Multiple Text Segmentation for Statistical Language Modeling

Sopheap Seng, Laurent Besacier, Brigitte Bigi, Eric Castelli

INTERSPEECH 2009 BRIGHTON

Multiple Text Segmentation for Statistical Language Modeling

Sopheap Seng[1,2], *Laurent Besacier*[1], *Brigitte Bigi*[1], *Eric Castelli*[2]

[1]LIG Laboratory, CNRS/UMR-5217, Grenoble, France
[2]MICA Center, HUT - CNRS/UMI-2954 - Grenoble INP, Hanoi, Vietnam
Sopheap.Seng@imag.fr

Abstract

In this article we deal with the text segmentation problem in statistical language modeling for under-resourced languages with a writing system without word boundary delimiters. While the lack of text resources has a negative impact on the performance of language models, the errors introduced by the automatic word segmentation makes those data even less usable. To better exploit the text resources, we propose a method based on weighted finite state transducers to estimate the N-gram language model from the training corpus on which each sentence is segmented in multiple ways instead of a unique segmentation. The multiple segmentation generates more N-grams from the training corpus and allows obtaining the N-grams not found in unique segmentation. We use this approach to train the language models for automatic speech recognition systems of Khmer and Vietnamese languages and the multiple segmentations lead to a better performance than the unique segmentation approach.

Index Terms: multiple segmentation, unsegmented language, statistical language modeling

1. Introduction

A language model is a probability assignment over all possible word sequences in a natural language. Its goal, loosely stated, is to assign a relatively large probability to meaningful, grammatical, or frequent word sequences and a low probability to nonsensical, ungrammatical or rare ones. The statistical approach used in N-grams language modeling requires a large amount of text data in order to make an accurate estimation of probabilities. These data are not available in large quantities for under-resourced languages and the lack of text data has a direct impact on the performance of language models. While word is usually a basic unit in statistical language modeling, word identification is not a simple task even for languages that separate words by a special character (a whitespace in general). For unsegmented languages, which have a writing system without obvious separation between words, the N-grams of words are usually estimated from the text corpus segmented into words employing automatic methods. Automatic segmentation of text is not a trivial task and introduces errors due to the ambiguities in natural language and the presence of out of vocabulary words in the text. A possible alternative is to calculate the probabilities from sub-lexical units. Among the existing works that use sub-lexical units for language modeling, we can mention [1] and [2] which used morphemes for the modeling of Finnish and Arabic. For other unsegmented languages such as Japanese, the logographic character is used in [3]. In our previous work on automatic speech recognition of Khmer language [4], we have exploited the different lexical and sub-lexical units in Khmer language modeling. We have proposed language models based on word, syllable and characters cluster. Our goal was to compare the performance of these lexical and sub-lexical units and try to exploit these units at the same time in language modeling. The results obtained have shown that the word remains the best unit for language modeling, even if word segmentation is error-prone.

In this article, we deal with the text segmentation problem in statistical languages modeling for under-resourced languages with a writing system without word boundary delimiters. While the lack of text resources has a negative impact on the performance of language models, the errors produced by the word segmentation make those data even less usable. The word N-grams not found in the training corpus could be due not only to the errors introduced by the automatic segmentation but also to the fact that a sequence of characters could have more than one correct segmentation. To better exploit the text resources, we propose a method to estimate the N-gram language model from the training corpus on which each sentence is segmented into multiple ways instead of a unique segmentation. The objective of multiple segmentation is to generate more N-grams from the training corpus to use in language modeling. We use this approach to train the language models for Automatic Speech Recognition systems (ASR) of two under-resourced and unsegmented languages: Khmer and Vietnamese. The multiple text segmentation leads to a better performance than the unique segmentation approach.

In the next section, the problem of unknown words in automatic text segmentation is presented. We describe our multiple text segmentation method based on weighted finite state transducers in section 3. The experimental framework and the results are presented in section 4. Section 5 concludes the work and gives some future perspectives.

2. Problem in automatic text segmentation

2.1. State-of-the-art methods

Text segmentation is one of the fundamental tasks in natural language processing (NLP). Many NLP applications require the input text segmented into words before making further process because word is considered as the basic semantic unit in natural languages. For unsegmented languages, the text segmentation into words is not a trivial task. Because of ambiguities in human languages, a sequence of characters may be segmented in more than one way to produce a sequence of valid words. The readability for humans, however, is not severely impacted by this apparent ambiguity, perhaps because incorrect segmentation usually leads to meaningless sentences. This is not to suggest that human readers will agree entirely on the segmentation of any particular sentence. The disagreement, when it occurs, is largely due to different segmentation conventions and to the fact

that the definition of word in a language is often ambiguous.

Text segmentation technique uses generally an algorithm, which search in the text the words corresponding to those in a dictionary. In case of ambiguity, the algorithm selects the one that optimizes a parameter dependent on the chosen strategy. The most common optimization strategies consists in maximizing the length of words (*"longest matching"*) or minimizing the number of words in the entire sentence (*"maximum matching"*).

These techniques rely heavily on the availability and the quality of the dictionaries and while it is possible to automatically generate a dictionary from an unsegmented text corpus using unsupervised methods, dictionaries are often created manually. The state-of-the-art methods use generally a combination of hand crafted dictionary and statistical techniques to obtain a better result. However, statistical methods need a large corpus segmented manually beforehand. Statistical methods and complex training methods are not appropriate in the context of under-resourced languages as the resources needed to implement these methods do not exist. For an under-resourced language, we seek segmentation methods that allow to best exploit the limited resources.

2.2. Problem of unknown words in dictionary-based text segmentation - example of Khmer language

To illustrate the impact of out-of-vocabulary words on the performance of dictionary-based automatic segmentation methods, we apply these methods on Khmer text. Two optimization criteria are tested: *"longest matching"* and *"maximum matching"*. Our test corpus is composed of 1000 sentences. After manual segmentation, we obtain 31k words and a dictionary of 4875 words is generated. To prepare the scenarios of segmentation with increasing rate of out-of-vocabulary word, we remove the least frequent words from the dictionary to create different dictionaries with the rate of out-of-vocabulary words, which varies from 5% to 50%. The segmentation performance is presented in table 1. We observe that when there are no out-of-vocabulary words, the performance is around 91% for both methods. The performance drops to 68% and 72% when there are 50% of out-of-vocabulary words in the text to segment.

Out-of-vocabulary word	Accuracy (%)	
	Longest Matching	Maximal Matching
0%	91.6	91.7
5%	90.1	90.2
10%	90.2	90.3
20%	86.3	86.9
30%	82.6	83.5
40%	75.7	77.2
50%	68.8	72.4

Table 1: *Accuracy of two dictionary-based segmentation methods Vs. rate of unknown words.*

For under-resourced languages, it is difficult to obtain a dictionary with a low rate of out-of-vocabulary words. In this case, we may have poor segmentation performance on the training text corpus and the performance of language models trained from this badly segmented corpus will be mediocre.

3. Multiple text segmentation

3.1. Why Multiple text segmentation?

Unlike the automatic segmentation method described in the previous section, which search in a sequence of characters the best segmentation according to a given optimization criteria, our multiple segmentation approach seeks to generate all possible segmentations from a sequence of characters. It is from those segmentations that N-grams are counted in order to train the language model.

Sentence	ព្រះពុទ្ធជាព្រះបរមគ្រូនៃយើង							3-grams found
Seg 1	ព្រះពុទ្ធ w_1	ជា w_2	ព្រះ w_3	បរមគ្រូ w_4		នៃ w_5	យើង w_6	$w_1\ w_2\ w_3$ $w_2\ w_3\ w_4$ $w_3\ w_4\ w_5$ $w_4\ w_5\ w_6$
Seg 2	ព្រះពុទ្ធ w_1	ជា w_2	ព្រះ w_3	បរម w_7	គ្រូ w_8	នៃ w_5	យើង w_6	$w_2\ w_3\ w_7$ $w_3\ w_7\ w_8$ $w_7\ w_8\ w_5$
Seg 3	ព្រះ w_3	ពុទ្ធ w_9	ជា w_2	ព្រះ w_3	បរម w_7	គ្រូ w_8	នៃ w_5	យើង w_6	$w_2\ w_9\ w_2$ $w_9\ w_2\ w_3$
Translation	The buddha is our great teacher							

Figure 1: *Example of multiple segmentation on a khmer sentence*

Figure 1 shows an example of three possible segmentations of a khmer sentence. The segmentation *Seg 1* corresponds well to the unique segmentation of type "longest matching". In this case, 4 tri-grams are found in the segmentation. If we make the count from these 3 possible segmentations, we have a total of 9 tri-grams. 5 new tri-grams are found in the other two segmentations *Seg 2* and *Seg 3*. Note that we count only one time a tri-gram that appears several times in the multiple segmentations.

Compared to unique segmentation, the multiple segmentation allows generating more N-grams. We can divide those new N-grams into two categories:

1. N-grams which are effectively present in the original training corpus but because of the errors introduced by the unique segmentation or because of the fact that a sequence of characters could have more than one correct segmentation (and only one segmentation was considered in the training corpus), those N-grams can not be found after the conventional segmentation process (unique segmentation)
2. N-grams which are not effectively present in the original training corpus even if the segmentation is done perfectly. Those N-grams are obtained because it is possible to segment partly a sequence of characters to obtain a sequence of valid words.

While the first category of N-grams is potentially useful for the language modeling, the second group could disturb the modeling. The goal of multiple segmentation is to generate those potentially useful N-grams and limit the generation of not relevant N-grams.

3.2. Finite state machine based multiple text segmentation

In this section we provide the mathematical details of the algorithm used to generate the multiple segmentations of a character sequence based on weighted finite state transducers (WFST). Many research works use WFST to do unique text segmentation. In [5], WFST is used for automatic unique segmentation of Arabic text into morphemes. Assume that the input string is described by the Q length characters sequence $S = c_1 c_2 ... c_Q$ and the L length word sequence $m_k = \{w_1^k, w_2^k, ... w_L^k\}$ is one

of the many possible segmentations M of S ($m_k \in M$). m_k is obtained from a lookup table of all words for a given language. Among all the possible segmentations M of the input string S, the best segmentation $\hat{m}$ is the one with the highest probability:

$$P(\hat{m}) = \max_k P(m_k) \quad (1)$$

The probability $P(m_k)$ is estimated using an N-gram language model on word sequence:

$$P(m_k) \simeq \prod_{i=1}^{L} P(m_i^k | m_{i-1}^k, m_{i-2}^k, ...m_{i-N+1}^k) \quad (2)$$

The $nbest$ (multiple) segmentations is the set of segmentations $\hat{M} = \{\hat{m}_1, \hat{m}_2, ...\hat{m}_{nbest}\}$ where $\hat{m}_i$ is the i^{th} best segmentation according to the value of the probability function $P(\hat{m}_i)$.

The formalization of the above algorithm in terms of composition of weighted finite state transducers is straightforward. Let's represent the input string with an acceptor I, each arc of I representing a single character. Given a finite list of words we can then build a finite state transducer M (or word transducer) that, once composed with I, generates a lattice of the words table entries that represent all of the possible segmentations. As with any vocabulary-based system, we need to handle out-of-vocabulary entries. For this reason, we make a model of any string of characters, which will constitute the unknown word model. A simple unknown word model using WFST can be represented by a $star$ closure operation over all the possible characters. Thus, the unknown word WFST can parse any sequence of characters and generate a unique unk word symbol. The word transducer can be therefore described in terms of the WFST operations as $M = (\texttt{WD} \cup \texttt{UNK})^+$ where WD is a WFST that represents the dictionary and UNK represents the unknown word WFST. Here, $\cup$ and + is the union and Kleene "+" closure operation.

A language model L is used to score the lattice of all possible segmentations obtained by the composition of our word transducer M and the input string I. A language model can be represented by a WFST. Briefly, a state is generated for each observed context. The number of states is bound by V^{N-1}, with V being the size of the dictionary and N the N-gram order. An arc exiting from state S_i represents an input symbol c in the context associated to S_i, the corresponding weight is thus the conditional probability $P(c|S_i)$. In our case, it is important to note that only a simple unigram language model must be used. The unigram model is estimated from a small training corpus segmented automatically into words using dictionary-based method. We use a unigram model because a higher order model trained on a corpus segmented automatically would influence the score so that the best segmentation obtained would be similar to the one generated by the dictionary-based method.

The composition of the sequence of input string I with the word transducer M yields a transducer that represents all possible segmentations. This transducer is then composed with the language model L, resulting into a transducer that represents all possible segmentations for the input string I, scored according to L. The highest scoring paths of the compound transducer is the segmentation $\hat{m}$ as defined in equation (1). The segmentation procedure can then be expressed formally as:

$$\hat{m} = \text{bestpath}(I \circ M \circ L) \quad (3)$$

where $\circ$ is the composition operator. The $nbest$ segmentations $\hat{M}$ are obtained by decoding the final lattice to output the $nbest$ highest scoring paths. These $nbest$ segmentations will be finally used in the N-gram count for language modeling as presented in figure 1.

The implementation of our multiple segmentation algorithm based on WFST is done using the AT&T FSM Library (Finite-State Machine Library) [6]. The language model in the form of WFST used to score the lattice is created using the GRM Library (Grammar Library).

4. Experiments and results

4.1. Experimental framework

The experiment is conducted on two under-resourced and non-segmented languages: Khmer and Vietnamese. In order to compare the performance of the multiple segmentation approach with the classical dictionary-based unique segmentation, we build N-gram language models from training corpus segmented using these two approaches. First of all, a language model named lm_unique is trained from the corpus segmented using dictionary-based, *"longest matching"* unique segmentation method. Then, other language models are trained from corpus segmented with multiple segmentation method by fixing the number of $nbest$ segmentations to generate for each sentence at 2, 5, 10, 50 and 100. After training we obtain 5 language models named respectively lm_2, lm_5, lm_10, lm_50 and lm_100. The performance of each language model is first evaluated in terms of the *tri-gram hits* and the *perplexity* on a development corpus (*dev*). The *perplexity* is comparable in our case because different language models are based on the same vocabulary. We evaluate then each model in automatic speech recognition systems to decode a *test* corpus. To show the potential of the multiple segmentation approach in statistical language modeling for under-resourced languages, the experiments are done on corpus of different sizes. The experiments on Khmer language are done on 100% of text of training corpus and also on 50% and 25% of text of the same training corpus.

4.2. ASR system setup

Our Khmer and Vietnamese automatic speech recognition systems use the CMU's Sphinx3 decoder. It uses Hidden Markov Models (HMM) with continuous output probability density functions. The model topology is a 3-state left-to-right HMM with 16 Gaussian mixtures per state. The pre-processing of the system consists in extracting a 39 dimensional feature vector of 13 MFCCs, the first and second derivatives. The CMU's SphinxTrain is used to train the acoustic models used in our experiment. The Khmer and Vietnamese acoustic models are trained from our Khmer speech corpus (6 hours of speech data from Khmer radio broadcast news, 8 speakers) and Vietnamese speech corpus (13 hours of speech data recorded in studio, 36 speakers).

While the evaluation metric WER (Word Error Rate) is generally used to evaluate and compare the performance of the ASR systems, this metric does not fit well for unsegmented languages because the errors introduced during the segmentation of the references and the output hypothesis may prevent a fair comparison of different ASR system outputs. For Khmer language, we propose to use Character-Cluster Error Rate (CCER) for evaluation metric as a character cluster (cc) in Khmer is a group of unseparated characters with a well-defined structure and the text segmentation into cc is trivial. In the case of Vietnamese, the Syllable Error Rate (SER) is used as Vietnamese text is composed of syllables naturally separated by whitespace.

4.3. Experimental results on Khmer language

Our training corpus of Khmer language contains 0.5 millions sentences from broadcast news domain. A vocabulary of 20k words from the *Chuon Nath* Khmer Dictionary of Cambodia Buddhist Institute is used in this experiment. The unique segmentation of this corpus with "longest matching" method gives a corpus of 15 millions words. The unique and multiple segmentations are based on the same dictionary. A development corpus (*dev*) of 370 sentences (11k word after automatic segmentation) is used to evaluate the *tri-gram hits* and the *perplexity*. The automatic speech recognition of Khmer language is done on a test corpus of 160 utterances (broadcast news domain). More details on our automatic speech recognition system of Khmer language can be found in [4]. Table 2 shows the result of the experiment on Khmer language.

Evaluation criteria	Language models trained with different segmentation					
	lm_unique	***lm_2***	***lm_5***	***lm_10***	***lm_50***	***lm_100***
Number of tri-grams in language model (million)	5.67	7.34	8.95	10.17	12.52	13.31
Tri-gram hits on *dev* (%)	31%	34.1%	34.6%	35.2%	36.6%	37%
Perplexity on *dev*	394.9	**322.5**	348.8	361.8	373.9	374.7
CCER of ASR on *test*	22%	21.7%	20.8%	**20.5%**	20.6%	20.7%

Table 2: *Results of experiments on Khmer language.*

Table 3 shows the influence of number of $nbest$ segmentations on the performance of the ASR systems with training corpus of different sizes.

LM	***CCER Corpus 100%***	***CCER Corpus 50%***	***CCER Corpus 25%***
lm_unique	22	22.8	23.9
lm_2	21.7	22.8	23.6
lm_5	20.8	22.6	23.5
lm_10	***20.5***	22.6	23.4
lm_50	20.6	22.4	23.5
lm_100	20.7	***22***	23.2
lm_500	20.9	22.6	***22.9***
lm_1000	21.0	22.8	23

Table 3: *ASR results of Khmer on corpus of different sizes.*

4.4. Experimental results on Vietnamese language

The training corpus of Vietnamese language contains 3 millions sentences from broadcast news domain. A vocabulary of 30k words extracted from Vietnamese-French bilingual dictionary is used in this experiment. The unique segmentation of this corpus with "longest matching" method gives a corpus of 46 millions words. A development corpus (*dev*) of 1000 sentences (44k word after automatic segmentation) is used to evaluate the the *tri-gram hits* and the *perplexity*. The automatic speech recognition of Vietnamese language is done on a test corpus of 270 utterances (broadcast news domain). More details on the automatic speech recognition system of Vietnamese language can be found in [7]. Table 4 shows the results of the experiment on Vietnamese language.

4.5. Discussion

The results of the experiments showed that the multiple segmentation allows to generate more tri-grams when we increase the number of $nbest$ segmentations. The increased number of tri-grams in the language models improves the percentage of tri-gram hits and the perplexity. This improvement shows that the new tri-grams generated by multiple segmentation are relevant

Evaluation criteria	Language models trained with different segmentation					
	lm_unique	***lm_2***	***lm_5***	***lm_10***	***lm_50***	***lm_100***
Number of tri-grams in language model (million)	20.32	24.06	28.92	32.82	34.2	34.9
Tri-gram hits on *dev* (%)	47.7%	48.6%	49.2%	49.4%	49.7%	49.7%
Perplexity on *dev*	118.9	**118.1**	125.9	129	133.4	134.8
SER of ASR on *test*	27.6%	**26.2%**	27%	26.5%	26.7%	26.9%

Table 4: *Results of experiments on Vietnamese language.*

for language modeling. In case of Khmer, the best error rate of ASR system (corpus 100%) is obtained with lm_10 and the performance drops if we continue to increase the number of $nbest$ segmentations. This could be explained by the fact that at some point, when we continue to generate more segmentations, those segmentations generate N-grams which are not relevant to the language and disturb thus the estimation of probability. Pruning out these incorrect N-grams using statistical or linguistic knowledge based methods would help to improve the performance. We can also observe that when the size of corpus is reduced, increasing the number of $nbest$ segmentations helps to improve the performance. In case of Vietnamese language, the best error rate is obtained at number of $nbest$ segmentations as little as 2. With deeper analyze of our Vietnamese training corpus, we see that nearly 80% of words in the corpus are monosyllabic words and only 20% are multi-syllabic words. This means that there are not much possible alternative good segmentations that can be generated compared to Khmer language.

5. Conclusions

We proposed in this article a method to make multiple segmentation on the training corpus in order to estimate statistical language models in the context of under-resourced and unsegmented languages. We showed that our approach allow generating more N-grams than classical dictionary-based unique segmentation method and those new generated N-grams are potentially useful and relevant in language modeling. The application of multiple segmentation in language modeling for Khmer and Vietnamese languages showed the improvement in terms of tri-gram hits, perplexity and the error rate in ASR systems compared to unique segmentation method. We plan to apply this approach in language modeling of Vietnamese for statistical machine translation system.

6. References

[1] M. Creutz and al, "Morph-based speech recognition and modeling of out-of-vocabulary words across languages," *ACM Trans. Speech Lang. Process.*, vol. 5, no. 1, pp. 1–29, 2007.

[2] M. Afify, R. Sarikaya, H.-K. Kuo, L. Besacier, and Y. Gao, "On the use of morphological analysis for dialectal arabic speech recognition," in *InterSpeech*, Pittsburg, PA, USA, 2006.

[3] E. Denoual and Y. Lepage, "The character as an appropriate unit of processing for non-segmenting languages," in *Proceedings of the Annual Meeting of the Association for Natural Language Processing*, vol. 12, Japan, 2006, pp. 731–734.

[4] S. Seng, S. Sam, V. Le, L. Besacier, and B. Bigi, "Which units for acoustic and language modelling for khmer automatic speech recognition?" in *SLTU'08*, Hanoi Vietnam, 2008, pp. 33–38.

[5] Y.-S. Lee and all, "Language model based arabic word segmentation," in *ACL'03*, Morristown, NJ, USA, 2003, pp. 399–406.

[6] M. Mohri, O. Pereira, and M. Riley, "A rational design for a weighted finite-state transducer library," in *Lecture Notes in Computer Science*. Springer, 1998, pp. 144–158.

[7] V. LE, L. Besacier, S. Seng, B. Bigi, and T. DO, "Recent advances in automatic speech recognition for vietnamese," in *SLTU'08*, Hanoi Vietnam, 2008, pp. 47–52.