Word Sense Disambiguation Algorithms in Hindi

Drishti Wali (13266) Nirbhay Modhe (13444)

Department of Computer Science and Engineering, IIT Kanpur

April 18, 2015

Abstract

Word sense disambiguation (WSD) is the task of automatic identification of the sense of a polysemous word in a given context. Quite a lot of work has been done for WSD in English, inspired mainly by the Senseval and SemEval tasks. In this project we use the model of word representation stated in the paper by Chen, Liu and Sun[3] for a Hindi corpus. This model represents the context and each sense of the target word as a vector in a high dimensional space and measures their similarity. The most similar sense is the chosen as the correct sense of the target word in the given context.

Contents

1 Motivation

Word sense disambiguation has lot of applications in improving search engines, machine translation, resolution of Anaphora etc. The first work in Hindi word sense disambiguation was done by Pushpak Bhattacharyya in 2004.[8] Following this paper other works using bilingual methods and graph based approaches have been researched. The dearth of resources in Hindi has prevented the successful application of supervised algorithms in Hindi. The introduction of new approaches in English including clustering based method[7] and Chen, Liu and Sun's method to improve word representation have motivated their application to Hindi Word Sense Disambiguation. Other methods in this field include Agirrie,[1] Yarowsky[9] and Lesk's[4] knowledge-based approach. Word vectors can be obtained for every word in Hindi given a large corpus, however the target word vector would consist of an agglomeration of the different senses. So, the aim is to find the most appropriate sense vector and context vector representation.

2 Resources and Corpus

2.1 Hindi Wordnet

The Hindi Wordnet is a lexical database created by the Natural Language Processing group at the CFILT (Center for Indian Language Technology) in the Computer Science and Engineering Department and IIT Bombay. We use it to fetch all the senses of a word to be disambiguated, along with its synonyms, hypernyms, homonyms and example sentences.[6]

2.2 Hindi Corpus

The HindMonoCorp 0.5 corpus is a monolingual Hindi corpus originally used for machine translation. It is freely available for research purposes, and contains 787 million tokens in 44 million sentences. The morphological tags are provided for each Hindi word, hence we processed the corpus to remove everything except the lemmatized forms of the Hindi words.[2]

3 Methodology

Our method consists of first training word vectors from the corpus using Mikolov's Skip-gram model, followed by vectorizing all the senses and the context of a word to be disambiguated (target word). Then we use cosine

distance as a metric for comparing similarity of the context vector with target word sense vectors, with the sense of highest similarity being allocated as the disambiguated sense.

3.1 Word2vec : Skip-Gram Model

Skip-gram model is similar to an n-gram model, with the difference being that the skip-grams need not have consecutive words from the text under consideration i.e. some words in between can be skipped. Google's Word2vec code, developed by Tomas Mikolov was used to train vector representations of all of the 70 million distinct words in our corpus, in a 100 dimensional space.[5]

3.2 Sense and Context Representation

The senses of a word, which are fetched from the Hindi Wordnet, are represented as a single vector in the 100 dimensional space using the following procedure:

- For each sense, a collection of words is made from its synonyms, hypernyms, homonyms and gloss.
- Only those words above a set similarity threshold δ are averaged to obtain the vector for the given sense

Now that the sense vectors of the word to be disambiguated are obtained, we will also represent the context in which the word occurs as a vector. A context window of ± 6 words is taken and those words with similarity greater than δ will be averaged out to obtain the vector representing the context.

3.3 Similarity using Cosine Distance

Measurement of similarity of two vectors is done using a cosine distance metric. Cosine distance between two vectors A and B is given in dot product form as

$$cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

In terms of magnitudes of vector components, it is given as

$$cos(\theta) = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

A and B are first normalized to get the unit vectors so we just need to calculate the dot product to get $cos(\theta)$. Normalization is done for all word vectors beforehand. Words which occur together frequently were observed to have a much higher cosine similarity that words which are unrelated.

4 Results

The unavailability of any sense tagged test data makes it difficult to evaluate the accuracy of the model. To get a rough estimate of accuracy, we have manually tagged 100 occurrences of the Hindi word "aam" from an alternate hindi corpus from CIIL (Central Institute of Indian Languages) were taken. This word has four meanings in the Hindi Wordnet - "mango", "mango tree", "normal" and "general". These 100 occurrences of the word were taken along with a context window of ± 6 words for disambiguation.

4.1 Threshold Analysis

The threshold δ was varied from -0.1 to 0.3 and different levels of accuracy were obtained as shown in the table 4.1

Table 1: Effect of changing δ, the cosine similarity threshold on accuracy

δ	Accuracy
-0.1	53%
0.0	60%
0.1	54%
0.2	55%
0.3	52%

4.2 Example Cases

A few cases where our model failed to output the correct sense have been shown in the table. The four senses (as mentioned earlier) are numbered from 0 to 3.

Hindi Context Window	Result	Correct Sense
गरम पानी ठण्डा कर पिलाना चाहिये आम के परिपाकार्थ रात्रि का भोजन बन्द	0	1
और रासायनिक नियंत्रण के इस्तेमाल पर आम विरोध भी ऐसी कठिनाई है जो	3	2
हाथ ऊँचा किये हुए है जैसे आम के पेड़ से कुछ तोड़ रही	1	2
किये जाने वाले मैदानी फलों में आम का प्रथम स्थान है यह देश	0	2
र आधिक क्षारीय भूमि को छोड़कर आम लगभग सभी प्रकार की भूमी में	1	3

4.3 Insights

We made the following notable observations from the results obtained.

1. Averaging word vectors to obtain sense and context vectors doesn't give importance to certain key words which may be important in determining the correct sense.

2. The vector of a word with many senses may not be reliable at times, especially for setting the similarity threshold δ.

3. If a sense from the Hindi Wordnet has insufficient number of words in its gloss or synset, the vector of that sense is inaccurate.

4. Rarely occurring senses (which also did not occur frequently in the corpus) are difficult to represent due to lack of sufficient training examples.

5 Future Work

This approach seems promising as the word vectors accurately capture the meaning of words. Future work may include learning the weights of the relevant words in the sense or context vector. This would increase the accuracy of vector representation for disambiguation. Learning the weights of the words can be made possible using a neural network and an adequate number of training examples for each sense. Furthermore, other distance metrics for similarity may be tested. A comparison can be performed with other unsupervised methods such as clustering in a bag of words model.[7]

6 Conclusion

The algorithm by Chen, Liu and Sun was implemented for word sense disambiguation in Hindi using a large corpus to train word vectors and using

the Hindi Wordnet for obtaining senses of words. It was observed that such knowledge-based methods depend heavily on the length of the gloss present in the Wordnet. The corpus used was vast enough to not cause any problems related to lack of generality or lack of certain senses of words. This method was a significant improvement over Lesk's algorithm. Verification was done on a manually sense tagged small test set. The word vectors obtained were accurate in representing different senses for comparison of similarity.

Code : The code for our project is available here.

References

[1] E Aguirre. G. rigau (1996). word sense disambiguation using conceptual density. In *Proc. 16th international conference on COLING. Copenhangen.*

[2] Ondřej Bojar, Vojtěch Diatka, Pavel Rychlỳ, Pavel Straňák, Vít Suchomel, Aleš Tamchyna, Daniel Zeman, et al. Hindencorp–hindi-english and hindi-only corpus for machine translation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, 2014.

[3] Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035, 2014.

[4] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26. ACM, 1986.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[6] Dipak Narayan, Debasri Chakrabarti, Prabhakar Pande, and Pushpak Bhattacharyya. An experience in building the indo wordnet-a wordnet for hindi. In *First International Conference on Global WordNet, Mysore, India*, 2002.

[7] Joseph Reisinger and Raymond J Mooney. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics, 2010.

[8] Manish Sinha, Mahesh Kumar, Prabhakar Pande, Laxmi Kashyap, and Pushpak Bhattacharyya. Hindi word sense disambiguation. In *International Symposium on Machine Translation, Natural Language Processing and Translation Support Systems, Delhi, India*, 2004.

[9] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.