Unsupervised Word Sense Disambiguation in Sanskrit

Divya Agarwal
MTech, CSA, IISc
divyaagarwak@iisc.ac.in

Abstract

We look at various unsupervised Word Sense Disambiguation Algorithms based on the Lesk Algorithm and implement them for the Sanskrit language and compare the results to find out which one gives the best results.

1 Introduction

Words in Natural languages more often than not have multiple meanings attached to them. The correct meaning is figured out using the context of the words near them and the context of the sentence. As an example, the word *clear* has multiple meanings, clear as in, obvious, or clear as in, transparent.

This process of finding out the meaning or the *sense* of the word from a list of possible senses is called Word Sense Disambiguation. As it happens, humans are exceedingly good at Word Sense Disambiguation due to the vast knowledge of the world around.

WSD is not an easy problem for computers to solve and as such is an open problem in the field of Artificial Intelligence and Natural Language Understanding.

For Sanskrit, in particular, the problem of Word Sense Disambiguation is compounded by the fact that Sanskrit, like many other Indic languages doesn't have a natural word ordering in the sentence, say Subject-Object-Verb and so on.

We implement two algorithms, the Lesk Algorithm and the Cosine Lesk Algorithm for doing unsupervised WSD on a Sanskrit Database.

2 Tools Used

We have used `pyiwn` which is a Python based API for accessing the Indian Language WordNet. ()

We also used the SaWN (Sanskrit WordNet) created by CFILT, IIT-B (Centre for Indian Language Technology) to tag the test data. We did the tagging ourselves.

3 The Lesk Algorithm

The Lesk Algorithm introduced in 1986 () was among the first algorithms for unsupervised WSD. The idea behind the Lesk algorithm is that it will input phrases and that words occuring close to each other in a certain window are expected to refer to "similar" meanings and hence would have close senses to each other.

The algorithm looks at the multiple meanings of the word to be disambiguated and then compares all of them with the meanings of the words in a neighbourhood. The Word Sense is chosen as the meaning with the maximum intersection with the all the other possible Word Senses.

As an example, consider the phrase `Pine Cone` (the original example used by Lesk). The dictionary definitions for `Pine` and `Cone` are as follows.

PINE
1. kinds of evergreen tree with needle-shaped leaves
2. waste away through sorrow or illness

CONE
1. solid body which narrows to a point
2. something of this shape whether solid or hollow
3. fruit of certain evergreen trees

As can be seen, the best intersection is Pine-1 ∩ Cone-3 which contains two overlapping words - `evergreen trees` Hence the Word Sense for `Pine` as disambiguated by the Lesk Algorithm

will be meaning number `1`

4 The Cosine Lesk Algorithm

The original Lesk algorithm faces inherent problems because it relies heavily on the exact definition and the words used by the definitions in the WordNet. As an example, for `Pine Cone`, if the word `evergreen` is replaced by `coniferous` in *exactly one* of the two places, then the word `Pine` will stay ambiguous as multip,e words occur at least once in multiple definitions. This problem can be solved by using the cosine as a measure of similarity. ()

We measured the similarity in the vectors of the sentence around the word to be disambiguated and the *gloss*, i.e. the meaning as given in the WordNet. The meaning with the maximum similarity is chosen as the appropriate word sense.

5 Results

For Testing purposes, we have used a manually sense annotated dataset sense annotated for 257 words. The Results are as follows.

Indicator	Lesk	Cosine Lesk
Correctly Classified	96	100
Wrongly Classified	67	63
Unclassified	94	94
Accuracy (in percentage)	**37.35**	**38.91**

The Cosine Lesk gives slightly better results than the Lesk Algorithm, as is expected as the Cosine is not a binary measure of similarity like the intersection.

6 Github Link

https://github.com/rama21094/WSD$_{Sanskrit}$

References

[1] Word sense disambiguation using WordNet and the Lesk algorithm .Jonas EKEDAHL2004

[2] Word Sense Disambiguation for Hindi language Rohan SharmaIPASJ International Journal of Computer Science 2007

[3] Ritesh Panjwani, Diptesh Kanojia and Pushpak Bhattacharyya, pyiwn: A Python based API to access Indian Language WordNets, GWC 2018, Singapore, January, 8-12, 2018.