It Makes Sense: A Wide-Coverage Word Sense Disambiguation System for Free Text

Zhi Zhong and **Hwee Tou Ng**
Department of Computer Science
National University of Singapore
13 Computing Drive
Singapore 117417
{zhongzhi, nght}@comp.nus.edu.sg

Abstract

Word sense disambiguation (WSD) systems based on supervised learning achieved the best performance in SensEval and SemEval workshops. However, there are few publicly available open source WSD systems. This limits the use of WSD in other applications, especially for researchers whose research interests are not in WSD.

In this paper, we present IMS, a supervised English all-words WSD system. The flexible framework of IMS allows users to integrate different preprocessing tools, additional features, and different classifiers. By default, we use linear support vector machines as the classifier with multiple knowledge-based features. In our implementation, IMS achieves state-of-the-art results on several SensEval and SemEval tasks.

1 Introduction

Word sense disambiguation (WSD) refers to the task of identifying the correct sense of an ambiguous word in a given context. As a fundamental task in natural language processing (NLP), WSD can benefit applications such as machine translation (Chan et al., 2007a; Carpuat and Wu, 2007) and information retrieval (Stokoe et al., 2003).

In previous SensEval workshops, the supervised learning approach has proven to be the most successful WSD approach (Palmer et al., 2001; Snyder and Palmer, 2004; Pradhan et al., 2007). In the most recent SemEval-2007 English all-words tasks, most of the top systems were based on supervised learning methods. These systems used a set of knowledge sources drawn from sense-annotated data, and achieved significant improvements over the baselines.

However, developing such a system requires much effort. As a result, very few open source WSD systems are publicly available – the only other publicly available WSD system that we are aware of is SenseLearner (Mihalcea and Csomai, 2005). Therefore, for applications which employ WSD as a component, researchers can only make use of some baselines or unsupervised methods. An open source supervised WSD system will promote the use of WSD in other applications.

In this paper, we present an English all-words WSD system, IMS (It Makes Sense), built using a supervised learning approach. IMS is a Java implementation, which provides an extensible and flexible platform for researchers interested in using a WSD component. Users can choose different tools to perform preprocessing, such as trying out various features in the feature extraction step, and applying different machine learning methods or toolkits in the classification step. Following Lee and Ng (2002), we adopt support vector machines (SVM) as the classifier and integrate multiple knowledge sources including parts-of-speech (POS), surrounding words, and local collocations as features. We also provide classification models trained with examples collected from parallel texts, SEMCOR (Miller et al., 1994), and the DSO corpus (Ng and Lee, 1996).

A previous implementation of the IMS system, NUS-PT (Chan et al., 2007b), participated in SemEval-2007 English all-words tasks and ranked first and second in the coarse-grained and fine-grained task, respectively. Our current IMS implementation achieves competitive accuracies on several SensEval/SemEval English lexical-sample and all-words tasks.

The remainder of this paper is organized as follows. Section 2 gives the system description, which introduces the system framework and the details of the implementation. In Section 3, we present the evaluation results of IMS on SensE-

Proceedings of the ACL 2010 System Demonstrations, pages 78–83,
Uppsala, Sweden, 13 July 2010.

val/SemEval English tasks. Finally, we conclude in Section 4.

2 System Description

In this section, we first outline the IMS system, and introduce the default preprocessing tools, the feature types, and the machine learning method used in our implementation. Then we briefly explain the collection of training data for content words.

2.1 System Architecture

Figure 1 shows the system architecture of IMS. The system accepts any input text. For each content word w (noun, verb, adjective, or adverb) in the input text, IMS disambiguates the sense of w and outputs a list of the senses of w, where each sense s_i is assigned a probability according to the likelihood of s_i appearing in that context. The sense inventory used is based on WordNet (Miller, 1990) version 1.7.1.

IMS consists of three independent modules: preprocessing, feature and instance extraction, and classification. Knowledge sources are generated from input texts in the preprocessing step. With these knowledge sources, instances together with their features are extracted in the instance and feature extraction step. Then we train one classification model for each word type. The model will be used to classify test instances of the corresponding word type.

2.1.1 Preprocessing

Preprocessing is the step to convert input texts into formatted information. Users can integrate different tools in this step. These tools are applied on the input texts to extract knowledge sources such as sentence boundaries, part-of-speech tags, etc. The extracted knowledge sources are stored for use in the later steps.

In IMS, preprocessing is carried out in four steps:

- Detect the sentence boundaries in a raw input text with a sentence splitter.
- Tokenize the split sentences with a tokenizer.
- Assign POS tags to all tokens with a POS tagger.
- Find the lemma form of each token with a lemmatizer.

By default, the sentence splitter and POS tagger in the OpenNLP toolkit[1] are used for sentence splitting and POS tagging. A Java version of Penn TreeBank tokenizer[2] is applied in tokenization. JWNL[3], a Java API for accessing the WordNet (Miller, 1990) thesaurus, is used to find the lemma form of each token.

2.1.2 Feature and Instance Extraction

After gathering the formatted information in the preprocessing step, we use an instance extractor together with a list of feature extractors to extract the instances and their associated features.

Previous research has found that combining multiple knowledge sources achieves high WSD accuracy (Ng and Lee, 1996; Lee and Ng, 2002; Decadt et al., 2004). In IMS, we follow Lee and Ng (2002) and combine three knowledge sources for all content word types[4]:

- *POS Tags of Surrounding Words* We use the POS tags of three words to the left and three words to the right of the target ambiguous word, and the target word itself. The POS tag feature cannot cross sentence boundary, which means all the associated surrounding words should be in the same sentence as the target word. If a word crosses sentence boundary, the corresponding POS tag value will be assigned as *null*.

  For example, suppose we want to disambiguate the word *interest* in a POS-tagged sentence "My/PRP$ brother/NN has/VBZ always/RB taken/VBN a/DT keen/JJ interest/NN in/IN my/PRP$ work/NN ./.". The 7 POS tag features for this instance are $<$*VBN*, *DT*, *JJ*, *NN*, *IN*, *PRP$*, *NN*$>$.

- *Surrounding Words* Surrounding words features include all the individual words in the surrounding context of an ambiguous word w. The surrounding words can be in the current sentence or immediately adjacent sentences.

  However, we remove the words that are in a list of stop words. Words that contain no alphabetic characters, such as punctuation

[1] `http://opennlp.sourceforge.net/`
[2] `http://www.cis.upenn.edu/~treebank/tokenizer.sed`
[3] `http://jwordnet.sourceforge.net/`
[4] Syntactic relations are omitted for efficiency reason.

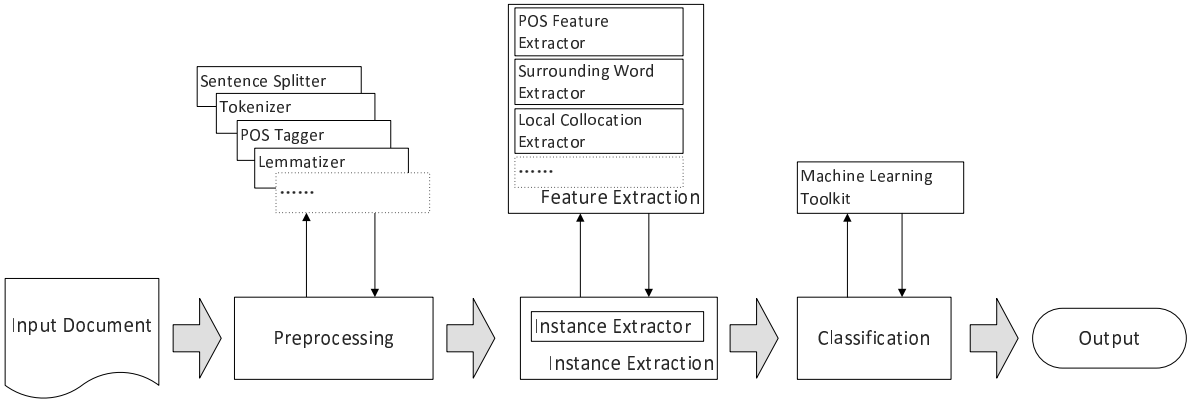

Figure 1: IMS system architecture

symbols and numbers, are also discarded. The remaining words are converted to their lemma forms in lower case. Each lemma is considered as one feature. The feature value is set to be 1 if the corresponding lemma occurs in the surrounding context of w, 0 otherwise.

For example, suppose there is a set of surrounding words features {*account, economy, rate, take*} in the training data set of the word *interest*. For a test instance of *interest* in the sentence "My brother has always taken a keen interest in my work .", the surrounding word feature vector will be <0, 0, 0, 1>.

- *Local Collocations* We use 11 local collocations features including: $C_{-2,-2}$, $C_{-1,-1}$, $C_{1,1}$, $C_{2,2}$, $C_{-2,-1}$, $C_{-1,1}$, $C_{1,2}$, $C_{-3,-1}$, $C_{-2,1}$, $C_{-1,2}$, and $C_{1,3}$, where $C_{i,j}$ refers to an ordered sequence of words in the same sentence of w. Offsets i and j denote the starting and ending positions of the sequence relative to w, where a negative (positive) offset refers to a word to the left (right) of w.

  For example, suppose in the training data set, the word *interest* has a set of local collocations {*"account ."*, *"of all"*, *"in my"*, *"to be"*} for $C_{1,2}$. For a test instance of *interest* in the sentence "My brother has always taken a keen interest in my work .", the value of feature $C_{1,2}$ will be *"in my"*.

As shown in Figure 1, we implement one feature extractor for each feature type. The IMS software package is organized in such a way that users can easily specify their own feature set by implementing more feature extractors to exploit new features.

2.1.3 Classification

In IMS, the classifier trains a model for each word type which has training data during the training process. The instances collected in the previous step are converted to the format expected by the machine learning toolkit in use. Thus, the classification step is separate from the feature extraction step. We use LIBLINEAR[5] (Fan et al., 2008) as the default classifier of IMS, with a linear kernel and all the parameters set to their default values. Accordingly, we implement an interface to convert the instances into the LIBLINEAR feature vector format.

The utilization of other machine learning software can be achieved by implementing the corresponding module interfaces to them. For instance, IMS provides module interfaces to the WEKA machine learning toolkit (Witten and Frank, 2005), LIBSVM[6], and MaxEnt[7].

The trained classification models will be applied to the test instances of the corresponding word types in the testing process. If a test instance word type is not seen during training, we will output its predefined default sense, i.e., the WordNet first sense, as the answer. Furthermore, if a word type has neither training data nor predefined default sense, we will output "U", which stands for the missing sense, as the answer.

[5] http://www.bwaldvogel.de/liblinear-java/

[6] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[7] http://maxent.sourceforge.net/

2.2 The Training Data Set for All-Words Tasks

Once we have a supervised WSD system, for the users who only need WSD as a component in their applications, it is also important to provide them the classification models. The performance of a supervised WSD system greatly depends on the size of the sense-annotated training data used. To overcome the lack of sense-annotated training examples, besides the training instances from the widely used sense-annotated corpus SEMCOR (Miller et al., 1994) and DSO corpus (Ng and Lee, 1996), we also follow the approach described in Chan and Ng (2005) to extract more training examples from parallel texts.

The process of extracting training examples from parallel texts is as follows:

- Collect a set of sentence-aligned parallel texts. In our case, we use six English-Chinese parallel corpora: Hong Kong Hansards, Hong Kong News, Hong Kong Laws, Sinorama, Xinhua News, and the English translation of Chinese Treebank. They are all available from the Linguistic Data Consortium (LDC).
- Perform tokenization on the English texts with the Penn TreeBank tokenizer.
- Perform Chinese word segmentation on the Chinese texts with the Chinese word segmentation method proposed by Low et al. (2005).
- Perform word alignment on the parallel texts using the GIZA++ software (Och and Ney, 2000).
- Assign Chinese translations to each sense of an English word w.
- Pick the occurrences of w which are aligned to its chosen Chinese translations in the word alignment output of GIZA++.
- Identify the senses of the selected occurrences of w by referring to their aligned Chinese translations.

Finally, the English side of these selected occurrences together with their assigned senses are used as training data.

We only extract training examples from parallel texts for the top 60% most frequently occurring polysemous content words in Brown Corpus (BC), which includes 730 nouns, 190 verbs, and 326 adjectives. For each of the top 60% nouns and adjectives, we gather a maximum of 1,000 training examples from parallel texts. For each of the top 60% verbs, we extract not more than 500 examples from parallel texts, as well as up to 500 examples from the DSO corpus. We also make use of the sense-annotated examples from SEMCOR as part of our training data for all nouns, verbs, adjectives, and 28 most frequently occurring adverbs in BC.

POS	noun	verb	adj	adv
# of types	11,445	4,705	5,129	28

Table 1: Statistics of the word types which have training data for WordNet 1.7.1 sense inventory

The frequencies of word types which we have training instances for WordNet sense inventory version 1.7.1 are listed in Table 1. We generated classification models with the IMS system for over 21,000 word types which we have training data. On average, each word type has 38 training instances. The total size of the models is about 200 megabytes.

3 Evaluation

In our experiments, we evaluate our IMS system on SensEval and SemEval tasks, the benchmark data sets for WSD. The evaluation on both lexical-sample and all-words tasks measures the accuracy of our IMS system as well as the quality of the training data we have collected.

3.1 English Lexical-Sample Tasks

	SensEval-2	SensEval-3
IMS	**65.3%**	72.6%
Rank 1 System	64.2%	**72.9%**
Rank 2 System	63.8%	72.6%
MFS	47.6%	55.2%

Table 2: WSD accuracies on SensEval lexical-sample tasks

In SensEval English lexical-sample tasks, both the training and test data sets are provided. A common baseline for lexical-sample task is to select the most frequent sense (MFS) in the training data as the answer.

We evaluate IMS on the SensEval-2 and SensEval-3 English lexical-sample tasks. Table 2 compares the performance of our system to the top

two systems that participated in the above tasks (Yarowsky et al., 2001; Mihalcea and Moldovan, 2001; Mihalcea et al., 2004). Evaluation results show that IMS achieves significantly better accuracies than the MFS baseline. Comparing to the top participating systems, IMS achieves comparable results.

3.2 English All-Words Tasks

In SensEval and SemEval English all-words tasks, no training data are provided. Therefore, the MFS baseline is no longer suitable for all-words tasks. Because the order of senses in WordNet is based on the frequency of senses in SEMCOR, the WordNet first sense (WNs1) baseline always assigns the first sense in WordNet as the answer. We will use it as the baseline in all-words tasks.

Using the training data collected with the method described in Section 2.2, we apply our system on the SensEval-2, SensEval-3, and SemEval-2007 English all-words tasks. Similarly, we also compare the performance of our system to the top two systems that participated in the above tasks (Palmer et al., 2001; Snyder and Palmer, 2004; Pradhan et al., 2007). The evaluation results are shown in Table 3. IMS easily beats the WNs1 baseline. It ranks first in SensEval-3 English fine-grained all-words task and SemEval-2007 English coarse-grained all-words task, and is also competitive in the remaining tasks. It is worth noting that because of the small test data set in SemEval-2007 English fine-grained all-words task, the differences between IMS and the best participating systems are not statistically significant.

Overall, IMS achieves good WSD accuracies on both all-words and lexical-sample tasks. The performance of IMS shows that it is a state-of-the-art WSD system.

4 Conclusion

This paper presents IMS, an English all-words WSD system. The goal of IMS is to provide a flexible platform for supervised WSD, as well as an all-words WSD component with good performance for other applications.

The framework of IMS allows us to integrate different preprocessing tools to generate knowledge sources. Users can implement various feature types and different machine learning methods or toolkits according to their requirements. By default, the IMS system implements three kinds of feature types and uses a linear kernel SVM as the classifier. Our evaluation on English lexical-sample tasks proves the strength of our system. With this system, we also provide a large number of classification models trained with the sense-annotated training examples from SEMCOR, DSO corpus, and 6 parallel corpora, for all content words. Evaluation on English all-words tasks shows that IMS with these models achieves state-of-the-art WSD accuracies compared to the top participating systems.

As a Java-based system, IMS is platform independent. The source code of IMS and the classification models can be found on the homepage: `http://nlp.comp.nus.edu.sg/software` and are available for research, non-commercial use.

Acknowledgments

This research is done for CSIDM Project No. CSIDM-200804 partially funded by a grant from the National Research Foundation (NRF) administered by the Media Development Authority (MDA) of Singapore.

References

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic.

Yee Seng Chan and Hwee Tou Ng. 2005. Scaling up word sense disambiguation via parallel texts. In *Proceedings of the 20th National Conference on Artificial Intelligence (AAAI)*, pages 1037–1042, Pittsburgh, Pennsylvania, USA.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007a. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 33–40, Prague, Czech Republic.

Yee Seng Chan, Hwee Tou Ng, and Zhi Zhong. 2007b. NUS-PT: Exploiting parallel texts for word sense disambiguation in the English all-words tasks. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 253–256, Prague, Czech Republic.

Bart Decadt, Veronique Hoste, and Walter Daelemans. 2004. GAMBL, genetic algorithm optimization of memory-based WSD. In *Proceedings of the Third*

	SensEval-2	SensEval-3	SemEval-2007	
	Fine-grained	Fine-grained	Fine-grained	Coarse-grained
IMS	68.2%	**67.6%**	58.3%	**82.6%**
Rank 1 System	**69.0%**	65.2%	**59.1%**	82.5%
Rank 2 System	63.6%	64.6%	58.7%	81.6%
WNs1	61.9%	62.4%	51.4%	78.9%

Table 3: WSD accuracies on SensEval/SemEval all-words tasks

International Workshop on Evaluating Word Sense Disambiguation Systems (SensEval-3), pages 108–112, Barcelona, Spain.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

Yoong Keok Lee and Hwee Tou Ng. 2002. An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 41–48, Philadelphia, Pennsylvania, USA.

Jin Kiat Low, Hwee Tou Ng, and Wenyuan Guo. 2005. A maximum entropy approach to Chinese word segmentation. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, pages 161–164, Jeju Island, Korea.

Rada Mihalcea and Andras Csomai. 2005. SenseLearner: Word sense disambiguation for all words in unrestricted text. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL) Interactive Poster and Demonstration Sessions*, pages 53–56, Ann Arbor, Michigan, USA.

Rada Mihalcea and Dan Moldovan. 2001. Pattern learning and active feature selection for word sense disambiguation. In *Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SensEval-2)*, pages 127–130, Toulouse, France.

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. 2004. The SensEval-3 English lexical sample task. In *Proceedings of the Third International Workshop on Evaluating Word Sense Disambiguation Systems (SensEval-3)*, pages 25–28, Barcelona, Spain.

George Miller, Martin Chodorow, Shari Landes, Claudia Leacock, and Robert Thomas. 1994. Using a semantic concordance for sense identification. In *Proceedings of ARPA Human Language Technology Workshop*, pages 240–243, Morristown, New Jersey, USA.

George Miller. 1990. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–312.

Hwee Tou Ng and Hian Beng Lee. 1996. Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 40–47, Santa Cruz, California, USA.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 440–447, Hong Kong.

Martha Palmer, Christiane Fellbaum, Scott Cotton, Lauren Delfs, and Hoa Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SensEval-2)*, pages 21–24, Toulouse, France.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. 2007. SemEval-2007 task-17: English lexical sample, SRL and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic.

Benjamin Snyder and Martha Palmer. 2004. The English all-words task. In *Proceedings of the Third International Workshop on Evaluating Word Sense Disambiguation Systems (SensEval-3)*, pages 41–43, Barcelona, Spain.

Christopher Stokoe, Michael P. Oakes, and John Tait. 2003. Word sense disambiguation in information retrieval revisited. In *Proceedings of the Twenty-Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 159–166, Toronto, Canada.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco, 2nd edition.

David Yarowsky, Radu Florian, Siviu Cucerzan, and Charles Schafer. 2001. The Johns Hopkins SensEval-2 system description. In *Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems (SensEval-2)*, pages 163–166, Toulouse, France.