Using salience to rank documents

Patricia Grudziecka, Björn Isaksson
Department of Computer Science
Lund Institute of Technology
Box 118
SE-221 00 Lund, Sweden
d98ng@efd.lth.se, d98bis@efd.lth.se

Abstract

The idea described in this paper is to improve searches on the Internet by using the syntactic structure of sentences. We implemented a method which with help of context factors is able to capture the importance of a word better than a simple occurrence count. The prototype developed is slow and with the functionality limited to simple cases. Further suggestions of improvement of the method are given.

1 Introduction

Often when you search on the Internet you will be frustrated when you can't find the information you want. The search engines will give you popular documents that contain the words you search for. But you don't search for words, you search for content. I.e how the words are used in the document is more important than how often they are used.
The idea is to use the syntactic structure of the sentences in documents to rank the importance of words in a document. E.g. a word that is the subject of a sentence is more important to the content of the document than an object is.
A way to capture that idea is described in this paper.

2 Vector Space Model

A common way to rank documents is to use the Inverse Document Frequency which is based on the Vector Space Model. The idea of the vector space model is to represent documents and queries in a multi-dimensional space. Semantic equivalence of the query and document is said to be correlated with the proximity of the query and document vectors.
The coordinates or term weights are derived from occurrence counts as described below.

2.1 Term weights

The important question is how to weight words in the vector space model. The essential information used in term weighting is term frequency and document frequency.
The term frequency shows how salient a word is

Quantity	Symbol	Definition
term freq.	$\mathrm{tf}_{i,j}$	number of occurrences of w_i in d_j
document freq.	df_i	number of documents in a collection that w_i occurs in

Figure 1: Two commonly used quantities in information retrieval. w_i stands for word i and d_j stands for document j

in a given document. The higher term frequency means a higher likleyhood that the word is a good description of the content of the document. The relative importance of a word is often not a linear function of the occurrences of the word, but is taken as a logarithmic function(or another dampening function) of the term frequency. A docu-

ment with three occurrences of a word is more important than a document with one occurrence, but not three times as important.
Document frequency indicates the informativeness of the word. If a word occurs in many documents in the collection its relative importance is less than if it occurs only in a few documents. Therefore one can take the importance of the word as an inverse function of the document frequency.
A way to combine a word's term frequency and document frequency into a single weight is as follows:

$$weight(i,j)=\begin{cases}(1+\ln(tf_{i,j}))\ln\frac{N}{df_i} & \text{if } tf_{i,j}\geq 1\\ 0 & \text{if } tf_{i,j}=0\end{cases}$$

where N is total number of documents in the collection. This form $\ln\frac{N}{df_i}$ is often called *inverse document frequency.*

3 The use of salience to give term weights

In the idea of improving information retrieval, given in the introduction, the syntactic structure of a sentence is the deciding factor of how salient a word is instead of the term frequency. A new number describing the salience of the word instead of the term frequency is used in the calculation of the term weights.
This new number calculated by the weight of the context factors, we call the *aggregated context factor weights* (ACFW). When a word in a sentence is within the scope of the context factor its weight is added to the ACFW of the word. The salience value of an individual word in a document is obtained by adding the weights of the context factors which have that word in their scope:

$$SV(word)=\sum_i weight(CF_i^{word})$$

where SV is the salience value, CF is the context factor (see Figure 3).

After each sentence the word's ACFW is updated by the salience value of that sentence. Figure 2 shows how the ACFW of two individual words changes in the short text.

	cat	dog
The cat and the dog ran.	3+2=5	3+2=5
The dog chased the cat.	5+3=8	5+3+2=10

Figure 2: Example of ACFW calculation.

Context Factors	Objects in scope	Weights
Major-constituent	Subjects and objects	3
Subject	Subject	2
Nested-term	Noun phrase modifiers	1
Relation	Relative clause	3

Figure 3: Context factors and their weights

4 Implementation

A prototype has been developed to rank documents according to the method above. The prototype, called SalRank written in Java, is at this stage not able to rank documents on the Internet, but you have to supply it with text-files. This has been a conscious choice to avoid all the technical pitfalls of the real world and to concentrate on the basic idea. For the same reason there are only rudimentary user interfaces.
The program consists of two major parts, the data processing part and the user interfaces.

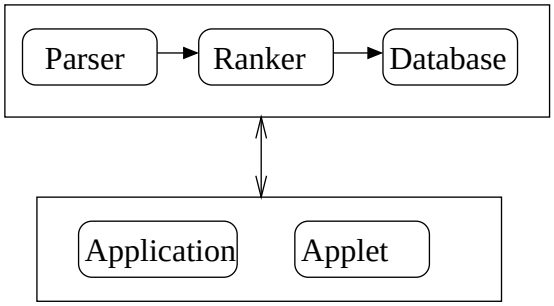

Figure 4: The data processing part above and the user interfaces below.

The data processing part consists of the Parser, Ranker and Database.

4.1 Parser

The parser parses a given text into sentences. At present it can only handle uncomplicated sentences and has trouble with abbreviations. It is

suitable for text-files and it can't read HTML-tags. Thus it is the practical obstacle to run the prototype on the Internet.

4.2 Ranker

In order to rank a document the Ranker has to obtain the context factors in a sentence. This it does with a grammatical parser, Link grammar.
Link grammar treats the words of a sentence as blocks with connectors. Every block has connectors pointing to the right or to the left, every connector is of specific type. A left-pointing connector connects with a right-pointing connector of the same type. The two connectors together form a "link". These links are used to decide the context factor of a word (see Figure 5).

Context Factors	Link types
Major-constituent	S, SI, J, O
Subject	S, SI
Nested-term	AN
Relation	R

Figure 5: Link types and corresponding context factors

Figure 6 gives an example of a parsed sentence with link types. The lower-case letters are connector subscripts that are not used in the implementation of SalRank (link type D connects a determiner with a noun).

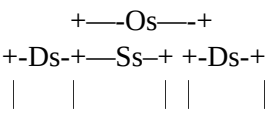

Figure 6: Example of link grammar representation of a sentence.

Depending on the link type of word it a different weight is added to its ACFW. In the example of Figure 6 3+2 (major+subject) is added to the ACFW of "dog" and 3 (major) is added to the ACFW of "cat". The ACFW is the weight of a word in a specific document.

4.3 Database

The Database is implemented as a hashtable with linked lists. Every entry in the hashtable corresponds with one word and contains a linked list. Every link in the list consists of an url and a rank for the specific word in that document.
The rank of every word is computed by the inverse document frequency using the ACFW instead of the term frequency. If a document does not contain a queried word the rank for that word is 0. The document is ranked by summing the ranks of the words according to:

$$r_D = \sum_i \begin{cases} (1 + \ln(a_{D,i})) \ln \frac{N}{df_i} & \text{if } a_{D,i} \geq 1 \\ 0 & \text{if } a_{D,i} = 0 \end{cases}$$

Where r_D is the rank of document D, i runs over the queried words and $a_{D,i}$ is the ACFW for word i in document D. Thus the document with the highest total rank is the best match for the query.
It is possible to save and load the Database.

4.4 Application & Applet

The application itself, SalRank, is mainly a user interface. The collection of text documents to be indexed will have to be in a specified directory. The documents are fetched by the application and are then passed on to the data processing part. When the data processing is finished one is able to query the database and get a presentation of the result, in form of a simple list with rank and url for the documents matching the query.
The applet is only able to query a database previously created by the application. The results are presented by the applet in a similar way to the application.

5 Evaluation

The data processing takes extremely long time. In our experiments we have had a collection of 33 files with a total of about 47000 words. It takes several hours to process this collection of documents on a Sparc 167MHz machine. The main bottleneck here is the Link grammar parser.
We tested the ACFW against the term frequency using inverse document frequency and an example of the result is presented in Figure 7. In the figure

there is also a comparison with Google.

Ranking results on: house decoration

ACFW	tf	Google*	Filename
13.81	12.01	4	chinadaily.txt
10.94	6.61	2	speel.txt
7.71	8.70	1	burrows.txt
7.07	8.14	3	philamuseum.txt
4.96	9.96	5	cleanairgardening.txt
3.03	3.96	-	house5.txt
1.79	3.19	-	house3.txt

∗Only relative order. “house5” and “house3” are not in the top 100 on Google.

chinadaily.txt: An article in China Daily about the mishaps of a customer trying to hire a house decoration company.

speel.txt: An historical review of Lord Leighton’s decoration of Arab Hall, Leighton House.

burrows.txt: Oscar Wilde’s lecture on house decoration given in 1882.

philamuseum.txt: A presentation of House Decoration Themes by the Park House Guides of Philadelphia Museum of Art.

cleanairgardening.txt: A commercial page selling bird- and bat-houses. The houses are “not decoration” according to the text.

house5.txt: Encyclopedia Brittanica’s entry on the house sparrow.

house3.txt: Encyclopedia Brittanica’s entry on the house mouse.

Figure 7: Test result of ranking with ACFW and term frequency with inverted document frequency.

Our subjective ordering of the documents is entirely consistent with the ordering given by SalRank. Although the first and the two last documents are rated in the same order there are some differences inbetween. The second highest rated document using term frequency, “cleanairgardening”, does not contain any information about “house decoration”. The term “not decoration” increases the rating for this document with term frequency. With the way the term is mostly used in the text e.g. “Bluebird house for bluebirds, not decoration!”, this does not increase the ACFW for “decoration”.

When it comes to the “speel”-text it is rated high with ACFW mostly because the word “house” in the ’right’ position in the sentences.

The differences in ratings between documents can intuitively be described as the relative importance of the documents. The difference between the highest and lowest ranking documents is larger with the ACFW method than with the term frequency method implying that the ACFW method is better in lifting important documents and suppressing irrelevant documents. But this is a fairly untested assumption.

6 Conclusions

The idea of SalRank was to improve the perceived correlation between the page content and the search query, on the Internet. We feel we have achieved this, but we would have benefited from a larger test collection to draw this conclusion for certain. At this stage SalRank most often returns a ranking list in the expected order of *our* subjective rating of the documents.

Some improvements of the program can obviously be made. The Parser should be able to fetch real Internet documents and to crawl the net for new documents, like a real search engine. To rank the documents one could exchange the Link grammar parser to a specially written sentence parser to extract the context factors. The database could be replaced by a regular database, like SQL.

A further improvement of our method to calculate the salience could be to take references of the words into account. One’s scope would then no longer be just one sentence at a time, but several on each other following sentences. A word with many references in following sentences would then receive a higher ACFW, than with the present method. The advantage of this method is that even if a word is only used once (or seldom) in a page, but referred to a lot, it is still important to the content of the page.

7 Acknowledgements

In our program we have used java-files written by Ola Åkerberg and Hans Svennson. The most important file was the java interface to Link grammar that is written in C.

We have also used Link grammar from http://www.link.cs.cmu.edu/link/.

References

Christopher D. Manning and Hinrich Schütze, 1999, *Foundations of statistical natural language processing*, MIT Press.

Carla Huls, Edwin Bos and Wim Claassen, 1995, *Automatic Referent Resolution of Deictic and Anaphoric Expressions*, Association for Computational Linguistics

Megumi Kameyama, 1997, *Recognizing Referential Links: An Information Extraction Perspective*, http://acl.ldc.upenn.edu/W/W97/W97-1307.pdf

Pierre Nugues, 2002, *Introduction to Language Processing and Computational Linguistics, Lecture notes*, Lund Institute of Technology

Davy Temperly, 1999, *An Introduction to the Link Grammar Parser*, http://www.link.cs.cmu.edu/link/dict/introduction.html

John Lafferty, 2000, *The Link Parser Application Program Interface (API)*, http://www.link.cs.cmu.edu/link/api/index.html