Parsa Yadollahi
COMP 550 - Assignment 2
260869949

Problem setup:

This experiment evaluates the accuracy of four different knowledge-based approaches against the publicly available SemEval 2013 Shared Task #12 (Navigli and Jurgens, 2013).

The dataset was preprocessed and fine tuned for performance by following the first assignments basic preprocessing methods, where I cleaned the data (sentences) by removing all extra white spaces, replacing all capitalized characters for their respective lower case. Using the Natural Language toolkit (nltk) library, I tokenized the sentences and removed all stopwords as well as all punctuations.

Experimental Procedure:

Each knowledge-based approach applied the same methods to load the key and instances. For the experiment, the four knowledge-based approaches that were tested on the dataset were: Yarowsky's Algorithm with bootstrapping, Simplified Lesk's Algorithm along with the two baseline methods, Lesk's Algorithm and the most frequent sense. Yarowsky's Algorithm implementation was created for more than binary-sense classification. For each lemma in the test instance, a decision list classifier was used. Which were then used to evaluate the word sense of a lemma for a context by the one-discourse-per-category property discussed in the lectures. The Simplified Lesk's Algorithm was inspired by the pseudocode presented on the wikipedia page for simple Lesk's Algorithm. This Algorithm calculates the signature for overlap by finding the set of words in the definition examples of each candidate sense. The overlap function returns the number of words in common between the two sets, whereas the original Lesk Algorithm defines the context in a more complex way. The pseudocode for the Algorithm is presented in figure 1 which was also presented in the Wikipedia page.

Figure 1: Simplified Lesk Algorithm
Algorithm from Figure 1 in A.H & Abbas 2015.

```
Function Simplified_Lesk_Algorithm(word, sentence):
best-sense <- most frequent sense for word
max-overlap <- 0
context <- set of words in sentence
for each sense in sense of word do
  signa; <- set of words in the gloss_examples of sense
  overlap <- compute_overlap(signa, context)
  If overlap > max-overlap
      max-overlap <- overlap
      best-sense <- sense
end
return(best-sense){returns best sense of word}
```

Results:

For this experiment, the best performing model for Word Sense Disambiguation was the second Algorithm performed, that being Most frequent baseline count with an accuracy of 62.3%, which is more than twice the accuracy of Lesk and more than three times more accurate than Simplified Lesk and bootstrap. Where they all scored an accuracy of 31.1%, 29.7% and 27.6% respectively. The testing and dev performed relatively the same with accuracy ranging only +- 5-10%.

The following results are presented from the code where the first row presented with numbers is the test and the second row is the results of the code run on the dev keys instances and keys.

	Standard Lesk	Most Frequent Baseline	Yarowsky	Simplified Lesk
Test	0.311034482758 62	0.623448275862 06	0.275862068965 51	0.296551724137 93
Dev	0.386597938144 32	0.675257731958 76	0.324742268041 24	0.226804123711 34

Test Most frequent baseline count = 0.623448275862069
Test Lesk count = 0.31103448275862067
Test Simplified Lesk count = 0.296551724137931
Accuracy for bootstrap is 0.27586206896551724

Dev Most frequent baseline count = 0.6752577319587629
Dev Lesk count = 0.3865979381443299
Dev Simplified Lesk count = 0.2268041237113402
Accuracy for bootstrap is 0.3247422680412371

Conclusion:
In conclusion, the Most frequent Baseline showed the best results compared to all other models. Every other algorithm presented very bad results. A possible reason as to why, this Algorithm did so well compared to the rest is that the dataset seemed mostly generalized, and which would make sense that the most frequent senses for words will naturally perform well. Also, possible reasons the other algorithms did so badly, from my assumption, is caused by bad implementation on my part due to my lack of knowledge of how bootstrapping and Lesk is implemented. A possible reason why both Lesk Algorithms return such low performance and that they return results that are close to each other are due to the complexity of the sentences in the dataset and of the way the algorithm chooses the Synset. A reason why Yarowsky's Algorithm could be so low is that the accuracy of the seed set is much more important to the performance of the model. Since the Yarowsky Algorithm relies on the idea of one sense per collocation, any collocation that is defined incorrectly with the wrong sense by the seed set can have catastrophic effects in disambiguation.

Reference:
Aliwy, A.H., & Abbas, A.R. (2015). IMPROVEMENT WSD DICTIONARY USING ANNOTATED CORPUS AND TESTING IT WITH SIMPLIFIED LESK ALGORITHM.