2017 14th IAPR International Conference on Document Analysis and Recognition

# An Open Vocabulary OCR System with Hybrid Word-Subword Language Models

Meng Cai, Wenping Hu, Kai Chen, Lei Sun, Sen Liang, Xiongjian Mo and Qiang Huo
Microsoft Research Asia, Beijing, China
Email: {meng.cai, wenh, kaic, lsun, sen.liang, xiomo, qianghuo}@microsoft.com

***Abstract*—The accuracy of a typical state-of-the-art optical character recognition (OCR) system benefits greatly from using a language model (LM). However, a conventional LM has a limited vocabulary, resulting in out-of-vocabulary (OOV) words that cannot be recognized by the OCR system. In this paper, we present an open vocabulary OCR system based on a hybrid LM. The vocabulary of the hybrid LM consists of both words and subwords. OOV words can be generated by combinations of subwords. A refined hybrid LM training scheme is applied by interpolating a standard hybrid LM, a word-based LM and a subword-based LM. An efficient word combination method is performed by modeling optional space symbols in a decoding network. The overall system deals with OOV words in a general, data-driven and language-independent way. We conduct experiments on an English handwriting OCR task. Evaluations on three testing sets demonstrate that the OCR system with the proposed method achieves a word error rate of 33.4% on an OOV-only testing set, yet without degrading the recognition accuracies on the other two testing sets mainly consisting of in-vocabulary words.**

## I. Introduction

Language models (LMs) are widely used in sequence classification problems such as optical character recognition (OCR), automatic speech recognition (ASR), machine translation, etc. An LM aims to estimate the probability of a word $w_t$ given the word history $\{w_{t-1}, w_{t-2}, \ldots\}$, in which any word $w_t$ belongs to a predefined vocabulary $V$. The words in the vocabulary $V$ are called in-vocabulary (IV) words, while words not in the vocabulary $V$ are called out-of-vocabulary (OOV) words. The OOV words cannot be modeled in standard LMs, which causes recognition errors of OCR or ASR systems. The OOV problems are especially important for OCR as the OOV words such as abbreviations or email addresses are often more informative than IV words in written materials.

A straightforward way to deal with the OOV problem is to increase vocabulary size $|V|$. However, increasing the vocabulary size will also cause the increase of system latency because of larger search space during decoding. In addition, a natural language obeys the well-known Zipf's law [1] so that the frequency of words has a long-tailed distribution. Increasing the vocabulary size can thus only obtain marginal gains for the OOV problem and may even be useless in recognizing abbreviations or email addresses for the OCR task. Another straightforward way to deal with the OOV problem is to use character-level LMs. This method is applicable for some specific tasks such as handwritten address recognition [2]. But a general OCR system often requires a high-order character-level $n$-gram LM to achieve the same accuracy as a system with a low-order word-level $n$-gram LM. The high-order character-level LM brings increased footprint and latency compared with the low-order word-level LM and may not be feasible for a practical OCR task.

Previous works have been done to address the OOV problem in OCR tasks. These works mainly fall into two categories. The first category tries to detect OOV words during recognition process and then recognizes the OOV words using an LM with a small linguistic granularity. For example, in [3], [4] and [5], a special word $w_{\text{oov}}$ or `<unk>` is modeled in word-level LM and decoding network. A character-level LM is used in conjunction with the word-level LM when an OOV word is detected. There are some complex issues in these methods, such as dynamically expanding the weighted finite-state transducer (WFST) [6] decoding network to make use of the character-level LMs. Some methods try to detect OOV words using word lattices and phone lattices for ASR task. Either confidence measures [7] or joint alignments [8] are performed on the lattices. A drawback of these methods is that two decoding passes are needed in recognition stage, which increases the complexity and latency of the systems.

The other category of methods for the OOV problem do not perform explicit OOV detection. Instead, an LM with different levels of linguistic granularity is often used. This LM is referred to as a hybrid LM. For example, the method in [9] uses a character-level LM with some word-level constraints to achieve an open vocabulary OCR system. The method in [10] builds a language model with both words and sub-lexical graphones for ASR. These methods do not introduce extra computation compared with the detection-based OOV recognition methods, thus little latency increase is observed.

Inspired by the previous works, we study the application of hybrid LMs for an English handwriting OCR task in this paper. We mainly conduct the exploration in two aspects. First, a refined training scheme for the hybrid $n$-gram LM is used, in which we interpolate three individual LMs to achieve improved performance. Second, a general subword combination method is applied based on the modeling of `<space>` symbols in the WFST decoding network.

The remainder of the paper is organized as follows. In Section II, we give an overview of our OCR system. In Section III, the method of applying the hybrid LM in OCR is presented. The detailed experiments are reported in Section IV. Finally, the paper is summarized in Section V.

2379-2140/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDAR.2017.91

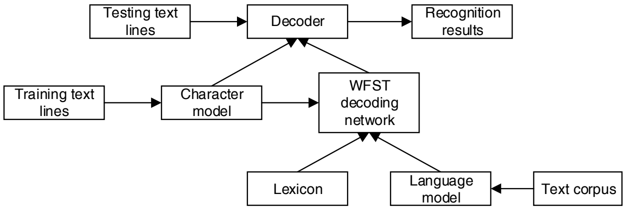


Fig. 1. Architecture block diagram of our OCR system.

## II. System Overview

Our OCR system has a similar architecture as an ASR system, which is illustrated in Fig. 1. There are four key components: character model, lexicon, LM and decoder.

### A. Character Model

The character model is used to calculate likelihood score $p(\mathbf{o}|\mathbf{c})$, where $\mathbf{o}$ represents a sequence of observation feature vectors and $\mathbf{c}$ represents a sequence of characters. A deep bidirectional long short-term memory (DBLSTM) recurrent neural network (RNN) is used for character modeling (e.g., [11], [12], [13], [14], [15]). A softmax layer is used as the output of the character model to produce posterior probability $p(\mathbf{c}|\mathbf{o})$, from which the likelihood score $p(\mathbf{o}|\mathbf{c})$ is calculated by using Bayes rule. The character model is trained in an end-to-end fashion with a so-called connectionist temporal classification (CTC) objective function [16], which allows a label `<blank>` to optionally appear between two adjacent labels to represent no character output. The softmax output of the character model thus encodes encodes all the characters plus the `<blank>` label. The `<space>` symbol that represents a white space is modeled in the character model. Readers are referred to [13], [14], [15] for the details of our DBLSTM-CTC model and its training recipe.

### B. Lexicon

The lexicon consists of a set of mappings from words to the corresponding character sequences. Each word in the LM vocabulary has three variations of character sequences in the lexicon, namely the sequence with all the lower-case characters, the sequence with the first character in upper case and the other characters in lower cases, and the sequence with all the upper-case characters. The `<blank>` symbol is added between adjacent characters in the character sequences to accommodate CTC training.

### C. Language Model

The language model is trained from a large collection of text corpora. Several pre-processing steps are performed to normalize and tokenize raw English text, aiming to minimize the number of OOV words given the vocabulary. Specifically, all words are normalized to lower cases. Punctuation is separated from a word whenever it appears. Each digit string is also separated into single digits. If a word is not in the vocabulary, we check whether the word is a compound word by trying to split it into two IV words. The *n*-gram LM is built using the SRILM toolkit [17] with the modified KN-smoothing [18] after the text normalization and tokenization. The vocabulary of the LM includes punctuation but does not include the `<space>` symbol that separates the tokens. After the *n*-gram LM is built, it is converted to a WFST format using a standard process as described in [6], [19].

### D. Decoder

The character model, the lexicon and the LM are all built during system training stage. At recognition stage, a decoder loads the models and an unknown text line to generate a recognition result. The topology of the character model, the lexicon and the LM are first converted to the WFST format, denoted as $H$, $L$ and $G$, respectively. The $H$, $L$ and $G$ are then composed and optimized to generate a single decoding network $S$ using operations such as composition, determinization and minimization [6]. When building the lexicon WFST $L$, we model an optional `<space>` symbol to enable the flexibility of combining IV words to produce OOV words. The details are presented in Subsection III-B.

The decoding process aims to find the path that has the optimal cost. The cost is a weighted combination of the character scores produced by the character model and the graph scores in the WFST decoding network $S$. An LM scaling factor $\lambda$ is used to control the weight of the graph scores. Searching is performed based on the Viterbi algorithm. The beam pruning and the histogram pruning [20] are used to speed-up the decoding. The Kaldi toolkit [21] is used for the WFST construction, and a compact and efficient WFST-based decoder (i.e., the cloud version in [22]) is used for decoding.

## III. Hybrid Language Models for OCR

In order to deal with the OOV problem for English OCR without increasing much the system complexity and latency, we use hybrid language models that contain both words and subwords in the vocabulary. Basically any methods for subword discovery and segmentation can be used to generate the set of subwords. There are unsupervised [23] or supervised [24] methods to perform subword segmentation. The segmented subwords are also referred to as *morphs* in the literature because the subwords are similar to morphemes, which are the smallest meaningful units of a language. Among the subword segmentation methods, we choose the unsupervised data-driven method in [23] as it is robust and language-independent. The recursive segmentation method with a minimum description length (MDL) cost is used. This method tries to maximize the likelihood while minimize the model complexity. The LM training schemes in this work are applied after the set of subwords are generated.

### A. Hybrid LM Training Schemes

The hybrid LM can be trained in different schemes. The standard scheme is to keep all the IV words and split the OOV words to subwords in the text normalization and tokenization step, then build the hybrid LM [25]. A possible drawback of

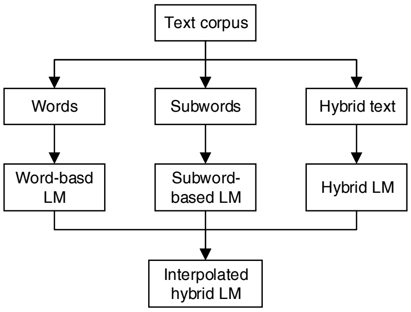


Fig. 2. Refined training scheme for a hybrid LM.

this scheme is that the OOV words are rare compared with the IV words. The probabilities of the subwords may not be well-estimated due to the rare occurrences of OOV words. Some works use subword-based LMs to deal with OOV words [26]. The subword-based LMs are able to better estimate the probabilities of the subwords. But a subword-based LM is weaker compared with a word-based LM if they have the same number of $n$-grams. The subword-based LMs have been known to be suitable for morphologically rich languages such as Arabic, Finnish, Turkish, etc.

In this work, we explore different hybrid LM training schemes for a general English OCR task. In addition to the standard hybrid LM and the morph-based LM, we also try a refined hybrid LM training scheme that interpolates three individual LMs. This training scheme is illustrated in Fig. 2. Given the training text corpus, we conduct three different kinds of text normalization and tokenization. The first processed text corpus has all the IV words but the OOV words are replaced by the symbol `<unk>`. The second processed text corpus is a corpus of subwords for which all the words are split. The third corpus contains hybrid text for which all the IV words remain but the OOV words are split into subwords. Three individual $n$-grams are built using each of the processed corpora, respectively. The three LMs share the same vocabulary consisting of all the IV words and all the subwords. Finally, the three LMs are interpolated to generate the refined hybrid LM. The probabilities of both words and subwords can be well-estimated in the refined hybrid LM training scheme, so the refined hybrid LM is expected to work well for both IV words and OOV words.

### *B. Word Combination*

The recognition results with the hybrid LM contain both words and subwords. A post-processing step is needed to combine the subwords to generate the real OOV words. We perform the word combination in a general data-driven way in the WFST decoding framework.

There are four components on a transition $e$ of a WFST: the input label $i[e]$, the output label $o[e]$, the weight $w[e]$ and the pointer to the next state $n[e]$. The input labels are the characters and the output labels are the words or subwords in the OCR decoding network $S$. The decoding process finds the best path $\pi = \{e_1, e_2, \ldots, e_n\}$ and the word sequence is obtained by removing the $\varepsilon$ symbols from the output label sequence $\{o[e_1], o[e_2], \ldots, o[e_n]\}$. We discover that the word combination can be efficiently performed using only the input label sequence of the decoding path $\pi$, as long as we model the optional `<space>` symbol in the lexicon WFST. Specifically, two paths are created for every lexical item when building the lexicon WFST $L$. One path contains the `<space>` symbol as the last symbol. The other path does not contain the `<space>` symbol. This lexicon WFST $L$ is used to compile the WFST decoding network $S$ with the character model WFST $H$ and the LM WFST $G$ using the standard recipe. The decoding path $\pi$ with such a WFST naturally contains the `<space>` symbol in the input label sequence $\{i[e_1], i[e_2], \ldots, i[e_n]\}$. The `<space>` symbol splits the input sequence to $m$ subsequences. We apply $\varepsilon$-removal and CTC mapping function [16] (i.e., removing consecutive duplicate symbols, removing `<blank>` symbols) to each of the $m$ subsequences to generate $m$ words. The sequence of the $m$ words forms the final decoding result.

### *C. Relation with Previous Works*

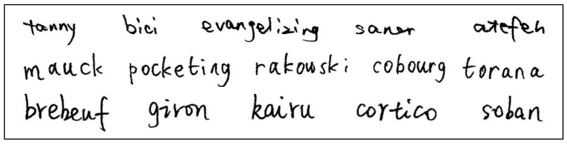
The linguistic granularity in detection-free OOV recognition methods is important. When using a smaller linguistic granularity such as the character, the linguistic constraints are weakened so that the overall recognition results get worse. But when a larger linguistic granularity is used, there are fewer chances that the OOV words can be recognized. Recent studies in the field of ASR make use of subwords for spoken keyword spotting tasks (e.g., [27], [28], [29]). The application of subword units is a good trade-off between the overall accuracy and the ability to recognize OOV words. These works inspire us to study subword-based methods for OCR.

The morph-based LMs have also been applied to Arabic OCR tasks (e.g., [30], [31]). The application of morphs for Arabic is somewhat straightforward as Arabic is a morphologically rich language and some specific characteristics are available to perform subword decompositions and combinations. Our method is general, data-driven and language independent. The refined hybrid LM training scheme and the word combination method are different from the methods designed for Arabic. The effectiveness of our method is verified on an English handwriting OCR task.

## IV. Experiments

### *A. Data Sets*

The LMs in this work are trained from large English data sets. The majority of the training data are from the Linguistic Data Consortium (LDC) and some data are self-collected. The training data sets include the corpora LDC2008T15, LDC2011T07, LDC2015T13, LDC2004T19, LDC2005T19 and self-collected data. The training sets contain about 7 billion words. The self-collected text corpus contains about 300 million words. The corpora LDC2008T15, LDC2011T07 and LDC2015T13 consist of news-style text. The corpora



tanny bici evangelizing saner atefeh
mauck pocketing rakowski cobourg torana
brebeuf giron kairu cortico soban

Fig. 3. Some example text lines of the OOV testing set.

LDC2004T19 and LDC2005T19 consist of spoken-style text. The self-collected corpus mainly consists of written-style text.

There are three offline handwriting OCR testing sets used in experiments. The first testing set is the benchmark IAM evaluation set [32]. The IAM set contains 1,861 text lines. The second testing set is a self-collected data set consisting of whiteboard and handwritten notes data. This testing set is referred to as E2E (end-to-end) set in this work. The E2E set contains 4,028 text lines. The contents of the IAM testing set and the E2E testing set are both natural text so the OOV rates are not high. We also want to evaluate the ability of the systems to deal with OOV words in extreme cases so the third OOV-only testing set is created. This testing set is referred to as the OOV set in this work. The OOV set is created by 10 writers, each of them writes 200 text lines with 5 OOV words per line. So a total of 10,000 different OOV words are in the OOV set. Some example text lines of the OOV testing set are shown in Fig. 3.

The data sets are first normalized and tokenized according to Subsection II-C. In order to decide the vocabulary size for the LM training, we vary the vocabulary size and calculate the OOV rates on the LDC2008T15 training set, the IAM testing set and the E2E testing set. The vocabularies are generated by selecting the words with top frequencies on the training set. The relationship between the vocabulary size and the OOV rate is plotted in Fig. 4. This figure shows the long-tailed distribution of the words. The OOV rate quickly drops when the vocabulary size increases from 10,000. But the drop of OOV rate becomes slow if the vocabulary size is larger than 100,000. Even when a huge vocabulary size of 1 million is used, the OOV rate on the E2E set is still 0.74%. Fig. 4 indicates that it is not economic to use a very large vocabulary to reduce the OOV rate. The OOV rate of the E2E testing set is higher than the IAM testing set because the data of the E2E testing set are mostly natural scene images containing free-style writings. The OOV rate on the training set is the lowest because the vocabularies are generated from the training data. Based on the relationship in Fig. 4, we choose a vocabulary of 100,000 words for the follow-up experiments.

### B. Experimental Setup

The OCR experiments are based on the framework in Section II. In the image pre-processing step, each text line image is normalized with de-skewing, de-slanting, and size-normalization [33] by using an in-house text line normalizer. The height of each normalized text line image is fixed at 60 pixels. For feature extraction, each line is first split into frames by a sliding window of 30 pixels with a frame shift of 3 pixels. Then each frame is smoothed by applying a horizontal cosine window to derive a 1,800-dimensional raw feature vector. The feature dimension is then reduced to 50 by principle component analysis (PCA). Finally, these 50-dimensional feature vectors are normalized to zero mean and unit variance at each dimension. The character model has 5 BLSTM hidden layers. Each BLSTM layer contains 128 memory cells for the forward recurrent connection and 128 memory cells for the backward recurrent connection. There are 96 nodes in the softmax output layer, which represent the 95 printable ASCII characters plus the `<blank>` symbol. The character model is trained using a corpus containing about 283 thousand text lines, including the standard IAM training set and a large collection of in-house data.

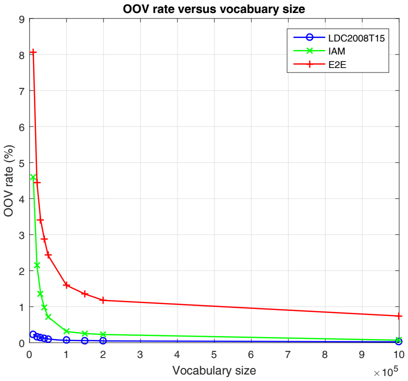


Fig. 4. The relationship between the vocabulary size and the OOV rate.

TABLE I
Examples of Words and Corresponding Subword Sequences

| Word | Subword sequence |
|---|---|
| exboyfriend | ex boy friend |
| affectionately | affection ate ly |
| broomsticks | broom stick s |

Before language model training, the set of subwords are first generated as mentioned in Section III. We use 1 million different words to train the subword segmentation model. A total of 50,900 subwords are discovered. Any single character is included in the subword vocabulary so there are no OOVs in terms of subwords. Some examples of words and the corresponding subword sequences are shown in Table I. The subword vocabulary is merged with the 100,000-word vocabulary in the previous subsection, forming the hybrid vocabulary containing 131,428 unique items.

The accuracy of the OCR system is measured by word error rates (WERs). We also record the decoding speed of some systems for comparison. The decoding beam is set to 13, which is reasonably large as this study mainly focuses on the accuracies. The LM scaling factor $\lambda$ is set to 0.7 unless specifically mentioned. Decoding is performed on a Linux server with a 2.8 GHz Intel Xeon E5-2680 v2 CPU.

TABLE II
The Performance of Standard Hybrid LMs. The Decoding Speed is Recorded on IAM Testing Set.

| LM | # $n$-grams | WER (%) | | | Speed (ms/line) |
|---|---|---|---|---|---|
| | | IAM | E2E | OOV | |
| Trigram | 2,517,719 | 12.9 | 18.7 | 34.2 | 12.83 |
| Trigram | 1,957,166 | 12.9 | 18.8 | 34.3 | 12.38 |
| Trigram | 604,748 | 13.4 | 18.9 | 35.8 | 8.52 |
| Bigram | 1,948,536 | 13.5 | 19.2 | 34.2 | 10.28 |
| Bigram | 602,824 | 13.7 | 19.3 | 35.7 | 7.97 |

TABLE III
Comparison of Different LM Training Schemes.

| LM | Vocabulary | WER (%) | | |
|---|---|---|---|---|
| | | IAM | E2E | OOV |
| None | None | 27.7 | 34.8 | 38.3 |
| Word-based | Word | 13.0 | 19.2 | 51.8 |
| Word-based | Word & Subword | 12.9 | 19.0 | 39.0 |
| Hybrid | Word & Subword | 12.9 | 18.8 | 34.3 |
| Subword-based | Word & Subword | 13.4 | 19.1 | 32.3 |
| Refined hybrid | Word & Subword | 13.1 | 18.5 | 33.4 |

### C. *Standard Hybrid LMs*

The standard hybrid LM training scheme [25] is first tried to determine the hyperparameters of the LMs. The OOV words in the training corpora are split using the subword segmentation model while the IV words remain. A separate LM is built for each training corpus. Then, all the LMs are interpolated to generate a single hybrid LM. We try different LM orders and pruning hyperparameters for the hybrid LM. The results are given in Table II.

The results show that the trigram LMs have better accuracies on the IAM and E2E testing sets compared with the bigram LMs with the same number of $n$-grams. The results of the trigrams on the OOV testing set are a little worse than the corresponding bigrams as the contents of the OOV testing set have less linguistic constraints than the IAM or E2E testing set. The trigram LM with 1,957,166 $n$-grams is a good trade-off between efficiency and accuracy if memory footprint is not critical. When using an even larger trigram LM, there is a marginal gain in accuracy but the size of the corresponding WFST decoding network is unacceptably large (465 MB). In the following experiments, we control the pruning hyperparameter so that all the trigram LMs contain approximately 2 million $n$-grams.

### D. *Comparison of LM Training Schemes*

We study different LM training schemes in this subsection. The results are summarized in Table III. The CTC best path decoding [16] results are first generated for comparison. These results are obtained by applying the CTC mapping function to the symbol sequences of the maximum softmax activations of the character model. Since neither vocabulary nor LM is used, the CTC best path decoding might be suitable for OOVs intuitively. The results are in the first line of Table III. We then train two word-based LMs. One uses the word vocabulary of 100,000 items. The other uses the hybrid vocabulary of 131,428 items. We apply the word combination strategy in Subsection III-B so that the word-based systems can still recognize some OOV words. Comparing the results in the second and the third line of Table III, it is shown that much better performance on the OOV set is obtained by replacing the word vocabulary to the hybrid vocabulary. This is because the LM with the hybrid vocabulary assigns some probabilities to the subwords by the modified KN-smoothing. However, the performances of the word-based LMs on the OOV set are worse than the result with CTC best path decoding.

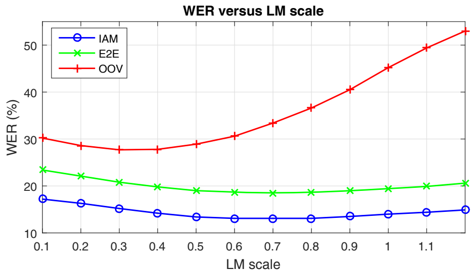


Fig. 5. The relationship between the LM scaling factor and the WER.

The results of the standard hybrid LM, the subword-based LM and the refined hybrid LM are shown in the last three lines of Table III. For the subword-based LM training, all the words in the training corpora are split using the subword segmentation model. The individual subword-based LMs are built from each training corpus and then all the LMs are interpolated. For the refined hybrid LM, we build individual LMs for each corpus according to the scheme illustrated in Fig. 2 and then interpolate all the LMs to generate a single LM. The interpolation weights for the hybrid LM, the word-based LM and the subword-based LM for each corpus are empirically set to 0.5, 0.3, and 0.2, respectively.

The results show that all the systems with an LM produce dramatically lower WERs on the IAM and E2E testing sets compared with the CTC best path decoding results. Even for the OOV testing set, the systems with hybrid LMs produce better results. This is because the subword strategy is able to discover the statistical properties within the structures of the OOV words. Compared with the results of the standard hybrid LM, the result of the refined hybrid LM is better on the E2E set and the OOV set but is a little worse on the IAM set. This is because the OOV rate is low on the IAM testing set, as illustrated in Fig. 4. The relative WER reduction of the refined hybrid LM on the OOV set is 2.62% compared with the standard hybrid LM. The results show that the refined hybrid LM training scheme is effective to deal with OOV words.

### E. *Effects of the LM Scaling Factor*

Since the nature of the OOV testing set is different from the IAM set or the E2E set, the LM scaling factor $\lambda$ may have different impacts on the testing sets. We plot the relationship between the LM scaling factor $\lambda$ and the WERs on the three testing sets in Fig. 5. The figure shows that the optimal $\lambda$ for the IAM or E2E testing set is 0.7, while the optimal $\lambda$ for the OOV testing set is around 0.3. This result suggests that better performance on the OOV testing set can be achieved if the nature of the testing set is known in advance. The lowest possible WER on the OOV testing set is 27.7%.

## V. Summary

We have presented an open vocabulary OCR system with hybrid word-subword language models. The system framework is similar to an ASR system and DBLSTM-CTC model is employed for character modeling. The details of subword segmentation and hybrid LM training schemes have been discussed and compared. A refined hybrid LM training scheme is explored to leverage the complementary information in word-based and subword-based LMs. A general word combination method is applied in a WFST framework, which determines word boundaries by the <space> symbol. Two normal testing sets and an OOV-only testing set are used for experiments. The systems with hybrid LMs achieve better results on all the testing sets compared with the system without an LM. The WER of the system with the refined hybrid LM is 33.4% on the OOV testing set, yet without degrading the recognition accuracies on the other two testing sets mainly consisting of in-vocabulary words.

## Acknowledgment

We would like to thank our colleagues, Michael Levit and Shawn Chang in Microsoft for sharing their universal language models (ULMs), which facilitate our study in this paper.

## References

[1] W. Li, “Random texts exhibit Zipf’s-law-like word frequency distribution,” *IEEE Transactions on Information Theory*, vol. 38, no. 6, pp. 1842–1845, 1992.

[2] A. Brakensiek, J. Rottland, and G. Rigoll, “Handwritten address recognition with open vocabulary using character n-grams,” in *Proc. IWFHR*, 2002, pp. 357–362.

[3] M. Kozielski, D. Rybach, S. Hahn, R. Schluter, and H. Ney, “Open vocabulary handwriting recognition using combined word-level and character-level language model,” in *Proc. ICASSP*, 2013, pp. 8257–8261.

[4] M. Kozielski, M. Matysiak, P. Doetsch, R. Schluter, and H. Ney, “Open-lexicon language modeling combining word and character levels,” in *Proc. ICFHR*, 2014, pp. 343–348.

[5] R. Messina and C. Kermorvant, “Over-generative finite state transducer n-gram for out-of-vocabulary word recognition,” in *Proc. IWDAS*, 2014, pp. 212–216.

[6] M. Mohri, F. Pereira, and M. Riley, “Speech recognition with weighted finite-state transdusers,” in *Springer Handbook on Speech Processing and Speech Communication*. Springer Berlin Heidelberg, 2008, pp. 559–584.

[7] L. Burget, P. Schwarz, P. Matejka, M. Hannemann, A. Rastrow, C. White, S. Khudanpur, H. Hermansky, and J. Cernocky, “Combination of strongly and weakly constrained recognizers for reliable detection of OOVs,” in *Proc. ICASSP*, 2008, pp. 4081–4084.

[8] H. Lin, J. Bilmes, D. Vergyri, and K. Kirchhoff, “OOV detection by joint word/phone lattice alignment,” in *Proc. ASRU*, 2007, pp. 478–483.

[9] I. Bazzi, R. Schwartz, and J. Makhoul, “An omnifont open-vocabulary OCR system for English and Arabic,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 6, pp. 495–504, 1999.

[10] M. Bisani and H. Ney, “Open vocabulary speech recognition with flat hybrid models,” in *Proc. Interspeech*, Lisbon, Portugal, 2005, pp. 725–728.

[11] T. Bluche, H. Ney, J. Louradour, and C. Kermorvant, “Framewise and CTC training of neural networks for handwriting recognition,” in *Proc. ICDAR*, 2015, pp. 81–85.

[12] K. Chen, Z.-J. Yan, and Q. Huo, “A context-sensitive-chunk BPTT approach to training deep LSTM/BLSTM recurrent neural networks for offline handwriting recognition,” in *Proc. ICDAR*, 2015, pp. 411–415.

[13] Q. Liu, L.-J. Wang, and Q. Huo, “A study on effects of implicit and explicit language model information for DBLSTM-CTC based handwriting recognition,” in *Proc. ICDAR*, 2015, pp. 461–465.

[14] H. Ding, K. Chen, Y. Yuan, M. Cai, L. Sun, S. Liang, and Q. Huo, “A compact CNN-DBLSTM based character model for offline handwriting recognition with Tucker decomposition,” in *Proc. ICDAR*, 2017.

[15] W. Hu, M. Cai, K. Chen, H. Ding, L. Sun, S. Liang, X. Mo, and Q. Huo, “Sequence discriminative training for offline handwriting recognition by an interpolated CTC and lattice-free MMI objective function,” in *Proc. ICDAR*, 2017.

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

[17] A. Stolcke, “SRILM—an extensible language modeling toolkit,” in *Proc. ICSLP*, 2002.

[18] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” *Computer Speech and Language*, vol. 13, no. 4, pp. 359–394, 1999.

[19] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiat, S. Kombrink, P. Motlicek, Y. Qian, K. Riedhammer, K. Vesely, and N. T. Vu, “Generating exact lattices in the WFST framework,” in *Proc. ICASSP*, 2012, pp. 4213–4216.

[20] G. Saon, D. Povey, and G. Zweig, “Anatomy of an extremely fast LVCSR decoder,” in *Proc. Interspeech*, Lisbon, Portugal, 2005, pp. 549–552.

[21] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, “The Kaldi speech recognition toolkit,” in *Proc. ASRU*, 2011.

[22] M. Cai and Q. Huo, “Compact and efficient WFST-based decoders for handwriting recognition,” in *Proc. ICDAR*, 2017.

[23] M. Creutz and K. Lagus, “Unsupervised discovery of morphemes,” in *Proc. ACL*, 2002, pp. 21–30.

[24] T. Ruokolainen, O. Kohonen, S. Virpioja, and M. Kurimo, “Supervised morphological segmentation in a low-resource learning setting using conditional random fields,” in *Proc. CoNLL*, 2013, pp. 29–37.

[25] A. Yazgan and M. Saraclar, “Hybrid language models for out of vocabulary word detection in large vocabulary conversational speech recognition,” in *Proc. ICASSP*, 2004, pp. 745–748.

[26] M. Creutz, T. Hirsimaki, M. Kurimo, A. Puurula, J. Pylkkonen, V. Siivola, M. Varjokallio, E. Arisoy, M. Saraclar, and A. Stolcke, “Morph-based speech recognition and modeling of out-of-vocabulary words across languages,” *ACM Transactions on Speech and Language Processing*, vol. 5, no. 1, 2007.

[27] K. Narasimhan, D. Karakos, R. Schwartz, S. Tsakalidis, and R. Barzilay, “Morphological segmentation for keyword spotting,” in *Proc. EMNLP*, 2014, pp. 880–885.

[28] Y. He, B. Hutchinson, P. Baumann, M. Ostendorf, E. Fosler-Lussier, and J. Pierrehumbert, “Subword-based modeling for handling OOV words in keyword spotting,” in *Proc. ICASSP*, 2015, pp. 7914–7918.

[29] C. Ni, C.-C. Leung, L. Wang, N. F. Chen, and B. Ma, “Unsupervised data selection and word-morph mixed language model for Tamil low-resource keyword search,” in *Proc. ICASSP*, 2015, pp. 4714–4718.

[30] M. Hamdani, A. E.-D. Mousa, and H. Ney, “Open vocabulary Arabic handwriting recognition using morphological decomposition,” in *Proc. ICDAR*, 2013, pp. 280–284.

[31] M. F. BenZeghiba, J. Louradour, and C. Kermorvant, “Hybrid word/part-of-Arabic-word language models for Arabic text document recognition,” in *Proc. ICDAR*, 2015, pp. 671–675.

[32] U.-V. Marti and H. Bunke, “The IAM-database: an English sentence database for offline handwriting recognition,” *International Journal on Document Analysis and Recognition*, vol. 5, pp. 39–46, 2002.

[33] A. Vinciarelli and J. Luettin, “A new normalization technique for cursive handwritten words,” *Pattern Recognition Letters*, vol. 22, pp. 1043–1050, 2001.