Integrating Ngram Model and Case-based Learning For Chinese Word Segmentation

Chunyu Kit Zhiming Xu Jonathan J. Webster
Department of Chinese, Translation and Linguistics
City University of Hong Kong
Tat Chee Ave., Kowloon, Hong Kong
{ctckit, ctxuzm, ctjjw}@cityu.edu.hk

Abstract

This paper presents our recent work for participation in the First International Chinese Word Segmentation Bakeoff (ICWSB-1). It is based on a general-purpose ngram model for word segmentation and a case-based learning approach to disambiguation. This system excels in identifying in-vocabulary (IV) words, achieving a recall of around 96-98%. Here we present our strategies for language model training and disambiguation rule learning, analyze the system's performance, and discuss areas for further improvement, e.g., out-of-vocabulary (OOV) word discovery.

1 Introduction

After about two decades of studies of Chinese word segmentation, ICWSB-1 (henceforth, the bakeoff) is the first effort to put different approaches and systems to the test and comparison on common datasets. We participated in the bakeoff with a segmentation system that is designed to integrate a general-purpose *ngram model* for probabilistic segmentation and a *case-* or *example-based learning* approach (Kit et al., 2002) for disambiguation.

The ngram model, with words extracted from training corpora, is trained with the EM algorithm (Dempster et al., 1977) using unsegmented training corpora. Originally it was developed to enhance word segmentation accuracy so as to facilitate Chinese-English word alignment for our ongoing EBMT project, where only unsegmented texts are available for training. It is expected to be robust enough to handle novel texts, independent of any segmented texts for training. To simplify the EM training, we used the uni-gram model for the bakeoff and relied on the Viterbi algorithm (Viterbi, 1967) for the most probable segmentation, instead of attempting to exhaust all possible segmentations of each sentence for a complicated full version of EM training.

The case-based learning works in a straightforward way. It first extracts case-based knowledge, as a set of context-dependent transformation rules, from the segmented training corpus, and then applies them to ambiguous strings in a test corpus in terms of the similarity of their contexts. The similarity is empirically computed in terms of the length of relevant common affixes of context strings.

The effectiveness of this integrated approach is verified by its outstanding performance on IV word identification. Its IV recall rate, ranging from 96% to 98%, stands at the top or the next to the top in all closed tests in which we have participated. Unfortunately, its overall performance is not sustainable at the same level, due to the lack of a module for OOV word detection.

This paper is intended to present the implementation of the system and analyze its performance and problems, aiming at exploration of directions for further improvement. The remaining sections are organized as follows. Section 2 presents the ngram model and its training with the EM algorithm, and Section 3 presents the case-based learning for disambiguation. The overall architecture of our system is given in Section 4, and its performance and problems are analyzed in Section 5. Section 6 concludes the paper and previews future work.

2 Ngram model and training

An ngram model can be utilized to find the most probable segmentation of a sentence. Given a Chinese sentence $s = c_1 c_2 \cdots c_m$ (also denoted as c_1^n), its probabilistic segmentation into a word sequence $w_1 w_2 \cdots w_k$ (also denoted as w_1^k) with the aid of an ngram model can be formulated as

$$\mathrm{seg}(s) = \underset{s = w_1 \circ w_2 \circ \cdots \circ w_k}{\arg\max} \prod_i^k p(w_i | w_{i-n+1}^{i-1}) \quad (1)$$

where $\circ$ denotes string concatenation, w_{i-n+1}^{i-1} the *context* (or *history*) of w_i, and n is the order of the ngram model in use. We have opted for uni-gram for the sake of simplicity. Accordingly, $p(w_i | w_{i-n+1}^{i-1})$ in (1) becomes $p(w_i)$, which is commonly estimated as follows, given a corpus $\mathcal{C}$ for training.

$$p(w_i) \doteq f(w_i) / \sum_{w \in \mathcal{C}} f(w) \quad (2)$$

In order to estimate a reliable $p(w_i)$, the ngram model needs to be trained with the EM algorithm using the available training corpus. Each EM iteration aims at approaching to a more reliable $f(w)$ for estimating $p(w)$, as follows:

$$f^{\mathrm{k}+1}(w) = \sum_{s \in \mathcal{C}} \sum_{s' \in \mathcal{S}(s)} p^{\mathrm{k}}(s')\, f^{\mathrm{k}}(w \in s') \quad (3)$$

where k denotes the current iteration, $\mathcal{S}(s)$ the set of all possible segmentations for s, and $f^{\mathrm{k}}(w \in s')$ the occurrences of w in a particular segmentation s'.

However, assuming that every sentence always has a segmentation, the following equation holds:

$$\sum_{s' \in \mathcal{S}(s)} p^{\mathrm{k}}(s') = 1 \quad (4)$$

Accordingly, we can adjust (3) as (5) with a normalization factor $\alpha = \sum_{s' \in \mathcal{S}(s)} p^{\mathrm{k}}(s')$, to avoid favoring words in shorter sentences too much. In general, shorter sentences have higher probabilities.

$$f^{\mathrm{k}+1}(w) = \sum_{s \in \mathcal{C}} \sum_{s' \in \mathcal{S}(s)} \frac{p^{\mathrm{k}}(s')}{\alpha}\, f^{\mathrm{k}}(w \in s') \quad (5)$$

Following the conventional idea to speed up the EM training, we turned to the Viterbi algorithm. The underlying philosophy is to distribute more probability to more probable events. The Viterbi segmentation, by utilizing dynamic programming techniques to go through the word trellis of a sentence efficiently, finds the most probable segmentation under the current parameter estimation of the language model, fulfilling (1)). Accordingly, (6) becomes

$$f^{\mathrm{k}+1}(w) = \sum_{s \in \mathcal{C}} p^{\mathrm{k}}(\mathrm{seg}(s))\; f^{\mathrm{k}}(w \in \mathrm{seg}(s)) \quad (6)$$

and (5) becomes

$$f^{\mathrm{k}+1}(w) = \sum_{s \in \mathcal{C}} f^{\mathrm{k}}(w \in \mathrm{seg}(s)) \quad (7)$$

where the normalization factor is skipped, for only the Viterbi segmentation is used for EM reestimation. Equation (7) makes the EM training with the Viterbi algorithm very simple for the unigram model: iterate word segmentation, as (1), and word count updating, via (7), sentence by sentence through the training corpus until there is a convergence.

Since the EM algorithm converges to a local maxima only, it is critical to start the training with an initial $f^0(w)$ for each word not too far away from its "true" value. Our strategy for initializing $f^0(w)$ is to assume all possible words in the training corpus as equiprobable and count each of them as 1; and then $p^0(w)$ is derived using (2). This strategy is supposed to have a weaker bias to favor longer words than maximal matching segmentation.

For the bakeoff, the ngram model is trained with the *unsegmented* training corpora together with the test sets. It is a kind of *unsupervised* training. Adding the test set to the training data is reasonable, to allow the model to have necessary adaptation towards the test sets. Experiments show that the training converges very fast, and the segmentation performance improves significantly from iteration to iteration. For the bakeoff experiments, we carried out the training in 6 iterations, because more iterations than this have not been observed to bring any significant improvement on segmentation accuracy to the training sets.

3 Case-based learning for disambiguation

No matter how well the language model is trained, probabilistic segmentation cannot avoid mistakes on ambiguous strings, although it resolves most ambiguities by virtue of probability. For the remaining unresolved ambiguities, however, we have to resort to other strategies and/or resources. Our recent study (Kit et al., 2002) shows that *case-based learning* is an effective approach to disambiguation.

The basic idea behind the case-based learning is to utilize existing resolutions for known ambiguous strings to do disambiguation if similar ambiguities occur again. This learning strategy can be implemented in two straightforward steps:

1. Collection of correct answers from the training corpus for ambiguous strings together with their contexts, resulting in a set of context-dependent transformation rules;

2. Application of appropriate rules to ambiguous strings.

A transformation rule of this type is actually an *example* of segmentation, indicating how an ambiguous string is segmented within a particular context. It has the following general form:

$$C^l \alpha \, C^r : \; \alpha \;\rightarrow\; w_1 \, w_2 \cdots w_k$$

where α is the ambiguous string, C^l and C^r its *left* and *right* contexts, respectively, and $w_1 \, w_2 \cdots w_k$ the correct segmentation of α given the contexts. In our implementation, we set the context length on each side to two words.

For a particular ambiguity, the example with the most similar context in the *example* (or, *rule*) *base* is applied. The similarity is measured by the sum of the length of the common suffix and prefix of, respectively, the left and right contexts. The details of computing this similarity can be found in (Kit et al., 2002) . If no rule is applicable, its probabilistic segmentation is retained.

For the bakeoff, we have based our approach to ambiguity detection and disambiguation rule extraction on the assumption that only ambiguous strings cause mistakes: we detect the discrepancies of our probabilistic segmentation and the standard segmentation of the training corpus, and turn them into transformation rules. An advantage of this approach is that the rules so derived carry out not only disambiguation but also error correction. This links our disambiguation strategy to the application of Brill's (1993) transformation-based error-driven learning to Chinese word segmentation (Palmer, 1997; Hockenmaier and Brew, 1998).

4 System architecture

The overall architecture of our word segmentation system is presented in Figure 1.

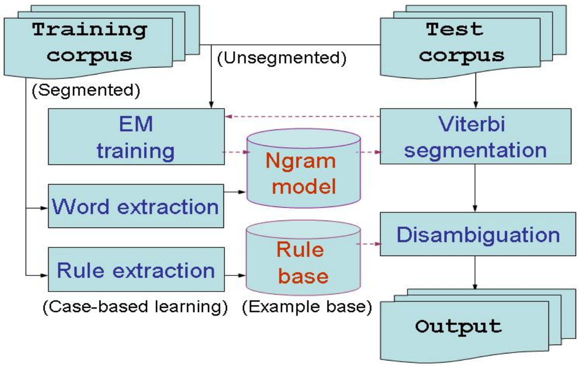

Figure 1: Overall architecture of the system

5 Performance and analysis

The performance of our system in the bakeoff is presented in Table 1 in terms of precision (P), recall (R) and F score in percentages, where "c" denotes closed tests. Its IV word identification performance is remarkable.

However, its overall performance is not in balance with this, due to the lack of a module for OOV word discovery. It only gets a small number of OOV words correct by chance. The higher OOV proportion in the test set, the worse is its F score. The relatively high R_{oov} for PK_c track is, mostly, the result of number recognition with regular expressions.

Test	P	R	F	OOV	R_{oov}	R_{iv}
SA_c	95.2	93.1	94.2	02.2	04.3	97.2
CTB_c	80.0	67.4	73.2	18.1	07.6	95.9
PK_c	92.3	86.7	89.4	06.9	15.9	98.0

Table 1: System performance, in percentages (%)

5.1 Error analysis

Most errors on IV words are due to the side-effect of the context-dependent transformation rules. The rules resolve most remaining ambiguities and correct many errors, but at the same time they also corrupt some proper segmentations. This side-effect is most likely to occur when there is inadequate context information to decide which rules to apply.

There are two strategies to remedy, or at least alleviate, this side-effect: (1) retrain probabilistic segmentation – a conservative strategy; or, (2) incorporate Brill's error-driven learning with several rounds of transformation rule extraction and application, allowing mistakes caused by some rules in previous rounds to be corrected by other rules in later rounds.

However, even worse than the above side-effect is a bug in our disambiguation module: it always applies the first available rule, leading to many unexpected errors, each of which may result in more than one erroneous word. For instance, among 430 errors made by the system in the SA closed test, some 70 are due to this bug. A number of representative examples of these errors are presented in Table 2, together with some false errors resulting from the inconsistency in the standard segmentation.

Errors	Standard	False errors	Standard
中的 (8)	中 的	第二次大戰	第二 次 大戰
只 是 (7)	只是	公元兩千年	公元 兩千年
不 能 (7)	不能	天然資源	天然 資源
個 人 (5)	個人	套裝軟體	套裝 軟體
只 有 (4)	只有	美其名爲	美其名 爲
才能 (4)	才 能	不虛此生	不 虛 此 生

Table 2: Errors and false errors

6 Conclusion and future work

We have presented our recent work for participation in ICWSB-1 based on a general-purpose ngram model for probabilistic word segmentation and a case-based learning strategy for disambiguation. The ngram model is trained using available unsegmented texts with the EM algorithm with the aid of Viterbi segmentation. The learning strategy acquires a set of context-dependent transformation rules to correct mistakes in the probabilistic segmentation of ambiguous substrings. This integrated approach demonstrates an impressive effectiveness by its outstanding performance on IV word identification. With elimination of the bug and false errors, its performance could be significantly better.

6.1 Future work

The above problem analysis points to two main directions for improvement in our future work: (1) OOV word detection; (2) a better strategy for learning and applying transformation rules to reduce the side-effect. In addition, we are also interested in studying the effectiveness of higher-order ngram models and variants of EM training for Chinese word segmentation.

Acknowledgements

The work is part of the CERG project "EBMT for HK Legal Texts" funded by HK UGC under the grant #9040482, with Jonathan J. Webster as the principal investigator and Chunyu Kit, Caesar S. Lun, Haihua Pan, King Kuai Sin and Vincent Wong as investigators. The authors wish to thank all team members for their contribution to this paper.

References

E. Brill. 1993. *A Corpus-Based Approach to Language Learning*. Ph.D. thesis, University of Pennsylvania, Philadelphia, PA.

A. P. Dempster, N. M. Laird, and D. B.Rubin. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society, Series B*, 34:1–38.

J. Hockenmaier and C. Brew. 1998. Error-driven learning of Chinese word segmentation. In *PACLIC-12*, pages 218–229, Singapore. Chinese and Oriental Languages Processing Society.

C. Kit, H. Pan, and H. Chen. 2002. Learning case-based knowledge for disambiguating Chinese word segmentation: A preliminary study. In *COLING2002* workshop*: SIGHAN-1*, pages 33–39, Taipei.

D. Palmer. 1997. A trainable rule-based algorithm for word segmentation. In *ACL-97*, pages 321–328, Madrid.

A. J. Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13:260–267.