1. Introduction

A fundamental characteristic of language is that words can have more than one distinct meaning. The lexical ambiguity is especially obvious to anyone who gets the joke when they hear a pun. The 121 most frequently used English nouns, which account for about 20% occurrences in real text, have on average 7.8 meanings each (1). However, for anyone who can fluently use English, the potential for ambiguous readings tends to be completely unnoticeable. This fact demonstrates that even though words have multiple meanings in principle, there is very little ambiguity to a real person in actual text. Step into the time of information explosion, nowadays people are trying to make machines “smart” enough to do anything for human beings. Making the machines to determine the meaning of every word in context is not a trivial task, computer scientists and software engineers thus came up with different algorithms to realize this goal. This type of problem in computational linguistics filed is called word sense disambiguation (WSD), which is essentially a task of classification. Word senses are classes, the context provides the evidence, and each occurrence of a word is assigned to one or more classes based on the evidence (2). Some common algorithms that currently used to handle WSD problem are including HyperLex algorithm, Lesk algorithm, Yarowsky algorithm, etc. In this project, we will focus on explaining (section 2), implementing (section 3) and evaluating (section 4) Yarowsky algorithm. We will implement Yarowsky algorithm in Spark with both Scala (for CS651) and Python (for CS631) drivers. Dataset for testing and code can be found in GitHub <https://github.com/aixeuy/DI-C-project>, and will be discussed in detail in section 3.

1. Original Yarowsky algorithm

The original Yarowsky algorithm is an unsupervised learning algorithm for WSD built by Professor David Yarowsky from University of Pennsylvania. He claims that this algorithm rivals the performance of supervised techniques that require hand annotations when trained on unannotated English text. This algorithm is based on two properties of human language:

1. One sense per collocation: some collocations are excellent features of the context that can be strong predictors for one sense or another.
2. One Sense Per Discourse: ambiguous tokens of the same type tend to be **correlated** within a document (i.e. discourse)

Moreover, as mentioned above, since a small proportion of English nouns are frequently used and account for about one fifth of occurrences in real text, we can at least conclude that English language is highly redundant. Therefore, the sense of a word is effectively overdetermined by 1) and 2) above. Section 2.1 and 2.2 discuss in detail how the two properties help to build Yarowsky Algorithm, section 2.3 explains the original algorithm step by step.

* 1. One sense per collocation

Prof. Yarowsky observed and quantified the strong tendency that words will exhibit only one sense in a given collocation in his work in 1993. The distance and adjacency of collocation, the predicate-argument relationship, and the burstiness of the words, all affect the influence

1. Princeton WordNet (Miller 1990). <https://wordnet.princeton.edu/>
2. Word Sense Disambiguation (Agirre, Eneko, Edmonds, Philip (Eds.) 2007). <https://www.springer.com/gp/book/9781402048081>
3. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods (David Yarowsky) <https://www.aclweb.org/anthology/P95-1026>