Evaluation of Neural Network Language Models in Handwritten Chinese Text Recognition

Yi-Chao Wu, Fei Yin, Cheng-Lin Liu
National Laboratory of Pattern Recognition (NLPR)
Institute of Institute of Automation,Chinese Academy of Sciences
Beijing, China
{yichao.wu, fyin, liucl}@nlpr.ia.ac.cn

***Abstract*—Handwritten Chinese text recognition based on over-segmentation and path search integrating contexts has been demonstrated successful, where language models play an important role. Recently, neural network language models (NNLMs) have shown superiority to back-off N-gram language models (BLMs) in handwriting recognition, but have not been studied in Chinese text recognition system. This paper investigates the effects of NNLMs in handwritten Chinese text recognition and compares the performance with BLMs. We trained character-level language models in 3-, 4- and 5- gram on large scale corpora and applied them in text line recognition system. Experimental results on the CASIA-HWDB database show that NNLM and BLM of the same order perform comparably, and the hybrid model by interpolating NNLM and BLM improves the recognition performance significantly.**

Keywords—handwritten Chinese text recognition; higher order character language model; neural network language model; hybrid language model

I. Introduction

Handwritten Chinese text recognition has been intensively studied in the past forty years. However, it remains a challenging problem due to the diversity of writing styles, the character segmentation difficulty, large character set and unconstrained language domain. The framework of the recognition method based on over-segmentation by integrating character classifier, geometric and linguistic context models has been demonstrated successful in handwritten string recognition [1], among which the linguistic context model (i.e., language model) is of great importance.

Statistical language models, which give the prior probability of a sequence of words, play an important role in many applications such as character and speech recognition, machine translation and information retrieval, etc. Although back-off N-gram language models (BLMs) were proposed more than twenty years ago [2],[3] and have been used in handwritten text recognition for more than ten years, they are still considered as a favorable choice. A system for the reading of totally unconstrained handwritten text is presented in [4], which combines a hidden Markov model with a statistical language model to improve the performance. By combining with multiple models including BLMs, effective approaches for both off-line Chinese handwriting recognition [1] and on-line Chinese handwriting recognition [5] obtained encouraging performance. Bissacco et al. [6] adopted a two-level language model with a 8-gram character model and a 4-gram word model in a system called PhotoOCR for text recognition in camera-based images, which outperformed all previously reported results on public benchmark datasets.

Generally, higher order language models can capture longer context patterns so as to estimate the sequence probability more accurately. Carpenter [7] found that the performance of character N-gram can be significantly improved until 8-gram, given sufficient training samples. However, traditional BLMs suffer from the data sparseness problem, as the number of parameters increases exponentially with the length of the context (i.e., the curse of dimensionality), preventing these models from estimating context stably.

Recently, a new type of language model called neural network language model (NNLM), also known as continuous space language model (CSLM) in [8], has been proposed to overcome the data sparseness based on a continuous representation of the words [9]. Inspired by that work, many extensions of NNLMs and related algorithms have been proposed, which either aim to improve the model performance [10],[11] or to reduce time complexity [12],[13]. NNLMs, which are complementary to standard N-gram models, have been successfully applied in speech recognition [8],[11], statistical machine translation [14],[15], and English off-line handwriting recognition [16]. However, to the best of our knowledge, NNLMs have never been evaluated in handwritten Chinese text recognition.

In this study, we evaluate the effects of three types of character-level language models (since it is relatively difficult to incorporate higher order word-level LMs into our system), say, higher order BLMs, NNLMs and hybrid language models (HLMs), under the general integrated segmentation-and-recognition framework. Experimental results on the CASIA-HWDB database show that NNLM and BLM of the same order perform comparably, and HLM by interpolating NNLM and BLM improves the recognition performance significantly.

The rest of this paper is organized as follows: Section II gives an overview of the handwritten Chinese text recognition system, Section III describes the neural network language models, Section IV presents the experimental results. Finally, the paper is concluded in Section V.

II. System Overview

The diagram of our handwritten Chinese text recognition system is shown in Fig. 1. First, the input text line image is over-segmented into a sequence of primitive image segments using the method of [17] (Fig. 2(a)). Then, consecutive segments are combined to generate candidate character patterns, forming a segmentation candidate lattice as shown in Fig. 2(b). After that, each character pattern in a sequence is classified to assign several candidate character classes, and all the candidate patterns in a candidate segmentation path generate a character

978-1-4799-1805-8/15/$31.00 ©2015 IEEE

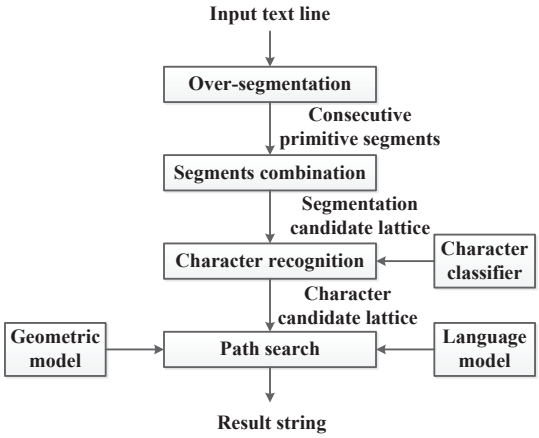

Fig. 1. Diagram of handwritten Chinese text recognition system.

candidate lattice, which is shown in Fig. 2(c). The rest of the task is to find the optimal path with minimum cost or maximum score.

We denote a sequence of candidate characters as $X = x_1...x_m$. Each candidate character is assigned candidate class (denoted as c_i) by a character classifier, and then the result of string recognition is a character string $C = c_1...c_m$. We adopt the path evaluation criterion presented in [1] which formulates string recognition from the view of Bayesian decision and integrates multiple contexts including character classification, geometric context [18] and linguistic context. For saving space, we give the criterion directly and more details can be found in [1].

Denote the score of classifying character x into class c as $P(c|x)$. The linguistic context is given by an N-gram language model, denoted as $P(c_i|h_i)$, where h_i denotes the history of c_i and an N-gram model only considers the N-1 history characters (see Section III). The unary class-dependent geometric (**ucg**) score, unary class-independent geometric (**uig**) score, binary class-dependent geometric (**bcg**) score and binary class-independent geometric (**big**) score are denoted as $P(c_i|g^{ucg})$, $P(z_i^p = 1|g^{uig})$, $P(c_{i-1}, c_i|g^{bcg})$, and $P(z_i^g = 1|g^{big})$, respectively, where g denotes corresponding geometric feature and the output scores are given by geometric models classifying on features extracted. We obtain a log-likelihood function $f(X, C)$ for the segmentation-recognition path:

$$f(X,C) = \sum_{i=1}^{m}(w_i \cdot \log P(c_i|x_i) + \lambda_1 \cdot \log P(c_i|g_i^{ucg}) + \lambda_2 \cdot \log P(z_i^p = 1|g_i^{uig}) + \lambda_3 \cdot \log P(c_{i-1}, c_i|g_i^{bcg}) + \lambda_4 \cdot \log P(z_i^g = 1|g_i^{big}) + \lambda_5 \cdot \log P(c_i|h_i)), \quad (1)$$

where w_i is the word insertion penalty which is used to overcome the bias to short strings, for which we use Weighting with Character pattern Width (WCW) [1] in the system, λ_1-λ_5 are the weights to balance the effects of different models optimized with Maximum Character Accuracy (MCA) criterion [1]. Via confidence transformation, the six models, namely, one character classifier, four geometric models and one character linguistic model, are combined to evaluate the segmentation paths. As for path search, a refined frame-synchronous beam search algorithm [1] is used to retain a limited number of partial paths with maximum scores at each frame, and finally, gives a number of global paths of maximum score.

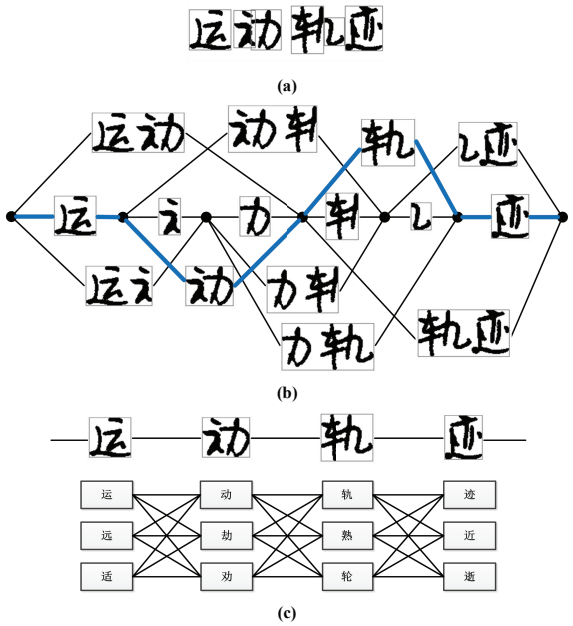

Fig. 2. (a) Over-segmentation of a text line; (b) Segmentation candidate of (a); (c) Character candidate lattice of the thick path in (b).

III. Neural Network Language Models

In this section, we present an alternative way of using the N-gram statistics by training and using NNLMs. If the sequence C contains m characters, $p(C)$ can be decomposed by

$$p(C) = \prod_{i}^{m} p(c_i|c_1^{i-1}), \quad (2)$$

where $c_1^{i-1} =< c_1, ..., c_{i-1} >$ denotes the history of character c_i. An N-gram model only considers the N-1 history characters in (2):

$$p(C) = \prod_{i=1}^{m} p(c_i|c_{i-N+1}^{i-1}) = \prod_{i=1}^{m} p(c_i|h_i), \quad (3)$$

where $h_i = c_{i-N+1}^{i-1} =< c_{i-N+1}, ..., c_{i-1} >$ (h_1 is null).

A. Basic Models

In order to attack the data sparseness problem, NNLMs were proposed to project the words into a continuous space to perform an implicit smoothing and estimate the probability of a sequence. Both the projection and estimation can be jointly performed by a multi-layer neural network [9]. The basic architecture of the NNLM with one hidden layer is illustrated in Fig. 3.

NNLMs are statistical N-gram models, and the inputs are the N-1 previous characters h_i, while the outputs are the posterior probabilities of all words in the vocabulary:

$$p(c_i = \omega_j|h_i) \qquad \forall j \in [1, V], \quad (4)$$

where V is the size of the vocabulary. Since we use character-level language model in our system, vocabulary consists characters as "words". Each input character is initially encoded using the "1-of-V" scheme. After training, each column of the $P \times V$ dimensional projection matrix corresponds to the

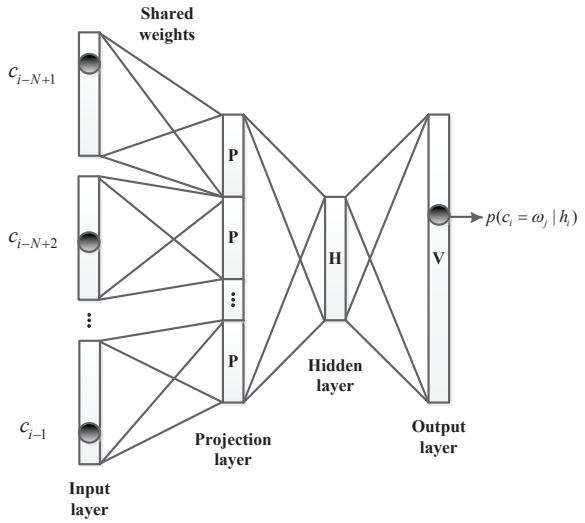

Fig. 3. Architecture of the NNLM with one hidden layer. P is the size of one projection, and H, V are the sizes of the hidden and output layer, respectively.

distributed representation of a word in the vocabulary denoted as $\boldsymbol{r}$, since all the weights of the projection layer are shared.

Using the column-major form, denote the weights between the projection layer and the hidden layer as $\boldsymbol{W}_{P,H}$, the N-1 history concatenate distributed representations $[\boldsymbol{r}_{i-N+1}^T, ..., \boldsymbol{r}_{i-1}^T]^T$ as $\boldsymbol{R}$, and the vector of the hidden layer biases as $\boldsymbol{B}_H$, then the hidden layer activities $\boldsymbol{D}_H$ can be computed as:

$$\boldsymbol{D}_H = \tanh(\boldsymbol{W}_{P,H} * \boldsymbol{R} + \boldsymbol{B}_H), \tag{5}$$

where $\tanh(\cdot)$ is the hyperbolic tangent activation function performed element wise. The neural network predicts all the words in the vocabulary using the following functions:

$$\boldsymbol{M} = \boldsymbol{W}_{H,O} * \boldsymbol{D}_H + \boldsymbol{B}_O, \tag{6}$$

$$\boldsymbol{O} = \exp(\boldsymbol{M}) / \sum_{i=1}^{V} \exp(m_i), \tag{7}$$

where $\boldsymbol{W}_{H,O}$ is the weight matrix of output layer, $\boldsymbol{B}_O$ is the vector of the output layer biases, $\boldsymbol{M}$ is the vector of the activation values calculated before softmax normalization, m_i is the ith element of $\boldsymbol{M}$. The $\exp(\cdot)$ function as well as the division function are performed element wise. After the above operations, the jth component of $\boldsymbol{O}$, denoted as o_j, corresponds to the probability $p(c_i = \omega_j | h_i)$.

The standard back-propagation algorithm is used in training to minimize the regularized cross entropy criterion:

$$E = -\sum_{j=1}^{V} t_j \log o_j + \beta(\|\boldsymbol{W}_{P,H}\|_2^2 + \|\boldsymbol{W}_{H,O}\|_2^2), \tag{8}$$

where t_j denotes the desired output, which should be 1.0 for the next character in the training sentence, and 0.0 for all the others.

B. Acceleration

Although many systems achieved success with the help of NNLMs, which have a much higher complexity than BLMs, it is not trivial to speed up both training and testing steps when using NNLMs. Since the softmax operation dominates the processing time, we adopt the method called short-list introduced by [15] to simplify the model at little loss of system performance.

The original short-list method was proposed in [8], where Schwenk chose to limit the output of the neural network to the s most frequent words, $s \ll V$, referred to as a short-list. Then the probability can be formalized as:

$$\hat{P}(c_i|h_i) = \begin{cases} \hat{P}_N(c_i|h_i, L) \cdot P_B(h_i|L), & \text{if } \mathrm{c_i} \in \mathrm{short-list} \\ \hat{P}_B(c_i|h_i), & \text{otherwise} \end{cases} \tag{9}$$

where $\hat{P}_N$ denotes the probability of characters in the short-list calculated by NNLMs, $\hat{P}_B$ is the probability given by standard BLMs, the random variable L defines the event that the word to be predicted is in the short-list, and $P_B(h_i|L)$ is given by:

$$P_B(h_i|L) = \sum_{c_i \in \mathrm{short-list}} \hat{P}_B(c_i|h_i). \tag{10}$$

In our system, we further simplify the above model by following [15]. One extra output neuron is added for all words that are not in the short-list, whose probability is learned by the neural network, but not used to renormalize the output distribution. We simply assume that it is sufficiently close to the probability mass reserved by the BLM. In general, (9) can be modified as:

$$\hat{P}(c_i|h_i) = \begin{cases} \hat{P}_N(c_i|h_i), & \text{if } \mathrm{c_i} \in \mathrm{short-list} \\ \hat{P}_B(c_i|h_i), & \text{otherwise} \end{cases} \tag{11}$$

It has been observed that there is no significant difference between the methods with and without renormalization [15].

IV. Experimental Results

We evaluated the performance of our handwritten Chinese text recognition system on the CASIA-HWDB [19] test set containing $1,015$ pages. The system was implemented on a desktop computer of Intel Core i5-2400 3.10 GHz CPU, programming using C++ in Microsoft Visual Studio 2008.

A. Experimental Setup

To make the comparison fair, we applied the same character classifier and geometric context models trained on CASIA-HWDB as in [1] to Chinese text line recognition.

The character classifier was trained on 4,198,494 isolated character images of 7,356 classes from isolated characters and unconstrained texts. It extracts gradient direction features from gray-scale images using the method of normalization cooperated gradient feature (NCGF) [20]. The obtained 512D feature vector was reduced to 160D by Fisher linear discriminant analysis (FLDA), and then input into the Modified Quadratic Discriminant Function (MQDF) [21] classifier. We used 4/5 samples of the training character set for training classifiers, and the remaining 1/5 samples for confidence parameter estimation.

As for the geometric context models [18], we extracted geometric features from 41,781 text lines of training text pages for parameter estimation of the corresponding four models (**ucg**, **uig**, **bcg**, and **big**).

All the language models were trained on a text corpus containing about 50 million characters, which is the same as that in [1]. In addition, we collected a development set containing 3.8 million characters from the People's Daily corpus [22] and ToRCH2009 corpus [23], to verify the trained language models.

We report the recognition performance using two character-level accuracy metrics following [24]: Correct Rate (CR) and Accurate Rate (AR):

$$CR = (N_t - D_e - S_s)/N_t,$$
$$AR = (N_t - D_e - S_s - I_e)/N_t, \quad (12)$$

where N_t is the total number of characters in the transcript of test strings. The numbers of substitution errors (S_e), deletion errors (D_e) and insertion errors (I_e) are calculated by aligning the recognition result string with the transcript by dynamic programming.

B. Comparison of Language Models

In the following, we present and discuss the recognition performance using the proposed language models. It should be noted that **cls**, **g**, **cti**, **cfour** and **cfive** denote character classifier (MQDF), the union of all geometric models, the character trigram language model, the character 4-gram language model, and the character 5-gram language model, respectively. In addition to the AR and CR, we also evaluated the perplexity (PPL) of language models on the development set.

1) Higher Order Back-off Language Models: We trained 4-gram and 5-gram BLMs with the SRI Language Model (SRILM) toolkit [25] with the default smoothing technique (Katz smoothing) and entropy-based pruning. The thresholds of the pruning for both character 4-gram and 5-gram are set empirically as 10^{-7}. We also trained a trigram BLM using the same parameters as in [1] to make sure that the realization of our system is correct.

TABLE I. EFFECTS OF HIGHER ORDER BLMS. TIME DENOTES THE RECOGNITION TIME ON ALL THE TEST PAGES.

Combination	AR (%)	CR (%)	Time (h)	PPL
cls+cti [1]	89.03	90.24	11.03	-
cls+cti+g [1]	90.20	90.80	11.73	-
cls+cti	89.05	90.26	8.08	82.97
cls+cti+g	90.21	90.81	8.25	82.97
cls+cfour	89.08	90.25	8.01	73.72
cls+cfour+g	90.23	90.82	8.49	73.72
cls+cfive	89.08	90.26	8.17	73.09
cls+cfive+g	90.23	90.82	8.50	73.09

The effects of different combinations are shown in Table I. Obviously, the performance of our system and the one in [1] give nearly the same performance with **cti**, which verifies the correct realization. It is observed that both **cfour** and **cfive** yield slight improvements compared with **cti**, but there is little difference between **cfour** and **cfive** because of the data sparseness problems, especially when the order of the language models gets higher. In summary, the capability of higher order BLMs trained on our training corpus is saturated. We also notice that it takes less time to recognize all the test images even combined with more complex models, which may attribute to the increase in hardware power.

2) Neural Network and Hybrid Language Models: The NNLMs were trained with a free software called CSLM toolkit [26], which provides full supports for short-list and GPU implementation, both convenient and efficient. We trained NNLMs on GPU of NVIDIA Tesla C2075, and also used Intel Math Kernel Library (MKL) to speed up the matrix operations of the neural network. The model with the lowest perplexity on the development set was chosen as the final one.

To explore the full potential of NNLMs, we compared three types of NNLMs as shown in Table II, which were all trained with batch size of 128 examplers, weight decay 10^{-7}, and 20 iterations. The short-list of NNLM-1 covers all the characters in the training corpus, NNLM-2 and NNLM-3 use a smaller short-list. NNLM-1 and NNLM-2 have two hidden layers, while NNLM-3 has only one hidden layer.

TABLE II. THREE TYPES OF NNLMS.

Type	Projection Size	Hidden Layer Size	Short-list Length	Initial Learning Rate
NNLM-1	320	1024×512	8330	0.06
NNLM-2	320	1024×512	1023	0.06
NNLM-3	320	512	1023	0.10

As Schwenk pointed out that the neural network is never used alone for large vocabulary tasks [8], it is a common practice to linearly interpolate an NNLM with a standard BLM for further improvement, which we call a hybrid language model (HLM). The weights of this linear combinations were computed by the compute-best-mix-tool from SRILM toolkit, minimizing the perplexity on the development set. Corresponding to the three types of NNLMs, we have three hybrid models HLM-1, HLM-2 and HLM-3. It should be pointed out that all the BLMs used in this subsection are trained with the same parameters mentioned above.

TABLE III. EFFECTS OF NNLMS AND HLMS.

Language Type	Combination	AR (%)	CR (%)	Time (h)	PPL
BLM	cls+cti[1]	89.03	90.24	11.03	-
	cls+cti+g [1]	90.20	90.80	11.73	-
NNLM-1	cfive	-	-	-	68.60
HLM-1	cls+cfive+g	**90.69**	**91.24**	143.44	59.44
NNLM-2	cfive	-	-	-	71.64
HLM-2	cls+cfive+g	90.51	91.09	27.23	63.03
NNLM-3	cls+cti	88.84	90.08	11.77	87.18
	cls+cti+g	90.00	90.64	12.43	87.18
	cls+cfour	88.93	90.13	12.58	79.63
	cls+cfour+g	90.05	90.67	12.98	79.63
	cls+cfive	88.97	90.20	13.94	76.75
	cls+cfive+g	90.12	90.75	14.50	76.75
HLM-3	cls+cti	89.31	90.43	12.08	76.35
	cls+cti+g	90.33	90.92	12.42	76.35
	cls+cfour	89.41	90.50	12.49	66.83
	cls+cfour+g	90.40	90.99	12.99	66.83
	cls+cfive	89.48	90.57	14.27	64.66
	cls+cfive+g	90.49	91.08	15.24	64.66

The results of different combinations are shown in Table III. Since it is quite time-consuming to process all the test samples with NNLM-1 and NNLM-2, we only evaluated the perplexity of both 5-gram NNLMs on the development set. It can be seen that the PPL of 5-gram NNLM-1 is 6.5% lower than the 5-gram BLM, and although 5-gram NNLM-2 PPL is slightly higher, it is still better than the 5-gram BLM. Since an decrease of PPL will lead to an improvement of the system performance in most situations observed in our work, we could expect an improvement in accuracies with NNLM-1 and NNLM-2.

Since it can be found in Table II and III that HLMs always perform better than the corresponding NNLMs and BLMs, we

integrated the 5-gram HLM-1 with the system to give a deep insight into the best performance our system could achieve. The 5-gram HLM-1 improves the PPL by 18.7% relative to the 5-gram BLM. Accordingly, the AR is improved to 90.69%, which is the best result we could obtain in this paper and is even higher than the best result without any candidate character augmentation techniques (90.53%) in [1] , at the cost of higher time complexity. We then restricted the length of the character list to 1023, and it is obvious that the time is greatly reduced by 81.0% to HLM-2 compared with HLM-1. The performance of 5-gram HLM-2 is not as good as that of HLM-1 in terms of both accuracies and PPL, due to the small short-list. However, it still outperforms the BLMs as in Table I.

The NNLM-3 models, which is restricted to both a simpler architecture and a smaller short-list, perform worse than BLMs with the same order. However, by interpolation with standard BLMs, the HLM-3 models not only lead to encouraging improvement, but also reduce the time to an acceptable range. This verifies the complementarity between NNLMs and BLMs. For example, the accuracies of 5-gram HLM-2 and HLM-3 are almost the same, but 5-gram HLM-3 reduces the test time by 44.0% compared to HLM-2. Moreover, the superiority of 5-gram HLM-3 compared to 4-gram HLM-3 implies that there still could be some room for improvement with even higher order grams.

V. Conclusion

In this paper, we trained character-level language models in 3-, 4- and 5- gram on large scale corpora and applied them in handwritten Chinese text recognition system. We compared BLMs and NNLMs in different network structures. Although it was observed that full NNLMs can obtain significant performance improvement, as for the time complexity, we adopted the short-list method to accelerate the processing. Our experimental results show that simple NNLMs with short-list yield comparable performance with BLMs of the same order, and using HLMs by interpolating NNLMs with BLMs can improve the performance significantly. It is our future work to further improve the time efficiency of NNLMs and try NNLMs of higher order and deeper structure.

Acknowledgment

This work has been supported by the National Natural Science Foundation of China (NSFC) grants 61305005, 61175021 and 61273269.

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] S. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. 35, no. 3, pp. 400–401, 1987.

[3] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," in *Proc. 34th Annual Meeting on Association for Computational Linguistics*, pp. 310–318, 1996.

[4] 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.

[5] X.-D. Zhou, D.-H. Wang, F. Tian, C.-L. Liu, and M. Nakagawa, "Handwritten Chinese/Japanese text recognition using semi-markov conditional random fields," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 10, pp. 2413–2426, 2013.

[6] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proc. ICCV*, pp. 785–792, Dec 2013.

[7] B. Carpenter, "Scaling high-order character language models to gigabytes," in *Proc. the Workshop on Software*, pp. 86–99, 2005.

[8] H. Schwenk, "Continuous space language models," *Computer Speech & Language*, vol. 21, no. 3, pp. 492–518, 2007.

[9] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, no. 2, pp. 1137–1155, 2003.

[10] A. Mnih and G. Hinton, "Three new graphical models for statistical language modelling," in *Proc. 24th International Conference on Machine Learning*, pp. 641–648, 2007.

[11] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model.," in *Proc. INTERSPEECH*, pp. 1045–1048, 2010.

[12] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *Proc. AISTATS*, vol. 5, pp. 246–252, 2005.

[13] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," in *Proc. 29th International Conference on Machine Learning*, pp. 1751–1758, 2012.

[14] H. Schwenk, "Continuous space translation models for phrase-based statistical machine translation.," in *Proc. COLING*, pp. 1071–1080, 2012.

[15] H. Schwenk, A. Rousseau, and M. Attik, "Large, pruned or continuous space language models on a gpu for statistical machine translation," in *Proc. NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pp. 11–19, 2012.

[16] F. Zamora-Martínez, V. Frinken, S. España-Boquera, M. Castro-Bleda, A. Fischer, and H. Bunke, "Neural network language models for off-line handwriting recognition," *Pattern Recognition*, vol. 47, no. 4, pp. 1642–1652, 2014.

[17] C.-L. Liu, M. Koga, and H. Fujisawa, "Lexicon-driven segmentation and recognition of handwritten character strings for Japanese address reading," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1425–1437, 2002.

[18] F. Yin, Q.-F. Wang, and C.-L. Liu, "Transcript mapping for handwritten Chinese documents by integrating character recognition model and geometric context," *Pattern Recognition*, vol. 46, no. 10, pp. 2807–2818, 2013.

[19] 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.

[20] C.-L. Liu, "Normalization-cooperated gradient feature extraction for handwritten character recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1465–1469, 2007.

[21] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discriminant functions and the application to Chinese character recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 149–153, 1987.

[22] S. Yu, H. Duan, B. Swen, and B.-B. Chang, "Specification for corpus processing at Peking University: Word segmentation, pos tagging and phonetic notation.," *Journal of Chinese Language and Computing*, vol. 13, no. 2, 2003.

[23] http://www.bfsu-corpus.org/channels/corpus.

[24] 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.

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

[26] H. Schwenk, "CSLM - A modular open-source continuous space language modeling toolkit.," in *Proc. INTERSPEECH*, pp. 1198–1202, 2013.