WSD for SENSEVAL-3 dataset using TiMBL

Milind Gokhale
Indian University
Bloomington
Indiana
USA
mgokhale@indiana.edu

Renuka Deshmukh
Indian University
Bloomington
Indiana
USA
renudesh@umail.iu.edu

Abstract

Word Sense Disambiguation is a challenge faced in multiple fields involving natural language processing. We present a word sense disambiguation approach that uses memory-based learning to train and test the machine learner. We use TiMBL as our machine learner and the dataset provided in the SENSEVAL-3 as test and train set. With use of simple features like context words, word collocations, Part of Speech tags and keyword information, we could achieve an average accuracy of 66.69%.

1 Introduction

The task of Word Sense Disambiguation entails assigning a polysemous word the most appropriate meaning based of its usage in the current context. A large number of approaches like Naïve Bayesian models, parallel corpora, SVD, combination of knowledge sources [3], hierarchical decision lists [2], etc. have been proposed for Word Sense Disambiguation. Recent trends in the Natural Language Processing community has seen a tremendous interest in Word Sense Disambiguation, especially since SENSEVAL-1 in 1999.

Word Sense Disambiguation has, since, become a motivation for more and more researchers to come up with an approach to solve this problem. SENSEVAL is a word sense evaluation series organized by ACL-SIGLEX. SENSEVAL 1, 2 and 3 were focused on word sense disambiguation for growing number of languages. Since 2007, SEMEVAL evaluation series was started, which also includes semantic analysis in addition to WSD.

The most common approaches to Word sense Disambiguation as discussed below. The *Knowledge-based* approach uses information from an explicit lexicon or a knowledge- source. This is one of the most common methodologies used in WSD. It makes use of existing knowledge sources like WordNet. *Corpus-based* approach does not use an explicit knowledge source. Instead, it uses information gained from a disambiguated or raw training corpus. The *Disambiguated Corpora-based* approach uses a disambiguated training set. It applies some machine learning algorithm to a set of features extracted from the training set. This trained model is then applied to new examples to disambiguate them. The *Raw-Corpora-based* approach uses raw dataset as opposed to the disambiguated dataset used in the Disambiguation-Corpora-based approach as it is difficult to obtain annotated datasets in most of the cases. This is an unsupervised model and is used for Word Sense Discrimination rather than Disambiguation. The last approach is *Hybrid Approach*, which is a combination of knowledge and corpus based approach. [4]

In this paper we describe a memory-based approach for *word sense disambiguation* (WSD) as defined in the SENSEVAL task: the association of a word in context with its contextually appropriate sense tag. We propose a WSD technique based on disambiguated corpora provided in the SENSEVAL-3. We also incorporate features like having context words as features, word collocations, POS Tagging and count of keywords to improve accuracy. We will discuss our approach in more detail in the following sections.

In this paper, we propose a WSD technique based on disambiguated corpora provided in the

SENSEVAL-3. We also incorporate features like having context words as features, word collocations, POS Tagging and count of keywords to improve accuracy. We will discuss our approach in more detail in the following sections.

The rest of the paper is organized as follows. Section 2 gives an overview of the related work. Section 3 describes our methodology and feature selection process. Section 4 gives the accuracy obtained by our approach, followed by Discussion in section 5, Future Work in section 6 and Conclusion on section 7.

2 Related Work

One of the first attempts for word sense disambiguation was done by Lesk in 1986 using dictionary definitions. This first attempt consisted of counting the number of overlaps between the dictionary definitions of the target word and the words near the target word contextually [6]. Banerjee, Patwardhan and Pederson in 2003 later extended this approach by including the dictionary definitions of the words related to the target words. These related words were determined according to the structure of WordNet. This they later called as determination of semantic relatedness and the determination of the most related sense of a target word is in fact disambiguation of the target word. They implemented their idea in a freely available software package called WordNet::Similarity.

[5] presents a corpus-based approach that results in high accuracy by combining a number of very simple classifiers into an ensemble that performs disambiguation via a majority vote. Since enhancement of feature set or learning algorithm used in a corpus-based approach does not usually improve disambiguation accuracy beyond what can be attained with shallow lexical features or simple algorithm.

In [7], Veenstra, Bosch et al. follow a memory based approach to word sense disambiguation in which relies on only a small number of annotated examples for each sense of the target word. There is based on on a POS-tagged corpus examples and selected dictionary entries. Since it depended on less number of annotated examples, it was portable and easily adaptable. They also constructed a separate classifier called the word expert for each of the target words.

There have been supervised methods for word sense disambiguation and [8] argues that typical supervised approach relies on feature vector constructed from the sense-tagged occurrences of target word, however there is a shortcoming in supervised approaches since they are only applicable to those words whose sense tagged data is available. Thus their accuracy is dependent on the amount of labeled data on target words available. So Rada Mihalcea focuses on an approach which can work when only few annotated examples are available for the target word using co-training and self-training methods.

3 Methodology

3.1 Classifier

Memory based learning is a supervised learning approach used for machine learning. A memory based learning system, has two components – a memory-based *learning component* and a similarity-based *performance component*. The learning component is used to train the learner, and involves storing the training instances in the memory along with their classifier information. The memory storage is done without any data processing or analysis. Hence, memory based learning is also known as lazy learning. The performance component takes new instances that need to be classified, as input, and classifies the input based on the similarity with the instances stored in memory.

Among two popular choices for MBL- TiMBL and SVM, we decided to use TiMBL due to its ease of use and ample documentation available online. Additionally, TiMBL gives us the ability of varying algorithms and settings in order to get better accuracy. We did a sample test using both TiMBL and SVM for three words – arm, difficulty and interest, and found that TiMBL performed better for all the three words, as compared to SVM. A probable cause for this could the limited number of examples in the train set. Hence, looking at the positive results from TiMBL, we have decided to use TiMBL as the machine learner for the given task of WSD.

3.2 Dataset

Our data set is taken from the SENSEVAL-3, which was held in 2004. The dataset was part of the SENSEVAL project organized by ACL-SIGLEX as part of a competition for evaluating various WSD systems. These systems are trained and tested on the dataset provided in the SENSEVAL for various words and languages.

In this paper, we present the best obtained results for disambiguation of every target word in the dataset, by using the words in a widow of ±3 words as feature set, and then adding more features to the feature set (as described in section 3.3) for better accuracy.

3.3 Feature Set

TiMBL requires a set of features along with the classifier as the last column in the training set. To disambiguate the 57 words given in the SENSEVAL-3 dataset for WSD, we performed our experiments in four stages. Each stage consists of adding new features to the already existing feature set, and then running this feature set on TiMBL to compare the accuracy obtained in each set. Our feature types can be broadly classified into four main categories:

Context Words: Words occurring in the immediate context of the target word, also called *local context*, are very helpful in discerning the sense of the target word. Most of the approaches use a ±2, ±3 or ±4 window size. The context words in our approach comprise of the ±3-word window around the target word. The context words form the basic feature for our WSD system. Additional features are built on top of these features.

Word Collocations: The word collocation are the bigrams and trigrams formed by appending the words around the target word. Each collocation is represented as one feature in the feature set. We consider word collocations, because they often contribute to the disambiguation accuracy in a big way, as opposed to just having context words in the feature set [5].

POS Tags: Parts of Speech tagging and WSD are very closely related to each other. Parts of Speech tagging is carried out to add more meaning to the extracted feature set, comprising of words and collocations. [9] and [10] show that adding POS tags to the feature set improves the accuracy significantly. We used the Stanford POS tagger to tag our file with the correct POS, and then added these extracted POS tags to the feature set.

Count of Keywords: Considering the keywords occurring in the given example can also improve the accuracy of the WSD system. Since our machine learner needs a fixed number of columns in the feature set, finding keywords of each example might not be the correct solution. In our implementation, we have the total count of the keywords as the extended feature. These keywords include the total number of nouns, verbs, adjectives, and adverbs occurring in the given example, as flagged by the Stanford POS tagger.

3.4 Representation

We have divided our implementation into four stages. This is to get a better idea of the contribution each type of feature viz. POS tags, context words, keywords and word collocations makes towards disambiguation of the target words. This also gives us an insight into the effect each kind of feature has on the accuracy of the learner.

The baseline setting for our approach comprises of the words in the ±3-word window, around the target word. Whenever a word is encountered that falls out of the boundaries of the sentence for any example, we assign it a default value of '_' [11]. Word collocations in terms of bigrams and trigrams are added after the context words. The last column is the classifier, or the sense id of the target word. In our approach we trained a different learner for each of the 57 words in the SENSEVAL-3 dataset, to be able to test the results obtained for individual words.

One of the earliest approaches to WSD involved directly assigning the most frequent sense to the ambiguous word. This was done with the understanding that there will be errors, but they will be well under the acceptable limit. In the first stage, called the *basic setting*, we test the accuracy of this baseline feature set with the classifier as the most frequent sense. The accuracy obtained in this step also serves as the benchmark to evaluate the results with extended feature set.

```
other plans not now were also otherplans plans-
not notnow nowwere werealso otherplansnot
plansnotnow notnowwere nowwerealso 38201
```

Table 1: Basic Feature Representation

In the second stage, the most frequent sense is replaced by the actual sense of the target word as provided in the train set of the SENSEVAL-3 dataset. TiMBL is again trained with this updated train set and the obtained accuracy is compared with the accuracy obtained in the previous stage. This stage is also serves as the *baseline setting*.

```
other plans not now were also otherplans plans-
not notnow nowwere werealso otherplansnot
plansnotnow notnowwere nowwerealso 38203
```

Table 2: Baseline Feature Representation

In the third stage, called the POS stage, POS tags generated by the Stanford POS tagger are also incorporated in the feature set. The feature set is represented by adding a POS tag to each of the context words, separated by '/'. For the word collocations, collocations of POS of the respective words are formed and included in the feature set in a fashion similar to the context word POS tagging. TiMBL is trained on this feature set and the obtained accuracy is compared with the baseline accuracy.

other/JJ plans/NNS not/RB now/RB were/VBD also/RB otherplans/JJNNS plansnot/NNSRB notnow/RBRB nowwere/RBVBD werealso/VBDRB otherplansnot/JJNNSRB plansnotnow/NNSRBRB notnowwere/RBRBVBD nowwerealso/RBVBDRB 38203

Table 3: POS Feature Representation

In the fourth and final stage, the count of keywords is added as the last column of the feature set, preceding the classifier column. The count here is the total number of nouns, verbs, adjectives and adverbs occurring in the context. This information is also obtained from POS tagging the test and train set and then counting the keywords for each example. TiMBL is trained and the obtained results as compared with the accuracy of the baseline setting.

other/JJ plans/NNS not/RB now/RB were/VBD also/RB otherplans/JJNNS plansnot/NNSRB notnow/RBRB nowwere/RBVBD werealso/VBDRB otherplansnot/JJNNSRB plansnotnow/NNSRBRB notnowwere/RBRBVBD nowwerealso/RBVBDRB 38203

Table 4: Keyword Count Feature Representation

We tabulate our findings in section 4.

3.5 Metrics

We use various algorithms, weighing vectors and distance metrics as provided by TiMBL. For algorithms we use IB1, IGTREE, TRIBL. For distance metrics we use Overlap (O), mvdm and Jeffry Divergence (JD). We use No Weight (NW), Information Gain (IG), Chi Squared (CS) and Gain Ratio (GR) as distance metrics. We vary k (nearest neighbor) from 1 to 5 for each setting. We tabulate our findings in the following section.

4 Results

We tested the data with classification algorithms – IB1, IGTREE and TRIBL along with the distance metrics – Overlap, MVDM and Jeffrey Divergence and feature weighing vectors – No weight, Gain ratio, Information gain and Chi Squared. Then we train the classifiers using the train files and test it against the test files. We tried to optimize the classifier accuracy by adjusting the parameters for the classifiers. Out of the derived outputs we selected the maximum accuracy TiMBL setting and represent it in table 5. The average accuracy obtained by our approach is 66.69% which is an improvement over baseline average of 62.41%. Thus we show that a combination of context words, word collocations, POS tags and keyword information is a good approach towards word sense disambiguation. Figure 1 shows the average accuracy obtained in our approach at each stage.

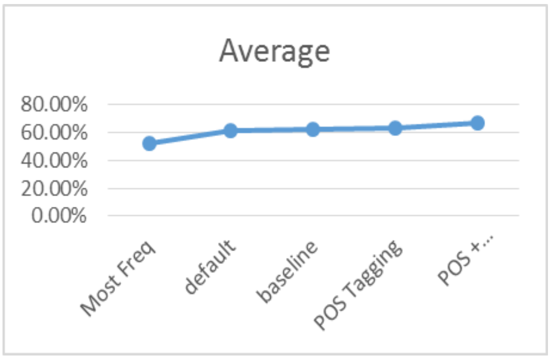

Figure 1: Average Accuracy

5 Discussion

We notice that IB1 algorithm gives best accuracy for most of the words. Addition of POS tags did not have a significant effect in improving the overall accuracy. Introducing Keyword count increased the overall accuracy of the classification. We noticed significant improvements in accuracy for some words and slight deterioration in other words. Default setting also performs better than basic setting which was followed in older approaches. Baseline setting shows significant improvement over default setting for most of the words. This is because the classification algorithm, distance metrics and weighing vectors can be varied to achieve better accuracy.

Words like *use, remain, provide, and receive* recorded best accuracy when the POS tags are added to the feature set. The same words showed best accuracy in baseline which shows that the POS tags do not change the accuracy much, in fact the accuracy deteriorated in some cases on introducing POS tags in the feature set. Interestingly for the word *eat*, the accuracy improved significantly just by adding POS tags. Although some of the words were outliers and showed fall in accuracy on adding new features to the feature set, we saw an overall accuracy improvement in most of the words as we added more features. For example words like *difficulty, paper, im-*

portant, performance and solid showed significant improvement while the words like *eat, wash, expect* showed a severe drop in accuracy. On analysis we found that the increase or decrease in accuracy in adding keyword count is not dependent on the number of test or train examples. It is more dependent on the coverage of the train set. However we can include features like the actual keywords in the feature set to improve the results which can be considered as future work.

Table 5. Best accuracy obtained during the four stage evaluation of our WSD approach.

Word	Most Freq	Default	Baseline	POS Tagging	POS + Keywords
activate	69.12%	62.50%	69.12% (IB1 O k3 default)	69.12% (IB1 O k5 GR)	69.12% (IB1 O k5 GR)
add	41.50%	70.07%	71.26% (IB1 O k3 IG)	70.75% (IB1 O k1 NW)	54.74% (IB1 O k3 CS)
appear	40.97%	59.72%	65.97% (IB1 mvdm k5 CS)	65.97% (IB1 JD k5 GR)	67.65% (IGTREE O GR)
argument	44.19%	44.96%	45.74% (IB1 JD k5 GR)	47.29% (IB1 mvdm k5 GR)	77.78% (IB1 O k3 CS)
arm	80.74%	85.93%	85.93% (IB1 O k1 NW)	85.93% (IB1 O k1 NW)	69.29% (IB1 JD k1 NW)
ask	22.98%	50.93%	53.42% (IB1 JD k3 IG)	48.45% (IB1 mvdm k3 NW)	45.38% (IB1 O k3 CS)
atmosphere	52.94%	46.08%	52.94% (IB1 O k3 NW)	52.94% (IB1 O k5 GR)	53.85% (IB1 mvdm k3 NW)
audience	62.62%	71.03%	78.50% (IB1 mvdm k3 CS)	77.57% (IB1 mvdm k3 CS)	39.13% (IB1 mvdm k3 NW)
bank	64.49%	71.01%	75.36% (IB1 mvdm k3 GR)	75.36% (IB1 mvdm k3 NW)	42.20% (IB1 mvdm k1 NW)
begin	50.00%	51.06%	55.32% (IB1 O k3 GR)	54.74% (IB1 O k3 CS)	87.64% (IGTREE O NW)
climb	54.41%	66.18%	70.59% (IB1 JD k3 GR)	67.65% (IB1 mvdm k3 GR)	69.23% (IB1 mvdm k3 IG)
decide	66.67%	73.02%	79.37% (IB1 O k3 IG)	77.78% (IB1 mvdm k5 NW)	70.07% (IB1 O k1 CS)
degree	55.71%	65.00%	71.43% (IB1 mvdm k3 GR)	70.00% (IB1 JD k1 NW)	71.26% (IB1 JD k5 GR)
difference	35.38%	41.54%	44.62% (IB1 JD k5 GR)	45.38% (IB1 O k1 CS)	66.67% (IB1 O k5 GR)
different	48.08%	48.08%	53.85% (IB1 mvdm k3 NW)	55.77% (IB1 mvdm k3 NW)	62.50% (IB1 mvdm k3 NW)
difficulty	17.39%	34.78%	34.78% (IB1 mvdm k3 NW)	39.13% (IB1 mvdm k3 NW)	79.55% (IGTREE O IG)
disc	34.86%	36.70%	42.20% (IB1 O k3 CS)	40.37% (IB1 O k3 CS)	60.00% (IB1 mvdm k1 GR)
eat	86.52%	82.02%	66.15% (IB1 mvdm k5 NW)	87.64% (IB1 O k1 CS)	37.04% (IB1 JD k1 IG)
encounter	36.92%	60.00%	70.07% (IB1 mvdm k5 NW)	66.15% (IB1 mvdm k5 NW)	62.11% (IB1 O k5 GR)
expect	66.67%	68.97%	66.67% (IB1 O NW)	70.11% (IB1 O k1 NW)	43.75% (IB1 O k1 NW)
express	66.67%	61.40%	66.67% (IB1 O NW)	66.67% (IB1 O k1	55.56% (IB1 O k3

				NW)	NW)
hear	46.88%	56.25%	59.38% (IB1 O k3 CS)	62.50% (IB1 mvdm k5 CS)	67.50% (IB1 mvdm k3 GR)
hot	77.27%	75.00%	79.55% (TRIBL O k1 CS)	79.55% (IGTREE O IG)	65.97% (IB1 mvdm k3 GR)
image	36.00%	52.00%	53.33% (IB1 O k1 GR)	53.33% (IB1 O k1 CS)	50.00% (IB1 mvdm k3 IG)
important	22.22%	25.93%	29.63% (IB1 O k1 GR)	33.33% (IB1 mvdm k1 NW)	63.24% (IB1 O k5 GR)
interest	61.05%	62.11%	62.11% (IB1 O k1 GR)	63.16% (TRIBL mvdm k3 IG)	61.11% (IB1 JD k5 GR)
judgement	28.13%	43.75%	50.00% (IB1 O k1 CS)	50.00% (IB1 mvdm k3 NW)	73.68% (IB1 O k1 CS)
lose	52.78%	44.44%	58.33% (IB1 JD K3 IG)	55.56% (IB1 JD k3 GR)	47.06% (IB1 JD k5 CS)
mean	52.50%	65.00%	65.00% (IB1 O k1 NW)	65.00% (IB1 JD k5 GR)	64.80% (IB1 mvdm k3 NW)
miss	33.33%	46.67%	53.33% (IB1 mvdm k5 GR)	53.33% (IB1 mvdm k5 GR)	66.00% (IB1 O k1 NW)
note	55.88%	57.35%	67.65% (IB1 O k3 GR)	64.71% (IB1 O k3 CS)	70.00% (IB1 O k3 CS)
operate	38.89%	61.11%	66.67% (IB1 JD K5 NW)	61.11% (IB1 JD k5 GR)	48.42% (IB1 O k1 NW)
organization	71.93%	70.18%	77.19% (IB1 mvdm k5 IG)	77.19% (IB1 mvdm k5 GR)	58.95% (IB1 mvdm k3 GR)
paper	22.06%	44.12%	48.53% (IB1 JD k5 NW)	46.32% (IB1 mvdm k5 NW)	84.51% (IB1 O k5 GR)
party	57.60%	59.20%	65.60% (IB1 mvdm k5 GR)	66.00% (IB1 mvdm k5 GR)	88.89% (IB1 O k5 GR)
performance	25.56%	31.11%	32.22% (IB1 O k3 GR)	33.33% (IB1 O k3 CS)	88.89% (IB1 O k5 CS)
plan	69.00%	65.00%	71.00% (IB1 O k3 GR)	71.00% (IB1 O k3 CS)	90.14% (IB1 JD k1 GR)
play	46.15%	44.23%	51.92% (IB1 mvdm k5 NW)	46.15% (IB1 O k5 GR)	70.00% (IB1 mvdm k1 NW)
produce	51.58%	56.84%	62.11% (IB1 JD k5 GR)	57.89% (IB1 JD k5 NW)	54.78% (IB1 JD k5 NW)
provide	83.10%	88.73%	91.55% (IB1 O k1 GR)	93.32% (IB1 mvdm k3 NW)	66.07% (IB1 O k1 NW)
receive	88.89%	74.07%	88.89% (IB1 O k3 NW)	88.89% (IB1 O k5 GR)	75.00% (IB1 mvdm k1 GR)
remain	77.46%	83.10%	87.32% (IB1 O k3 GR)	87.32% (IB1 mvdm k1 GR)	66.97% (IB1 O k1 NW)
rule	40.00%	66.67%	66.67% (IB1 O k1 NW)	66.67% (IB1 O k1 NW)	66.97% (TRIBL O k1 GR)
shelter	38.26%	52.17%	53.04% (IB1 JD k3 CS)	53.04% (IB1 O k1 NW)	85.93% (IB1 O k1 NW)
simple	25.00%	25.00%	35.00% (IB1 mvdm k1 NW)	38.00% (IB1 mvdm k1 NW)	85.93% (IB1 mvdm k1 NW)
smell	39.29%	69.64%	76.79% (IB1 mvdm k3 NW)	78.57% (IB1 JD k5 CS)	72.60% (IB1 O k5 GR)

solid	26.47%	23.53%	29.41% (IB1 O k5 IG)	33.41% (IB1 mvdm k3 GR)	75.34% (IB1 mvdm k1 GR)
sort	57.80%	66.06%	66.06% (IB1 O k1 NW)	66.97% (TRIBL O k1 GR)	66.67% (IB1 O k1 NW)
source	60.00%	48.57%	60.00% (IB1 O k3 NW)	62.74% (IB1 O k5 GR)	93.33% (IGTREE O GR)
suspend	35.94%	45.31%	50.00% (IB1 mvdm k5 NW)	54.56% (IB1 mvdm k5 GR)	76.47% (IB1 O k1 NW)
talk	72.60%	72.60%	76.71% (IB1 JD k3 GR)	76.71% (IB1 mvdm k3 NW)	78.43% (IB1 mvdm k3 NW)
treat	28.07%	36.84%	47.37% (IB1 mvdm k3 NW)	48.86% (IB1 mvdm k3 NW)	56.52% (IB1 O k1 NW)
use	66.67%	66.67%	93.33% (IGTREE GR)	93.33% (IGTREE O GR)	60.87% (IB1 JD k3 GR)
wash	69.70%	60.61%	72.73% (TRIBL O q1 no gain)	78.73% (TRIBL O k1 NW)	52.94% (IB1 O k5 GR)
watch	74.51%	78.43%	80.39% (IB1 mvdm k1 NW)	78.43% (IB1 O k1 NW)	71.03% (IB1 O k1 NW)
win	43.59%	53.85%	56.41% (IB1 JD k3 NW)	56.41% (IB1 O k3 CS)	75.70% (IB1 mvdm k3 CS)
write	34.78%	52.17%	52.17% (IB1 O k1 NW)	58.52% (IB1 O k1 CS)	76.09% (IB1 mvdm k3 NW)
Average	**51%**	**57%**	**62.41%**	**62.96%**	**66.69%**

6 Future Work

There are a number of extensions to our work. Implementing forward or backward selection can be considered for future work. Since the SENSEVAL-3 dataset for English language is not exhaustive, we believe implementing backward selection will be a good approach to further improve the accuracy. In the current approach, we use count of the keywords as a feature value. Going one step further, and finding the actual keywords related to the target word, using the TF-IDF approach can be worked upon in the future.

Publicly available lexical data sources, like the WordNet, can be leveraged to extract additional contextual information about the target word. Online dictionaries and thesaurus can be used to glean alternative words for the target and context words, and these words can be added as features. Same word might appear in different forms in the test and train file misleading TiMBL to believe that they are different words. This compromises the accuracy of the WSD system. As future work, we plan to substitute the context and target words with the morphological root word. We believe that this will have a positive effect on the results.

Lastly, we would like to add senses of words to the feature set. Each sense of the target word can be added as a separate feature, where the feature-value is the total number of keywords present in the example which correspond to the given sense. Thus, the feature-value with highest number would be the most probable sense of the target word.

7 Conclusion

We believe that the memory-based learning methodology is well-suited for the problem of Word Sense Disambiguation, as it can work well with both dense and sparse data. Memory-based learning can also easily work with exceptional and corner-cases.

In this paper, we present our approach for WSD using memory-based learning. We argued that using TiMBL over SVM was a better choice for the current dataset. We also show, how a choice of feature set affects the accuracy of the learner. We propose and subsequently evaluate the improvements made by: 1) adding POS tags to the feature set; and 2) adding keyword information to the feature set. Lastly, we are confident that having senses of the target word as part of the feature set with weights as value of the feature will improve the outputs significantly.

8 Acknowledgement

We would like to extend our sincere thanks to Dr. Prof. Sandra Kuebler, for her guidance and direction, without which this paper would not have been a success. We would also like to thank Prof. Kuebler for all the study material and encouragement that lead to the completion of this project.

References

[1] Daelemans, W., Zavrel, J., Van der Sloot K., and Van den Bosch, A.: TiMBL: Tilburg Memory-Based Learner. Computational Linguistics Tilburg University version 5.0.1 December ILK Technical Report - ILK 04–02 (2004).

[2] D. Yarowsky. 2000. Hierarchical decision lists for word sense disambiguation. Computers and the Humanities, 34(1{2).

[3] Y. Wilks and M. Stevenson. 1998. Word sense disambiguation using optimised combinations of knowledge sources. In Proceedings of COLING/ACL-98.

[4] "Word Sense Disambiguation." - ACL Wiki. Accessed May 5, 2015. http://aclweb.org/aclwiki/index.php?title=Word_sense_disambiguation.

[5] Pedersen, Ted. "A simple approach to building ensembles of Naive Bayesian classifiers for word sense disambiguation." In Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference, pp. 63-69. Association for Computational Linguistics, 2000.

[6] Patwardhan, Siddharth, Satanjeev Banerjee, and Ted Pedersen. "SenseRelate:: TargetWord: a generalized framework for word sense disambiguation." In Proceedings of the ACL 2005 on Interactive poster and demonstration sessions, pp. 73-76. Association for Computational Linguistics, 2005.

[7] Veenstra, Jorn, Antal Van den Bosch, Sabine Buchholz, and Walter Daelemans. "Memory-based word sense disambiguation." Computers and the Humanities 34, no. 1-2 (2000): 171-177.

[8] Mihalcea, Rada. "Co-training and self-training for word sense disambiguation." In Proceedings of the Conference on Computational Natural Language Learning (CoNLL-2004). 2004.

[9] Gaustad, Tanja. "The importance of high quality input for WSD: An application-oriented comparison of part-of-speech taggers." In Proceedings of the Australasian Language Technology Workshop (ALTW 2003), pp. 65-72. 2003.

[10] Moreno-Monteagudo, Lorenza, Rubén Izquierdo-Beviá, Patricio Martínez-Barco, and Armando Suárez. "A Study of the Influence of PoS Tagging on WSD." InText, Speech and Dialogue, pp. 173-179. Springer Berlin Heidelberg, 2006.

[11] Escudero, Gerard, Lluís Màrquez, and German Rigau. "Naive Bayes and exemplar-based approaches to word sense disambiguation revisited." arXiv preprint cs/0007011 (2000).