Comparison of New Simple Weighting Functions for Web Documents against Existing Methods

Byurhan Hyusein, Ahmed Patel, and Ferad Zyulkyarov

Computer Networks and Distributed Systems Research Group,
Department of Computer Science,
University College Dublin,
Belfield, Dublin 4,
Ireland
{bhyusein, apatel, feradz}@cnds.ucd.ie

Abstract. Term weighting is one of the most important aspects of modern Web retrieval systems. The weight associated with a given term in a document shows the importance of the term for the document, i.e. its usefulness for distinguishing documents in a document collection. In search engines operating in a dynamic environment such as the Internet, where many documents are deleted from and added to the database, the usual formula involving the inverse document frequency is too costly to be computed each time the document collection is updated. This paper proposes two new simple and effective weighting functions. These weighting functions have been tested and compared with results obtained for the PIVOT, SMART and INQUERY methods using the WT10g collection of documents.

1 Introduction

An important question in text retrieval is how to assign weights to the terms either in the documents or in the queries. Weighting is usually an automatic process. Manual weighting or indexing of documents would be more precise, but it requires a huge amount of human resources. The most successful and widely used function for automatic generation of weights for static databases are so-called $tf \cdot idf$ (or *TF-IDF*) weighting functions [1], where the abbreviations *tf* and *idf* denote term frequency and inverse document frequency.

The term frequency *tf* is the number of occurrences of the given term t within the given document D. It is a document-specific statistic and can vary from one document to another. Document frequency for a term t in the collection of documents C is the number of documents in that collection which contain the term t. Inverse document frequency *idf* is usually defined as a natural logarithmic function $ln\frac{N}{df}$ where N is the number of documents in the entire collection and *df* is the document frequency for the given term t.

A vector T is used to represent the set of unique terms t_j in the collection C, i.e. $T = (t_1, t_2, \ldots, t_{N_C})$ (dictionary vector), where N_C is the number of unique terms in C. Then each document D_i in the collection C is represented by set of terms t_j and/or their frequency tf_{ij} (or associated weight w_{ij} of the term) i.e.

A. Yazici and C. Şener (Eds.): ISCIS 2003, LNCS 2869, pp. 236–243, 2003.

$$D_i = \{(t_1, tf_{i1}), (t_2, tf_{i2}), \ldots, (t_{N_C}, tf_{iN_C})\}$$
$$(\text{or } D_i = (tf_{i1}, tf_{i2}, \ldots, tf_{iN_C}))$$

or

$$D_i = \{(t_1, w_{i1}), (t_2, w_{i2}), \ldots, (t_{N_C}, w_{iN_C})\}$$
$$(\text{or } D_i = (w_{i1}, w_{i2}, \ldots, w_{iN_C}))$$

The terms with frequency zero are usually omitted and the document is represented only by the terms which occur in the document with associated term frequency or weight. (We use this assumption throughout the rest of the paper.)

Two main problems arise from using *TF-IDF* weighting functions. The first problem is if a document D_i is represented by a set of terms t_j and/or their frequency tf_{ij} then the weighting has to be done at search time. If a document is represented by the terms and associated weights, then when new documents are added to or deleted from the collection the weights of the terms in the documents will change because *df* for the collection will be changed.

The second problem arises when a collection contains documents from several different topics in which case the document frequency factor may be large for the *topic specific terms*, decreasing the average contribution of these terms towards the query-document similarity. This problem has been found to be the case in the *Adaptive Distributed Search & Advertising* (*ADSA*) search engine (SE) which is a topic specific SE [2].

Typically the search process consists of checking the similarity between the query and documents in order to find the relevant documents. We assume that a document is represented as a weight vector $D_i = (w_{i1}, w_{i2}, \ldots, w_{iN_c})$, and a user's query $Q = (q_1, q_2, \ldots, q_{N_C})$, where q_i is a real number (usually $q_i \in [0, 1]$) which shows the importance of the term t_j for the query. Then the similarity between the query and the document is usually checked using the *inner product* function between the query vector and the given document vector:

$$sim(Q, D_i) = \sum_{j=1}^{N_c} q_j w_{ij}$$

When the similarity between the query and the documents in the collection is checked, the top documents (with the highest similarity to the query) are returned to the user.

2 Weighting

Three main components that affect the importance of a term in a text [3] are:

- the term frequency factor;
- the inverse document frequency factor; and
- document length normalization.

In SEs operating in a dynamic environment such as the Internet where many documents are deleted from and added to the SEs' databases in real time. In such instances if the weights of the terms in the documents depend on the *idf* then existing indexed documents need to be re-indexed everytime a new document (that has even just one term in common) is indexed simply because the *idf* of those terms has changed.

Document length normalization of term weights is used to remove the advantage that the long documents have in retrieval over the short documents. However many recent experiments show that document length normalization has little or no effect on Web document retrieval [4]. One of the main reasons that document length normalization is not successful for the Web is that such documents contain significantly less terms than ordinary text documents.

Having tested by experiments, we observed that weighting functions involving maximum term frequency normalization give very good results. The SMART and INQUERY systems define the weights by applying augmented maximum term frequency normalization [5,6]. However this may also cause problems when documents contain terms with unusually high frequency. In order to reduce the impact of the terms with unusually high frequency, the PIVOT function[7] applies logarithmic scale to term frequencies.

Based on the observation of the results of SMART, INQUERY and PIVOT and their shortcomings, we created the following weighting functions:

$$(W1) \qquad c_1 + \frac{1 + \ln(tf)}{1 + \ln(tf_{max})}$$

$$(W2) \qquad c_2 - \frac{1}{1 + \ln(tf)}$$

where $tf \geq 1$, $tf_{max} \geq 1$, $c_1 \geq 0$ and $c_2 > 1$ are constants (belief coefficients). The belief coefficients ensure that documents containing more keywords from the query will be considered to be more relevant than documents containing less keywords even though these terms appear more frequently.

The proposed new functions W1 and W2 are simple and do not depend on document requency. The purpose is that the weight (the importance) of a term in a given document should depend on its absolute frequency of occurrence but at the same time the terms with high frequency of occurrence should not drag the weights of other terms with less frequencies way down the scale. This is achieved by using the combination of *ln* function and belief coefficients that restricts the weights to a maximum values of $(c_1, c_1 + 1]$ for W1 and $[c_2 - 1, c_2)$ for W2.

The main advantages of the proposed weighting functions W1 and W2 are that they are suitable for use in dynamic document databases. Adding or deleting documents from a collection does not affect the terms' weights because the functions do not involve *idf*. W1 is similar to the function used for the calculation of the term frequency factor in PIVOT [7]. However, W1 can be used without *idf*. The second proposed weighting function W2 is simpler than W1 and does not depend on the term frequency of other terms belonging to the document.

From all our experiments for SMART, INQUERY, PIVOT, W1 and W2, we show in the next section how the proposed weighting functions achieve better results when querying documents.

3 Experiments and Results

In our experiments, we use the 10 gigabyte collection of documents (WT10g) [8] available for participants of Web track main task in the TREC-9 [9] experiments. This collection is a subset of the 100 gigabyte VLC2 collection created by Internet Archive in 1997 [10].

Queries are created using only the title part of Web track topics 451-500. The full text of all the documents was indexed. Stop word's lists of 595 English, 172 German and 352 Spanish words were used. Porter's stemming algorithm [11] was applied to both documents and queries. Average index size per document was 132.54 words. All terms in the query were equally weighted by one. Inner product is used as the similarity function in the experiments.

Many variations of PIVOT, SMART and INQUERY weighting functions have been used with different document collections by other researchers. In our experiments we used the functions shown below to calculate the term frequency factor for a term t within a document D. From each of the three families of weighting functions, the following gave the best results for the WT10g collection of documents:

$$(PIVOT) \qquad 0.4 + 0.6\frac{1 + \ln(tf)}{1 + \ln(tf_{max})}$$

$$(SMART) \qquad 0.5 + 0.5\frac{tf}{tf_{max}}$$

$$(INQUERY) \qquad 0.4 + 0.6\frac{tf}{tf_{max}}$$

Table 1. Summary statistics

	W1	W2	PIVOT	SMART	INQUERY
Retrieved	50000	50000	50000	50000	50000
Relevant ret.	1260	1223	1268	1220	1137
Relevant	2617				
Num. of queries	50				

The results obtained for W1, W2, PIVOT, SMART and INQUERY weighting functions are given in Tables 1, 2 and 3. The explanation of the results in the tables are as follows:

Table 2. Recall level precision averages

Recall	Precision				
	W1	W2	PIVOT	SMART	INQUERY
0.00	0.4260	0.4593	0.4260	0.4086	0.3696
0.10	0.3199	0.3068	0.3199	0.2893	0.2460
0.20	0.2593	0.2644	0.2592	0.2298	0.1897
0.30	0.2323	0.2234	0.2325	0.2004	0.1626
0.40	0.1837	0.1859	0.1830	0.1542	0.1289
0.50	0.1647	0.1705	0.1640	0.1363	0.1080
0.60	0.1236	0.1321	0.1228	0.0950	0.0846
0.70	0.1033	0.0970	0.1036	0.0830	0.0562
0.80	0.0664	0.0769	0.0675	0.0509	0.0288
0.90	0.0591	0.0689	0.0588	0.0413	0.0240
1.00	0.0362	0.0473	0.0360	0.0237	0.0175
Average precision over all relevant documents					
	W1	W2	PIVOT	SMART	INQUERY
non-interpolated	0.1592	0.1673	0.1591	0.1368	0.1126

– *Summary statistics*:
  - Retrieved - number of documents retrieved.
  - Relevant ret. - total number of relevant documents returned for all queries.
  - Relevant - total possible relevant documents within a given task.
  - Num. of queries - the number of queries used in the search runs.
– *Recall Level Precision Averages*:
  - Precision at eleven standard recall levels.
    The precision averages at eleven standard recall levels are used to compare the performance of the weighting functions and as the input for plotting the recall-precision graph (see Figure 1).
  - Average precision over all relevant documents, non-interpolated.
    This is a single-valued measure that reflects the performance over all relevant documents. The measure is not an average of the precision at standard recall levels. Rather, it is the average of the precision value obtained after each relevant document is retrieved.
– *Document Level Averages*:
  - Precision at nine document cut-off values.
    Each document precision average is computed by summing the precisions at the specified document cut-off value and dividing by the number of queries (50).
  - R-Precision - precision after R documents have been retrieved, where R is the number of relevant documents for the queries. The average R-Precision is computed by taking the mean of the R-Precisions of the individual queries.

The results are evaluated using the trec_eval package written by Chris Buckley of Sabir Research (available at ftp://ftp.cs.cornell.edu/pub/smart/).

Table 3. Document level averages

	Precision				
	W1	W2	PIVOT	SMART	INQUERY
At 5	0.2400	0.2480	0.2360	0.2000	0.1840
At 10	0.2060	0.2000	0.2060	0.1800	0.1720
At 15	0.1813	0.1640	0.1840	0.1587	0.1520
At 20	0.1660	0.1490	0.1660	0.1480	0.1390
At 30	0.1527	0.1387	0.1540	0.1367	0.1300
At 100	0.1034	0.0912	0.1030	0.0928	0.0802
At 200	0.0739	0.0707	0.0740	0.0691	0.0578
At 500	0.0419	0.0400	0.0418	0.0398	0.0366
At 1000	0.0252	0.0245	0.0254	0.0244	0.0227
R-Precision					
	W1	W2	PIVOT	SMART	INQUERY
Exact:	0.1909	0.1895	0.1934	0.1678	0.1380

4 Discussion of Results

To soften the influence of the terms with a high frequency of occurrence, the natural logarithm function was used in W1 and W2 weighting functions. Independence of document frequency ensures that adding or deleting documents from a collection does not affect the term's weights. Varying the weights of the indexed terms in the range of $(c_1, c_1+1]$ for W1 and in $[c_2-1, c_2)$ for W2 and using inner product similarity function, the documents containing more keywords from the query are considered to be more relevant than documents containing less keywords even if these terms appears more frequently.

We also found that the W1 and W2 proposed weighting functions are not sensitive to the belief coefficients c_1 and c_2. We achieved approximately the same precision and recall varying c_1 in the interval of *(0, 0.9]* and c_2 in the interval of *(1, 2.5]*. The best results are achieved using constant $c_1 = 0.9$ and $c_2 = 2.5$. We obtained 1260 *relevant* documents with W1 (48.14%), 1223 with W2 (46.73%), 1268 with PIVOT (48.45%), 1220 with SMART (46.61%) and 1137 with INQUERY (43.44%) weighting function of the total possible 2617 relevant documents (Table 1).

The recall-precision graph in Figure 1 compares the results obtained by newly proposed weighting functions W1 and W2 against the results obtained for PIVOT, SMART and INQUERY weighting functions. This graph is created using the eleven recall levels from the recall level precision averages (Table 2).

Figure 1 also shows that W1 and W2 give significantly better precision than SMART and INQUERY weighting functions in all of the eleven standard recall levels. For most of the recall levels W1 and W2 give better precision than PIVOT as well. In two recall levels W1 has the same precision as PIVOT and in six recall levels W1 is superior. W2 is superior in eight of possible eleven recall levels.

W1 and W2 weighting functions also give better results than PIVOT, SMART and INQUERY with respect to average precision over all relevant doc-

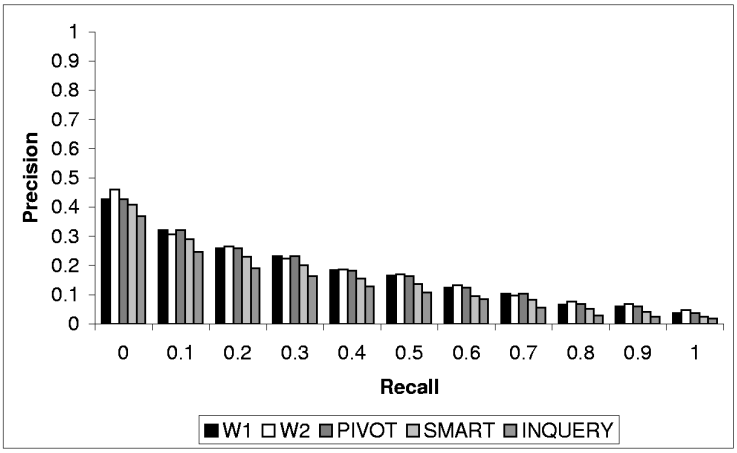

Fig. 1. Recall-precision graphs

uments (Table 2), significantly better R-Precision and precision at the nine document cut-off values than SMART and INQUERY (Table 3). Because of the greater number of retrieved documents for PIVOT it gives better R-Precision and better precision in the most of the document cut-off values.

5 Conclusions and Future Work

In this paper we proposed two new simple and effective weighting functions for Web document retrieval. The weighting schemes are tested and compared with results obtained for the PIVOT, SMART and INQUERY methods on the WT10g collection of documents.

Experiments showed that our weighting functions perform better than SMART, INQUERY and PIVOT weighting functions with respect to average precision and recall. The proposed weighting functions are computationally fast and suitable for use in SEs. Weighting function W2 is currently in use in the ADSA search engine because it gives better precision than W1 (which was used in the previous version of the ADSA search engine).

From the experimental results, our current research work is concentrating on the development of a more complex indexer which will allow any attribute of Web documents to be indexed. This will permit us to investigate in detail how the different parts of the documents contribute to the average precision in the process of search in a much more comprehensive manner.

Acknowledgement. The research was funded by Enterprise Ireland as part of the Enterprise Ireland Informatics Research Initiative.

References

1. G. Salton and C. Buckley. Term Weighting Approaches in Automatic Text Retrieval. *Information Processing and Management*, 24(5):513–523, 1988.
2. R. Khoussainov, T. O'Meara, and A. Patel. Independent Proprietorship and Competition in Distributed Web Search Architectures. In *Proceeding of the Seventh IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2001)*, pages 191–199. IEEE Computer Society Press, 2001.
3. G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, NY, 1983.
4. C. Buckley and J. Walz. SabIR Research at TREC 9. In *Proceeding of the 9th Text REtrieval Conference (TREC-9)*, pages 475–477. The National Institute of Standards and Technology, 2000.
5. R. Larson. Term Weighting in Smart, October 1998 Available from: http://www.sims.berkeley.edu/courses/is202/f98/Lecture18/sld021.htm [Accessed July 14th, 2003].
6. J. Broglio, J. P. Callan, W. B. Croft, and D. W. Nachbar. Document Retrieval and Routing Using the Inquery System. In *Proceeding of the Third Text REtrieval Conference (TREC-3)*, pages 29–38. The National Institute of Standards and Technology, 1995.
7. A. Singhal, C. Buckley, and M. Mitra. Pivoted Document Length Normalization. In Hans-Peter Frei, Donna Harman, Peter Schäuble, and Ross Wilkinson, editors, *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29, New York, 1996. ACM Press.
8. P. Bailey, N. Craswell, and D. Hawking. Engineering a Multi-Purpose Test Collection for Web Retrieval Experiments. *Information Processing and Management*, 2002.
9. D. Hawking, CSIRO Mathematical, and Information Sciences. Overview of the TREC-9 Web Track. In *Proceeding of the 9th Text REtrieval Conference (TREC-9)*, pages 87–102. The National Institute of Standards and Technology, 2000.
10. Internet Archive: Building an Internet Library, http://www.archive.org.
11. M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, July 1980.