Polysemy Detection

Natural language is characterized by a high degree of polysemy, and the majority of content words accept multiple interpretations. NLP applications, such as the automated word sense disambiguation, require the ability to identify correctly context elements that activate each sense.

Approach

Word2vec model captures word semantics by using its distribution a large corpus. Given a large text corpus, word2vec gives us a dictionary where each definition is just a row of, say, 300 floating-point numbers. To find out whether two entries in the dictionary are similar, we use vector similarity.

- I/PRON am/VERB going/VERB back/ADV to/ADP my/ADJ diet/NOUN
- I/PRON have/VERB back/NOUN pain/NOUN

Consider a word like "back". No individual usage of the word "back" refers to the concept "return, or the rare side". But that's the concept that word2vec is trying to model because it smashes all senses of the words together. What we want to do is have different vectors for the different senses of the same word.

Sense2vec: Word2vec model - words with POS for more precise vectors
Sense2vec model is built by doing little modification on top of the Word2vec model generation code. In Sense2vec words combined with the POS tags are the key in the model. For example, back_NOUN and back_ADV will be two separate entries in the Sense2vec model.

Packages / methodologies we can use

Polysemy detection is still a very open ended problem. There is no standard package/methodology available for this. As I explained we can build a Sense2vec model by extending standard Word2vec model. We can use Word2vec model training code and make the modification to include the POS tag with the word token to represent distinct senses of same word.

Problem with this approach and that can be overcome

to build effective Sense2vec model based on Word2vec we need large text corpus in the domain we will be working. If we are working in nutrition domain then lack of large corpus might be a problem for this approach.