Adding syntactical information to the vector space model

Johanna Högberg
Umeå University
Umeå, Sweden
johanna@cs.umu.se

Abstract

We describe an on-going research effort that studies a generalisation of the vector space model, in which the target domain is syntactically annotated text and the index set consists of fractions of parse trees.

1 Introduction

The creation and manipulation of information is currently replacing the production of physical goods as the principle carrier of the global economy. The trend is visible almost everywhere we look. This year, for instance,

- the indexable world wide web has grown to contain more than 65 billion pages,
- EU member states receive about 55000 documents from the EU, the majority of which must be translated to each of the unions 23 official languages; and
- CERN's new particle accelerator is predicted to produce 15 petabytes of data. To store this data in CDs, one would need a stack more than 20km high.

Never before has so much data been available, but to make use of it, we need efficient methods of extracting the fraction of the information that we are actually interested in. In other words, we need good search algorithms. In this paper we focus on algorithms that operate against the backdrop of a large but finite set of documents, and that when given a query q computes a listing of the n documents that are most relevant with respect to q.

The performance of a search algorithm can be evaluated in terms of *recall*, *precision* and *rank*. Good recall means that there are few false negatives, good precision that there are few false positives. If the algorithm assigns weights to the documents returned, then the rank tells us how well a documents weight reflects its relevancy to the search query.

Many search algorithms precalculate an index of the document collection before they start to accept queries. A popular way of doing this is through the *vector space model* (Salton et al., 1975). Here, documents are identified with points in a vector space in which each dimension corresponds to a separate term. The number of times a term t appears in a document d determines the size of the t-component in the vector representation of d. Documents are considered to be semantically related if the distance between them is small, and clustered documents are seen to represent concepts. When the user formulates a query, the query is treated as a pseudo document q and mapped into the vector space. Each document in the data collection is weighted according to its distance to q, and the n documents with smallest weight are returned to the user. Two popular extensions of the vector space model are latent semantic analysis (LSA) (Deerwester et al., 1990) and random indexing (Sahlgren, 2005), both of which are methods to reduce the rank of the vector space while striving to preserve the relative distance between document vectors.

2 Adding syntactical information

A nice feature of the vector space model is that it works equally well if other objects than terms are used to index dimensions. All that is required is that one fixes a domain D of objects, and that indexing is made over a finite set of quantifiable properties of the objects in D. Alternative index sets described in literature include n-grams and sets of terms.

In the setting that we investigate, the domain is syntactically annotated text and the index set consists of fractions of parse trees.

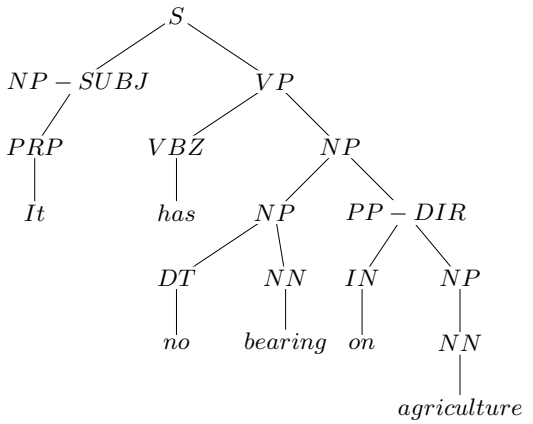

Figure 1: A sample from the Penn Treebank.

To avoid an exponential surge in the number of dimensions, the fractions are required to be of size at most k, where k is a user-provided constant. This combination of domain and index set is thus a proper generalisation of the vector space model which is regained by letting $k = 1$ and assigning the weight 0 to every fraction labelled with a part-of-speech tag.

We see a number of benefits of incorporationg syntactial information into the algorithm. As arguend in (Levin, 1985), the syntactical constructions in which a term appears give a strong clue about its semantics, so using this additional information may help to mitigate over-fitting when the data set is small.

It is also known that syntactic filtering of search results can improve precision (Chandrasekar and Srinivas, 1997) and in our setting the filtering is incorporated into the core algorithm itself. The impact that the occurrence of a term has can now be weighted to reflect the context in which it appears. For example, it seems reasonable to assume that if a term functions as the agent of a sentence, then this should matter more than if it is just one element in a long list of nouns.

Another potential advantage has to do with the the fact that different people tend to select different keywords for the same object. This lack of consensus leads to bad recall, but matters improve if queries are extended through paraphrasing prior to the invocation of the search algorithm. As shown in (Pang et al., 2003), paraphrasing is a natural language processing task in which adding syntactic information can lead to a better solution.

A disadvantage of the tree-based approach is that syntactically annotated text is still rare, but this may change in a not too-distant future. Suppose that an EU commission authors one document of legal text which must be translated into the 22 other official languages. One solution would be to hire a single linguist to add part-of-speech tags to the text, and then use machine translation to turn it into reasonably good translations in the target languages. Each of these rough translations can then be polished by a speaker that is an expert in one target language, but may be completely ignorant of the source language. If the original text is archived together with the syntactic information, then the algorithm described above can be used to search the archive.

3 Project status and future work

We are currently working on an implementation of the tree-based algorithm in Java/Flex. Input will be taken from the Penn Treebank of syntactically annotated English news text, a sample of which is shown in Figure 1. The performance of the algorithm will be compared with a reference implementation of the term-based vector space model, and results should be available towards the end of 2008.

References

Chandrasekar, R. and Srinivas, B. (1997). Gleaning information from the web: Using syntax to filter out irrelevant information. In *World Wide Web*. Stanford University.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41:391–407.

Levin, B. (1985). Lexical semantics in review: An introduction. In Levin, B., editor, *Lexicon Project Working Papers, No. 1*. Cambridge, MA: MIT Center for Cognitive Science.

Pang, B., Knight, K., and Marcu, D. (2003). Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proc. NAACL-HLT*.

Sahlgren, M. (2005). An introduction to random indexing. In *Proc. of MASIW, Int. Conf. on Terminology and Knowledge Engineering*, Copenhagen, Denmark.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613 – 620.