Senseval-2 system

Ildar Nurgaliev, Innopolis university

***Abstract*—Word sense disambiguation (WSD) is the task of determining which sense of an ambiguous word is being used in a particular context. The solution to this problem impacts other NLP related problems such as machine translation and document retrieval.**

***Keywords*—WSD, SENSEVAL, NLP, Machine Learning**

I. Introduction

There are several types of approaches to WSD, including dictionary-based methods, semisupervised methods, supervised methods and unsupervised methods. Supervised methods based on sense-labeled corpora are the best-performing methods for sense disambiguation. But such labeled data is expensive and limited. In contrast, dictionary-based methods and unsupervised methods require no labeled texts and are more efficient, but have lower accuracy. In the following part, we will see how the Lesk algorithm (unsupervised) make its work on WSD . Distinguishing homographs and polysemy is important.

II. Problem Statement

Given a sentence $s\ inS$ and a word $w \in s$ define its sense from the set of its synsets $E(w)$. Results evaluation carried out as follows:

- $Precision = \dfrac{\wedge(M)}{\wedge(M) + \Psi M}$, where $\wedge(M)$ is correct and $\Psi(M)$ is incorrect sense prediction.
- $Recall = \dfrac{\wedge(M)}{n}$, where n is overall number of tests.
- $F_{1-score} = 2\dfrac{Precision \times Recall}{Precision + Recall}$

The value of $F_{1-score}$ is between 0 and 1. When $F_{1-score}$ is close to 1, we can conclude that (i) $Precision$ is close to 1, i.e., the fraction of errors in the set of guesses M is close to 1, and (ii) $Recall$ is close to 1, i.e., a large fraction of words from the whole set are defined right. Our task is to maximize the objective function $F_{1-score}$.

III. Testing data

We used test data from senseval dataset from NLTK. For simplification of the testing and verification the real sense of a word we chose just 3 labels HARD1, HARD2, HARD3.

The resulting synset that we obtain from lesk algorithm has another form, thus we mapped for every 'HARD' appropriate list of synset word. Wordnet by the given morpheme given a list of synsets, we have correctly choose one.

IV. Approaches

Collocational features are those that bear a position-specific relation to the target word, i.e. they encode information about the lexical items in specific positions to the left/right of the target word.

There we will use unsupervised algorithm Lesk from broad known NLTK library. We will analyze its behavior by executing with different parameters on the same test set $\{\forall s \in S : hard \in s\}$.

The Simplified Lesk chooses the word sense which has the most in common between: its dictionary definition and examples, and the context of the target word. Wse made data cleaning we removed stop words and punctuation from tokens of a sentence.

A. Improve algorithm

Lesk is based just on collocational features thus it would be useful to reduce this set of features from closed group words and stop words. In contrast to content words, function words are words that have little lexical meaning or have ambiguous meaning, but instead serve to express grammatical relationships with other words within a sentence. Function words are closed-class words. Languages do not easily add new words to this set. They are always relatively few and resistant to change. They are lexically unproductive and are generally invariable in form.

We increased the search space by the capability of wordnet system and it's data stored. As we revealed collocational features are not enough, thus we extended search in every syntec's hyponims (hypo from Greek is translated as 'sub'). From every hyponim of a syntec we take examples if it has one, afterwards we do its filtration by functional and stop words thus collect all words to glossary set. Finally we conduct the intersection of the sentence and a full glossary of hyponim tree of a syntec (s / functional words) $\wedge$ glossary. The resulting synset is that one which has bigger number of intersection words.

V. Results

A. Original lest from nltk

	pos=None
Precision	0.507
Recall	0.507
F1-score	0.253

B. updated lesk2 algorithm

	pos=None	pos='a'
Precision	0.556	0.567
Recall	0.497	0.497
F1-score	0.262	0.265

VI. Conclusion

As we could reveal in the experiments conducted, stop words and functional words have bad impact to the overall score of lesk algorithm as it could be considered as noise.Thus removing them in our updated implementation of lesk2 algorithm gave us better performance.