Knowledge-based Word Sense Disambiguation with Feature Words Based on Dependency Relation and Syntax Tree

Heyan Huang, Wenpeng Lu

1, School of Computer, Beijing Institute of Technology, Beijing, China
Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Application (BIT), Beijing, China, hhy63@bit.edu.cn

2, School of Computer, Beijing Institute of Technology, Beijing, China
School of Science, Shandong Polytechnic University, Jinan, China, luwpeng@bit.edu.cn

Abstract

Context feature words are important for word sense disambiguation (WSD). There are two kinds of methods to extract feature words: window-based and dependency relation method. Both of them have some defects. In order to solve the problems of the existing methods, this paper proposes a knowledge-based WSD method which obtains context feature words by dependency relation and syntax tree. Firstly, according to dependency relation between context words and the ambiguous word, the layer relation and path distance in the phrase structure syntax tree, direct distance in the sentence, feature words are selected from the context and are assigned different WSD weights. Secondly, Based on WordNet (WN), semantic relatedness between each sense of the ambiguous word and feature words are computed and the sense with most semantic relatedness is selected as the right sense. Evaluation is performed over a publicly available lexical sample dataset. The results show that our WSD method is better than the methods that obtain feature words with window or dependency relation. The method is a preferred strategy to select feature words to disambiguate the target words.

Keywords: Word Sense Disambiguation, Dependency Relation, Phrase Structure Parsing, Syntax Tree

1. Introduction

Word sense disambiguation (WSD) is the technology that automatically chooses the intended sense of a word based on its context. It has been a central and difficult issue in natural language processing (NLP) for years, and it has been shown to be useful in several NLP tasks such as machine translation[1], information retrieval[2] and question answering[3,4,5]. WSD is one of basic problems in computational linguistics, which is crucial for natural language understanding.

As Firth said, “You shall know a word by the company it keeps.”[6] The sense of an ambiguous word is related to the words in its context. The context of the ambiguous word includes the various features that are crucial for WSD, such as lexical feature, syntactic feature, semantic feature and discourse feature[7]. How to choose the effective features and assign the appropriate weights to them is a key problem, which would directly affect the performance of WSD. In this paper, we focus on the selection of lexical feature. In general, we can select feature words with context window or dependency relation as described in section 2. The former is easy to be implemented, but it may induce some noise words. The latter can exactly select feature words, but it is confused by the paucity of feature words.

In order to solve the defects of existing methods, this paper proposes a knowledge-based WSD method which selects feature words based on dependency relation and syntax tree. The key premise of our work is that the words that have dependency relation and the words that are adjacent in syntax tree (that is phrase structure parsing tree) are more related than others. Different with the existing methods, our method selects feature words based on dependency relation and syntax tree, and assigns them different WSD weights. With the method of McCarthy et al.[8], based on WordNet3.0[9], feature words are used to compute the relatedness of each sense of the ambiguous words. The sense with the most relatedness is selected as the right sense.

In order to evaluate the effectiveness of our method, experiments have been done on a publicly available lexical sample dataset of 41 words[10]. The experimental results demonstrate the good performance of our method, which is better than the methods that select feature words based on dependency relation or based on context window.
The rest of this paper is structured as follows. Section 2 presents an overview of existing work on WSD. Section 3 introduces our method for WSD with feature words that are selected based on dependency relation and syntax tree. The experiments and results are presented in Section 4. Finally, the conclusions are drawn and further work is mentioned.

2. Related work

Methods of WSD can be divided into knowledge-based, supervised and unsupervised method. Knowledge-based methods[2, 8, 11] mainly use dictionary knowledge like gloss overlaps or sense relatedness measures[12] to choose the sense with the maximum relatedness with its context. Supervised systems[13] need to be trained with sense-tagged corpus, learn the relationship between the specific sense and the context. Unsupervised approaches[2, 14] exploit the notion that words with similar sense would be likely to occur in the similar context, cluster words based on their contexts and use the clusters as sense label. Supervised approaches have the best accuracy, but it must be trained with large-scale sense-tagged corpus which is expensive and hard to be found. This makes that supervised approaches are difficult to be applied in large-scale WSD task. Unsupervised approaches use the clusters as sense inventory, which are often used for word sense identification. In this paper, we utilize WordNet3.0[9] as sense inventory and focus on knowledge-based method. The selection of feature words is the main work of the paper.

Patwardhan et al.[15] and Pederson et al.[16] utilize the context words to disambiguate the ambiguous word. Both of them collect context words with window method in which $2N$ content words surrounding the ambiguous word are selected as feature words. Then, based on WordNet, they calculate the score of relatedness between each sense of the ambiguous word and the senses of the feature words. The sense with the maximum sum of relatedness score is selected as the right sense of the ambiguous word.

McCarthy et al.[17] present a WSD method based on distributional similarity and semantic relatedness. In the method, feature words are selected based on dependency relation. The method needs to do dependency parsing for the sentences in the corpus in order to get the dependency relations among the words. According to the dependency relations, the method proposed by Lin[18] is used to compute the distributional similarity and obtain the most top-$N$ similar words as feature words. The distributional similarity is used as the WSD weight of the feature word. Then, the scores of relatedness between each sense of the target word and the senses of the feature words are calculated, and the weighted sum of relatedness is computed. Based on the weighted sum of relatedness, the right sense is selected. Similar with McCarthy et al, Agirre et al.[11] also collect the feature words based on dependency parsing, but they compute the relative importance of each sense of the target word with personalized PageRank to disambiguate the target word. Lu et al.[19] propose a WSD method based on dependency relation and Bayes model, in which the feature words are also selected by dependency relation.

To summarize previous works, the methods to select feature words can be divided into either window-based or dependency relation method. The window-based method simply collects content words surrounding the ambiguous word within the window as feature words. The dependency relation method only selects content words which have direct dependency relation with the ambiguous word as feature words. The window-based method is easy to implement, but it is likely to induce some irrelevant noise words which would affect the accuracy of disambiguation. The dependency relation method can exactly select the feature words, as it takes advantage of the results of dependency parsing. But it often obtains few feature words and some effective words with important WSD information may be missing. This would lead to the failure of disambiguation.

In order to solve the problems that exist in the selection of context feature words in knowledge-based WSD, this paper proposes a novel WSD method which obtains the context feature words with dependency relation and syntax tree.
3. Proposed method

3.1. Framework of WSD method

The proposed knowledge-based WSD method can be divided into two steps. The framework of the method is as Figure 1. The first step is the selection of context feature words. According to dependency relations between context words and the ambiguous word, the hierarchy relation and path distance in the phrase structure parsing tree, direct distance in the sentence, feature words are selected from the context and are assigned different WSD weights. The second step is to select the right sense of the ambiguous word. We calculate the semantic relatedness between each sense of the ambiguous word and the senses of the feature words, and choose the sense with most semantic overall relatedness as the right sense of the ambiguous word. In this paper, we focus on the first step, as our objective is mainly to propose a novel way to select the effective feature words.

![Figure 1: Framework of WSD Method](image)

3.2. Feature words selection

As is shown in Fig.1, there are two different sources to select feature words. Firstly, based on results of dependency parsing, the content words that have direct dependency relation with the ambiguous word are selected as feature words. Secondly, according to hierarchy structure of phrase structure parsing tree, from the layer where the leaf node of the ambiguous word lies to the layer of root node, the neighbor words are collected layer by layer as feature words. According to relative position between the feature word and ambiguous word, which includes hierarchy relation, path distance in syntax tree and direct distance in the sentence, the weight of the feature word is assigned. Then, feature words are sorted in descending order by weight. Top-N feature words are selected to disambiguate the target word.

3.2.1. Selection of feature words based on dependency relation

The theory of dependency grammar has been established by Tesnière[20]. As dependency grammar can get the dependency relations among the words in the sentence, the theory has been paid more and more attention to in recent years. Syntax structure is represented by dependency relation between a word (known as a governor or a head) and its dependents. The results of dependency parsing can be represented with the triples. The form is as follow: relation ( governor, dependent ).

Here is an example sentence: *1,700 coaches and 25 special trains brought workers from Italy to Roma*. The results of dependency parsing for the example are as follow:

```
num(coaches-2, 1,700-1)
nsubj(brought-7, coaches-2)
um(trains-6, 25-4)
amod(trains-6, special-5)
cc(coaches-2, and-3)
conj (coaches-2, trains-6)
dobj(brought-7, workers-8)
prep(brought-7, from-9)
pobj(from-9, Italy-10)
prep(brought-7, to-11)
```
According to these dependency triples, we can exactly extract the content words that have direct dependency relation with the target word as feature words. In this example, for the target word “coach”, “bring” and “train” are its feature words. Based on the semantic relatedness between “coach” and “train”, the sense of “coach” can be determined.

In our method, based on the results of dependency parsing, we firstly extract feature words that have direct relation with the ambiguous word. In the view of syntax relation, this kind of feature words has the most relatedness with the ambiguous word among all of the words in the sentence, so they would be assigned the maximum WSD weight.

### 3.2.2. Selection of feature words based on dependency relation and syntax tree

The method to select feature words based on dependency relation can exactly extract feature words from a sentence. But, the kind of feature words that have direct dependency relation is few, so the method only gets a small number of feature words. In some cases, the method may fail to extract effective feature words, or the number of feature words is not enough for WSD. This would lead to the failure of WSD.

In order to improve the defect, feature words should be expanded with the words that have close relations with the ambiguous word. The syntax relations among leaf nodes that have the same ancestor and are adjacent in the syntax tree are closer than the relations among others. So, we propose the method that selects feature words based on dependency relation and syntax tree, which is as follow.

---

**The algorithm to select feature words based on dependency relation and syntax tree**

**Input:** the phrase structure syntax tree of the sentence *ParseTree*, the ambiguous word *TargetWord*, the feature words that have been selected based on dependency relation *DependSet*

**OutPut:** the feature words that are selected with method based on syntax tree *SyntaxSet*

**Step1.** Get the leaf node(*TargetWordNode*) of *TargetWord* in *ParseTree*, and initialize the variable *CurNode* with it. Besides, initialize *SyntaxSet* with NULL.

**Step2.** Based on syntax tree, expand from current node to its father node. Get the father node(*Father*) of *CurNode*, if *Father* is null, return *SyntaxSet*; otherwise, go to step 3.

**Step3.** Get the set of the child nodes of *Father* – {child}. If the size of {child} is 1, save *Father* to *CurNode*, go to step 2; otherwise, add 1 to the layer number variable – layer, go to step 4. (Note that, For layer, only the layer that has more than one child is counted.)

**Step4.** Traverse {child} to select context feature words in this iterator of expansion.

  For each child in {child}
    Initialize *LeafSet* with the leaf nodes of child
    If *TargetWordNode* ∉ *LeafSet*
      For each leafnode in *LeafSet*
        If leafnode ∉ *DependSet*
          Record the three kinds of information for leafnode (layer number, path distance between the word and *TargetWord* in syntax tree, direct distance between the word and *TargetWord* in the sentence)
          Add leafnode to *SyntaxSet*
        Endif
      EndFor
    Endif
  EndFor

**Step5.** Go to Step 2.

---

Here is another example sentence: *1,700 coaches produced by heavy duty truck corporation brought workers from Italy to Roma*. For the sentence, only “produce” and “bring” can be extracted as feature words with the method based on dependency relation. But, the two words have little help to distinguish the sense of “coach”. Obviously, “trunk” is the best feature word to “coach”. Figure 2 shows the phrase structure syntax tree of the sentence. In this example, “trunk” can be collected after two layers are expanded. Its layer number is 2, path distance is 8 and direct distance is 5. With the algorithm, all of
the content words in the sentence would be collected as feature words. According to the difference of layer number, path distance and direct distance of the feature words, they would be assigned different WSD weights.

### 3.2.3. Assignment of WSD weight for feature words

Feature words are collected by method based on dependency relation and syntax tree. Feature words based on dependency relation have closest semantic relation with ambiguous word, so they are assigned the maximum WSD weight which is 1 in our method. According to layer relation, path distance and direct distance, for a feature word \( f \), its WSD weight is assigned with Equation 1.

\[
weight(f) = \alpha' \cdot \frac{1}{1 + \log p} \cdot \frac{1}{1 + \log d}
\]

In the equation, \( l \) represents layer number, \( p \) represents path distance in syntax tree, \( d \) represents direct distance. \( \alpha', \beta, \gamma \) are tuning factors. \( 0 < \alpha < 1 \), this means that if more layers are expanded, feature words in that layer would be assigned smaller weight. \( \gamma > \beta > 1 \), this means that path distance has more affect to WSD weight than direct distance. In our method, we set \( \alpha, \beta, \gamma \) to 0.9, 2, 4 which get the best performance in our experiments.

With the method, feature words are assigned different WSD weights and are sorted in descending order. The top-\( N \) feature words would be selected to disambiguate the target words.

### 3.3. Sense selection

With the method mentioned in section 3.2, context feature words are collected and sorted by WSD weights. The next work is to select the right sense of target word with these feature words. The right sense of target word should be the one that has most relatedness with context feature words. We adopt the method described in McCarthy et al.[8] to select the most related sense. WordNet 3.0[9] is used as sense inventory. Each sense of target word is assigned a relatedness score which sums the semantic relatedness scores between feature word and the sense of target word. The semantic relatedness is weighted by the WSD weight of the target word and is normalized by the sum of semantic relatedness scores between all senses of the target word and the senses of the feature words that maximizes this score. For each sense, the relatedness score is computed with Equation 2.
relatedness score\( (w_{s_i}) = \sum_{f_j \in F_w} \text{weight}(f_j) \times \frac{\text{wnss}(w_{s_i}, f_j)}{\sum_{w_{s_j} \in \text{senses}(w)} \text{wnss}(w_{s_j}, f_j)} \) \hspace{1cm} (2)

Where:

\[ \text{wnss}(w_{s_i}, f_j) = \max_{f_s \in \text{senses}(f_j)} (\text{wnss}(w_{s_i}, f_s)) \] \hspace{1cm} (3)

In Equation 2, \( w_{s_i} \) is the \( i \)-th sense of the target word \( w \), \( \text{senses}(w) \) is the sense set of \( w \), \( w_{s_i} \in \text{senses}(w) \); \( f_j \) is the \( j \)-th feature word of \( w \), \( F_w \) is the set of feature words of \( w \), \( f_j \in F_w \); \( \text{weight}(f_j) \) is the WSD weight of the \( j \)-th feature word. Equation 3 means that when \( \text{wnss}(w_{s_i}, f_j) \) is computed, the sense of feature word that maximize the relatedness score with \( w_{s_i} \) is selected.

### 4. Experiment

In order to verify the performance of our method, we evaluate the method on the dataset published by Koeling et al.[10] We have done the experiments to compare the performance of the three WSD approaches, which are the method based on context window, the method based on dependency relation and our method based on dependency relation and syntax tree.

#### 4.1. Dataset and evaluation measure

The evaluation dataset is a test dataset for lexical sample task and is used widely for the specific domain WSD, which consists of instances retrieved from the sport and finance sections of Reuter Corpus, and from the balanced British National Corpus (BNC). There are 41 words in the dataset. Each word has about 100 instances for each specific domain and there are 11426 instances in total. The words are quite polysemous and difficult to disambiguate, with an average polysemy of 6.7 senses, ranging from 2 to 13 senses. Each of instances is annotated by two or three taggers with senses from WordNet1.7.1, yielding an inter-tagger agreement of 65\%. In our experiments, we only select the 3216 instances that are retrieved from BNC and have gotten sense agreement by the majority of taggers. As the original dataset are tagged with WordNet1.7.1, we map the original senses to the senses of WordNet3.0.

We have done the syntax parsing with Stanford parser[21], by which the dependency triples and syntax tree can be easily gotten. Based on the triples and syntax tree, we can get the context feature words with the method mentioned in section 3.2. The semantic relatedness is computed by WN similarity package (v2.05)[22]. The package provides six similarity measures and four relatedness measures. The similarity measures are used to the comparisons of the words with same POS (such as, noun and noun or verb and verb). The relatedness measures are used to the comparisons of the words with different POS. In our experiments, we need to compute the semantic relatedness with different POS, so we adopt the context vector relatedness measure[23]. With the method mentioned in section 3.3, based on semantic relatedness, the right sense of ambiguous word is selected.

The evaluation measures for WSD are accuracy, recall, coverage and F-Measure. Among them, recall is the most important. In our experiments, we use recall as evaluation measure, which is computed with Equation 4.

\[ R = \frac{M}{N} \times 100\% \] \hspace{1cm} (4)

In Equation 4, \( N \) is the total number of occurrences and \( M \) is the number of correct occurrences.
4.2. Experiment results

In the experiments, we utilize three different methods to get the feature words for the target word, and compare their performances of WSD.

The first method (Win) is the WSD method that gets feature words based on context window. In the experiment, the size of windows is set to $2N$. Centered with the target word, $N$ content words before and $N$ content words following the target words are selected as feature words. The feature words are unordered, whose weights of WSD are uniformly set to 1.

The second method (Depend) is the WSD method that gets feature words based on dependency relation. In the sentence, all the content words that have direct dependency relation with target word are selected as feature words, whose weight of WSD is also uniformly set to 1. Because this kind of feature words is few, we take all of them as feature words without restricting the number of them.

The last method (DepTree) is our WSD method that gets feature words based on dependency relation and syntax tree. As mentioned in section 3.2, the method selects all the content words in the sentence as feature words and assigns different WSD weight for each of them. In the experiment, only top-$N$ feature words are chosen to disambiguate the target word.

Based on Equation 2, for each sense of the target word, we compute its relatedness with each of the feature words and select the sense with the maximum weighted sum as the right sense. The performance of Win and DepTree are affected by the number of feature words. For Win, the best performance is achieved when windows size is 16. For DepTree, the best performance is achieved when $N$ is 8. We compare the best performance of the three methods. The recall is as Table 1.

<table>
<thead>
<tr>
<th>Word</th>
<th>Win</th>
<th>Depend</th>
<th>DepTree</th>
<th>Word</th>
<th>Win</th>
<th>Depend</th>
<th>DepTree</th>
</tr>
</thead>
<tbody>
<tr>
<td>bank</td>
<td>74.19</td>
<td>55.91</td>
<td>74.19</td>
<td>phase</td>
<td>30.26</td>
<td>26.31</td>
<td>28.94</td>
</tr>
<tr>
<td>bill</td>
<td>60.64</td>
<td>45.74</td>
<td>54.26</td>
<td>pitch</td>
<td>85.19</td>
<td>74.07</td>
<td>86.42</td>
</tr>
<tr>
<td>bond</td>
<td>57.61</td>
<td>50.00</td>
<td>55.43</td>
<td>receiver</td>
<td>20.00</td>
<td>23.16</td>
<td>25.26</td>
</tr>
<tr>
<td>check</td>
<td>15.63</td>
<td>31.25</td>
<td>15.63</td>
<td>record</td>
<td>8.00</td>
<td>16.00</td>
<td>13.33</td>
</tr>
<tr>
<td>chip</td>
<td>81.61</td>
<td>68.97</td>
<td>80.46</td>
<td>reserve</td>
<td>42.11</td>
<td>28.95</td>
<td>40.79</td>
</tr>
<tr>
<td>club</td>
<td>72.60</td>
<td>47.95</td>
<td>69.86</td>
<td>return</td>
<td>30.14</td>
<td>20.55</td>
<td>30.14</td>
</tr>
<tr>
<td>coach</td>
<td>41.57</td>
<td>34.83</td>
<td>43.82</td>
<td>right</td>
<td>36.05</td>
<td>34.88</td>
<td>37.21</td>
</tr>
<tr>
<td>competition</td>
<td>38.64</td>
<td>35.23</td>
<td>38.64</td>
<td>running</td>
<td>55.88</td>
<td>38.24</td>
<td>48.53</td>
</tr>
<tr>
<td>conversion</td>
<td>27.85</td>
<td>26.58</td>
<td>26.58</td>
<td>score</td>
<td>10.59</td>
<td>17.65</td>
<td>15.29</td>
</tr>
<tr>
<td>country</td>
<td>7.53</td>
<td>11.83</td>
<td>5.38</td>
<td>share</td>
<td>30.34</td>
<td>32.58</td>
<td>31.46</td>
</tr>
<tr>
<td>crew</td>
<td>27.85</td>
<td>29.11</td>
<td>29.11</td>
<td>star</td>
<td>16.67</td>
<td>16.67</td>
<td>19.05</td>
</tr>
<tr>
<td>delivery</td>
<td>15.05</td>
<td>15.05</td>
<td>18.28</td>
<td>strike</td>
<td>24.68</td>
<td>29.87</td>
<td>29.87</td>
</tr>
<tr>
<td>division</td>
<td>27.40</td>
<td>28.77</td>
<td>28.77</td>
<td>strike</td>
<td>24.68</td>
<td>29.87</td>
<td>29.87</td>
</tr>
<tr>
<td>fan</td>
<td>56.63</td>
<td>53.01</td>
<td>56.63</td>
<td>strike</td>
<td>5.00</td>
<td>6.00</td>
<td>5.00</td>
</tr>
<tr>
<td>fishing</td>
<td>58.75</td>
<td>57.50</td>
<td>58.75</td>
<td>target</td>
<td>57.53</td>
<td>41.10</td>
<td>57.53</td>
</tr>
<tr>
<td>goal</td>
<td>51.02</td>
<td>44.90</td>
<td>50.00</td>
<td>tie</td>
<td>23.17</td>
<td>30.49</td>
<td>30.49</td>
</tr>
<tr>
<td>half</td>
<td>22.83</td>
<td>31.52</td>
<td>29.35</td>
<td>title</td>
<td>18.97</td>
<td>31.03</td>
<td>27.59</td>
</tr>
<tr>
<td>level</td>
<td>44.00</td>
<td>33.33</td>
<td>44.00</td>
<td>top</td>
<td>33.33</td>
<td>33.33</td>
<td>38.33</td>
</tr>
<tr>
<td>manager</td>
<td>70.10</td>
<td>54.64</td>
<td>69.07</td>
<td>transfer</td>
<td>60.00</td>
<td>42.86</td>
<td>54.29</td>
</tr>
<tr>
<td>market</td>
<td>9.84</td>
<td>11.48</td>
<td>18.03</td>
<td>will</td>
<td>53.06</td>
<td>30.61</td>
<td>48.98</td>
</tr>
<tr>
<td>package</td>
<td>37.21</td>
<td>33.72</td>
<td>37.21</td>
<td>Average</td>
<td>38.99</td>
<td>34.55</td>
<td>39.70</td>
</tr>
</tbody>
</table>

The results demonstrate the good performance of our method. Our method based on dependency relation and syntax tree is the best, the method based on window follows and the method based on dependency relation is the worst. With dependency relation, feature words can be extracted exactly. But the recall of the method is lowest. The reason is as follow. Feature words that have direct dependency relations with the target word are often few in the sentence. In some cases, with the method, we can’t get any effective feature words. Comparing the number of instances that fail to get more than one feature word, there are 9 instances for Win, 255 instances for Depend and 9 instances for DepTree. Apparently, the method based on dependency relation fails to get effective feature words for many instances. This leads to the lower recall than the other methods.
Compared with the method based on context window, our method based on dependency relation and syntax tree obtains the better results with the average improvement of 0.71%. In general, our method is more effective than method based on context windows. Among the 41 sample words, the recalls of 24 words are better than the method based on context window. When feature words are selected, our method considers dependency relation, hierarchy relation and path distance on the phrase structure parsing tree, direct distance in the sentence, and assigns different WSD weights for them. The method based on context window only simply select the words surrounding the ambiguous word within the window as feature words without considering syntax relations and may induce some irrelevant noise feature words.

Though our method is better than the other methods, some results are not very satisfied. For a part of words, such as check, fishing, tie and title, the performance is worse greatly than the method based on context window. The four words have more close relatedness with directly adjacent words than with syntax-related words. For this kind of words, we need to try more effective way to improve the effectiveness. Besides, for country and striker, all of three methods only get very low recall. This may be caused by the measure of semantic relatedness, which is not fit for the two words.

5. Conclusions and future work

The paper proposes the WSD method with feature words based on dependency relation and syntax tree. For the selection of feature words, the method is different with the method based on context window and the method based on dependency relation. Our method considers dependency relation, syntax relation and position relation to select feature words and assigns them different weights based on their relations. The semantic relatedness between each sense of the ambiguous word and feature words are computed. The sense with maximum relatedness is chosen as the right sense. The results of experiments demonstrate the good effectiveness of the proposed method in the paper. The method is a preferred strategy to select feature words to disambiguate the ambiguous word.

The method in the paper focuses on the selection of feature words. There are two aspects in the future works. On the one hand, we would try to find the collocation words that can indicate the sense of target word more clearly, and try to avoid the noise feature words. On the other hand, we would like to try to consider more information to the selection of the sense of the ambiguous word. The selection of the sense is based on the WN semantic relatedness in our method. Word frequency and domain information also play important roles for WSD, so we would try to integrate them into the selection of the sense of the target word.

6. Acknowledgement

This work is supported by research fund of “The Theory and Method of Intelligent Processing for Mass Language Information” and “Research on Coreference Resolution for Chinese Text” of Beijing Institute of Technology and Primary Funds of National Defense. Besides, we would like to thank greatly to Rob Koeling and Diana McCarthy for kindly providing the dataset and assistance, and thank Siddharth Patwardhan and Ted Pedersen for making the WN Similarity package publically available.

7. References


