Building WSD systems with MBL

Dana Dannélls

Department of Swedish language
Section of Natural Language Processing
Göteborg University
SE-405 30 Gothenburg, Sweden

1 Introduction

This paper describes an attempt to build a supervised word sense disambiguation (WSD) system using memory based learning (MBL) algorithm. This task was one of the requirements for passing the NLP (Natural Language Processing) course for graduate students.

We designed two features sets to be the base of two WSD systems, we trained a MBL algorithm and tested each of the system's ability to disambiguate a word in its context. We describe the resources we used, motivate the choice of the features, present and compare the results of the systems.

2 Experiment

2.1 Data

Our data is taken from the Semeval 2007 task.[1] This task was a part of the senseval project[2], that aims at evaluating WSD systems with respect to different words, different varieties of language, and different languages.

Among other resources, the Semeval has released the Semcor, sense-tagged corpus[3] which contains nouns, verbs, adjectives and adverbs tagged with both syntactic and semantic information, i.e., each word class is annotated with POS (part-of-speech) tag, lemma, and WordNet synset.

In this experiment we picked 5 verbs and 5 nouns from the Semcor corpus. The selection of these words was based on the amount of training data and the polysemy degree of each word.

2.2 Algorithm

The TiMBL software (Daelemans et al., 2004) allows an access to various Memory-Based Learning (MBL) algorithms that can be used to detect patterns and regularities in a data set. This machine learning paradigm was used to build the WSD systems.

[1] SemEval-2007, the 4th International Workshop on Semantic Evaluations http://ixa2.si.ehu.es/semeval-clir/
[2] http://www.senseval.org/
[3] http://ji.ehu.es/eneko/resources.html#_Toc136420101

Although we were asked to use a single train/test partition (train 80%, test 20%) to build the systems on, we chose to perform our experiment using the *leave-one-out* method. This choice was made mainly because of the small amount of data which is available for each example. Using this method no test file is provided, testing is done on each pattern of the training file, by treating each pattern of the training file in turn as a test case and the whole remainder of the file as training cases. Accuracy of the classifier is the number of data items correctly predicted.

The algorithm that was used to build the WSD systems is IB1, since it is the only algorithm which is combined with the leave-one-out test. Due to the short time for completing this assignment either distance metrics nor feature weighting were tested.

2.3 Feature sets

Table 1. Feature sets of the WSD systems

Feature nr.	Feature set 1	Feature set 2
1	word-form	word-form
2	$big\_lem\_cont\_+1$	$big\_lem\_cont\_+1$
3	$big\_lem\_cont\_-1$	$big\_lem\_cont\_-1$
4	big_pos_+1	big_pos_+1
5	big_pos_-1	big_pos_-1
6	unigram	unigram
7	*post_J_lem*	*post_J_lem*
8	*post_N_lem*	*post_N_lem*
9	*post_V_lem*	*post_V_lem*
10	*prev_J_lem*	*prev_J_lem*
11	*prev_N_lem*	*prev_N_lem*
12	*prev_V_lem*	*prev_V_lem*
13		$trig\_lem\_cont\_+1$
14		$trig\_lem\_cont\_-1$
15		$trig\_pos\_+1$
16		$trig\_pos\_-1$
13/17	The word sense (i.e., the class to predict)	

Our feature selection is merely based on previous observations and conclusions made by other authors (Agirre et al., 2006; Audibert, 2004; Pedersen, 2001; Yarowsky et al., 2001; Yarowsky, 1995).

Agirre et al. (2006) point out the importance of using large amount of feature types, they also show that local features constituted by lemmas and POS tags are beneficial features for WSD systems. We followed Audibert (2004) assertions, i.e., (1) indicators for verbs tend to be mainly situated after the verb, (2) adjectives

and nouns are good indicators for determining the sense of a verb, and (3) bigrams and trigrams are seldom used alone and should be taken in conjunction with unigrams. Furthermore, Yarowsky (1995) show content words are powerful and have strong effect during the learning process.

In this experiment we wanted to clear out the significancy of trigram features by testing how they effect the disambiguation accuracy. Therefore we designed two similar feature sets of which only one contains trigram features. The selected features are shown in Table 1.[4]

Systematically we converted each example into feature vectors, this was done for each example in our data, in total 10 files of which 5 verbs and 5 nouns. An example of one instance of the verb "seem" presented as a feature vector according to feature set 1 is:

"seem, statesman-seem, seem-intent, NN-VB, VB-JJ, seem, intent, footstep, follow, nuclear, statesman, arouse, 01460069-v."

3 Results

Table 2 specifies the words used in the experiment, their part-of-speech (POS), their polysemy degree (Senses), the amount of examples (Instances) that the systems were build on and the results (Accuracy) obtained by each system.

The words which consist of rather large amount of examples (more than 400) show that the results obtained by the classifier which was trained on feature set 2 gave in most cases better accuracy, and hence using trigram features seem to improve the disambiguation system.

It is rather hard to compare the results between the two systems in cases where the polysemy degree of the word is high and the amount of instances are low. When these factors are combined, e.g., "life", "day", accuracy is reduced, in these cases trigram features don't seem to improve disambiguation results. A known fact is that the polysemy degree of a word reduces disambiguation accuracy, e.g. "take", "find", it is of course easier to obtain high accuracy when the polysemy degree of the word is low, e.g., "seem", "school".

The reason why the accuracy for the noun "thing" is lower than for the noun "life" even though it is more polysemous and contains larger amount of instances can be explained by the occurrences of zero values which we observed in the feature vectors for "life".

4 Conclusions

This small experiment showed trigram features improve accuracy results. The results agree well with the assertions made by Agirre et al. (2006), Audibert

[4] Feature type abbreviations: prev (previous), post (posterior), big (bigram), tri (trigram), J (adjective), N (noun), V (verb), pos (part-of-speech), lem (lemma), cont (content word).
Interpretation example: the feature "*prev_V_lem*" indicates the verb that precedes by target word.

Table 2. The data and results of the WSD systems

POS	Word	Senses	Instances	Accuracy % (System 1)	Accuracy % (System 2)
V	take	42	694	24.4 %	26.1 %
V	keep	22	318	55 %	54.7 %
V	find	18	673	27.9 %	30.4 %
V	show	12	435	28.2 %	33.5 %
V	seem	4	585	84.7 %	87.5 %
N	life	14	233	46.7 %	45 %
N	thing	12	270	28.8 %	28.5 %
N	man	11	647	68.6 %	70 %
N	day	10	330	57.8 %	55.1 %
N	school	7	147	83 %	80.2 %

(2004) and Yarowsky (1995), concerning large amount of features including content words, previous/posterior lemma/word forms etc. In order to build an accurate system the data must contain a large amount of instances for each word the system is trained on.

It will be interesting to expand our systems with Bag-of-Words features and features that provide information about syntactic dependencies. We also think it will be profitable to include function words features since we noticed many of the instances in the data don't contain information about content words which resulted in zero values. Experimenting with different algorithms may also lead to improvement. As Pedersen (2001) points out: "an informative feature set will result in accurate disambiguation when used with a wide range of learning algorithms".

References

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).

Agirre, E., Lopez de Lacalle, O., and Martinez, D.: Exploring feature set combinations for WSD. In proceedings of the annual meeting of the SEPLN, Spain (2006).

Audibert, L.: Word sense disambiguation criteria: a systematic study. In Proceedings of the 20th international conference on Computational Linguistics (2004).

Florian, R., Cucerzan, S., Schafer, C., and Yarowsky, D.: Combining classifiers for word sense disambiguation. Natural Language Engineering, **4** (8) (2002) 327–341.

Pedersen, T.: A Decision Tree of Bigrams is an Accurate Predictor of Word Sense. In Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL-01), Pittsburgh, PA (2001).

Yarowsky, D., Cucerzan, S., Florian, R., Schafer, C., and Wicentowski, R.: The Johns Hopkins senseval2 system descriptions. Proceedings SENSEVAL2, (2001) pp. 163–166.

Yarowsky, D.: Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. Meeting of the Association for Computational Linguistics (1995) 189–196.