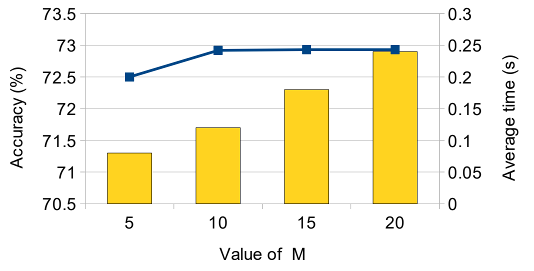
According to our experiment results, it is observed that the best performing method is the combination of the beam search transcription, selection and correction, which improves the absolute accuracy by 3.8% with IC13 dataset and 12.9% with CH10K dataset over the baseline approach. Applying beam search and directly returning the highest ranked candidate shows negligible improvement over the baseline approach. But a considerable improvement is obtained (3.7% in IC13 and 11.1% in CH10K) when beam search and selection stages are both used. In addition, the correction stage could boost the accuracy by 1.8% in CH10K dataset. These results clearly demonstrate the gains of integrating the recognition and the correction processes and potentials of improving accuracy for highly demanding text recognition applicaitons.

To further evaluate the effectiveness of the corrector, we compare the CERs with and without the correction stage, computed by the CER of b^* and b^{**} in the CH10K test set, respectively. According to the experiment results, the CER decreases from 6.98% to 3.21%, which confirms the effectiveness of the proposed corrector.

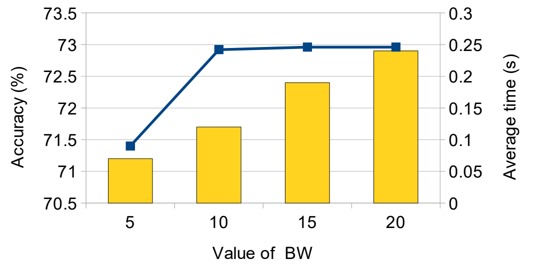
Lastly we conduct experiments on CH10K dataset by adjusting the settings of parameters M, N_{bw} and N to examine their impact on accuracy and running time. When one parameter is investigated, the others are set to their default values. Fig. 3 shows the impact of each parameter on accuracy and average running time. In general the running time increases linearly with increased value of these parameters. Parameter N has a relatively smaller impact on running time than others. There is about 0.06 seconds increase of running time when N changes from 5 to 20, while the corresponding time is about 0.17 seconds for the same change on both N_{bw} and M. On the other hand, recognition accuracy increases with increased value of the parameters up to 10 and then saturates. For example, accuracy is 71.4% with N_{bw} = 5, 72.9% with N_{bw} = 10 and 73.0% with N_{bw} = 20. The results indicate that the default setting of 10 for these parameters are good enough for the given recognition tasks. It is also observed that there is only a slight accuracy drop (0.4%) by changing M from 10 to 5, but it could save 0.04 seconds for running time. Considering the trade-off between running time and accuracy performance for practical applications, we recommend to set M to 5.

V. CONCLUSION

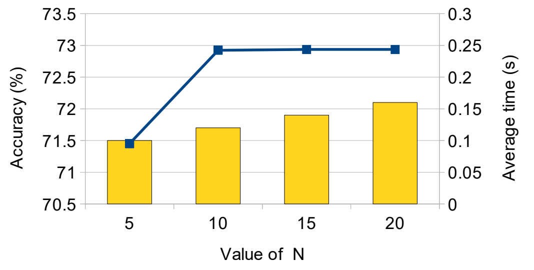
In this paper we proposed an integrated approach of text recognition and correction from images, which are particularly important for the text recognition applications with high



(a)



(b)



(c)

Fig. 3. Performance comparison with varying parameter values on recognition accuracy (blue line) and average running time (yellow bar)

accuracy requirements. In the proposed approach the recognition and correction processes are tightly integrated. It has four operation stages, including CNN-RNN prediction, beam search transcription, lexicon selection and encoder-decoder RNN model correction. The CNN-RNN network predicts the per-frame character probabilities through time-steps. Beam search iteratively extends and scores possible beams in each time-step and generates a set of top ranked candidates at last. Among the set of candidates, a pre-defined lexicon is adopted to check the correctness of them and select the best one. The recognized candidate texts and the corresponding ground truths of the real images are taken as the training samples for corrector. This enables a specifically customized corrector that could learn the hidden inclination of the recognizer and maintain a low level of false positives with a candidates-supervised scheme. We conducted experiments on ICDAR 13 and CH10K benchmarks that belong to different languages and domains. Experiment results demonstrated the superior performance over the conventional baseline approach with significant recognition accuracy improvement and the large potential gains of the integrated recognition and correction design.

Acknowledgment

This project has received funding from the European Unions Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 824019.

References

[1] H. Scheidl, S. Fiel and R. Sablatnig, "Word Beam Search: A Connectionist Temporal Classification Decoding Algorithm," *In 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 253-258, 2018.

[2] B. Shi, X. Bai and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE transactions on pattern analysis and machine intelligence*, 39(11), 2298-2304, 2017.

[3] A. Graves, S. Fernndez, F. Gomez and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," *In Proceedings of the 23rd International Conference on Machine learning*, 369-376, 2006.

[4] F. Borisyuk, A. Gordo and V. Sivakumar, "Rosetta: Large scale system for text detection and recognition in images," *In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 71-79, 2018.

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

[6] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao and X. Bai, "Aster: An attentional scene text recognizer with flexible rectification," *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[7] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu and S. Zhou, "Focusing attention: Towards accurate text recognition in natural images," *In International Conference on Computer Vision*, 5086-5094, 2017.

[8] I. Kissos, and N. Dershowitz, "Ocr error correction using character correction and feature-based word classification," *In 12th IAPR Workshop on Document Analysis Systems*, 198-203, 2016.

[9] K. Chen, L. Tian, H. Ding, M. Cai and Q. Huo, "A Compact CNN-DBLSTM Based Character Model for Online Handwritten Chinese Text Recognition," *In 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Vol. 1, 1068-1073, 2017.

[10] Y.C. Wu, F. Yin, Z. Chen and C.L. Liu, "Handwritten Chinese Text Recognition Using Separable Multi-Dimensional Recurrent Neural Network," *In 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Vol. 1, 79-84, 2017.

[11] Q.F. Wang, F. Yin and C.L. Liu, "Integrating language model in handwritten Chinese text recognition," *In 10th International Conference on Document Analysis and Recognition (ICDAR)*, 1036-1040, 2009.

[12] X. Chen, T. Tan, X. Liu, P. Lanchantin, M. Wan, M.J. Gales and P.C. Woodland, "Recurrent neural network language model adaptation for multi-genre broadcast speech recognition," *In 16th Annual Conference of the International Speech Communication Association*, 2015.

[13] G. Klein, Y. Kim, Y. Deng, J. Senellart and A.M. Rush, "Opennmt: Open-source toolkit for neural machine translation," *arXiv preprint* arXiv:1701.02810, 2017.

[14] 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*, 65, 251-264, 2017.

[15] K. Mokhtar, S.S. Bukhari and A. Dengel, "OCR Error Correction: State-of-the-Art vs an NMT-based Approach," *In 13th IAPR International Workshop on Document Analysis Systems*, 429-434, 2018.

[16] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L.G.I. Bigorda, S.R. Mestre and L.P. De Las Heras, "ICDAR 2013 robust reading competition," *In 12th International Conference on Document Analysis and Recognition (ICDAR)*, 1484-1493, 2013.

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

[18] MXNet. http://mxnet.incubator.apache.org/.