Soochow University: Description and Analysis of the Chinese Word Sense Induction System for CLP2010

Hua Xu Bing Liu Longhua Qian* Guodong Zhou
Natural Language Processing Lab
School of Computer Science and Technology
Soochow University, Suzhou, China 215006
Email:
{20094227034,20084227065055,qianlonghua,gdzhou}@suda.edu.cn

Abstract

Recent studies on word sense induction (WSI) mainly concentrate on European languages, Chinese word sense induction is becoming popular as it presents a new challenge to WSI. In this paper, we propose a feature-based approach using the spectral clustering algorithm to this problem. We also compare various clustering algorithms and similarity metrics. Experimental results show that our system achieves promising performance in F-score.

1 Introduction

Word sense induction (WSI) is an open problem of natural language processing (NLP), which governs the process of automatic discovery of the possible senses of a word. WSI is similar to word sense disambiguation (WSD) both in methods employed and in problem encountered. In the procedure of WSD, the senses are assumed to be known and the task focuses on choosing the correct one for an ambiguous word in a context. The main difference between them is that the task of WSD generally requires large-scale manually annotated lexical resources while WSI does not. As WSI doesn't rely on the manually annotated corpus, it has become one of the most important topics in current NLP research (Pantel and Lin, 2002; Neill, 2002; Rapp, 2003). Typically, the input to a WSI algorithm is a target word to be disambiguated. The task of WSI is to distinguish which target words share the same meaning when they appear in different contexts. Such result can be at the very least used as empirically grounded suggestions for lexicographers or as input for WSD algorithm. Other possible uses include automatic thesaurus or ontology construction, machine translation or information retrieval. Compared with European languages, the study of WSI in Chinese is scarce. Furthermore, as Chinese has its special writing style and Chinese word senses have their own characteristics, the methods that work well in English may not perform effectively in Chinese and the usefulness of WSI in real-world applications has yet to be tested and proved.

The core idea behind word sense induction is that contextual information provides important cues regarding a word's meaning. The idea dates back to (at least) Firth (1957) (“You shall know a word by the company it keeps”), and underlies most WSD and lexicon acquisition work to date. For example, when the adverb phrase occurring prior to the ambiguous word “把握”, then the target word is more likely to be a verb and the meaning of which is “to hold something”; Otherwise, if an adjective phrase locates in the same position, then it probably means “confidence” in English. Thus, the words surrounds the target word are main contributor to sense induction.

The bake off task 4 on WSI in the first CIPS-SIGHAN Joint Conference on Chinese Language Processing (CLP2010) is intended to promote the exchange of ideas among participants and improve the performance of Chinese WSI systems. Generally, our WSI system also adopts a clustering algorithm to group the contexts of a target word. Differently, after generat-

* Corresponding author

ing feature vectors of words, we compute a similarity matrix with each cell denoting the similarity between two contexts. Furthermore, the set of similarity values of a context with other contexts is viewed as another kind of feature vector, which we refer to as similarity vector. Both feature vectors and similarity vectors can be separately used as the input to clustering algorithms. Experimental results show our system achieves good performances on the development dataset as well as on the final test dataset provided by the CLP2010.

2 System Description

This section sequentially describes the architecture of our WSI system and its main components.

2.1 System Architecture

Figure 1 shows the architecture of our WSI system. The first step is to preprocess the raw dataset for feature extraction. After that, we extract "bag of words" from the sentence containing a target word (feature extraction) and transform them into high-dimension vectors (feature vector generation). Then, similarities of every two vectors could be computed based on the feature vectors (similarity measurement). the similarities of an instance can be viewed as another vector—similarity vector. Both feature vectors and similarity vectors can be served as the input for clustering algorithms. Finally, we perform three clustering algorithms, namely, k-means, HAC and spectral clustering.

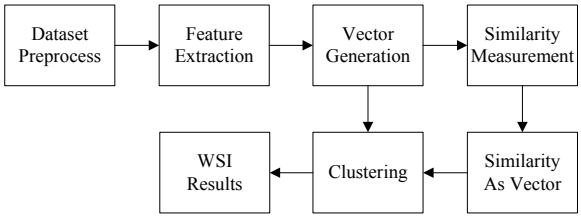

Figure 1 Architecture of our Chinese WSI system

2.2 Feature Engineering

In the task of WSI, the target words with their topical context are first transformed into multi-dimensional vectors with various features, and then applying clustering algorithm to detect the relevance of each other.

Corpus Preprocessing

For each raw file, we first extract each sentence embedded in the tag <instance>, including the <head> and </head> tags which are used to identify the ambiguous word. Then, we put all the sentences related to one target word into a file, ordered by their instance IDs. The next step is word segmentation, which segments each sentence into a sequence of Chinese words and is unique for Chinese WSI. Here, we use the software from Hylanda[1] since it is ready to use and considered an efficient word segmentation tool. Finally, since we retain the <head> tag in the sentence, the <head> and </head> tags are usually separated after word segmentation, thus we have to restore them in order to correctly locate the target word during the process of feature extraction.

Feature Extraction

After word segmentation, for a context of a particular word, we extract all the words around it in the sentence and build a feature vector based on a "bag-of-words" Boolean model. "Bag-of-words" means that we don't consider the order of words. Meanwhile, in the Boolean model, each word in the context is used to generate a feature. This feature will be set to 1 if the word appears in the context or 0 if it does not. Finally, we get a number of feature vectors, each of them corresponds to an instance of the target word. One problem with this feature-based method is that, since the size of word set may be huge, the dimension is also very high, which might lead to data sparsity problem.

Similarity measurement

One commonly used metric for similarity measurement is cosine similarity, which measures the angle between two feature vectors in a high-dimensional space. Formally, the cosine similarity can be computed as follows:

$$\cos ine\ similarity < \mathbf{x}, \mathbf{y} > = \frac{\mathbf{x} \cdot \mathbf{y}}{|\mathbf{x}| \cdot |\mathbf{y}|}$$

where $\mathbf{x}$, $\mathbf{y}$ are two vectors in the vector space and $|\mathbf{x}|, |\mathbf{y}|$ are the lengths of $\mathbf{x}$, $\mathbf{y}$ respectively.

[1] http://www.hylanda.com/

Some clustering algorithms takes feature vectors as the input and use cosine similarity as the similarity measurement between two vectors. This may lead to performance degradation due to data sparsity in feature vectors. To avoid this problem, we compute the similarities of every two vectors and generate an $N * N$ similarity matrix, where N is the number of all the instances containing the ambiguous word. Generally, N is usually much smaller than the dimension size and may alleviate the data sparsity problem. Moreover, we view every row of this matrix (i.e., an ordered set of similarities of an instance with other instances) as another kind of feature vector. In other words, each instance itself is regarded as a feature, and the similarity with this instance reflects the weight of the feature. We call this vector similarity vector, which we believe will more properly represent the instance and achieve promising performance.

2.3 Clustering Algorithm

Clustering is a very popular technique which aims to partition a dataset into such subgroups that samples in the same group share more similarities than those from different groups. Our system explores various cluster algorithms for Chinese WSI, including K-means, hierarchical agglomerative clustering (HAC), and spectral clustering (SC).

K-means (KM)

K-means is a very popular method for general clustering used to automatically partition a data set into k groups. K-means works by assigning multidimensional vectors to one of K clusters, where K is given as a priori. The aim of the algorithm is to minimize the variance of the vectors assigned to each cluster.

K-means proceeds by selecting k initial cluster centers and then iteratively refining them as follows:

(1) Choose k cluster centers to coincide with k randomly-chosen patterns or k randomly defined points.
(2) Assign each pattern to the closest cluster center.
(3) Recompute the cluster centers using the current cluster memberships.
(4) If a convergence criterion is not met, go to step 2.

Hierarchical Agglomerative Clustering (HAC)

Different from K-means, hierarchical clustering creates a hierarchy of clusters which can be represented in a tree structure called a dendrogram. The root of the tree consists of a single cluster containing all objects, and the leaves correspond to individual object.

Typically, hierarchical agglomerative clustering (HAC) starts at the leaves and successively merges two clusters together as long as they have the shortest distance among all the pair-wise distances between any two clusters.

Given a specified number of clusters, the key problem is to determine where to cut the hierarchical tree into clusters. In this paper, we generate the final flat cluster structures greedily by maximizing the equal distribution of instances among different clusters.

Spectral Clustering (SC)

Spectral clustering refers to a class of techniques which rely on the eigen-structure of a similarity matrix to partition points into disjoint clusters with points in the same cluster having high similarity and points in different clusters having low similarity.

Compared to the "traditional algorithms" such as K-means or single linkage, spectral clustering has many fundamental advantages. Results obtained by spectral clustering often outperform the traditional approaches, spectral clustering is very simple to implement and can be solved efficiently by standard linear algebra methods.

3 System Evaluation

This section reports the evaluation dataset and system performance for our feature-based Chinese WSI system.

3.1 Dataset and Evaluation Metrics

We use the CLP2010 bake off task 4 sample dataset as our development dataset. There are 2500 examples containing 50 target words and each word has 50 sentences with different meanings. The exact meanings of the target words are blind, only the number of the meanings is provided in the data. We compute the system per-

formance with the sample dataset because it contains the answers of each candidate meaning. The test dataset provided by the CLP2010 is similar to the sample dataset. It contains 100 target words and 5000 instances in total. However, it doesn't provide the answers.

The F-score measurement is the same as Zhao and Karypis (2005). Given a particular class L_r of size n_r and a particular cluster S_i of size n_i, suppose n_{ir} in the cluster S_i belong to L_r, then the F value of this class and cluster is defined to be

$$F(L_r, S_i) = \frac{2 \times R(L_r, S_i) \times P(L_r, S_i)}{R(L_r, S_i) + P(L_r, S_i)}$$

$$R(L_r, S_i) = n_{ir} / n_r$$

$$P(L_r, S_i) = n_{ir} / n_i$$

where $R(L_r, S_i)$ is the recall value and $P(L_r, S_i)$ is the precision value. The F-score of class L_r is the maximum F value and F-score value follow:

$$F\text{-}score(L_r) = \max_{S_i} F(L_r, S_i)$$

$$F - score = \sum_{r=1}^{c} \frac{n_r}{n} F - score(L_r)$$

where c is the total number of classes and n is the total size.

3.2 Experiment Results

Table 1 reports the F-score of our feature-based Chinese WSI for different feature sets with various window sizes using K-means clustering. Since there are different results for each run of K-means clustering algorithm, we perform 20 trials and compute their average as the final results. The columns denote different window size *n*, that is, the *n* words before and after the target word are extracted as features. Particularly, the size of infinity (∞) means that all the words in the sentence except the target word are considered. The rows represent various combinations of feature sets and similarity measurements, currently, four of which are considered as follows:

F-All: all the words are considered as features and from them feature vectors are constructed.

F-Stop: the top 150 most frequently occurring words in the total "word bags" of the corpus are regarded as stop words and thus removed from the feature set. Feature vectors are then formed from these words.

S-All: the feature set and the feature vector are the same as those of F-All, but instead the similarity vector is used for clustering (c.f. Section 2.2).

S-Stop: the feature set and the feature vector are the same as those of F-Stop, but instead the similarity vector is used for clustering.

Feature/ Similarity	3	7	10	∞
F-All	0.5949	0.6199	0.6320	0.6575
F-Stop	0.6384	0.6500	0.6493	0.6428
S-All	0.5856	0.6044	0.6186	0.6843
S-Stop	**0.6532**	**0.6696**	**0.6804**	**0.7320**

Table 1 Experimental results for different feature sets with different window sizes using K-means clustering

This table shows that S-Stop achieves the best performance of 0.7320 in F-score. This suggests that for K-means clustering, Chinese WSI can benefit much from removing stop words and adopting similarity vector. It also shows that:

- As the window size increases, the performance is almost consistently enhanced. This indicates that all the words in the sentence more or less help disambiguate the target word.
- Removing stop words consistently improves the F-score for both similarity metrics. This means some high frequent words do not help discriminate the meaning of the target words, and further work on feature selection is thus encouraged.
- Similarity vector consistently outperforms feature vector for stop-removed features, but not so for all-words features. This may be due to the fact that, when the window size is limited, the influence of frequently occurring stop words is relatively high, thus the similarity vector misrepresent the context of the target word. On the contrary, when stop words are removed or the context is wide, the similarity vector can better reflect the target word's context, leading to better performance.

In order to intuitively explain why the similarity vector is more discriminative than the feature vector, we take two sentences containing

the Chinese word “把握” (hold, grasp) as an example (Figure 2). These two sentences have few common words, so clustering via feature vectors puts them into different classes. However, since the similarities of these two feature vectors with other feature vectors are much similar, clustering via similarity vectors group them into the same class.

```
<lexelt item="把握" snum="4">
<instance id="0012">
当一个人有了跳槽想法后，很自然他（她）确需要找到一个合适的时机。实际上，<head>把握</head>“时”和“机”都非常重要。
</instance>
<instance id="0015">
无论是在球场上还是学习中，现在的李纵横都表现得非常自信，“人生要懂得<head>把握</head>机会，一次机会或许可以使你的一生发生转变，然而这样的前提就是参与。”
</instance>
</lexelt>
```

Figure 2 An example from the dataset

According to the conclusion of the above experiments, it is better to include all the words except stop words in the sentence as the features in the subsequent experiment. Table 2 lists the results using various clustering algorithms with this same experimental setting. It shows that the spectral clustering algorithm achieves the best performance of 0.7692 in F-score for Chinese WSI using the S-All setup. Additionally, there are some interesting findings:

- Although SC performs best, KM with similarity vectors achieves comparable results of 0.7320 units in F-score, slightly lower than that of SC.
- HAC performs worst among all clustering algorithms. An observation reveals that this algorithm always groups the instances into highly skewed clusters, i.e., one or two clusters are extremely large while others usually have only one instance in each cluster.
- It is surprising that S-All slightly outperforms F-All by only 0.0006 units in F-score. The truth is that, as discussed in the first experiment, KM using F-All doesn’t consider instance density while S-All does. On the contrary, SC identifies the eign-structure in the instance space and thus already considers the density information, therefore S-All will not significantly improve the performance.

Feature/ Similarity	KM	HAC	SC
F-All	0.6428	0.6280	0.7686
S-All	0.7320	0.6332	**0.7692**

Table 2 Experiments results using different clustering algorithms

3.3 Final System Performance

For the CLP2010 task 4 test dataset which contains 100 target words and 5000 instances in total, we first extract all the words except stop words in a sentence containing the target word, then produce the feature vector for each context and generate the similarity matrix, finally we perform the spectral cluster algorithm. Probably because the distribution of the target word in the test dataset is different from that in the development dataset, the F-score of our system on the test dataset is 0.7108, about 0.05 units lower than that we got on the sample dataset.

4 Conclusions and Future Work

In our Chinese WSI system, we extract all the words except stop words in the sentence, construct feature vectors and similarity vectors, and apply the spectral clustering algorithm to this problem. Experimental results show that our simple and efficient system achieve a promising result. Moreover, we also compare various clustering algorithms and similarity metrics. We find that although the spectral clustering algorithm outperforms other clustering algorithms, the K-means clustering with similarity vectors can also achieve comparable results.

For future work, we will incorporate more linguistic features, such as base chunking, parse tree feature as well as dependency information into our system to further improve the performance.

Acknowledgement

This research is supported by Project 60873150, 60970056 and 90920004 under the National Natural Science Foundation of China. We would also like to thank other contributors in the NLP lab at Soochow University.

References

Jain A, Murty M. 1999.Flynn P. *Data clustering : A Review* [J]. ACM Computing Surveys,1999,31 (3) :2642323

F. Bach and M. Jordan.2004. *Learning spectral clustering.* In Proc. of NIPS-16. MIT Press, 2004.

Samuel Brody and Mirella Lapata. 2009. *Bayesian word sense induction.* In Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009), pages 103–111.

Neill, D. B. 2002. *Fully Automatic Word Sense Induction by Semantic Clustering*. Cambridge University, Master's Thesis, M.Phil. in Computer Speech.

Agirre, E. and Soroa, A. 2007. *Semeval-2007 task 02: Evaluating word sense induction and discrimination systems.* In Proceedings of the 4th International Workshop on Semantic Evaluations:7-12

Ioannis P. Klapaftis and Suresh Manandhar. 2008. *Word sense induction using graphs of collocations.* In Proceedings of the 18th European Conference On Artificial Intelligence (ECAI-2008), Patras, Greece, July. IOS Press.

Kannan, R., Vempala, S and Vetta, A. 2004. *On clusterings: Good, bad and spectral*. J. ACM, 51(3), 497–515.

Reinhard Rapp.2004. *A practical solution to the problem of automatic word sense induction.* Proceedings of the ACL 2004 on Interactive poster and demonstration sessions, p.26-es, July 21-26, 2004, Barcelona, Spain

Bordag, S. 2006. *Word sense induction: Triplet-based clustering and automatic evaluation.* In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL, Trento, Italy). 137--144.

Ying Zhao, and George Karypis.2005. *Hierarchical Clustering Algorithms for Document Datasets.* Data Mining and Knowledge Discovery, 10, 141–168.