A Generalized Framework for Word Sense Diambiguation

Siddharth Patwardhan
School of Computing
University of Utah
Salt Lake City, UT 84112

1 Introduction

This document describes a generalized framework for a word sense disambiguation system using algorithms that use a bag-of-words approach. In this approach the an algorithm views a context simply as a bag of words, without any consideration for the sequence in which they appear.

The idea for this framework was born out of the *SenseRelate* package. The package is a highly specific algorithm for Word Sense Disambiguation, that uses the relatedness of senses of words in a context to perform the task. However, looking at the algorithm, it can clearly be seen that it consists of distinct stages. Each stage can be generalized to be done in one of many different ways. Thus, modularizing the code creates possibilities for research and experimentation with different ways of doing the various subtasks.

The following section will cover the generalized framework in some detail.

2 Modules

This section describes the various modules that appear in this framework. We work top-down, starting with input data, following the data flow up until the output (i.e. sense selection).

2.1 Format Filter Module

The filter takes as input annotated file(s) in a certain format. It parses the file creates instances, each containing a list of words and a pointer to the word that has to be disambiguated. Each instance is passed onto the disambiguation module for further processing.

The primary task of the filter module is to parse a formatted document. For example, the data we are working with is the SENSEVAL-2 data, which exists in XML formatted files. The filter would parse these files and build instances from these for the disambiguation module.

2.2 Disambiguation Module

The disambiguation module mainly exists to combine and co-ordinate the tasks of its submodules. Given an instance (list of words, with a pointer to an ambiguous word) the following tasks are performed:

1. **Preprocessing:** A preprocessing module would take a list of words, perform some preprocessing, and generate a processed list for the next stage. An example would be a module that detects compounds within the instances. Ideally, it would be convenient to have the option of having multiple preprocessing modules piped togather.

2. **Context Selection:** In a bag-of-words approach, a subset of words from the words around the ambiguous word are used by the algorithm for resolving the ambiguity. In the simplest case, we could select n words from around the ambiguous word in the context. However, for improving performance, we may wish to employ a more intelligent algorithm for context selection. Hence, the disambiguation module is free to choose a context selection module for its use.

3. **Sense Inventory:** The Sense Inventory attempts to determine the word senses of each of the words shortlisted during context selection. For each word, it decides the base form of the word and builds a list of senses for it. Currently, our sense inventory is WordNet, hence the list of senses is obtained from WordNet. However, in case the sense inventory changes, this module can be replaced with one designed for the new inventory.

4. **Optional Postprocessing:** Some optional processing can be performed on the data structures generated by the Sense Inventory module. This would include tasks such as *sense pruning*, which is my unofficial description of the process of removing some senses from the inventory, based on some simple heuristic algorithms. For example, we may decide (based on heuristics) that sense 3 of the noun form of the target word is not possible as the answer, and remove this from the sense inventory.

5. **Disambiguation Algorithm:** The disambiguation algorithm takes the list of senses of each of the context words and the of target word as input. It uses this information for selecting the correct sense of the target word. This module corresponds to the *local* and *global* disambiguation schemes of *SenseRelate*. But we could easily come up with other schemes (such as *random*)for selecting the target sense.

2.3 Preprocessing Modules

Any of the preprocessing modules should be able to take a list of words as input and return a list of (preprocessed) words as output for the next stage of the process.

Some possible preprocessing modules:

1. Compound Detection Module

2. Part of Speech Tagger

2.4 Context Selection Module

In order to disambiguate a word, the surrounding words are the only clues we have. In a bag-of-words approach the properties of the surrounding words are used to determine the correct sense of the target word. However, not all words in a context are indicative of the correct sense of the target word. An intelligent seletion of the context words used in the disambiguation process could yield much better results and generate a solution faster than if all the nearby words were used in the process.

The context selection module would take as input an ordered list of words, including the target word. The module would select, from this list, n words including the target word and send this list on to the next module.

2.5 Sense Inventory Module

At this point in the process, we have the list of words we plan to use for performing the disambiguation. However, the algorithm would need to know the possible senses of the target word. This list can be obtained from a resource, such as WordNet. However, a different source is possible.

In addition, the module can determine the senses for all the other words in the context, and pass this list to the next stage as well.

2.6 Postprocessing Modules

Some optional processing can be performed on the data structures generated by the Sense Inventory module. This would include tasks such as *sense pruning*.

2.7 Disambiguation Module

The disambiguation module takes the lists of senses of the target word and the context words and uses these to select the correct sense of the target word.

3 Implementation Details

This section describes the data structures used by the various modules.