A Semi-supervised Approach to Word Sense Disambiguation
(CS446 Class Project)

Haoruo Peng (hpeng7@illinois.edu)
Shyam Upadhyay (upadhya3@illinois.edu)

December 6, 2013

Abstract

We investigate the task of word sense disambiguation using a semi-supervised approach. Word sense disambiguation is the task of identifying the correct sense for a word which can possibly admit multiple senses. We use the Semeval-2007 dataset to evaluate the performance of our system. We use a graph based approach to disambiguate senses; the underlying assumption is that the sense of the neighboring words can assist in inferring the sense of the target word. In this respect, the senses are learnt in a somewhat joint-manner. Using appropriate weights in the graph constructed for all possible senses, we can use the graph to obtain measures of centrality, which can guide the sense-disambiguation.

1 Introduction

In natural language, a word may be associated with possibly multiple meanings, depending on the context in which the word occurs. For instance, the word pen has the following senses according to Wordnet [8]:

pen : a writing implement with a point from which ink flows

pen : an enclosure for confining livestock

In linguistic parlance such words are called polysemous. *Word sense disambiguation*(WSD) is the problem of determining the correct sense of a polysemous word in a given sentence. For example, determining the correct sense of the word "pen" in the following passage:

```
Little John was looking for his toy box. Finally he found it.
The box was in the pen. John was very happy.
```

WSD can be viewed as a multi-class classification problem, where each word admits several possible senses and the task is to identify the correct sense of a given word given its context and knowledge sources. The inherent difficulty of WSD is evident from the fact that the target classes change for each word in the lexicon. In this respect, WSD involves training n different classifiers,

one for each word in a lexicon of size n. WSD is a key component for natural language processing systems which involves semantic interpretation of text. Tasks such as machine translation, information retrieval, data mining, web data analysis can greatly benefit from text disambiguation tools.

2 Background

The task of WSD was first described by Warren Weaver in 1949. He encountered the problem while working on machine translation. In those days, researchers such as Bar-Hillel felt that WSD could not be solved without modelling world knowledge. WSD has been described as an AI-complete problem [5], which means that it is computationally as hard as solving central artificial intelligence problems like the Turing test. Until 1970s, semantic interpretation systems used rule based patterns to resolve the senses, which were hand-coded and hence error prone. It was only in the 1980s, when lexical resources like Oxford Advanced Learner's Dictionary of Current English (OALD) became available and dictionary-based methods were used for the first time.

Today, significant progress has been made in WSD and researchers have achieved sufficiently high levels of accuracy on a variety of word types and ambiguities. A rich variety of techniques have been used, from dictionary-based methods [7] that use the knowledge encoded in lexical resources, to supervised machine learning methods [6] in which a classifier is trained for each distinct word on a corpus of manually sense-annotated examples, to completely unsupervised methods [14] that cluster occurrences of words, thereby inducing word senses.

Supervised learning approaches have been remarkably successful for performing word sense disambiguation. Instance-based learning and SVM approaches have proven to be the best systems in several competition. The only issue with supervised approaches is the lack of sense-tagged data which poses a severe bottleneck.

To address this problem, researchers have resorted to semi-supervised learning algorithms. The Yarowsky algorithm [14] was an early example of such an algorithm. It uses the 'One sense per collocation' and the 'One sense per discourse' properties of human languages for word sense disambiguation. From observation, words tend to exhibit only one sense in most given discourse and in a given collocation. The common motif in approaches like [14], [4] is that they allows both labeled and unlabeled data. Training involve iteratively labeling the unlabeled data using an initial seeding labeled set. The seed labeled data is used to learn a classifier which is used to assign labels to unlabeled data, which is merged with the initial seed data to obtain a larger dataset for the next iteration. Finally, the classifier is learned from the extended labelled dataset. More recently, researchers [12] have leveraged word alignment information from parallel corporas to aid in obtaining coarse grained senses. The aim of word-alignment task is to align a word in a given sentence to its translated counterpart in another sentence in a different language. Unlike sense-tagged datasets, good quality parallel corpora are readily available.

Other semi-supervised techniques use large quantities of untagged corpora to provide co-occurrence information that supplements the tagged corpora. These techniques have the potential to help in the adaptation of supervised models to different domains. Today, some of the key problems in WSD are domain adaptation and using coarse grained senses. Another closely related task to WSD is word sense induction, where the task is to cluster the senses which are similar to a target word. The reader is encouraged to peruse a more general survey provided in [11].

3 Task and Data

3.1 The Task

We are given a a document D and sequence of target words $(w_1, w_2 \cdots w_n)$ in the document. Each word w_i admits a set of candidate senses $S_i = (s_{i1}, s_{i2} \cdots s_{in_i})$ where $|S_i| = n_i$. The task is to assign each word w_i the most appropriate sense from its context in D.

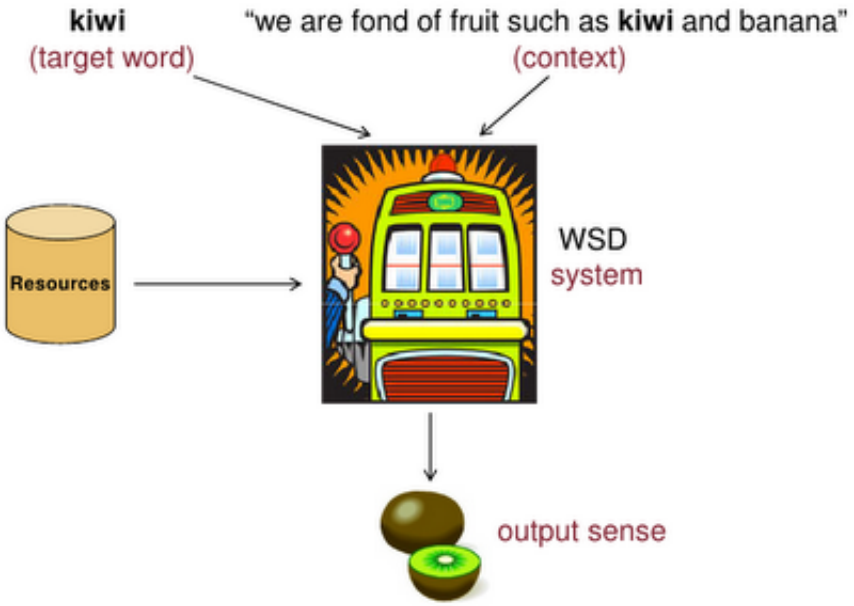

Figure 1: An example of word sense disambiguation system. Image courtesy [9]

3.2 Evaluation

WSD systems are evaluated by comparing in terms of their coverage,precision and recall with respect to a gold tagging(gold tags are provided by human). We allow a empty sense assignment in case the system is not able to disambiguate the word. So for each word w_i the output of the classifier lies in $S_i \cup \{\varepsilon\}$. We can now define our metrics as follows,

$$Precision = \frac{\text{\# of correct sense tags in the output}}{\text{Total \# of tags in the output}}$$

$$Recall = \frac{\text{\# of correct sense tags in the output}}{\text{Total \# of tags in the gold tagging}}$$

$$Coverage = \frac{\text{\# of tags in the output}}{\text{Total \# of correct tags in the gold tagging}}$$

3.3 The Data

We use datasets from **SemEval-2007** [3]. It is an ongoing series of evaluations of computational semantic analysis systems; it evolved from the Senseval word sense evaluation series. We choose a specific dataset from "Task # 7: Coarse-grained English all-words (Coarse AW)" to do both

the training and testing. The chosen dataset contains coarse grained tags for approximately 6,000 words from five running texts. The coarse senses were based on a clustering of the WordNet sense inventory obtained via a mapping to the Oxford Dictionary of English (ODE), a long-established dictionary which encodes coarse sense distinctions. The dataset also contains the coarse-grained sense inventory which is prepared semi-automatically: starting from an automatic clustering of senses produced by [10] with the Structural Semantic Interconnections (SSI) algorithm, and then manually validate the clustering for the words occurring in the text. For each content word the dataset provides lemma and part of speech.

Our work also requires some libraries such as "Wordnet" database and "WS4J" library. We use the the latest released version for Wordnet 3.0 from [2]. We compute the similarity metrics by utilizing WS4J library from [1].

4 The Models

4.1 Baseline Models

Random Baseline A naive baseline for word sense disambiguation. A randomly selected sense is made from the set of permissible senses for the target word.

First Sense Baseline (Most Frequent Sense) A more appropriate baseline model will be to output the sense of the word which is most frequently encountered. On coarse grained senses, using this baseline gives an accuracy ranging from 50-60%. As noted in [11], this baseline is often difficult to beat.

4.2 Our Model

We employ a graph-based approach to disambiguate senses of a word. Our method is similar to [13]. We define a fixed sized window around a target word in the document as its context. We first create a weighted graph of label dependencies for all candidate senses. For every candidate sense s_{ij} for word w_i, we have a node in the graph. This node has edges to the all candidate senses of words within a pre-defined window around word w_i. These edges are weighed by the dependency score for the pair of senses. The dependency score captures the relationship between two candidate senses for nearby words. Based on the edge weights we assign scores to each node based on a graph-based measure of centrality. The score of a node denotes the "importance" of the node in the graph, taking into account its relationship with neighboring nodes. Our method is different from [13] in that we make use of the coarse grain senses provided in the Semeval corpus to construct our graph. Consequently, our graph is less dense than the graph constructed in [13].

Algorithm 1: Algorithm to generate the graph (adapted from [13])

input : A document D and a list of ambiguous words $W = (w_1, w_2 \cdots w_n)$ such that $\forall i, w_i \in D$.
Each $w_i \in W$ admits a set of candidate senses $S_i = (s_{i1}, s_{i2} \cdots s_{in_i})$ where $|S_i| = n_i$

output: A graph G where a vertex represents a candidate sense of w_i and has edges to candidate senses of w_j, $i \neq j$. The cost of the edge is determined by a dependency scoring metric.

```
for word_1 ← w_1 to w_n do
    for word_2 ← w_1 to w_n do
        if word_1.position − word_2.position > window_max then
            break
        else
            for s_1 ∈ S_word_1 do
                for s_2 ∈ S_word_2 do
                    vertex_1 ← newVertex(s_1)
                    vertex_2 ← newVertex(s_2)
                    edgeCost ← getEdge(word_1, word_2, sense_1, sense_2, metric)
                    addEdge(vertex_1, vertex_2, edgeCost)
```

Algorithm 2: Algorithm to assign Senses

```
for word_i ← w_1 to w_n do
    sense_i ← max_{V_i ∈ S_i} score(V_i)
    where S_i corresponds to possible senses of word_i and V_i is the respective vertex
```

Algorithm 3: Algorithm to assign Senses

```
for word_i ∈ (w_1, w_2 ··· w_n) do
    sense_i ← max_{V_i ∈ S_i} score(V_i)
    where S_i corresponds to possible senses of word_i and V_i is the respective vertex
```

5 Experiments

5.1 Experimental results

6 Conclusion

Your current to-do list

Done

1. Implemented the code.
2. Obtained the Semeval 2007 dataset.
3. Added relevant related work.

Left to do

1. Running experiments with different similarity metrics.
2. Write-up.

References

[1] `https://code.google.com/p/ws4j/`.

[2] `http://wordnet.princeton.edu/wordnet/download/current-version/`.

[3] `http://www.senseval.org`.

[4] Anh-Cuong Le, Akira Shimazu, Van-Nam Huynh, and Le-Minh Nguyen. Semi-supervised learning integrated with classifier combination for word sense disambiguation. *Computer Speech & Language*, 22(4):330–345, 2008.

[5] John C Mallery. Thinking about foreign policy: Finding an appropriate role for artificially intelligent computers. In *Master's thesis, MIT Political Science Department*. Citeseer, 1988.

[6] Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[7] Rada Mihalcea. Using wikipedia for automatic word sense disambiguation. In *HLT-NAACL*, pages 196–203, 2007.

[8] George A. Miller. Wordnet: A lexical database for english. 1995.

[9] Roberto Navigli. Lecture 7. `http://naviglinlp.blogspot.com`.

[10] Roberto Navigli. Meaningful clustering of senses helps boost word sense disambiguation performance. In *In: Proceedings of COLING-ACL*, pages 105–112, 2006.

[11] Roberto Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10, 2009.

[12] Hwee Tou Ng, Bin Wang, and Yee Seng Chan. Exploiting parallel texts for word sense disambiguation: An empirical study. In *In Proceedings of ACL03*, pages 455–462, 2003.

[13] Ravi Sinha and Rada Mihalcea. Unsupervised graph-basedword sense disambiguation using measures of word semantic similarity. In *Proceedings of the International Conference on Semantic Computing*, ICSC '07, pages 363–369, Washington, DC, USA, 2007. IEEE Computer Society.

[14] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.