CS 4740 Introduction to NLP
Spring 2010
Word Sense Disambiguation

Proposal: Due via CMS by Monday, March 12, 11:59pm
Results and Report: Due via CMS by Tuesday, March 27, 11:59pm

1 Objectives

This is an open-ended assignment in which you are to implement a Word Sense Disambiguation (WSD) system of your choice. Briefly, the input to the system will be a target word along with its surrounding context; the output of the system will be the sense ID(s) of the target word as it is used in that context. You will use training and test data of the English Lexical Sample task from Senseval-3. We have minimally preprocessed these data and, as in the first project, we have set up Kaggle so that you can submit your predictions there. As noted above, you can implement any WSD method of your choice. Additional information on how to get started is below.

2 Programming Portion

We suggest that you work in groups of three to five. Students in the same group will get the same grade. You will need to form groups using CMS in order to submit as a team.

1. To get started, look at the Senseval-3 web site (http://www.senseval.org/senseval3) to get a sense of whats there. Read the overview of the English Lexical Sample task on pages 2528 in the proceedings. This is a paper by Mihalcea, Chklovski and Kilgarriff.

2. The files for the assignment are lightly preprocessed and shuffled versions of the English Lexical Sample task from Senseval-3. They are available via CMS and include training data, test data, and a dictionary that describes the senses of each word. For every word, we have mapped each of its dictionary senses to a number in order of appearence in the dictionary (the first sense of a word is assigned to 1, the second to 2, etc). For each word we have reserved sense 0 to correspond to a word sense annotation of "unclear" or "none of the above". The training and test data is composed of lines in the following format:

`word t0 t1 ... tk @ context @ target @ context`

- `word.p-o-s` is the base form (`word`) and part-of-speech (`p-o-s`) (e.g., noun, verb, adj) of the word whose sense we are predicting. "begin.v", for example, means that `word` is "begin" and its part-of-speech will be a verb.
- `t0 t1 ...tk` are 1 or 0 depending on whether the corresponding sense is used in this example. At least one of these values will be 1. Sometimes more than one value will be 1.
- `@` is used as a delimiter.
- `context` is the actual text around the word whose sense we are predicting.
- `target` is the actual form of the ambiguous word as it appears in the text. For example, `word` may be "begin.v" while `target` is "beginning".

3. Scoring will be based on **fine-grained** scores. This means that incorrect predictions won't be getting partial credit if the predicted sense(s) are close, but not identical to, the actual one. Getting the fine grained opinion correctly is a hard task but makes evaluation on Kaggle simple. We are also investigating if we can set up a second Kaggle site where you can obtain a coarse-grained score. As in the previous project, you will upload your predictions to Kaggle and you will receive your score on a subset of the test set. The submission website is
`http://inclass.kaggle.com/c/cornell-cs4740-word-sense-disambiguation`
Submitting your predictions to Kaggle is not optional. The score we will use is *Accuracy*. The predictions of your system should therefore be 0 or 1.

   The prediction file you will submit to Kaggle should contain one number (0 or 1) per line. For each example on the test set, the prediction file should contain multiple lines. If an example refers to a word with k senses your system has to output $k+1$ lines for that example. Each of these lines is the prediction 0 or 1 depending on whether your system thinks sense i is being used in this context from $i = 0$ up to $i = k$. For example, if the first word in the test data has 4 senses and your system thinks senses 1 and 4 are used, then the first 5 lines of your submission file should be

```
0
1
0
0
1
```

4. You probably want to train separate models for each word.

5. As noted above, you may use any WSD method that you desire, and any publicly available software tools (except, of course, another WSD system). (If you have questions about what source code is ok vs. not ok to use, ASK US.) Two potentially useful sources are the Weka collection of machine learning systems, the OpenNLP software package, and the NLTK toolkit. The URLs for these are available from the course home page. There's no need to write your own machine learning algorithm for this assignment!!! Instead you can use any "off-the-shelf" systems that you can find.

6. You need to figure out and implement some simple mechanism to find the most informative features among the features that you used. Report the top 3 features. Note: You don't need to implement anything complicated for this. The brute force way of doing it could be remove a feature and see how much performance drops. There are various other ways — e.g., in linear regression you can use the absolute value of coefficient of the feature as a measure of its importance (so long as all your features are on a similar scale).

7. To make things easier, your system does not need to operate directly on the given files. You can put those files into any format that you'd like. The system response file (i.e., the output of the WSD system) WILL, of course, need to be in the format required by Kaggle.

8. It is also not necessary to write a script to ensure that the system runs end-to-end without any manual intervention. It's ok, for example, to convert the test data into your required format (by hand or automat- ically), to manually invoke a series of programs to build the training data set (if you are using a machine learning algorithm), or to convert the output of your WSD system (manually or automatically) into the format required by Kaggle.

9. Training vs. test data. When developing your system, please be sure that you don't use the test data: you should only be using (and looking at) the training data. The provided labels for the test data are not the actual labels. They have been provided for uniform parsing of the two files. When you upload your predictions to Kaggle you will get back your score on a subset of the test data. You will be able to see your score on the full test set after the submission deadline. You might also want to reserve some of the training data for test purposes during development.

10. Analysis. Leave enough time to provide an analysis of your results. This is really really important for your grade on the assignment.

3 Part 1: Proposal

Draft a description of your planned WSD approach. Will you implement a particular machine learning method? Use existing ML systems? What features

will you include? If you are developing a novel approach to WSD, provide a clear description of the algorithm and some motivation for why you selected the approach. If you are using a standard machine learning algorithm, it is NOT necessary to explain how it works.

The proposal should also include the group's plans for evaluation and for analyzing the system's performance. E.g., what kind of experiments will you run?

A draft of this part of the report is due on **Monday, March 12, 11:59 pm**.

4 Project Report

You should submit a short document (5-6 pages will suffice) that contains the following sections. (You can include additional sections if you wish.)

1. Method. Describe the approach that you took to WSD. (You should have basically written this part of the report for the proposal.) If you are using a standard machine learning algorithm, it is NOT necessary to explain how it works.

2. Software. List any software that your system uses that you didn't write yourself. Make clear which part of the system relies on this code if it's not obvious.

3. Results. Summarize the performance of your system on the test data. Minimally, you should compare your results to the most frequent sense baseline. Please put the results into clearly labeled tables or diagrams and include a written summary of the results.

4. Discussion of the results. This section should include any observa- tions regarding your system's performance e.g. When did it work well, when did it fail, why? How might you improve the system? What features were the most important?

5 Grading Guide

- Proposal (part 1) [5%]
- General design and implementation of the WSD system as well as the experiments to evaluate it [50%]
- Quality and clarity of the report and its analysis of the WSD system [45%]

6 What to Submit

Submit the following in a .zip or compressed .tar file to CMS — one submission per group:

- Source code (for code that YOU wrote, not for any of the packages that you used).
- A file that shows your system's output as it processes the training or test data. If this file becomes huge, please just include a portion of it. We just want to see your system "in action".
- The report (in .pdf).