Word Segmentation of Micro Blogs with Bagging

Zhenting Yu, Xin-Yu Dai(✉), Si Shen, Shujian Huang, and Jiajun Chen

State Key Laboratory for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China
{yuzt,shens}@nlp.nju.edu.cn, {daixinyu,huangsj,chenjj}@nju.edu.cn

Abstract. This paper describes the model we designed for the Chinese word segmentation Task of NLPCC 2015. We firstly apply a word-based perceptron algorithm to build the base segmenter. Then, we use a Bootstrap Aggregating model of bagging which improves the segmentation results consistently on the three tracks of closed, semi-open and open test. Considering the characteristics of Weibo text, we also perform rule-based adaptation before decoding. Finally, our model achieves F-score 95.12% on closed track, 95.3% on semi-open track and 96.09% on open track.

Keywords: Weibo · Word segmentation · Word-based perceptron model · Bagging

1 Introduction

Since Chinese sentences are written in continuous characters without explicit word boundaries, Chinese word segmentation (CWS) is a critical and a necessary initial step for most NLP tasks such as syntax parsing, information extraction and machine translation. At present, there are two main models for Chinese word segmentation "word-based" approach and "character-based" approach. In this paper, we prefer to use a word-baed model rather than a character based model, because word-based model may use more contextual information (Zhang and Clark 2011)[1].

Microblog is a new kind of broadcast medium in the form of blogging. A microblog differs from traditional blog in its smaller size. Furthermore, microblog text contains a large number of new words, name entities, punctation patterns (such as "..."), structured symbols representing conversation ("@") and topics ("#...#") etc. These characteristics take more challenges to microblog text segmentation.

In this paper, we use an averaged perceptron model as the base segmenter. We train several segmenters on the several sampling of the training data. Then we apply a Bootstrap Aggregating model of bagging for voting the segmentation results. We use this bagging strategy for the closed track. For the semi-open track, we incorporate more statistic-based features in the bagging model. For the open track, we combine a bagging model with extra lexicon features, unsupervised statistical features. Experimentally, our method gets F-score 95.12% on the closed track, 95.3% on the semi-open track and 96.09% on the open track, respectively.

J. Li et al. (Eds.): NLPCC 2015, LNAI 9362, pp. 573–580, 2015.
DOI: 10.1007/978-3-319-25207-0_54

2 System Description

In this section, we describle the details of our system. We use a bagging model to combine multiple segmenters. In each segmenter, preprocessing are conducted to recognize URLs, Emails, numbers, latin letters. Optional features like PMI, MI, Lexicon features can be included to enhance the performance of our system. And the system architecture is illustrated in fig.1.

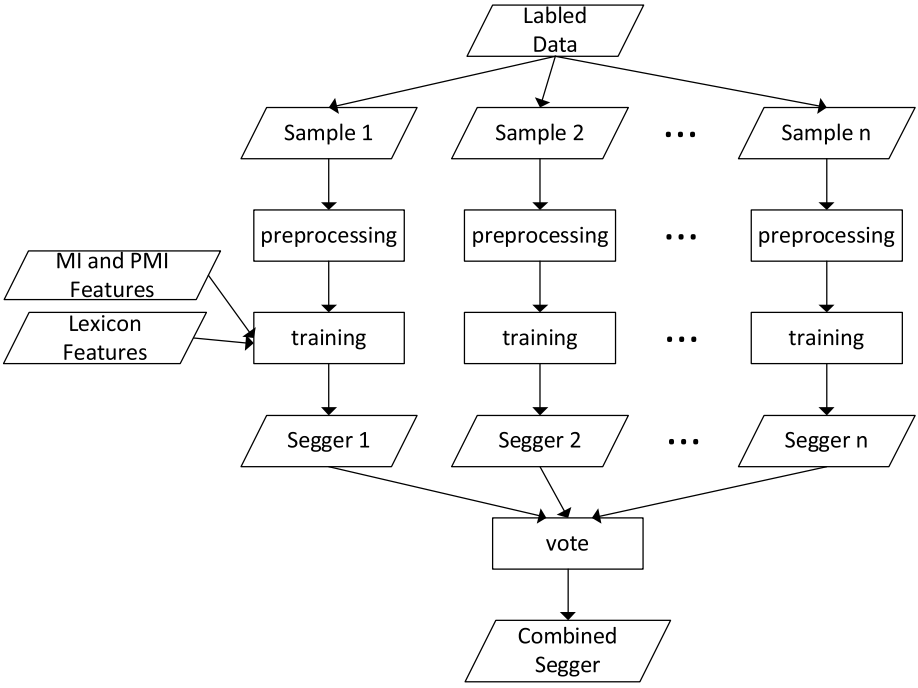

Fig. 1. System Architecture

2.1 Basic Model

In this paper, we use a word-based perceptron model. Word-based models read an input sentence from left to right and predict whether the current piece of continuous characters is a word. After a word is identified, this method moves on and searches for the next possible word. Zhang and Clark(2007)[2] firstly proposed a word-based approach for segmentation using a discriminative perceptron algorithm proposed by Michael Collins(2002)[3]. Compared with traditional character-based model, word-based perceptron model maps the CWS problem into an action sequence generation task (zhang 2012)[4] rather than a sequence labeling task (Xue 2003)[5]. There are two kinds of actions, **a** and **s**. The action a_i=**a** indicates the i-1-th character and the i-th character are in the same word. The action a_i=**s** indicates the *i-1*-th character and the i-th character are in two

separate words. The problem of finding a best segmented sentence is transformed into generating a best action sequence.

Given a sentence X, the output $\hat{y}$ is defined as the highest scored segment among the possible segments for X, shown as Equ.1:

$$\hat{y} = \underset{y \in GEN(X)}{\operatorname{argmax}} \; Score(y) \tag{1}$$

Where GEN(X) denotes the possible segments for an input sentence X and Score(y) is used to evaluate a segment.

Let us denote a possible action sequence as A=$\{a_0,a_1,...,a_{|x|}\}$. Then Score(y) can be computed as:

$$Score(y) = \sum_{i=0}^{|X|} \phi(X, s_i)w \tag{2}$$

Where $\phi(X, s_i)$ is feature vector generated by the input X, w is a parameter vector and s_i is the current state which is composed of the current index i, the current action a_i and the previous state with an action **s**.

For the decoding phase, we follow the beam search method presented by Zhang and Clark(2007)[2].

2.2 Feature Templates

We now describe the features used in our word based segmentation model.

Basic Features. We define three kinds of features as basic features, including character-based features, character-type-based features and word- based features. Particularly, we use full word templates only when the current action is **s**. The word-based features are mainly based on Zhang and Clark(2011)[1]. First we show atom features in Table 1 and specific basic feature templates are shown in Table 2. $|w|$ is the length of w. $w[0]$ if the first character of w and $w[-1]$ is the last character of w.

Totally, there are 6 character-based features, 6 character-type-based features and 12 word-based features.

Table 1. Atom Features

Atom Feautres	Description
x_i	the i-th character in X
a_i	the current action
c_i	the character type of x_i (alphabet, digit, punctuation or others)
w_{-1}	the last word before current character
w_{-2}	the word before last word

Table 2. Basic features for the word segmenter

character-based	$\langle$**c-1**$, x_i, a_i\rangle$, $\langle$**c-2**$, x_{i-1}, a_i\rangle$, $\langle$**c-3**$, x_{i+1}, a_i\rangle$
	$\langle$**c-4**$, x_{i-1}x_i, a_i\rangle$, $\langle$**c-5**$, x_ix_{i+1}, a_i\rangle$, $\langle$**c-6**$, x_{i-1}x_ix_{i+1}, a_i\rangle$
chartype-based	$\langle$**ct-1**$, c_i, a_i\rangle$, $\langle$**ct-2**$, c_{i-1}, a_i\rangle$, $\langle$**ct-3**$, c_{i+1}, a_i\rangle$
	$\langle$**ct-4**$, c_{i-1}c_i, a_i\rangle$, $\langle$**ct-5**$, c_ic_{i+1}, a_i\rangle$, $\langle$**ct-6**$, c_{i-1}c_ic_{i+1}, a_i\rangle$
word-based	$\langle$**w-1**$, w_{-1}\rangle$, $\langle$**w-2**$, \|w_{-1}\|\rangle$, $\langle$**w-3**$, w_{-1}, w_{-2}\rangle$
	$\langle$**w-4**$, w_{-1}[0], w_{-1}[-1]\rangle$, $\langle$**w-5**$, w_{-1}[0], \|w_{-1}\|\rangle$, $\langle$**w-6**$, w_{-1}[-1], \|w_{-1}\|\rangle$
	$\langle$**w-7**$, w_{-1}, w_{-1}[-1]\rangle$, $\langle$**w-8**$, w_{-2}, w_{-2}[-1]\rangle$, $\langle$**w-9**$, w_{-1}, \|w_{-2}\|\rangle$
	$\langle$**w-10**$, w_{-2}, \|w_{-1}\|\rangle$, $\langle$**w-11**$, w_{-1}, x_i\rangle$, $\langle$**w-12** $, w_{-1}[0], x_i\rangle$

Mutual Information. In probability theory and information theory, the mutual information of two random variables is a measure of the variables' mutual dependence. The large value of mutual information indicates two consecutive strings are more probable to be combined together, while small value of mutual information often means they are unlikely to be in a word.

In this paper, we follow Sun and Xu(2011)[7]'s definition of mutual information. For two continuous charaters x_ix_{i+1}, the mutual information between x_i and x_{i+1} is computed as below:

$$MI(x_i, x_{i+1}) = \log \frac{p(x_ix_{i+1})}{p(x_i)p(x_{i+1})} \tag{3}$$

For each character x_i, $MI(x_i, x_{i+1})$ and $MI(x_{i-1}, x_i)$ are computed and rounded to an integer. We include these two features as additonal feature template.

Accessor Variety. A string with various linguistic environments may be a meaningful word. This idea is first proposed as Accessor Variety by Feng(2004)[6] to extract meaningful words from unlabeled corpus. This criterion of Accessor Variety is used to evaluate how independently a string is. Sun and Xu(2011)[7] define Accessor Variety as a type of statistic-based feature. In this paper, we follow this study and the features are defined as follows.

For each character x_i, we have features $L^l_{AV}(x[i : i + l - 1])$,$L^l_{AV}(x[i + 1 : i + l])$,$R^l_{AV}(x[i - l + 1 : i])$,$R^l_{AV}(x[i - l : i])$ $(l = 2, 3, 4)$. Here left accessor variety $L^l_AV(s)$ means the number of distinct characters that precede s in a corpus and right accessor variety $R^l_AV(s)$ means the number of distinct characters that succeed s.

Lexicon Features. Empirical study shows lexicon features can enhance the performance of a segmenter. We use a feature template $Lex(s)$ where s is a string and Lex is a function to indicate whether s is in a lexicon or not.

We add several word lists to our lexicon, including SogouW[1] and a few finance, sports, entertainment related word lists from sogou's lexicon sharing website[2].

2.3 Bagging Model

Bootstrap aggregating (Bagging) is a machine learning ensemble meta-algorithm to improve classification and regression accuracy. It also reduces variance and helps to avoid overfitting. Given a training set D of size n, Bagging generates m new training set D_i of size $n' \leq n$, by samppling examples from D uniformly. The m models are fitted using the above m bootstrap samples and combined by voting (for voting) or average the output (for regression).

We use a Bagging model to combine multiple segmenters. In the training phase, given a training set D of size n, our model generates m new training sets D_i of size $80\% \times n$ by sampling sentences from D without replacement. We use each D_i to train a weak segmenter. Thus we can get m weak segmenters. In the segmentation phase, the m segmenters have m segmentation results, which are further transformed into action sequences. In other words, for each position we have m **a** or **s** actions. The final segmentation is the voting result of these m actions. We set m an odd number, because when m is even there may be equal number of **a** or **s** actions.

Our bagging model is mainly based on Sun(2010)[8] and the difference between us is we only use word-based weak segmenters.

2.4 Rule-Based Adaptation

It is worthy to note that, considering the characteristics of microblog text, We adapt a rule-based preprocessing before the statistical model.

URLs (like http://www.baidu.com), Emails (like nlp@nlp.nju.edu.cn) are first recognized. The boundaries of these components are assigned to **s**, while the inner character intervals of the URLs and Emails are assigned to **a**.

Likewise, the punctuations (such as Chinese full stop and comma) are recognized and the boundaries of these are assigned to **s**. The intervals between two Arabic numbers or two Latin letters are assigned to **a**.

By using preprocessing, we can assign some fixed action a_i to the certain positions of the recognized words before the decoding phase. Thus the search space of the statistical model can be reduced.

3 Experiments

In experiments, we firstly use 5-fold cross validation for the development and use the whole dataset to train the final model for the test data. The F-score

[1] https://www.sogou.com/labs/dl/w.html
[2] http://pinyin.sogou.com/dict/

is used to evaluate the performance of the word segmentation system. We follow Liu(2012)[9] and evaluate the effects of preprocessing, bagging and different feature templates.

3.1 Effect of Preprocessing and Bagging

In Table 3, we compare the results from our baseline model and baseline model with preprocessing. We can see that the performance is improved with the help of preprocessing. We also compare the results from baseline model and the bagging model. The results suggest that using the bagging method can obviously improve the performance.

Table 3. Effect of preprocessing and bagging

Model	Precision	Recall	F-score
Base	0.9405	0.9401	0.9403
Base + Pre	0.9418	0.9421	0.9419
Base + Pre + Bagging	0.9465	0.9466	0.9465

3.2 Effect of Statistic-Based Features

Table 4 shows the difference of combining different statistical features which are extracted from the given background data. It can be seen that all statistical features lead to the improvement on performance. But accessor variety features contribute to more improvement than mutual information features.

Table 4. Effect of Statistic-based features

Model	Precision	Recall	F-Score
Base + Pre	0.9418	0.9421	0.9419
Base + Pre + MI	0.9431	0.9427	0.9428
Base + Pre + AV	0.9463	0.9458	0.9460
Base + Pre + MI + AV	0.9471	0.9469	0.9470

3.3 Effect of Lexicon Features

We compare the results with lexicon features and without lexicon features. The results is shown in Table 5. With lexicon features, the model get an F-score 95.43% and an improvement of 1.24% comparing to the baseline model. As is expected, lexicon features greatly improve the performance and outperform all other statistical features.

Table 5. Effect of Lexicon features

Model	Precision	Recall	F-score
Base + Pre	0.9418	0.9421	0.9419
Base + Pre + MI + AV	0.9471	0.9469	0.9470
Base + Pre + Lexicon	0.9534	0.9552	0.9543

Table 6. Final Results

Track	Configuration	Precision	Recall	F-score
Closed	Pre+Bagging	0.9514	0.9509	0.9512
Semi-Open	Pre+Bagging+MI+AV	0.9530	0.9531	0.9530
Open	Pre+Bagging+MI+AV+Lexicon	0.9603	0.9615	0.9609

3.4 Final System

For the closed track, the configuration is set as "Pre+Bagging". For the semi-open track, the configuration is set as "Pre+Bagging+MI+AV". For the open track, the configuration is set as "Pre+Bagging+MI+AV+Lexicon". And the final results for the test data is shown in Table 6.

4 Conclusion

In this paper, we describe our system of Chinese Word Segmentation on microblog data. We exploit a bagging model which ensembles multiple weak segmenters. Rule-based preprocessing, statistic-based features and lexicon features are also used to enhance the performance of our model. Finally, our model achieves F-score 95.12% on closed track, 95.3% on semi-open track and 96.09% on open track, respectively.

In the future, we will try to apply novel words detection and name entities recognition to further enhance the performance on microblog text. Also some more complex enesemble methods can be appiled to our system.

Acknowledgments. We thank the anonymous reviewers for their insightful comments. This work was supported by the NSFC (61472183, 61333014) and the 863 program(2015AA015406).

References

1. Zhang, Y., Clark, S.: Syntactic processing using the generalized perceptron and beam search. Computational Linguistics **37**(1), 105–151 (2011)
2. Zhang, Y., Clark, S.: Chinese segmentation with a word-based perceptron algorithm. In: Proceedings of ACL, Prague, pp. 840–847 (2007)
3. Collins, M.: Discriminative training methods for hidden Markov models: theory and experiments with perceptron algorithms. In: Proceedings of EMNLP, Philadelphia, PA, pp. 1–8 (2002)
4. Zhang, K., Sun, M., Zhou, C.: Word segmentation on Chinese mirco-blog data with a linear-time incremental model. In: Second CIPS-SIGHAN Joint Conference on Chinese Language Processing (2012)

5. Xue, N.: Chinese word segmentation as character tagging. International Journal of Computational Linguistics and Chinese Language Processing **8**(1) (2003)
6. Feng, H., Chen, K., Deng, X., Zheng, W.: Accessor variety criteria for Chinese word extraction. Computational Linguistics **30**(1), 75–93 (2004)
7. Sun, W., Xu, J.: Enhancing Chinese word segmentation using unlabeled data. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics (2011)
8. Sun, W.: Word-based and character-based word segmentation models: comparison and combination. In: Coling 2010: Posters, Beijing, China, August, pp. 1211–1219. Coling 2010 Organizing Committee (2010)
9. Liu, Y., Che, W.: Micro blogs oriented word segmentation system. In: Second CIPS-SIGHAN Joint Conference on Chinese Language Processing (2012)