Word senses in information retrieval

Abstract

Information retrieval (IR) often ignores word senses in document relevance calculations. This is largely due to the fact that word sense disambiguation is not an easy task. However, in IR some of the strictest requirements of traditional approach to detailed disambiguation can be relaxed. This paper discusses a successfully implemented approach to using word senses in IR tasks, which can be combined with various IR methods.

Mirva Salminen
mirva.salminen@helsinki.fi

University of Helsinki
Department of Computer Science
Research Seminar on Intelligent Systems, Autumn 2004
Organizer: Petri Myllymäki
Helsinki 4th November 2004

1 Introduction

Information retrieval (IR) systems are used to retrieve documents from document collections that are relevant to a posed question. Questions are often composed of one or more words such as "Bush" or "George Bush" (looking for documents of president George Bush). These requests do retrieve the relevant documents, but they retrieve also documents with the other senses of word bush, like bush as plant or vegetation. This problem is not rare, since word sense ambiguity is always present in text. Nonetheless, many IR systems do not use semantical information when retrieving documents.

However, disambiguation is not an easy task, which is one of the reasons why word senses are often ignored in IR. Much research on disambiguation area has been done and methods have improved, so it is ever closer to benefit from disambiguation in IR. This paper will discuss different aspects of using word sense disambiguation in IR and of how to implement it effectively.

First, the paper will discuss different approaches to using word sense disambiguation (WSD) in IR and the reasons why WSD has failed [San00]. Second, a promising IR application is introduced [KSR04], its evaluation is presented, and it is discussed, why it has succeeded in using WSD while many others have failed.

2 Failures of word sense approaches

The disambiguation task in IR has been approached mainly from three different perspectives [San00]:

1. Dictionaries and synonym sets have been used to choose the correct interpretation for a word.
2. Word senses are derived from corpora.
3. Retrieval is done without identifying word senses.

The first approach assumes that a word has a predefined number of (dictionary) senses and the task of disambiguation is to decide, which of these senses is the right one. The decision can be done in various ways. For example one can use the dictionary descriptions of words, compare them with the word context in the text, and choose the (correct) sense. The chosen sense has the largest number of common words with the context. As another example, if one has a dictionary where words are organized into a hierarchical hyponym tree, such as WordNet (WN), one can calculate the semantic distance between any two words. Then given an ambiguous word appearing in text, all the synomym sets (senses) containing that word are looked up in WN, and the senses are scored based on the semantic distances between the context of the word and the sense. The highest scored sense is chosen. The problem of choosing the right sense for the word is that the other words appearing in the context can also be ambiguous. Thus the problem accelerates as the sense combinations increase exponentially. The precision of disambiguators needs to be high, if they are to be of benefit for IR systems, otherwise those documents that are assigned faulty word senses reduce the precision and recall of the result. Precision is often lower the more fine grained the senses are. For IR purposes a more coarse grained, but more accurate method might be useful.

The second approach assigns senses to words using their contexts taken from the document collection. The contexts can be clustered and the clusters then represent word senses. This approach leaves open the question of how the users could mark up their query with word "senses", since the senses are not "real" senses but are defined by the clusters of surrounding context words. Also, the problem of clustering is that it often requires heavy computation.

The third approach does not use the word senses for disambiguation when choosing the right documents, but instead uses the information of multiple senses in other ways. For example, query words can be assigned weights of query relevance based on ambiguousness: words with many senses are probably less useful in retrieval than words with only one sense, since they are more likely to bring irrelevant documents to the result set. The drawback is that, not all senses are covered by dictionaries, and also some queries have junk words that are irrelevant to the query despite their non-ambiguousness (e.g. "I want articles that").

3 Root sense tagging approach

[KSR04] presents an IR method using word senses that has promising results. The paper introduces the root sense tagging (RST) method, which is designed to overcome some of the problems faced in the previous IR systems using word senses.

3.1 Three principles

The RST approach is based on three principles, which all aim to solve some problems of disambiguation in IR:

1. The RST approach does not use fine grained senses but instead uses *coarse grained disambiguation* that is assigns words only to their root senses. For example, "actor" has two fine grained senses — actor as a doer and actor as a role player — , but only one coarse grained sense — person.
2. Accurate disambiguation is complicated, because the context of words in different documents — even the context of the words with same fine grained sense — differs some times, and also different senses can appear in similar context. The RST approach thus relies on *consistent disambiguation* instead of accurate disambiguation. Consistent disambiguation assumes that it is more important to assign senses in a consistent manner even though the consistent manner is sometimes faulty.
3. Word sense disambiguation does not yet reach high enough accuracy for IR needs. This problem is passed by using *flexible disambiguation* instead of strict one, which allows for multiple sense assignment for a word.

3.2 The disambiguation

RST approach uses WN to assign senses to words. WN (version 2.0) has some 110.000 unique strings of nouns organized into some 80.000 synsets i.e. synonymy sets that have their own sense [Fel01]. The synsets are

act	animal	artifact
attribute	body	cognition
communication	event	feeling
food	group	location
motive	object	person
phenomenon	plant	possession
process	quantity	relation
shape	state	substance
time		

Table 1: 25 unique beginners of WordNet.

organized into a hierarchical tree that has 25 roots (unique beginners, see table 1) based on relations such as hypernymy (is-a-kind-of) or meronymy (is-a-part-of). Through the hierarchy, all the synsets can be traced back to their root senses.

Disambiguation task of as high a number of senses as 80.000 would require a huge document collection, not to mention that the statistical information alone is often not enough. In comparison, the assignation of a sense within 25 senses is a much more easy task than that within 80.000 senses, which is the point of coarse grained disambiguation. Hence, the RST approach uses the 25 root senses for the nouns in its disambiguation.

In RST, if a word is non-ambiguous, i.e. it has only one root sense in WN, then the word is classified with this root sense. For example word *actor* has two senses but only one root:

```
Sense 1:
actor, histrion, player, thespian, role player
       => performer, performing artist
           => entertainer
               => person, individual, someone, somebody,
                  mortal, human, soul

Sense 2:
actor, doer, worker
       => person, individual, someone, somebody, mortal,
          human, soul
```

If a word is ambiguous, i.e. it has multiple root senses in WN, it is classified based on context words in the documents that appear within a small window. In practice, the disambiguation is done on MI-based root sense tagging using co-occurrence data. For clarification of the disambiguation method and co-occurrence data construction, let us consider a document with three phrases:

"On March 20, the Fed lowered interest rates by a half percentage point."
"Bank keeps interest rates on hold."
"Rate comparisons available on this site."

The construction of the co-occurrence data is done in five steps. First, all the non-ambiguous words are assigned a root sense: there are no non-ambiguous words. Second, for all second nouns of non-ambiguous compound words are assigned the root sense of the compound. E.g. the *interest rate* has only one root sense, so *rate* is assigned root sense *possession*:

```
interest rate, rate of interest
     => rate, charge per unit
         => charge
             => cost
                 => outgo, expenditure, outlay
                     => financial loss
                         => loss
                             => transferred property,
                                transferred possession
                                 => possession
```

Third, if any noun tagged in step two occurs alone, it is assigned the same root sense as in above. For instance, *rate* in third sentence will be assigned with the root sense *possession*.

Fourth, for each sense-assigned noun in the document, all *(context word, sense)* pairs within a predefined window are extracted. In experiments the window is set to 2. Only nouns, verbs and adjectives are considered as content words and to compose the window. For example, in the first sentence around the root sense assigned word *rate*, the extracted *(context word, sense)* pairs are *(lowered, possession)*, *(interest, possession)*, *(half, possession)* and *(percentage, possession)*.

Fifth, all *(word, word)* pairs are extracted within the same window as in step four. *(word, word)* pairs of sentence three are thus *(rate, comparisons)*, *(rate, available)*, *(comparisons, available)*, *(comparisons, site)*, *(available, site)*.

The *(context word, sense)* and *(word, word)* pairs compose the global co-occurrence data which is used in disambiguation with MI-based root sense tagging. The disambiguation is done in following way:

For an ambiguous word c the most related context word $c(w)$ is chosen with formula

$c(w) = argmax_{ci \in cw(w)} MI(cw, c_i)$,

where $cw(w)$ is the set of context words for w. For example, the set of context words for word *interest* in sentence one are {Fed, lowered, rates, half}. The most related context word is *rate*, since it has the highest MI value in the co-occurrence data because it occurs twice with *interest* in the document, while all the other content words occur only once.

Next, the highest MI-valued candidate root sense $s(w)$ with the selected $c(w)$ is chosen with formula:

$s(w) = argmax_{si \in cs(w)} MI(c(w), s_i)$,

where $cs(w)$ is the set of candidate root senses of w defined in WN. Since the $c(w)$ is *rate*, the $cs(w)$ is {possession, relation, attribute} according to WN. Of these senses, the sense *possession* has the highest mutual information (MI) value in the co-occurrence data, since no other of the senses are tagged to *rate* in the example document. Thus, the word *interest* is assigned word sense *possession*.

Sometimes the sense cannot be assigned: this is the case, when there is no co-occurrence data available for defining $cw(w)$ or $cs(w)$, or the word is not found in WN. These cases are tagged so that in the first one the sense $s(w)$ in set to *null* and in the latter to *unk*.

So, the sense is assigned according to only one context word having the highest mutual information with the given word, and also this way the most probable candidate root sense is found. Thus, the tagger always assigns the same root sense to the word when the word occurs with its frequently co-occurring word. This enables consistent disambiguation especially for collocations and compound nouns. Accurate disambiguation can experience a setback here, but since the same system is used both in the documents and in the query, the senses will match and the objective is fulfilled.

Now, if a word is tagged in same document with multiple senses, these senses are merged. The merging is done with bitwise-OR operation. For example, if the document was different and word *interest* would have been tagged with two senses, like *possession* and *cognition*, both of these senses are attached to the word *interest*, so that neither of the senses is lost. This multiple sense tagging enables flexible disambiguation.

3.3 Retrieval

The senses assigned to words are used in retrieval to rank the retrieved documents. This is the *sense-oriented term weighting method*.

The retrieval uses a sense index that includes all the words in the document and 26 bits for the senses. The first 25 bits of the sense tagging field indicate the 25 root senses: if the bit is 1, then the word is assigned with that sense, if it is 0 then the word does not have that sense (see figure 1). The last bit of the sense tagging field indicates the *unk* value, and for *null* senses all the bits are 0. If the word has more than one sense, and thus the senses are merged, then more than one bit has value 1 in that word's sense tagging field.

To calculate the sense weights for terms, we need to compare the senses assigned to the query terms and the document terms. This can be done with formula:

$$sw_{ij} = 1 + \alpha \cdot q(dsf_{ij}, qsf_i),$$

Word	Sense tagging field
bank – with one sense	0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
interest – with two senses	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0
word without context words	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
word not in WordNet	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1

Figure 1: Example of sense index.

where dsf_{ij} and qsf_i indicate the sense field of term t_i in document d_j and query q. The α indicates control of the impact of sense-matching result by function $q()$. In the evaluation, the parameter α is set to arbitrary value 0.5. The higher the α is, the more weight the sense-matching receives. Sense-matching function itself is defined with formula:

$$q(dsf_{ij}, qsf_i) = \begin{cases} 0 & \textit{if } (dsf_{ij} = 0) \textit{ or } (qsf_i = 0) \\ 1 & \textit{else if } (dsf_{ij} \textit{ and } qsf_i) \neq 0 \\ -1 & \textit{otherwise} \end{cases}$$

The function returns one of the three values: 0 if word has *null* or *unk* sense on either document or query, 1 if the word sense matches in query and document and -1 if the senses do not match. For example if the query has compound noun "interest rate" the term *interest* will be assigned sense *possession*, and hence the senses of term *interest* in the example document (introduced earlier) and in the query match and the function $q()$ returns 1. On the other hand, if the query does not include word *rate* and if query term *interest* is assigned e.g sense *state*, then the senses in the example document and in the query do not match and the function $q()$ returns -1.

To finally rank the retrieved documents and to calculate the final term weights, the sense weight sw_{ij} is multiplied by a term weight computed in the traditional non-sense regarding manner. The traditional term weight can be e.g. $tf \cdot idf$. The advantage is that the method does not ignore the traditionally used term frequencies, but instead takes them into concern and weights them with the sense. For example, the strongest weight will be given to term that has high $tf \cdot idf$ value and also has matching senses in both document and query.

3.4 Evaluation

The evaluation of RST method was done using TREC-7 and TREC-8 collections, and title and description queries [KSR04]. The document rankings were calculated in three different manners, so that sense weight was first combined with idf, then with $tf \cdot idf$ and last with $(1 + log(tf)) \cdot idf$.

The results show that root sense tagging mostly improves results, thought that is not the case every time. The improvements were at maximum 17,78% and at minimum -5,36%. The best results were perceived with idf in description queries, while the worst results with $(1+log(tf)) \cdot idf$ in title queries.

Also, the sense weighting was tested with pseudo relevance feedback and MB25 [RW00] methods. The pseudo relevance feedback provided similar results as the first tests, while MB25 method was used for testing different weights for parameter α in sense weight formula (for more detailed information about the evaluation, see [KSR04]).

Based on the results, it seems like sense weighting may deteriorate results in case of frequent terms, since frequent terms are more probable to be assigned with multiple senses within a document and thus match query terms too often. If this matching is also combined with higher term frequency (not-sense) weighting, like $tf \cdot idf$, the result may be worse than the original. The sense weighting might hence benefit from some kind of penalty for too frequent terms or alternatively it should be combined only with weights that do not reward for high term frequency.

4 Pros and cons

The pros and cons of the root sense tagging approach and the paper representing it [KSR04] are listed in table 2.

+ The root sense tagging method seems to be a useful approach to applying word senses in IR, since it mostly improves results. + The method takes the word senses into account, but does not require accurate and detailed disambiguation. + The method does not require heavy computation. + The method can be combined with different no-word-sense-using methods in document ranking.
- The root sense tagging method does not use verb and adjective senses of WN. - The method does not solve the problem of synonyms. - The paper does not test parameters with different values — except the parameter α in sense weight formula — e.g. tests with different window sizes would be interesting.

Table 2: Pros and cons of root sense tagging approach.

References

[Fel01] Fellbaum, C., editor, *WordNet. An electronic lexical database.* MIT-PRESS, 2001.

[KSR04] Kim, S.-B., Seo, H.-C. and Rim, H.-C., Information retrieval using word senses: root sense tagging approach. *SIGIR 2004*, Sheffield, UK, July 2004.

[RW00] Robertson, S. and Walker, S., Okapi/keenbow at trec-8. *Proceedings of TREC-8, 8th Text Retrieval Conference*, Gaithersburg, US, 2000, pages 151–161.

[San00] Sanderson, M., Retrieving with root sense. *Information Retrieval*, Vol. 2,Issue 1(2000), pages 49–69.