Spring 2005

Improving document representation by accumulating relevance feedback : the relevance feedback accumulation (RFA) algorithm

Razvan Stefan Bot
New Jersey Institute of Technology

ABSTRACT

IMPROVING DOCUMENT REPRESENTATION BY ACCUMULATING RELEVANCE FEEDBACK: THE RELEVANCE FEEDBACK ACCUMULATION (RFA) ALGORITHM

by
Razvan Stefan Bot

Document representation (indexing) techniques are dominated by variants of the term-frequency analysis approach, based on the assumption that the more occurrences a term has throughout a document the more important the term is in that document. Inherent drawbacks associated with this approach include: poor index quality, high document representation size and the word mismatch problem. To tackle these drawbacks, a document representation improvement method called the Relevance Feedback Accumulation (RFA) algorithm is presented. The algorithm provides a mechanism to continuously accumulate relevance assessments over time and across users. It also provides a document representation modification function, or document representation learning function that gradually improves the quality of the document representations. To improve document representations, the learning function uses a data mining measure called "support" for analyzing the accumulated relevance feedback.

Evaluation is done by comparing the RFA algorithm to other four algorithms. The four measures used for evaluation are (a) average number of index terms per document; (b) the quality of the document representations assessed by human judges; (c) retrieval effectiveness; and (d) the quality of the document representation learning function. The evaluation results show that (1) the algorithm is able to substantially reduce the document representations size while maintaining retrieval effectiveness parameters; (2) the algorithm provides a smooth and steady document representation learning function; and (3) the algorithm improves the quality of the document representations. The RFA

algorithm's approach is consistent with efficiency considerations that hold in real information retrieval systems.

The major contribution made by this research is the design and implementation of a novel, simple, efficient, and scalable technique for document representation improvement.

IMPROVING DOCUMENT REPRESENTATION BY ACCUMULATING RELEVANCE FEEDBACK: THE RELEVANCE FEEDBACK ACCUMULATION (RFA) ALGORITHM

by
Razvan Stefan Bot

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Information Systems

Department of Information Systems

May 2005

APPROVAL PAGE

IMPROVING DOCUMENT REPRESENTATION BY ACCUMULATING RELEVANCE FEEDBACK: THE RELEVANCE FEEDBACK ACUMULATION (RFA) ALGORITHM

Razvan Stefan Bot

Dr. Yi-fang Brook Wu, Dissertation Advisor — Date
Assistant Professor of Information Systems, NJIT

Dr. Murray Turoff, Committee Member — Date
Distinguished Professor of Information Systems, NJIT

Dr. Vincent Oria, Committee Member — Date
Assistant Professor of Computer Science, NJIT

Dr. Nicholas J. Belkin, Committee Member — Date
Professor of Library and Information Science, Rutgers University

Dr. Bartel A. Van de Walle, Committee Member — Date
Assistant Professor of Information Systems and Management, Tilburg University,
The Netherlands

BIOGRAPHICAL SKETCH

Author: Razvan Stefan Bot

Degree: Doctor of Philosophy

Date: May 2005

Undergraduate and Graduate Education:

- Doctor of Philosophy in Information Systems,
  New Jersey Institute of Technology, Newark, NJ, 2005

- Master of Science in Information Systems,
  New Jersey Institute of Technology, Newark, NJ, 2004

- Bachelor of Science in Computer Science,
  Technical University of Cluj-Napoca, Cluj-Napoca, Romania, 1999

Major: Information Systems

Presentations and Publications:

Yi-fang Brook Wu, Quanzhi Li, Xin Chen, and Razvan Stefan Bot,
"Learning by Examples: Identifying Key Concepts from Text Using Pre-Defined Inputs," The 2005 International Conference on Artificial Intelligence (ICAI 2005), Las Vegas, Nevada.

Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen,
"Finding Nuggets in Documents: A Machine Learning Approach,"
Journal of the American Society for Information Science and Technology, (JASIS&T), 2005.

Razvan Stefan Bot and Yi-fang Brook Wu,
"Improving Document Representations using Relevance Feedback: The RFA Algorithm," Proceedings of the 13th Conference on Information and Knowledge Management, CIKM 2004, Washington DC, pp. 270-278, November 2004.

Quanzhi Li, Yi-fang Brook Wu, Razvan Stefan Bot, and Xin Chen,
"Incorporating Document Keyphrases in Search Results,"
AMCIS 2004, New York, NY, August 2004.

Razvan Stefan Bot, Yi-fang Brook Wu, Xin Chen, and Quanzhi Li,
"A hybrid classifier approach for web retrieved documents classification,"
Information Technology: Coding and Computing, 2004. Proceedings sponsored
by IEEE Computer Society. International Conference on Information Technology,
Vol. 1, April 5-7, 2004, pp. 326-330.

Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen,
"KIP: A Keyphrase Identification Program with Learning Function,"
Information Technology: Coding and Computing, 2004. Proceedings sponsored
by IEEE Computer Society. International Conference on Information Technology,
Vol. 2, April 5-7, 2004, pp. 450-454.

Soției mele, Corina, pentru toată dragostea, rabdarea si ințelegerea cu care m-a susținut de-a lungul acestor ani.

To my wife, Corina, for all the love, patience and understanding she supported me with during the last years.

Parinților mei, pentru toata dragostea, increderea, efortul şi sustinerea depusă de-a lungul intregii mele vieți.

To my parents, for all the love, trust, effort and support during my whole life.

Fratelui meu, Dragoş, pentru dragostea, prietenia şi increderea acordată.

To my brother, Dragoş, for all his love, friendship and support.

ACKNOWLEDGMENT

I would like to express my deepest appreciation to Dr. Yi-fang Brook Wu, who not only served as my research supervisor, providing valuable and countless resources, insight, and intuition, but also constantly gave me support, encouragement, and reassurance.

Special thanks are given to Dr. Nicholas J. Belkin, Dr. Murray Turoff, Dr. Vincent Oria, and Dr. Bartel A. Van de Walle for actively participating in my committee.

I would like to express my appreciation for Dr. Starr Roxanne Hiltz, who guided and helped me throughout all my years at NJIT.

My fellow graduate students in the Information Retrieval Group within the Information Systems Department also deserve recognition for the support and valuable feedback they provided.

Last, but not least, my wife, my parents, and my brother deserve a great amount of recognition for their thorough love, support, encouragement, and patience.

TABLE OF CONTENTS

Chapter Page

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF TABLES
(Continued)

Table **Page**

LIST OF FIGURES

LIST OF FIGURES
(Continued)

CHAPTER 1

INTRODUCTION

This thesis investigates the possibility of exploiting a priori relevance feedback assessments in an Information Retrieval system, in order to improve the quality of the document representations.

This chapter begins with an overview of the most important concepts related to Information Retrieval and the present work. This is done in Section 1.1. Section 1.2 presents the motivation and the idea of relevance feedback accumulation in the context of an Information Retrieval system.

1.1 Preliminary Definitions and Overview of Main Concepts

1.1.1 Information Retrieval

Information Retrieval (IR) is a widely used term today. IR is defined as the activity of finding and retrieving relevant documents from a document collection, in response to an information-need request formulated as a query. Each of us needs information on an every day basis in order to solve problems. A student needs to find good references in order to write a project paper. A lawyer needs to find precedents in order to support his or hers client's cause. A manager needs information in order to assess the position of his/her company, within the economic environment.

In the old days information was only available in libraries, newspapers or archives. The whole process of information retrieval was slow and painful. Today we are living in the information age. Most of the information is digitized and available "at a push of a button". Due to ever evolving computer and telecommunication technology, information is now available online. One of the most important sources of information is the Internet, also called the Web. Connected to the Internet, we have libraries, online journals, domain specific databases, governmental databases and many more. These are all sources of information to satisfy individual needs. But finding the information one needs is not an easy task. There are several reasons why information is difficult to find: first of all information is scattered around the web; second, the amount of information is overwhelming; third, information is presented in many different formats like text, audio, video or combinations. The most usual source of information is textual content.

1.1.2 Information Retrieval Systems

Information Retrieval systems (IR systems) accomplish several tasks. They gather and store information from scattered repositories; they organize this information for fast and effective retrieval; they deliver information to individuals based on their explicitly formulated information need (Korfhage, 1997, p. 1-6). Some well known IR systems are: Yahoo! at www.yahoo.com, Google at www.google.com, AltaVista at www.altavista.com, WebCrawler at www.webcrawler.com, HotBot at www.hotbot.com, Lycos at www.lycos.com and many more. A more comprehensive list of general and topic IR systems is available at http://www.allsearchengines.com. The generic architecture for an IR system is depicted in Figure 1.1, adapted from (Korfhage, 1997, p. 3).

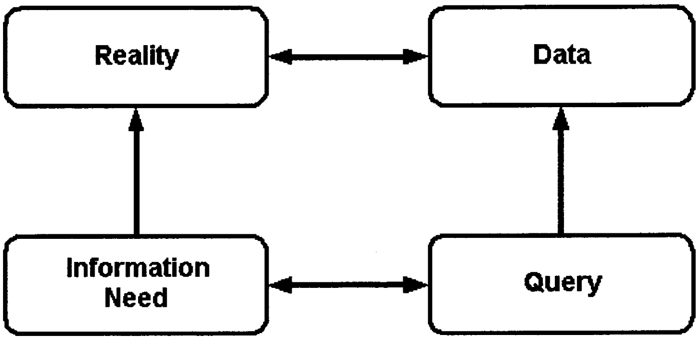


Figure 1.1 An IR System Abstraction.
Source: (Korfhage, 1997, p. 3)

Regardless of the method of implementation or of the model chosen, most IR systems conform to the generic architecture in Figure 1.1. The figure emphasizes the whole concept of retrieving information. On one side, the user has certain information needs from the real world. He/she expresses these information needs by means of queries. Such a query only approximates his/her real information needs. On the other side, the IR system works as a reality abstraction. It represents reality in terms of structured/unstructured documents ("Data" in Figure 1.1). Given the query formulated by a user, the system will then try to match it to the available data. The fit is judged in terms of relevance (see Section 1.1.4 for the definition of relevance).

1.1.3 Documents and Queries

In IR systems data/information is stored in documents. All the documents in the repository of a retrieval system form the document collection. A **document** is "*a stored data record in any form*" (Korfhage, 1997, p. 17). Thus, a document can be a text file, a sound file, a video file, a record in a relational database, an email etc. Everything that can be electronically stored in digital form can represent a document. In Figure 1.1, "Data" represents the document. The document is an abstraction of the reality. The most widely used document content type is text. The current work deals with textual documents. It is not necessary that large chunks of data, e.g. a book, be considered as single documents. In many cases the topic in such a document covers more than one aspect. It is better to divide the document into smaller sub-documents that model the sub-topics presented in the global document. These smaller representative documents are called ***surrogates***. Examples of document surrogates are abstracts, extracts, or reviews. An ***abstract*** is a few-paragraphs description of a document. From a well-written abstract a user can infer the topic of that document. An ***extract*** is a small document constructed out of important topic-related paragraphs, passages or sentences from the document. A ***review*** is similar to the abstract but is created by some other person than the original author, and it is usually critical in nature. One popular research thread is to use passages from documents as documents themselves, or to create extract-like surrogates (Salton et al. 1993; Mittendorf et al. 1994; Kaszkiel et al. 1999). There are also other methods like text summarization or text segmentation (Gong, 2001; Salton et al. 1996; Hearst, 1993; Hearst, 1994). Surrogates are useful for two reasons: first they are smaller and easier to handle, and second, they are more topic-oriented, emphasizing an important sub-topic of the bigger whole document (Fidel, 1986; Molina, 1995). This kind of analysis is called structure analysis since it discovers relationships between a document's sentences, paragraphs and chapters. The opposite technique is to perform full text retrieval. Some efforts in full-text retrieval can be found in (Martin, 1986; Miike et al. 1994).

The **query** expresses the information need to be retrieved by the IR system from its document collection. In Figure 1.1, the query is depicted as the observable representation of the information need. There is a debate in the literature if queries should be considered documents (smaller documents in general) or not. In general, researchers are in favor of considering queries as documents (Harman, 1993; Olsen et al. 1993; Harman, 1994; Harman, 1995). Those opposing this view, generally state that queries are much smaller than normal documents, and they are not suited for complex semantic content analysis (Bollmann-Sdorra and Raghavan, 1993). The position taken in this thesis is to consider queries to be documents.

1.1.4 Relevance and Relevance Feedback

Relevance is a central concept in information retrieval and information science. As an informal definition, relevance emphasizes to what extent the retrieved documents are satisfying the information need. This is just a shallow definition. Mizzaro (1997) presents an exhaustive study about relevance. He identifies several kinds of relevance, based on different points of view. For example, relevance can be viewed from a user's point of view, from an information need's point of view, from a ranking algorithm's point of view etc. Mizzaro describes relevance as a relationship between *two entities of two groups*:

The first group contains

- Document: the actual physical representation.
- Surrogate: a smaller version of the document.
- Information: the actual knowledge the user gains by reading the document.

The second group contains

- Problem: that needs to be solved by user.
- Information need: a way of representing the problem under user's understanding.
- Request: the representation of information need in user's language.
- Query: the representation of information need in machine language.

The mappings between the two groups identify several types of relevance. These types are further expanded on a third dimension that contains:

- Topic: the area in which the user is interested.
- Task: emphasizes what will the user do with the retrieved documents.
- Context: anything other than topic and task that can affect the searching process and the results.

There is also a fourth very important dimension of relevance: time. It is possible for example that a document is not relevant to the user at the beginning of a search process, but becomes relevant towards the end, and vice-versa. In conclusion, relevance can be interpreted as a point in the 4-dimensional space previously described. This general description of relevance is depicted in Figure 1.2 (time dimension not depicted here). In the figure the lines connecting the objects represent kinds of relevance. The gray arrows emphasize how close a relevance is to the best relevance (relevant information received by the problem).

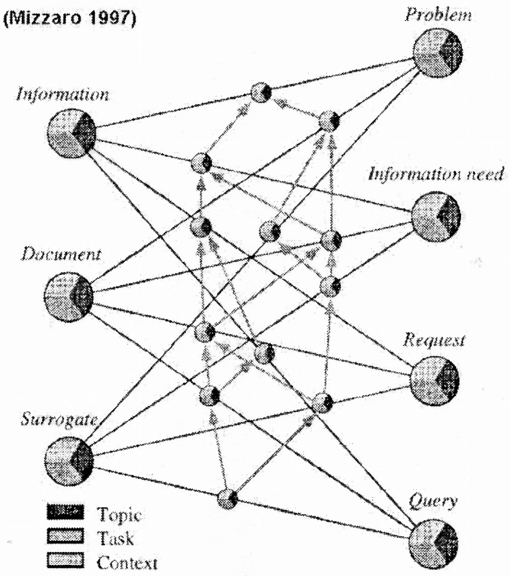


Figure 1.2 Kinds of Relevance.
Source: (Mizzaro, 1997)

Relevance Feedback (RF) consists of retrieved documents' relevance judgments, given by the user of the IR system. Relevance feedback is a judgment that describes to what degree a document's content satisfies the information need described in the query.

Few other definitions of RF gathered from the literature:

- "Relevance feedback is the user feeding back into the system decisions on relevance of retrieved documents" (Korfhage, 1997, p. 221).
- "Relevance feedback is an assessment of the actual (as opposed to predicted) relevance of a document to a query" (Bodoff, 2001, p. 785).
- A generic mathematical definition is given below (Bodoff, 2001, p. 787), where D-document, Q-query, R-relevance: $r : D \times Q \to R$

1.1.5 Information Retrieval Model

The model of an information retrieval system is given by the following aspects:

- What is the mathematical theory used (fuzzy, probabilistic, vector space)?
- How are documents and queries represented internally as data structures?
- Is relevance a discrete or continuous measure?
- How does the retrieval mechanism work?

In light of these criteria we can identify several well-known information retrieval models. They are the Boolean model, vector space model (VSM), probability model and fuzzy model. The current work is built on top of the vector space model, which is more explicitly described later, in Section 2.3.

1.1.6 Words, Terms and Concepts

A text document is composed of words. In IR contexts, words are also called terms. A concept is a term that semantically suggests the topic of a document. A concept term can be composed of one or more terms. For example, in a document talking about information retrieval, *information*, *retrieval*, and *information retrieval* are considered to be concepts because they suggest the semantic content of the document.

1.1.7 Query Expansion

Query expansion is a technique used in most retrieval systems. The initial query posed by the searcher is improved by adding additional terms. There are many algorithms for query expansion. The main problem is to identify the sources for finding new terms. Thesauri are good examples of new terms sources. Using a thesaurus, a retrieval system can add to the terms from the initial query, all their corresponding synonyms. Also, relevance feedback can be used to identify new terms, by selecting the most important ones from the documents assessed relevant to the initial query.

1.1.8 Document Representation Modification

A document representation is the data structure used by a retrieval system, to internally represent a document. In short, document representations are also called indexes. In Figure 1.1 for example, "Data" is the document representation. Document representation modification is the process of modifying this data structure (i.e. the values in the data structure). It is the analog process to query expansion, but for documents.

1.2 Motivation

The automatic generation of document representations is called automatic indexing. Automatic indexing takes a document collection as input, and generates a document representation space (e.g. a set of document representations) as output, using techniques such as stopped word removal and stemming, etc.

Traditional automatic indexing algorithms use term frequency (TF) to derive the importance of a word/index term in a document. These are called frequency-based algorithms. The term frequency of a term with respect to a document is given by the number of occurrences of the term throughout the document. The usage of the term frequency measure is based on the assumption that the more occurrences a term has in a document, the more important it is in that document (Salton, 1971). In general, to derive the importance of a term in a document, the TF of the term is weighted using another frequency-based measure called Inverse Document Frequency (IDF). The IDF of a term is a measure that depends logarithmically on the fraction of documents in the collection containing the term. In other words, if a term from a document occurs in many other documents from the collection, it will be assigned a lower score because it is considered to be a poor identifier. The formula for IDF is given by: $IDF_k=[\log_2 N-\log_2 d_k+1]$ (Korfhage, 1997, p. 116). In the previous formula N is the number of documents in the collection and d_k is the number of documents containing the term k. Using TF and IDF together, the most standard formula for the importance weighing of a term in a document is given by $w_{ik}=f_{ik}*[\log_2 N-\log_2 d_k+1]$ (Korfhage, 1997, p. 116).

Nevertheless, traditional automatic indexing algorithms are not perfect. Some of the inherent problems associated with these algorithms are: (a) poor quality: most index terms are not semantically related to the topic of the document from which they were selected; (b) static nature: resulted indexes are static structures. Once generated, they cannot be further modified in response to external environment changes and inputs (e.g. terminology change, relevance assessments); (c) size: the size of the generated indexes is usually very large, because of the great amount of non content bearing index terms; (d) word mismatch: many indexes suffer from the word mismatch or the synonymy syndrome. For example a document talking about “laptop” will never be retrieved in response to a query containing only the term “notebook,” because the index does not contain the latter term, which in fact is a synonym of “laptop.”

All these problems, constituted the motivation to design and evaluate a document representation improvement algorithm, namely the Relevance Feedback Accumulation (RFA) algorithm.

CHAPTER 2

BACKGROUND

This chapter is a literature review. It presents the background material used throughout the rest of the thesis. Section 2.1 presents the conceptual model of a retrieval system. Section 2.2 lists a classification of Information Retrieval Systems. Section 2.3 describes the well-known vector space model, used as the basis for developing the information retrieval system proposed by this thesis.

2.1 A Conceptual Model for Retrieval Systems

Fuhr and Buckley's conceptual model of an Information Retrieval system is the reference model for this thesis proposal (see Figure 2.1). This model represents a more explicit conceptualization of a retrieval system, than the one formerly presented in Section 1.1.2 (see Figure 1.1).

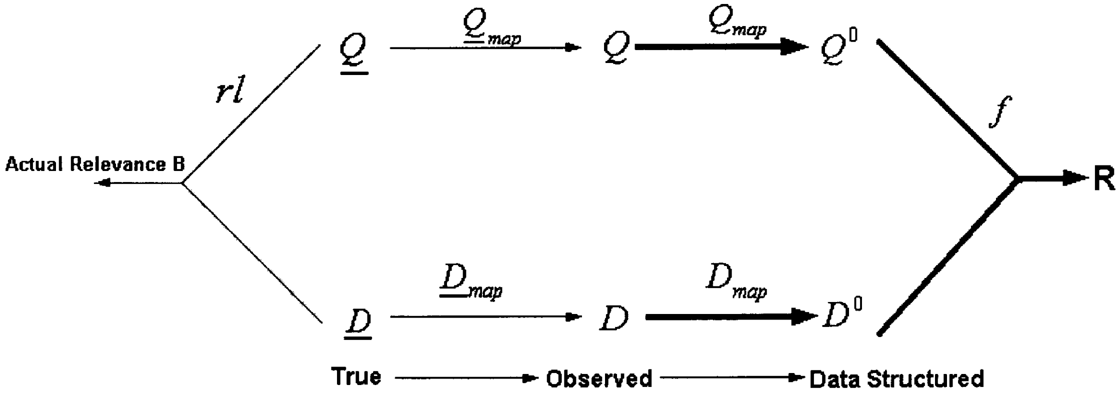


Figure 2.1 Fuhr and Buckley's IR Conceptual Model.
Source: (Fuhr and Buckley, 1991)

Logically, the model flows from left to right, from true to data structured, respectively. The left-hand side depicts the searcher's point of view. This is indicated in Figure 2.1 by the **true** side. The middle part, indicated by **observed**, refers to the communication protocols between a searcher and a retrieval system. The right-hand side, or the **data-structured** side, represents the system's view.

The main concepts on the left-hand side are:

- $\underline{Q}$ - the **real query**: represents the information need as it exists in the searcher's mind. It represents the information that the searcher thinks he/she needs, to solve his/her problem.
- $\underline{D}$ - the **real document**: represents the searcher's understanding of the content of a document, as it exists in his/her mind.
- The **actual relevance**: this is a measure of how relevant $\underline{D}$ is to $\underline{Q}$ in the searcher's understanding. It is also called **target relevance**. It is important to mention that relevance judgments made by the user are relative to his/her mental understanding of the queries and the documents at hand. When such an assessment made by a user is fed back into the system, the assessment is called **relevance feedback**. **rl** is Boolean, and is indicated as **B** in Figure 2.1. This means that usually, searchers say a document is either relevant or not relevant, to a particular query. The mathematical definition of **rl** is given by: $rl : \underline{Q} \times \underline{D} \rightarrow B$

The main concepts on the central buffer area of the model are:

- Q – observable query representation: represents how the searcher articulates in words (or any other query language) his/her information need from $\underline{Q}$.
- D – observable document representation: similar to Q but for documents.

The main concepts on the right-hand side of the model are:

- Q_0 - the internal system representation of the query: represents how does the IR system internally represent the query as a data structure.
- D_0 - the internal system representation of the document: represents how does the system internally represents a document as a data structure.

- R - the **predicted relevance**: the extent to which a document is judged relevant to a query by the IR system.

- f – the matching function: The matching function, also called the output ranking-function, is usually a real-valued function that evaluates the extent to which a document is relevant to a query. This is the **predicted relevance**. The matching function **f**, is algorithmically implemented by the IR system. It can be observed that f is defined as $f : Q_0 \times D_0 \to R$.

On the same model one can observe mappings between concepts on different sides.

The main mappings and their explanations are:

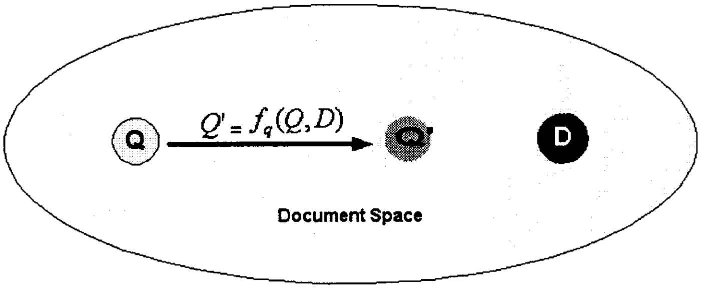
- From **true** to **observable**: $\underline{Q}_{map}$ and $\underline{D}_{map}$ map internal searchers metaphors for queries and documents in observable representations.

- From **observable** to **data-structured**: Q_{map} and D_{map} map observable representations of queries and documents into internal system representations.

From this model it follows that implementing an IR system consists of the definition of the three mappings: Q_{map}, D_{map} and f (Bodoff et al. 2001). Finding "optimal" mappings for an IR system implementation usually consists of two phases: **(a)** ***design phase***: some parameterized versions of Q_{map}, D_{map} and f are defined; and **(b)** ***tuning (training phase)***: the parameters that characterize Q_{map}, D_{map} and f are tuned by repeated runs on document test collections, using relevance feedback. In IR this process is called **parameter estimation**. There are two approaches to build an instance of the conceptual model from Figure 2.1. The **first** approach is to estimate parameters characteristic to Q_{map}, D_{map} and f. A good example is when one of these functions depends on an unknown parameter k, which is iteratively optimized/tuned using relevance feedback. The disadvantage in this case is that the estimated values for the parameters are just averages. The advantage is that only a small amount of feedback for all queries and/or documents is needed to optimize the parameters. The **second** approach is to individually estimate query and document representations using the relevance feedback specific only to that document or query. In this situation, the query and the document representations are considered to be the estimated parameters. The advantage is that their representations are better than averages. A good example is: having a query Q, and a document D judged as being relevant to Q, then Q's/D's representation can be modified according to the following generic formula(s) (Salton, 1989; Brauen, 1971):

$$Q' = f_q(Q,D) \text{ and } D' = f_d(Q,D) .$$

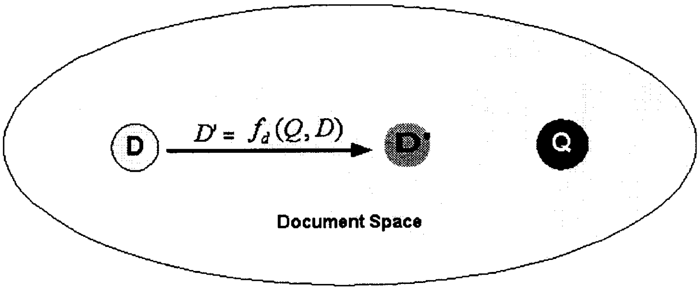
The first case when Q is modified into Q' is called the ***query-oriented view***. Conceptually, queries in the document space are moved closer to their relevant documents. Document representations are fixed. See Figure 2.2.



Title: Query-oriented View
Author: Razvan Stefan BOT

Figure 2.2 Query Oriented View.

The second case is called the ***document-oriented view***. Here, queries are fixed while relevant documents are moved closer to them (Figure 2.3). The hybrid case consists of combining both query and document oriented views. Such an approach is presented in (Bodoff et al. 2001).



Title: Documet-oriented View
Author: Razvan Stefan BOT

Figure 2.3 Document Oriented View.

In both cases the shifting operation is accomplished based on relevance judgment assessments. Most research focuses on the query-oriented approach. The disadvantage of the query-oriented approach is that it lacks improving document representation over time and across users, given their relevance judgments.

This thesis focuses on the document-oriented view. The algorithm proposed hereby is testing a technique that modifies the documents' representations by accumulating and analyzing relevance feedback assessments from searchers.

2.2 IR Systems Classification

This section presents a classification of retrieval systems based on their parameter estimation type, defined in Section 2.1. This classification allows the identification of the parameter estimation type for the algorithm described throughout this document.

From the point of view of the parameter estimation type, one can have:

- Retrieval systems built by estimating parameterized versions of Q_{map}, D_{map} and f (see Figure 2.1). There can be two estimation types: **exhaustive** and **heuristic** (Bartell et al. 1998). The **exhaustive methods** search the entire parameter space and are guaranteed to find the optimal solution. There are several problems with these methods: **first,** if the number of parameters is big the computational complexity is too large; and **second,** if the parameter space is not discrete it must be reduced by sampling. Actually when sampling the parameter space, the solution will not be guaranteed to be the optimal one. Algorithms that fall under this category are usually adaptive learning algorithms like ID3 or Simulated Annealing. Description of such algorithms can be found in (Chen, 1998). The **heuristic methods** search only a subspace of the parameter space. These methods focus on improving system performance. Very well known examples are genetic algorithms, hill-climbing algorithms or random start hill climbing algorithms. One disadvantage of heuristic methods is that they are generally not guaranteed to find the optimal solution.

- Retrieval systems built by estimating queries Q_0 and documents D_0 themselves. This type is also called **implicit parameter estimation** (Bartell et al. 1998), because the goal of the technique is not explicitly to improve retrieval effectiveness. The goal of this technique is to improve document representations. In general, the better the document representation, the better the retrieval effectiveness.

A comprehensive view of the classification, from the parameter estimation type point of view, is depicted in Figure 2.4.

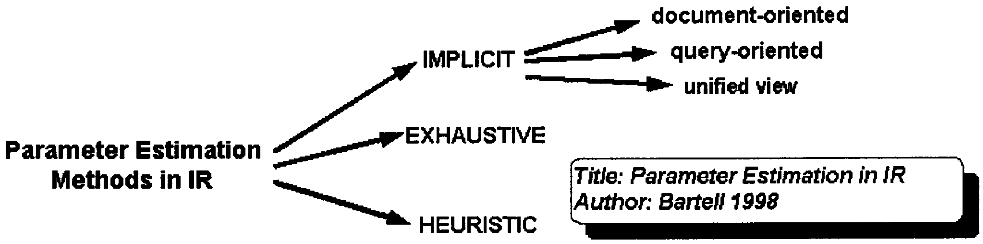


Figure 2.4 Parameter Estimation in IR.
Source: (Bartell, 1998)

The retrieval system proposed in this thesis falls under the ***implicit*** ➔ ***document-oriented*** category. Document representations are first estimated, and then improved over time and across users, using the Relevance Feedback Accumulation algorithm.

2.3 The Vector Space Model (VSM)

The Vector Space Model (VSM) serves as the foundation model for the system proposed by this thesis. The general characteristic of this model is that documents and queries are treated alike. They are represented as multi dimensional vectors, organized in an N-dimensional space, called the **document space**. N is the number of distinct terms (words) that characterizes the whole document collection. If document D contains term T_i ($1 \le i \le N$), then, in the vector representation of document D, we will have 1 or other frequency-based weight at position i (Korfhage 1997, p. 125). Usually, the vectors contain frequency-based weights (rather than binary values) that are normalized to the document length and/or to the whole document collection size. A **frequency-based** term weight represents an absolute or relative measure of a term's occurrence within a document and/or collection.

Given that there is no differentiation between documents and queries, similarity measures are used to assess relevance. Based on the similarity measures it is possible to have a ranked output. From this point on, the following notations are adopted: Q for the query vector, and D for the document vector. The ranking function takes two objects as arguments: the query, which is also a document, and a document from the collection. The result is a measure of how similar the two are. The more similar a document to a query, the more relevant it is considered by the system. Documents that are more similar to the query will be ranked higher. Usually the similarity measures are normalized, having values between [0,1]. Such a function is defined as:

$$0 \le sim(Q, D) \le 1$$

Some basic similarity metrics identified in the literature are:

(a) Distance-based measures: evaluate how close two documents are in the document space. A document D is represented in the vector model as:

D(term_1, term_2, … , term_N)

In the representation above, **term_i** would be 1 if the term exists in the document or 0 otherwise. It is possible to have the raw count of that term within the document rather than the value 1. It is important to mention here that the actual distance between two documents (more specific distance between two n-dimensional vectors) is actually a dissimilarity measure. The bigger the distance the more dissimilar the documents are. So, the similarity of two documents is inversely proportional to the distance between them. The most widely used distance based metrics are L_p metrics (Korfhage, 1997, p. 132):

- General L_p formula: $L_p(D_1,D_2) = \left[\sum_i^N |d_{1i} - d_{2i}|^p \right]^{1/p}$
  - o N – vector dimensionality
  - o d_{ij} – the value of the document's **i** vector position **j**
  - o p – parameter (see below)
- Manhattan Distance (also called City-block) for p=1
- Euclidean Distance for p=2
- Maximal Direction Distance when p= ∞

(b) Angle-based measures: are also called cosine similarity measures. Two documents are considered similar if they are situated along the same direction in the document space, starting from the origin. It is a quite different view from the distance-based metrics. It is possible to have documents that are similar when using a cosine metric, and dissimilar if using a distance metric instead, and vice versa.

The generic formula for a cosine measure is (Korfhage, 1997, p. 84):

- Generic formula: $\sigma(D,Q) = \dfrac{\sum_{k}^{N}(d_k \times q_k)}{\sqrt{\sum_{k}^{N}(d_k)^2} \times \sqrt{\sum_{k}^{N}(d_k)^2}}$
    - D – Document
    - Q – Query
    - d_k – value of term k in document D
    - q_k – value of term k in query Q

(c) Inner product: represents a similarity measure given by the inner product between the query and the document vectors. The generic formula is given below:

$$\sigma(D,Q) = \sum_{k}^{N}(d_k \times q_k)$$

The meaning of the operands is the same as above (b). Usually the document contains many more terms than the query. In this case, only the terms that appear in both the document and the query are taken into consideration. The inner product is a part of the cosine-similarity measure.

CHAPTER 3

RF IN QUERY ORIENTED VIEW VS. DOCUMENT ORIENTED VIEW

This chapter presents a comparison of the query-oriented view and the document oriented view. As a reminder, both the query oriented view and the document oriented view are implicit parameter estimation techniques. As mentioned in Section 2.2 their explicit goal is to improve query and document representations. By improving the query representation, the retrieval system helps the searcher to better express an information need. The mechanism is called query expansion (see Section 1.1.7). The mechanism for document representation improvement is called document representation modification (see Section 1.1.8).

Techniques pertaining to both views use relevance feedback as their primary source for improvement. Query expansion uses relevance feedback to identify new terms to be added to the initial query. Document representation modification uses relevance feedback to identify what terms to modify in a document's representation.

Sections 3.1 and 3.2 discuss in more detail, with examples from literature, the use of relevance feedback in the query/document-oriented views. Section 3.3 presents a comparison of the two views emphasizing their advantages/disadvantages.

3.1 The Query Oriented View (QOV)

Most of the work with relevance feedback found in the literature addresses this view. The relevance feedback is used to modify the initial query and to improve retrieval performance. Conceptually, in the document space, documents are fixed and queries are moved towards them (see Figure 2.2) (Salton and Buckley, 1990). The rest of this section presents a chronologically ordered list of several systems that address this view.

Rocchio Jr. (1971) was one of the first to develop a process of improving the user information needs formulation (query). This is probably the most acclaimed and most cited query modification algorithm. Consider Q_0 as the initial query posed by the user. Denote S_{RF+} as the set of documents assessed relevant by the user and S_{RF-} as the set of documents assessed as non-relevant by the user. The modification function should look like: $Q_1 = f(Q_0, S_{RF+}, S_{RF-})$, where Q_1 is the first modified query version. Given that n_1 represents the cardinality of S_{RF+} and n_2 represents the cardinality of S_{RF-}, the recommended modification function f is:

$$Q_{i+1} = Q_i + \frac{1}{n_1}\sum_{j=1}^{n_1} S_{RF+}^{j} - \frac{1}{n_2}\sum_{j=1}^{n_2} S_{RF-}^{j}$$

Salton (1971b) develops another query modification formula which is a variation of the one presented above. The variation is given by the normalization of the vector weights by the length of the vectors.

$$Q_{i+1} = Q_i + \frac{1}{n_1}\sum_{j=1}^{n_1} \frac{S_{RF+}^{j}}{\left|S_{RF+}^{j}\right|} - \frac{1}{n_2}\sum_{j=1}^{n_2} \frac{S_{RF-}^{j}}{\left|S_{RF-}^{j}\right|}$$

Based on the formula above, the query modification function is parameterized as follows:

$$Q_{i+1} = \alpha Q_i + \beta Q_0 + \gamma \sum_{j=1}^{n_1} c_i S_{RF+}^{j} + \delta \sum_{j=1}^{n_2} c_i S_{RF-}^{j}$$

In the equation above, c_i is either set to 1 for all documents or is a magnitude of the correlation between document S_{RF+}^{j} (or S_{RF-}^{j}) and initial query Q_0.

Salton et al. (1985) provides a query expansion mechanism using relevance feedback for the extended Boolean model described in (Salton et al. 1983). The basic idea is to represent queries in DNF (Disjunctive Normal Form – which consists of a disjunction of conjunctions), and to find new terms from relevance assessments to be added to this query. By terms the author means individual words or conjunction of two or three words (like term_{k1} AND term $_{k2}$ AND term $_{k3}$). Other similar work was done by (Dillon and Desper, 1980; Salton et al. 1984).

Allan et al. (1995) presents an overview of TREC experiments with query processing. The authors believe that better queries can be achieved by structure and multiple sources of evidence. TREC collections contain topics, documents and relevance assessments in the topicXdocument space. A topic is a more structured and extensive version of a query and much lengthier at the same time. The authors developed a system called InFinder that generates potential queries from these topics. The generated queries were then submitted for automatic query expansion. Here, the query expansion is using multiple evidence sources: at document level and at passage level.

Buckley et al. (1995) present an interesting query expansion approach. The number of terms that are added to the query is about 300. This methodology is also called massive query expansion. The explanation is that bad-terms (non-representative for a document) randomly co-occur within the whole collection while good-terns tend to co-occur constantly (or at least non-randomly). As the authors state, this massive query expansion creates a noise similarity, accounted for by the bad terms. At the same time good documents escape this noise similarity by containing good terms. Good terms co-occur non-randomly. As a result massive query expansion was most effective for routing experiments, but it was also effective in general for ad-hoc experiments. In the TREC routing experiments queries are formed in two phases: (a) concepts that occur many times in relevant documents are added to the query vocabulary; and (b) all the concepts in the final query vocabulary are weighted according to their frequency of occurrence in relevant and non-relevant documents (Buckley et al. 1994). As for the TREC ad-hoc experiments, the expansion and weighting of query-terms is done by parsing and analyzing the first top-ranked documents, without actually knowing which are relevant or not.

Mitra et al. (1998) focus their attention on search engines on the WWW (e.g. large collections). Users are usually posting queries that contain only a few terms, and they might omit important ones. The proposed idea tries to offer a competitive alternative/improvement to blind or ad-hoc feedback. In this case there is no need for user intervention to state which documents from the retrieved set are relevant. The system automatically considers the first few retrieved documents as relevant and analyzes them to identify new terms. The main flaw of this approach is when a large fraction of the documents assumed relevant, is not relevant. In this case many of the words added to the query are likely to be not related to the topic. The main idea of this research effort is to eliminate as much as possible the non-relevant documents from the top positions in order to avoid the drift of the query after expansion. A query is said to be drifting if after expansion more non-relevant documents are promoted to being top ranked.

The steps of the algorithm are:

- The initial set of retrieved documents is checked for additional relevance indicators. Some examples are: 1) Boolean and/or fuzzy filters (to check if they cover all topics required by the query - when the query asks for different topics) and 2) proximity constraints relative to a window of adjustable size. By proximity constraint, it is meant how far two terms can be from one another when co-occurring within a document.

- The documents are assigned new scores based on the new identified relevance indicators from the previous step.

- The documents are re-ranked according to the new scores.

- Expand the query using a few top ranked documents using an ad-hoc query expansion methodology (Rocchio in this case).

The evaluation tested the performance of manually created Boolean filters and automatically created Boolean filters. The automatic process was based on co-term occurrence analysis. A Boolean filter represents the topics formulated in the query using a CNF (Conjunctive Normal Form) for each query. The results showed significant increase in retrieval effectiveness as follows: between 7% and 22% for manually created Boolean filters and between 6% and 13% for automatically created Boolean filters.

Gauch et al. (1999) starts from the known fact that: the average number of query terms that users usually submit (in WWW context) is two. Adding other words to these short queries is a way to perform conceptual retrieval. The only problem is how to find the related terms to add to the initial query.

The authors describe three sources of new terms:

- Query specific: these terms are extracted from documents judged relevant by users.
- Corpus specific: these are terms identified by analyzing the whole document collection. The terms that are used in similar fashion, are considered conceptually close, and are selected. One well-known technique is Latent Semantic Indexing (LSI), where all documents are represented in a conceptual lower dimensional space. The corpus analysis is based on term co-occurrence patterns. These techniques though database specific, are tied to the specific terminology. One good example provided by the author is that the Congressional Record uses the term "*senior citizen*" as standard but a query might contain the synonym *elder*.
- Language specific: these terms are usually extracted from thesauri (e.g. Webster). The main disadvantage of these methods is that there are no comprehensive thesauri to cover every topic.

In this article, the authors use a corpus-based technique where new terms are identified using a co-occurrence analysis for the whole collection. The co-occurrence is evaluated based on a co-occurrence window (the maximum distance of a two term co-occurrence). First, the system identifies a set of terms (the most important), called target words, based on frequency measures. Afterwards, the system constructs a context vector for each such term. A context vector contains all term co-occurrences for a specific target word given a window of preset size. Based on these data structures, the system derives as a final step, a similarity matrix that contains for each word the most similar other words, where the similarity is above a certain threshold. As a result the methodology was able to achieve a 7.6% improvement for TREC-5 queries and a much better result of 28.5% improvement for domain specific database queries (namely the Cystic Fibrosis collection). The methodology was extended afterwards to multiple data collections, each of them having its own similarity matrix. Some techniques for selecting the right similarity matrix accounted for improvements of 4.8%.

Kekalainen and Jarvelin (2000) emphasize in their article that conceptual query expansion produces better results than a simple non-conceptual one. The whole query expansion mechanism is tested using InQuery. The context is a probabilistic IR Model. Several operators, such as SYNONYM for example, emphasized the conceptual relations, between terms in the query and other terms. The evaluation considered several levels of expansion: (a) no expansion (b) expansion based on synonyms from the thesaurus (c) narrowing the concept (d) associative concept and (e) all previously described expansion levels. The results showed that the best performance was achieved by using synonymy expansion. This result is somehow expected, but one disadvantage is the lack of comprehensive thesauri to describe every concept and relation from all domains. In this case, the authors developed their own thesaurus just for this experiment.

Chen et al. (2001) developed FEATURES, a system that is able to adaptively learn in real time from relevance feedback, improve retrieval performance, and speed up the process. The system is specifically designed for web searches. It addresses efficiency problems too. The authors set a threshold of 20% in retrieval performance increase after each feedback round from the user, and state that if such a threshold is met the users are willing to give several rounds of feedback. Considering q, the query posted by the user, the system builds the following sets D(q), F(q) and V(q). D(q) contains all the matches for query q that are retrieved from the internal database if any, or from AltaVista if none are in the internal database. F(q) contains all the keywords (also called indexing features) that are used to index documents from D(q). Each document is indexed in the internal database with at most 300 features. The documents retrieved from meta-search (AltaVista) are indexed by at most 64 automatically generated keywords. F(q) is also called the set of dynamic features. V(q) is the dynamic vector space consisting of documents from D(q), and they are indexed using terms from F(q). The system uses two learning algorithms afterwards: feature learning (FEX) and document learning. FEX starts with a small number of generated keywords to index documents and then uses term relevance feedback to refine this set by promoting and demoting features from F(q). The document-learning algorithm promotes all dynamic features of a document assessed relevant by the user. In this way it automatically promotes the document itself. At the end of the interactive feedback session the documents are ranked according to the learned weights and displayed for the user. Using a relative recall and precision as measures, the system outperformed AltaVista in the evaluation runs.

Evaluations of the query-oriented view emphasized its potential on improving retrieval performance (Efthimiadis, 2000). Spink and Saracevic (1997) also identified the efficiency of relevance feedback items to be high. It is meaningless to talk about evaluations of the query-oriented mechanisms since most of the commercial search engines are using them as state of the art.

3.2 The Document Oriented View (DOV)

Even if early results of document-oriented view techniques, obtained by pioneers like Brauen (1968), Brauen (1969) or Ide (1969), revealed high potential value, very little effort was spent by researchers afterwards (Bodoff, 2001; Kemp and Ramamohanarao, 2002). One explanation for this situation would be the fact that all available test collections are not suitable for document-oriented techniques (Bodoff, 2001). That is because not enough relevance data points are available for each document, and it is very hard to attempt collecting this data, using an experimental setting (Salton, 1989, p. 326). This section presents a compilation of the most important research falling under the document-oriented view umbrella. The techniques under this umbrella focus on the modification of the document representation based on relevance feedback assessments. Conceptually, this can be pictured as moving documents towards queries to which they are the most similar (see Figure 2.3).

The advantages of accumulating relevance feedback as permanent changes to document representation are (Salton, 1971):

- It allows the return of similar documents to a set of similar queries that are posted by different users.
- The document representation can be practically updated in response to vocabulary/terminology changes.
- The concept weights are allowed to fluctuate (Salton, 1971); when new documents are introduced in the database. They can start with an initial document representation (term weights) that will change after relevance feedback assessments.

The rest of this section lists the few research efforts focused on document-oriented techniques.

Friedman et al. (1967) describe an algorithm that creates and maintains a dynamic document space. The alteration of the document terms (or concepts) is based on their discrimination power. A good positive discriminator is characterized by the fact that it is strong in relevant documents and weak in non-relevant documents. The strength or weakness of a term is judged in terms of frequency-based importance measures. The algorithm uses prior knowledge in order to separate relevant and non-relevant documents. The prior knowledge is built-up from the relevance assessments history. The evaluation of the algorithm revealed better results when compared to a query expansion IR system.

The drawbacks of the algorithm are: (1) each time a search process occurs all the documents in the collection are modified. This is unrealistic and inefficient; (2) after the search process is over the modifications are dropped. In this case modifications are not permanent. This, again, denotes a waste of resources; (3) the evaluation was done on a very small collection.

Brauen et al. (1968) proposes an algorithm in which document vectors are modified for all relevant documents to a query. The modification is done in two steps:

STEP 1: The query vector is normalized against the document vector, in order to ensure that the length of the document vector remains unchanged after weight adjustment. The "length of query/document vector" is defined as the sum of term weights corresponding to all terms included in the query/document. The normalization is done according to the following formula:

$$q_0^{norm} = q_0 * \left(\frac{Ad_n}{Aq_0} \right)$$

In this formula q_0 represents the original query vector (weights), q_0^{norm} represents the normalized query vector, Ad_n represents the length of the document vector d_n, while Aq_0 represents the length of the query vector q_0.

STEP 2: the weights of concepts from document vectors are adjusted according to the formula below:

$$d_n^{new} = d_n + \alpha(q_0^{norm} - d_n)$$

The evaluation test was conducted using the Cranfield collection. After several iterations with different values for α, a value of 0.2 was found "optimal." For this value the methodology revealed significantly higher normalized recall and precision levels.

Critique points regarding this algorithm are: (1) the modification of document vectors was accomplished in a one-by-one sequential manner. If several documents were relevant to a query Q, each of the documents was modified according only to Q's content. It might be possible though that Q contains accidental terms whose weights do not deserve to be increased; (2) there is no classification scheme for terms that were modified. From a (Q, D) relevance assessment point of view terms can fall under three categories: (a) terms that occur only in Q, (b) terms that occur only in D and (c) terms that occur in both Q and D. They deserve different modification schemes.

Ide (1969) proposes a quite similar method to Brauen's et al. formerly described. The main difference is the fact that Ide's algorithm also modifies weights in non-relevant documents in order to move them farther from queries to which the documents were irrelevant. For this operation, only the high-ranking irrelevant documents were considered. The formulas for the positive/negative document vector modification phase are:

$$d_n^{new} = d_n \pm \alpha(q_0^{norm} - d_n)$$

The methodology was tested on the Cranfield collection and the results were similar to Brauen's. The levels of normalized recall and precision were significantly improved around the same value of 0.2 for α. Another problem, other than those already expressed for (Brauen et al. 1968), is the fact that altering terms in non-relevant document vectors does not represent a justified extra effort (Brauen, 1969).

Brauen (1969) presents an algorithm that fixes some of the problems formerly emphasized. He argues in favor of an extended vector modification scheme according to the following terms/concepts classification:

- Terms/concepts present only in the query vector
- Terms/concepts present only in the document vector
- Terms/concepts present in both query and document vector

Brauen proposed two new strategies. These methodologies showed slightly better performance than the previous two methodologies (Brauen et al. 1968; Ide, 1969) when compared for an α of 0.2.

Strategy I: The vector modification is done according to the following scheme:

- Case 1: If a concept i is present in the query q_0 but is not present in the relevant document d_n. Then: $d_n^{new}(i) = \beta$, where β is a parameter to be estimated. $d_n(i)$ stands for the ith term/concept of d_n.
- Case 2: Concept i is present in both q_0 and d_n. Then: $d_n^{new}(i) = d_n(i) + \gamma[120 - d_n(i)]$, where γ is a parameter to be estimated.
- Case 3: Concept i is present in relevant document d_n but is not present in the initial query q_0. Then: $d_n^{new}(i) = d_n(i) - \left(\frac{d_n(i)}{\delta} + 1\right)$, where γ is a parameter to be estimated.

This strategy showed significant improvement in the retrieval performance, while taking into consideration issues like synonyms or adaptive document representation. The author remark on the evaluation measurements was that they were somehow specific to the Cranfield collection. The collection contains short abstracts as documents. Still he expected that the results would be scalable.

Strategy II: the second strategy incorporated non-relevant document vector modification, as the next logical improvement. First, for all documents that the user considered as relevant, the positive modification takes place following exactly the same scheme as under Strategy I. Second, for all the non-relevant documents that were returned as a result of the initial query a negative modification is performed on the document representation. In this system, as opposed to Ide's system, the negative modified documents are only those corresponding to the initial query. The argument is that the intermediate non-relevant documents, corresponding to the expanded versions of the initial query q_0, are not suited to be modified with respect to q_0. The negative vector modification is done according to the following scheme:

- Case 1: If concept i is present in the initial query q_0 but is not present in the non-relevant document d_n, no modification is performed.
- Case 2: If concept i appears in both initial query q_0 and the non-relevant document d_n, then its corresponding weight is modified by the following formula: $d_n^{new}(i) = d_n(i) - \left(\frac{d_n(i)}{\theta} + 1 \right)$, where θ is a parameter to be estimated.
- Case 3: If the concept i is present in the non-relevant document d_n but is not present in the query q_0, its weight is modified as follows: $d_n^{new}(i) = d_n(i) + \varepsilon[120 - d_n(i)]$, where ε is a parameter to be estimated.

The result of the evaluation of this second strategy is that it also improved retrieval performance, but it did not improve overall performance compared to Strategy I. The immediate conclusion is that modifying non-relevant document vectors does not improve retrieval performance. At the same time it is more expensive.

Parker (1983) proposes a solution similar in essence to Brauen's previously described techniques. The idea is framed under the title of "document learning." In the author's formulation, document learning represents a process having queries, documents, and a set of relevance assessments as inputs. The output of this process is an altered document space. The alteration procedure is designed along the following rules: (a) if a term does not appear in the query, its weight is not affected in the document representation; (b) if the term appears in the query but not in the document and the document is not relevant to the query, then its weight is not affected; (c) if the term appears in the query but not in the document and the document is relevant to the query, then its weight is modified; and (d) if the term exists in both the query and the document the weight is increased/decreased if the document is relevant/not relevant to the query.

Belew (1989) presents a mechanism that uses connectionist networks (e.g. neural networks) to learn document representations and associations. The neural network generates nodes for documents, index terms and authors. These nodes are interconnected by weighted links. The weight of a link emphasizes the strength of association between two nodes. The weights are constantly adjusted using back propagation algorithms. The signal to adjust the weights is obtained from relevance feedback assessments. The system is one of the few to generate associative rules of the type term1 → term 2. For example the system is able to emphasize the association between "adaptive" and "adaptation" without using any generalization techniques like stemming. No detailed evaluation is presented in the article.

Kanazawa (1999) presents a technique that alters the documents' representations using relevance information from the documents themselves. The model is called Relevance-based Superimposition Model. Similar documents in the database are clustered together. A document might be part of several clusters. Then from within each cluster the system automatically generates a feature vector called the Representative Vector (RV). The features selected to build the RVs are taken from titles, abstracts and keywords given by author. The last step is the modification of the original document feature vectors using the RVs of the clusters to which the document belongs. Given the original document vector d=(d1, …, dn) and the feature vector of the cluster set to which d belongs s=(s1, …, sn), the modified original document feature vector is given by:

$$d'=(\max(d1,s1), \ldots , \max(dn,sn))$$

Figure 3.1 shows the process flow of the algorithm (Kanazawa, 1999). This algorithm does not actually use a priori relevance feedback. It looks for modification information indicators by analyzing the document collection. One weakness is the fact that it has to be repeated when new documents are available. Also, it does not work for documents that do not have abstracts or keywords. Without abstracts or keywords, there is nothing from which to extract features to build RVs.

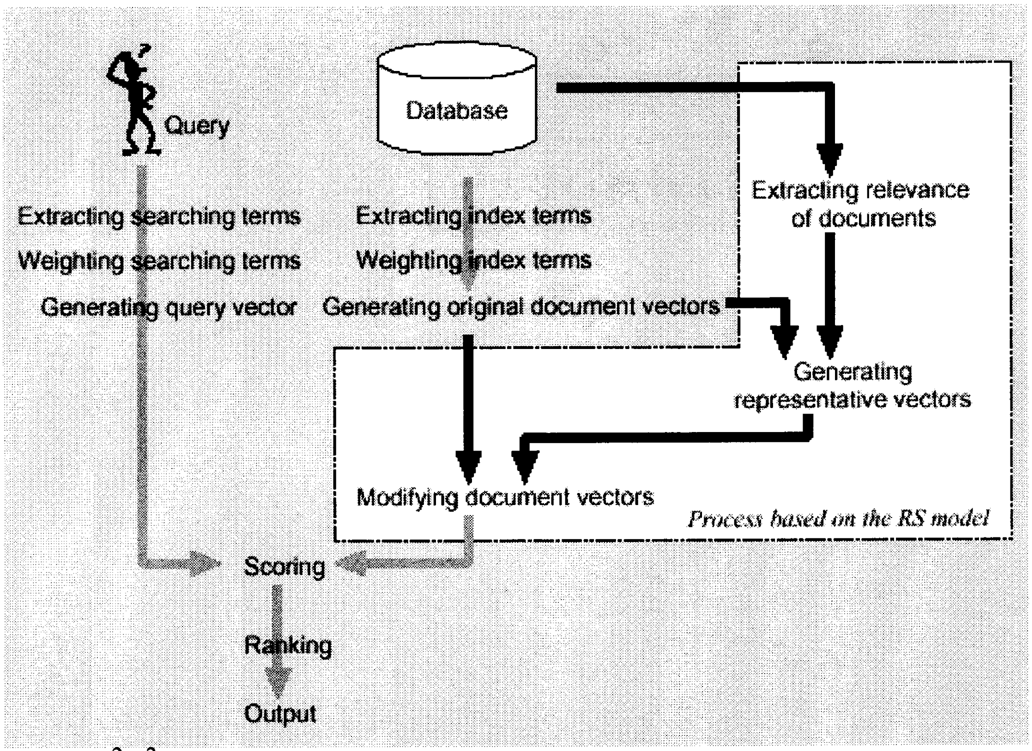

Query
Database
Extracting searching terms
Weighting searching terms
Generating query vector
Extracting index terms
Weighting index terms
Generating original document vectors
Extracting relevance of documents
Generating representative vectors
Modifying document vectors
Process based on the RS model
Scoring
Ranking
Output

Figure 3.1 R^{2}D^2 Process Flow.
Source: (Kanazawa, 1999)

Bodoff et al. (2001) present a hybrid approach called the unified view. It has the characteristics of both the query-oriented view and the document-oriented view. It uses a maximum likelihood approach in order to estimate both the query and document representations. Both document information and a priori relevance feedback are considered. The method suffers from several drawbacks: (1) too much modeling, drawback emphasized by the author(s) themselves, (2) the cost for parameter estimation is high, and (3) re-estimation of parameters is necessary from time to time.

Piwowarski (2000) presents a probabilistic document-centered method. The method is based on the probability of the anti-document of D. The anti-document of D is defined as the representation of the whole document collection, with document D removed. The probabilistic computations try to answer the following question: if document D is removed from the database, does the probability to find a relevant document increase or decrease? It is then straightforward to note that if this probability increases, document D can be doomed as probably irrelevant. The evaluation was performed on the Cranfield and CISI collections. The Cranfield collection consists of 1398 documents, 225 queries and 1837 relevance feedback judgments. The CISI collection consists of 1460 documents, 112 queries and 3114 relevance feedback judgments. The size of these collections is relatively small though, and their topics are domain specific: aeronautical topics for the Cranfield collection and information science for the CISI collection. The results on the Cranfield collection revealed precision improvements at all four recall levels that were tested (0.1, 0.4, 0.7 and 1.0). The results on the CISI collection revealed precision improvements at all the 0.4 and 0.7 recall levels and precision loss at the 0.1 and 1.0 recall levels.

Kemp and Ramamohanarao (2002) present a document transformation algorithm for web search engines, based on relevance feedback obtained from web logs. The test collection was built using web pages from the University of Melbourne's website. The size of the final test collection was of about 6644 web pages. The user click-through history and queries were obtained from the web server logs. The document representation modification strategy accounted for document vectors saturation. In other words, the strategy did not allow for the document vectors to grow without a limit. To avoid this effect, each document vector was composed of two parts: the original document vector and the learned document vector. The original document vector was left unchanged, while the learned document vector started as a zero vector, and it was then built up until certain limit was reached. The limit was adjustable.

The actual document transformation function was the following:

$$D = D_{original} + D_{learned}, \text{ where}$$
$$D_{learned} = \begin{cases} D_{learned} + \beta Q & \text{if } |D_{learned}| < l \\ (1-\alpha)D_{learned} + \alpha\frac{|D_{learned}|}{|Q|}Q & \text{otherwise} \end{cases}$$

In this formula D is the total document vector, $D_{original}$ is the original document vector, $D_{learned}$ is the learned part of the total document vector, Q is the query vector, |D| is the [1-norm_of_D] (norm_of_D is the sum of all weights in D), and l is the boundary parameter that limits the infinite growth of the learned part. Alpha and beta are adjustable learning parameters. Using precision(10) as the evaluation metric, the system augmented with the learning function yielded an improvement of 6% over the control system. Precision(10) is defined as the average number of relevant documents among the first ten retrieved documents.

3.3 QOV vs. DOV

It is clear from Section 3.1 and Section 3.2 that each of the two views has its advantages and disadvantages. On one hand, QOV methods need little relevance feedback assessments to improve the user's information need representation (the query). The drawback is that these methods do not accumulate in any way, the feedback across users and time. After each retrieval process ends, all relevance feedback assessments made during this process are simply discarded. Accumulating relevance feedback makes it possible to analyze and improve different aspects of the documents and related queries. On the other hand, DOV methods need many more relevance feedback assessments in order to be able to generalize any relationships between documents and queries. But, once this feedback is available, DOV methods can yield improved retrieval effectiveness and better document representations (both dimension and better indexing terms). The purpose of this thesis is to explore a novel technique of using prior accumulated relevance feedback to improve document representation quality (see goals in Section 4.1).

CHAPTER 4

THE RFA ALGORITHM

This chapter represents a detailed presentation of the *Relevance Feedback Accumulation (RFA) Algorithm*. It begins by enumerating the goals of the algorithm. It continues by listing the assumptions behind the algorithm. The chapter concludes with a detailed presentation of the algorithm's modules (the pseudo-code).

4.1 Goals

Goal 1: Reduce the size of the document representations. In this context, size is defined as the average number of index terms per document representation.

Goal 2: Improve document representation quality (find higher quality terms to index each document). These terms are called concept terms.

Goal 3: Improve retrieval effectiveness.

Chapter 5, "Evaluation", discusses in more detail the way in which each of the three goals is evaluated.

4.2 Assumptions

The design of the RFA algorithm is based on several assumptions about the real world.

They are:

- (A) *Every document in the searchable document space is best characterized by a small set of terms.* The meaning is that each document can be characterized by a few words that best emphasize its topicality. All other terms (non content bearing terms) are less important for document representation purposes.

- (B) A **term,** as used by the RFA algorithm can be of two types:

    - **Single term**: a term consisting of only one word. Example: "data" or "mining."

    - **Composite term**: *a term consisting of two words.* A good example is the term "information retrieval" which has a more powerful semantic meaning than the single terms "information" and "retrieval" considered separately.

- (C) *The terms characterizing a document might or might not occur in that document.* It is common sense to say that a term that does not appear throughout a document can be a good semantic descriptor of that document's topics. For example, the topic of a document talking about laptops can be emphasized using the term "notebook" along with "laptop." "Notebook" is a synonym for "laptop." Therefore, it is a quality descriptor, even if it does not appear in the document. The RFA algorithm makes it possible to improve documents representations by adding new terms.

- (D) *When many users judge that a particular document is relevant to their query, there will be a relatively small set of query terms that will be common to most users' queries.* This assumption implies that the set of few terms that best characterize a document can be obtained by analyzing the users' relevance feedback assessments. For example, suppose a number of searchers are given the task to find documentation about how to install a Microsoft Local Area Network (LAN), by means of a search engine. Analyzing their queries one can notice that some of the terms appear in almost all of them (for example "Microsoft" or "network" or "LAN" etc). It means that in the users' opinion, these terms best characterize the topic of the document they are looking for. This is the mechanism used by the RFA algorithm, to identify the best terms/concepts to represent a document.

- (E) *Queries consist of short natural language statements that are of maximum twenty five words.* Most users are not willing to create elaborate queries consisting of many terms. The general user behavior pattern is to use the least effort in order to accomplish their goals(s) (Marchionini, 1992; Jansen, 2000). The number of terms per query usually averages around 2 (Andrews, 1996; Spink et al. 1998; Spink et al. 2000; Jansen et al. 2000; Johnson et al. 2001; Toms et al. 2001; Spink et al. 2001) or slightly greater values that are smaller than ten (Belkin et al. 1996). The RFA algorithm assumes a maximum of twenty five terms per query.
- (F) *This study assumes that the relevance feedback assessments pool is readily available.* The study does not concern about how to obtain the relevance assessments.

4.3 Composite Terms Handling – Ordered Terms Pairing (OTP) Heuristic

As mentioned under the assumptions in Section 4.2 (B), RFA manipulates two types of terms: simple and composite. This section describes the mechanism by which the RFA algorithm looks for composite terms within a query Q, from a relevance assessment (Q, D).

Let's consider **Q(term_1, term_2, term_3, term_4)** to be a query composed of four terms. The single terms of **Q** are: **term_1**, **term_2**, **term_3** and **term_4**. The composite terms are derived from the original query **Q** by using a heuristic called ***ordered terms pairing***. This heuristic takes all pairs of adjacent single terms from a query (maintaining their ordering in the query), and builds composite terms by aggregating them. In the case of **Q** considered above the heuristic will generate the following composite terms: **[term_1, term_2]**, **[term_2, term_3]**, and **[term_3, term_4].** After generating the composite terms, Q is considered to be composed of the union of all single and composite terms. The rationale behind the idea is the fact that many times individual words (single terms) do not provide enough meaning with respect to the topics of a document. From this point on, the notion ***term*** denotes both single and composite terms, treated similarly by the RFA algorithm.

A more specific example emphasizing how the ordered terms pairing heuristic works, is presented below (Figure 4.1).

Example

Let's consider the query **Q(data mining algorithms)**. The terms extracted from this query are:

1. Single terms: **data**, **mining** and **algorithms**
2. Composite terms: **data mining**, and **mining algorithms**

So the query is considered to comprise **5** terms: ***data***, ***mining***, ***algorithms***, ***data mining***, and ***mining algorithms***.

Figure 4.1 Example: OTP Heuristic.

4.4 The Relevance Feedback Accumulation (RFA) Algorithm

The RFA algorithm is based on the assumption that every document is best characterized by a set of few terms. These terms are called concept terms, or simply, concepts. Then, a concept term is a term that suggests/represents the topics of a document. The difficult part is to identify these concepts. Most automatic indexing algorithms are focused on finding the importance of terms within documents by calculating frequency-based measures. This is also called lexicographic analysis. Their direct assumption is that if a term occurs several times within a document, that term is likely to be a concept in the document. This, of course, is not always the case. Also, the above-referred automatic indexing algorithms cannot support the case of a concept term that does not appear throughout the document: the *word mismatch* problem. The RFA algorithm creates and maintains a dynamic document representation space as a response to the above problems. The importance of concepts is not given any more by a simple lexicographic analysis of the documents' content. The importance of terms is derived from RF assessments over time and across users. A document concept in this context is identified as a term that has reasonable support among all queries from all relevance assessments of this document. Support is a data mining measure emphasizing the occurrence percentage of an item in a set of transactions. In this case, a query term is considered to be the item while the query is considered to be the transaction. See the example illustrated below (Figure 4.2).

Example

Suppose document D was assessed as relevant to the following queries:

- Q1(term_1, term_2, term_3)
- Q2(term_1, term_2)
- Q3(term_1, term_4) and
- Q4(term_1, term_2, term_3)

In this case the support for each of the terms is:

- support(term_1)=4/4 ➔ 1.0
- support(term_2)=3/4 ➔ 0.75
- support(term_3)=2/4 ➔ 0.5 and
- support(term_4)=1/4 ➔ 0.25

The interpretation is that if we set the support threshold at 0.5 than term_1 and term_2 can be considered concept terms because appeared in more than 50% of the queries to which D was judged as relevant. term_3 and term_4 are not considered concept terms.

Figure 4.2 Example: Term Support Among Queries.

The important thing to mention is that concept discovery process is user-driven. In time, the concepts from each document representation will reflect the users' general perception regarding which are the most important terms to describe the topics of the document. An immediate logic augmentation is to eliminate terms with low support in a document from that document's representation. In this way the size of each document's representation is reduced.

Figure 4.3 shows the steps of the RFA algorithm.

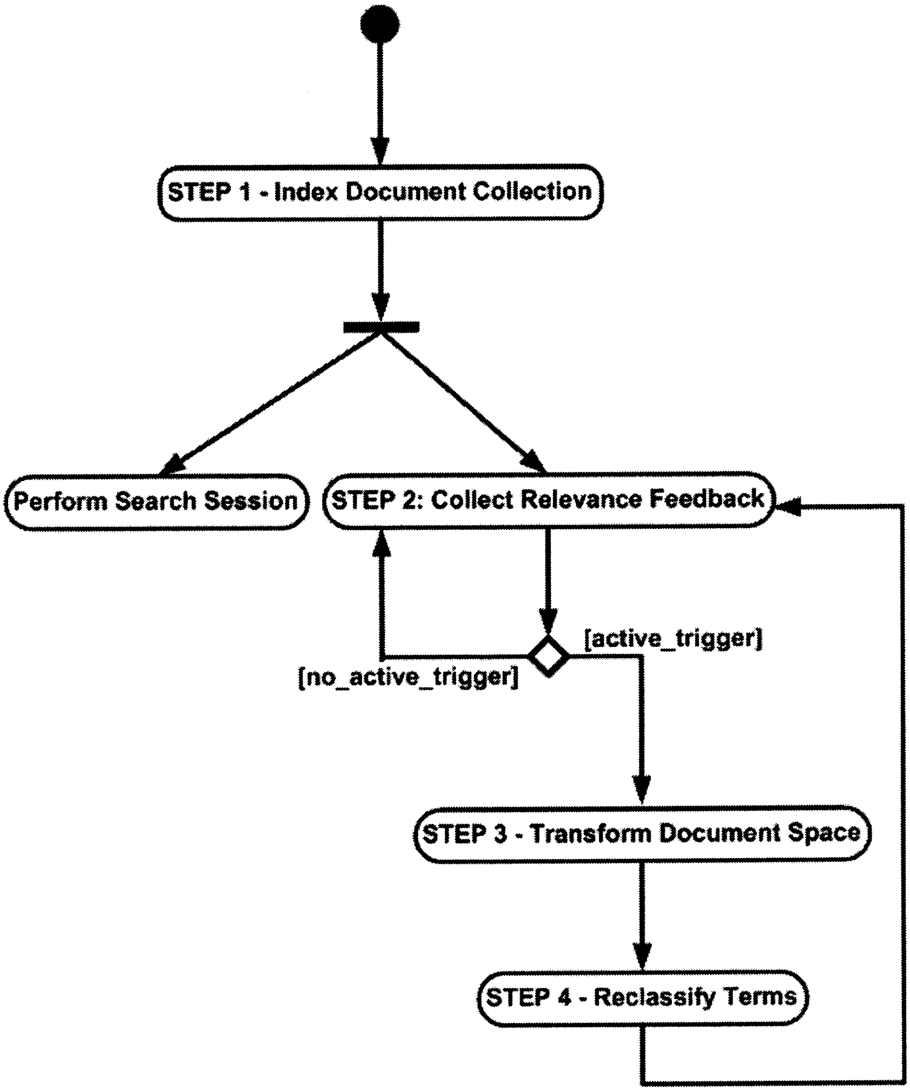

STEP 1 - Index Document Collection
Perform Search Session
STEP 2: Collect Relevance Feedback
[active_trigger]
[no_active_trigger]
STEP 3 - Transform Document Space
STEP 4 - Reclassify Terms

Figure 4.3 RFA Algorithm Steps (UML Activity Diagram).

STEP 1 - Automatic Indexing: during this step an initial document space is created by automatically indexing the whole document collection (see Figure 4.3). The indexing procedure uses standard lexicographic analysis (e.g. tf-idf based measures) complemented by additional improvement techniques like: stopped words removal and/or stemming. One very important mention to be made at this point is the fact that STEP 1 is performed only once, after the document collection is gathered. Afterwards, only STEP 2, 3 and 4 are infinitely repeated at pre-programmed intervals (see Figure 4.3), for as long as the system is up and running. The pre-programming is implemented using **activation triggers**. An activation trigger is defined as *a flag that becomes active whenever a certain condition relative to the relevance assessments collected so far, is satisfied*. An example of such an activation trigger would be a counter showing how many relevance assessments the system collected with respect to a certain document. Whenever this counter reaches a certain threshold (for example 1000 relevance assessments) the trigger becomes active.

From RFA algorithm's point of view, an activation trigger can be attached to:

- Individual documents
- Groups of documents
- Whole document collections

There are two types of activation triggers:

- Counters – counting items or events
- Timers – monitoring the time lapsed

Based on the retrieval system's internal and external characteristics, the designer can custom-configure the activation triggers to be used. Even more, the activation trigger design remains an open problem. The WISEarch system used to evaluate the RFA algorithm described by this thesis uses individual document counters as activation triggers. This counter, attached to each document, counts the number of relevance assessments collected for this document from all users.

STEP 2 – Collect Relevance Feedback: during this step the system will collect from searchers any relevance feedback assessment(s) of type (Q_0, D). One can observe on the UML activity diagram (Figure 4.3) that STEP 2 is a parallel/concurrent process with "Perform Search Session." That is because the process of collecting relevance feedback is part of the retrieval/browsing operation. During this step, the result might be a set of (query, document) tuples rather than a single tuple. This happens when the user judges more documents from the result set to be relevant to the same query.

For each of the (Q_0, D) tuples:

- Q_0 is transformed in Q using the ordered terms pairing heuristic, presented in Section 4.3. The relevance judgment (Q_0, D) then becomes (Q, D).
- (Q, D) is then used to update the data structure designed to accumulate the relevance feedback. This data structure is composed of two matrices called Term-Document Matrix (the document vector space), and Document Matrix. For all the terms that appear in both Q and D, the RF accumulation data structures are updated. The updating consists of increasing the relevance assessment counters for (term_Q, D) pairs in Term-Document Matrix, and for documents in Document Matrix. The relevance assessment counter for (term_Q, D) pairs in Term-Document Matrix, shows how many times the term term_Q and the document D were involved together, in a relevance assessment. The relevance assessment counter for documents in Document Matrix shows how many times document D was involved in a relevance assessment. Having computed these two values, it is easy to estimate the support for any (term_Q, D) pair. Any term in Q that does not appear in D, will be introduced as a new term in D's vector. Its initial weight is set to the minimum weight among all terms in D's vector. By this, new potentially high quality terms are added to one document's vector. This mechanism is aimed at tackling the word mismatch issue.

STEP 3 – Document Space Transformation: the document term modification takes place for each document whenever the attached activation trigger becomes active. In Figure 4.3, this step is marked as "Transform Document Space." This is the weight learning function.

For all (term, D) pairs:

Step 3.1: compute the new support value S_{NEW} (term, D) using information collected during previous STEPs 2.

Step 3.2: compute the support variance $\Delta_{SUP} = S_{NEW} - S_{OLD}$

Step 3.3: modify the weight of the tuple (term, D) as follows:

If $\Delta_{SUP} > 0$, $w_{NEW} = w_{OLD} + (1 - w_{OLD}) * \Delta_{SUP}$

The weight is increased proportionally with the increase in support Δ_{SUP}. The term $(1 - w_{OLD})$ makes sure the value of the weight will not exceed 1. Weight values range from 0 to 1.

If $\Delta_{SUP} < 0$, $w_{NEW} = w_{OLD} + w_{OLD} * \Delta_{SUP}$

The weight is decreased proportionally with the decrease in support Δ_{SUP}. In this case Δ_{SUP} is negative.

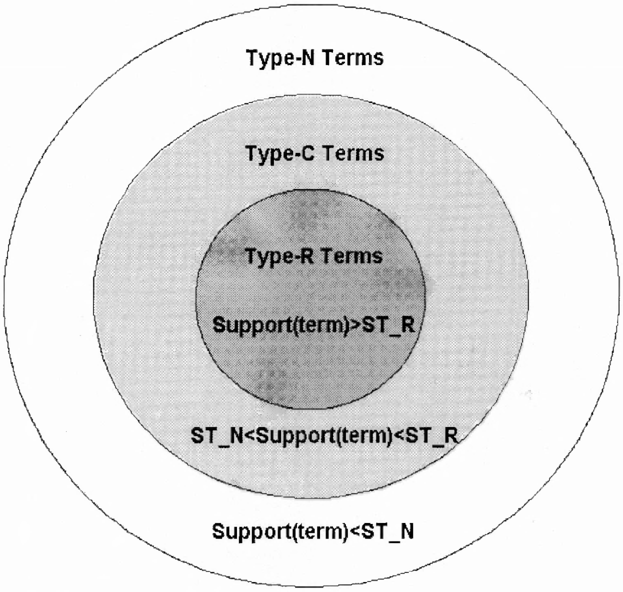
If $\Delta_{SUP} = 0$, $w_{NEW} = w_{OLD}$

No changes are made if the support of the term does not change.

STEP 4 – Term Classification: following Step 3, during this step, all the terms from D's vector are re-classified into three type categories, according to their support values. In Figure 4.3, this step is marked as "Reclassify Terms." Support is a data mining measure emphasizing the occurrence percentage of an item in a set of transactions. In this case, a query term is considered to be the item while the query is considered to be the transaction. Two threshold parameters are used for this task: ST_N and ST_R (0 ≤ ST_N, ST_R ≤ 1) (see Figure 4.4).

The term type categories are:

- **Type R terms**: relevant terms, having high support. (i.e. terms having support greater than ST_N).
- **Type C terms**: candidate terms, having moderate support (i.e. terms having support greater that ST_N but smaller than ST_R).
- **Type N terms**: non-relevant terms, having low support (i.e. terms having support smaller than ST_N).



Title: Types of index terms
Author: Razvan Stefan Bot

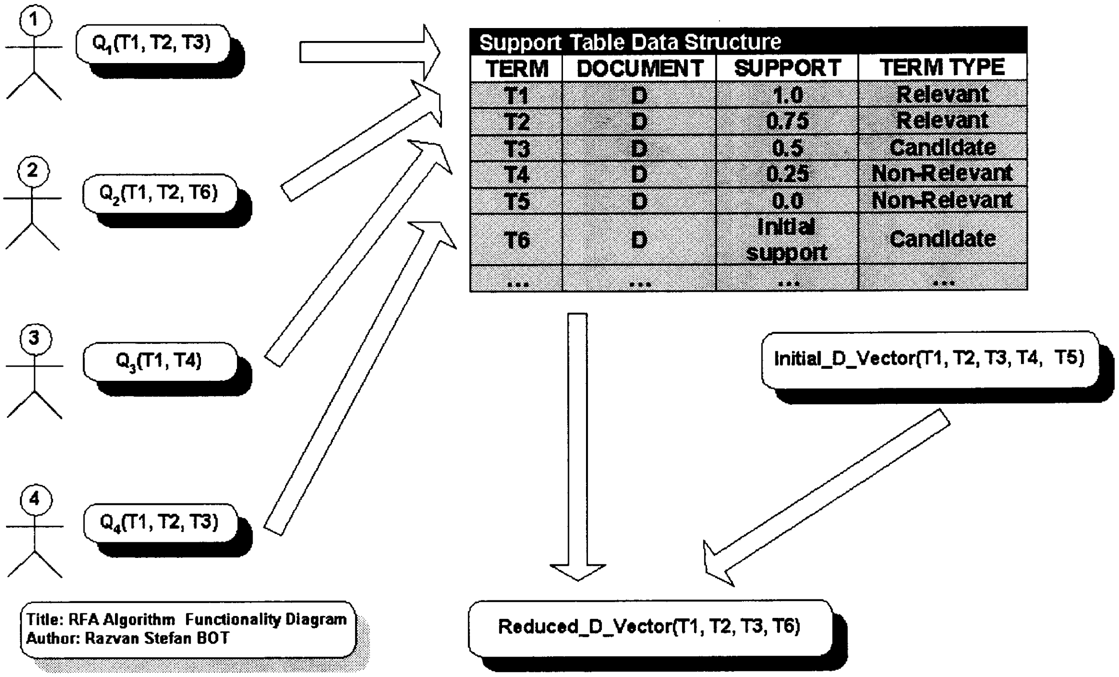
Figure 4.4 Types of Index Terms.

The type N terms will not be considered indexing terms anymore, but they will still be kept in the data structure because their support might increase with future relevance assessments. By this, the size of document representations is reduced.

A simplified illustration of the RFA algorithm is given in Figure 4.5. The figure only shows the support value for each (term, D) pair rather that the actual weight because this weight is directly proportional to the support value. Also, for simplification matters the figure only emphasizes the situation for one document. Suppose this document is D. **STEP 1** is depicted by the Initial_D_Vector, containing the terms: T1, T2, T3, T4 and T5. This is the corresponding VSM vector for D calculated during the automatic indexing phase. During **STEPs 2** and **3**, the relevance assessments given by users 1, 2, 3 and 4 are collected and support is calculated for each (term, D) pair. One can observe, for example, that term T2 appears in queries Q_1, Q_2 and Q_4 hence its support will be 0.75 since there are four queries in total. During **STEP 4** all terms of document D, are classified according to their support values.

In this case:

- Type R terms: the terms having the support greater than or equal to 0.75.
- Type C terms: the terms having the support greater than or equal to 0.5 but less than 0.75.
- Type N terms: the terms having the support less than 0.5.


1
Q_1(T1, T2, T3)
2
Q_2(T1, T2, T6)
3
Q_3(T1, T4)
4
Q_4(T1, T2, T3)
Initial_D_Vector(T1, T2, T3, T4, T5)
Reduced_D_Vector(T1, T2, T3, T6)
Title: RFA Algorithm Functionality Diagram
Author: Razvan Stefan BOT

Support Table Data Structure

TERM	DOCUMENT	SUPPORT	TERM TYPE
T1	D	1.0	Relevant
T2	D	0.75	Relevant
T3	D	0.5	Candidate
T4	D	0.25	Non-Relevant
T5	D	0.0	Non-Relevant
T6	D	Initial support	Candidate
...	...	...	...

Figure 4.5 RFA Algorithm Functionality Diagram – Steps 1, 2, 3 and 4.

Terms having type R and C are selected to be index terms for document D (see Reduced_D_Vector on figure consisting of only four terms rather than five as in the Initial_D_Vector). They are T1, T2, T3 and T6. One can observe that even though term T6 does not occur in document D, it was introduced as a new indexing term since it appeared in some of the queries. At the same time, terms T4 and T5 are removed from being indexing terms since they are not supported enough in the queries.

Before presenting the algorithm's pseudo-code it is necessary to describe the data structures and a set of notations used. The data structures are the Document-Matrix (Figure 4.6) and the TD-Matrix (Figure 4.7).

The Document Matrix

Document-Matrix		
Document	**No_Of_RF_Assessments**	**RF_Assessments_Counter**

Figure 4.6 The Document Matrix.

In this data structure the column names have the following meaning:

- **Document**: the document ID in the document collection.
- **No_Of_RF_Assessments**: the total number of queries to which the **Document** was assessed as being relevant. Used in the computation of **Support** from TD_Matrix.
- **RF_Assessments_Counter**: a counter indicating how many (query, **Document**) assessments have been made without re-classifying the **Terms**. Each time this counter is exceeded the **Terms** indexing this **Document** are re-classified as belonging to **Type R**, **C** or **N**. This is an activation trigger.

The Term-Document Matrix (TD-Matrix)

TD-Matrix					
Term	**Document**	**Weight**	**No_Of_Queries**	**Support**	**Type**

Figure 4.7 The Term-Document Matrix.

The columns in the data structure have the following meaning:

- **Term**: the word representing a term in a document.
- **Document**: the document ID in the document collection.
- **Weight**: the frequency-based weight for this (**Term, Document**) tuple.
- **No_Of_Queries**: the number of queries that contained **Term**, to which **Document** was assessed relevant by a user.
- **Support**: the ratio between **No_Of_Queries** in TD_Matrix and the total number of queries to which **Document** was assessed as relevant. This amount is represented by **No_Of_RF_Assessments** in the Document_Matrix. The total number includes also the queries that did not contain **Term**.
- **Type**: Indicates the type of this Term. There are three types for a term (see Figure 4.4)
- **R** (Relevant Term): all the terms having support (relative to **Document**) above a threshold **ST_R**. These are the "concepts" that best characterize the **Document**. They are used to index the document.

- C (Candidate Term): all terms having support (relative to **Document**) less than **ST_R** but greater than **ST_N**. These are terms that are potential "concepts" for **Document**. They are also used to index the **Document**.

- N (Non-Relevant Term): all terms having support less than **ST_N**. They are not used to index the **Document**. These are the terms we can get rid of when representing the **Document**, hence achieving a size reduction.

Notations used throughout the RFA algorithm:

- **Q_0** – the query posed by the user after it was cleaned. Cleaning is the process of removing the stopped-words.

- **Q** – the result of applying ordered terms pairing heuristic on **Q_0**.

- **RF(Q)** – the relevant set contains all the documents the user judged relevant, with respect to Q and R(Q).

- **DOCUMENT_COUNT** – the count threshold indicating how many (query, **Document**) assessments have been made without re-classifying the **Terms**. See **RF_Assessments_Counter** in Document-Matrix.

- **ST_R** – the support threshold above which terms are considered relevant terms.

- **ST_N** – the support threshold below which terms are considered non-relevant terms.

RFA Algorithm Pseudocode

The pseudo-code of the RFA algorithm consists of six procedures.

- **Accumulate_Relevance_Feedback**: represents the main procedure. It models in pseudo-code format the **Collect Relevance Feedback**, **Transform Document Space**, and **Reclassify Terms** activities from the UML activity diagram presented in Figure 4.3.

- **Ordered_Terms_Pairing**: the procedure takes a user query as input and generates an extended query using the ordered terms pairing heuristic described in Section 4.3.

- **Update_Document_Matrix**: this procedure updates the document matrix. It increases No_Of_RF_Assessments and decreases the RF_Assessments_Counter.

- **Update_TD_Matrix**: updates the TD_Matrix for a (term,doc) pair. A new entry is created for **term** if it does not already exist as an index term for doc. If it already exists the No_OF_Queries is incremented by 1.

- **Update_TD_Weight**: updates weights for all index terms of a document. It is the corresponding procedure for “Transform Document Space" in the UML activity diagram in Figure 4.3. It adjusts the weight of each index term of a document. The change is directly proportional to the variance in support of the term with respect to the document. If the support increased, the weight goes up. If the support decreased, the weight goes down.

- **Classify_Terms**: classifies the terms of a document. It is the corresponding procedure to “Reclassify Terms” in the UML activity diagram in Figure 4.3. Based on the support of a term with respect to a document, it classifies the term in one of three possible categories: Type R, Type C or Type N (see Figure 4.4).

```
Procedure Accumulate_Relevance_Feedback
PROCEDURE Accumulate_Relevance_Feedback($Q_0$, RF($Q_0$))
BEGIN
        //[1] apply ordered terms pairing heuristic on $Q_0$ and obtain Q
         Q = Ordered_Terms_Pairing($Q_0$);
        //[2] Collect Relevance Feedback on UML activity diagram
        //[2.1] update document matrix
         Update_Document_Matrix(RF(Q));
         //[2.1] update TD matrix
        FOR EACH doc WHERE $doc \in RF(Q)$
        BEGIN
          FOR EACH term WHERE $term \in Q$
          BEGIN
          Update_TD_Matrix(term, doc);
          END
          //check if activation trigger is true
          IF (Document_Matrix[doc]→ RF_Assessments_Counter $\leq 0$ )
          BEGIN
          //Transform Document Space on UML activity diagram
          Update_TD_Weight(TD_Matrix[doc]);
          //Reclassify Terms on UML activity diagram
          Classify_Terms(doc, ST_R, ST_N);
          END
        END
END
```

Figure 4.8 Procedure Accumulate_Relevance_Feedback.

```
Procedure Update_Document_Matrix
PROCEDURE Update_Document_Matrix(RF(Q))
BEGIN
    FOR EACH doc WHERE $doc \in RF(Q)$
        BEGIN
          Document_Matrix[doc]→Number_Of_RF_Assessments ++;
          Document_Matrix[doc]→RF_Assessments_Counter --;
        END
END
```

Figure 4.9 Procedure Update_Document_Matrix.

```
Procedure Classify_Terms
PROCEDURE Classify_Terms(doc, ST_R, ST_N)
BEGIN
FOR EACH term WHERE $term \in TD\_Matrix[term, doc]$
BEGIN
// case (a)
IF (TD_Matrix[term, doc]→Support $\geq$ ST_R)
  THEN TD_Matrix[term, doc]→Type := R;
// case (b)
IF (TD_Matrix[term, doc]→Support $\leq$ ST_N)
  THEN TD_Matrix[term, doc]→Type := N;
// case (c)
IF (TD_Matrix[term, doc]→Support  BETWEEN (ST_R, ST_N))
   THEN TD_Matrix[term, doc]→Type := C;
//reset counter
Document_Matrix[doc]→RF_Assessments_Counter :=  DOCUMENT_COUNT;
END
```

Figure 4.10 Procedure Classify_Terms.

Procedure Update_TD_Weight

```
PROCEDURE Update_TD_Weight(TD_Matrix[doc])
BEGIN
FOR EACH term WHERE $term \in TD\_Matrix[doc]$
BEGIN
$S_{OLD}$=TD_Matrix[term, doc]; // retain old support
$TD\_Matrix[term, doc] \rightarrow Support := \frac{TD\_Matrix[term, doc] \rightarrow Number\_Of\_Queries}{Document\_Matrix[doc] \rightarrow No\_Of\_RF\_Assessments}$
$w := TD\_Matrix[term, doc] \rightarrow Weight;$
$s_{NEW} := TD\_Matrix[term, doc] \rightarrow Support;$
$\Delta_{SUP} = S_{NEW} - S_{OLD}$
IF( $\Delta_{SUP} > 0$ ) THEN
BEGIN //increase weight because support increased
    $TD\_Matrix[term, doc] \rightarrow Weight := w + (1 - w) * \Delta_{SUP};$
END
ELSE IF( $\Delta_{SUP} < 0$ )
BEGIN // decrease weight because support decreased
    $TD\_Matrix[term, doc] \rightarrow Weight := w + w * \Delta_{SUP};$
END
END
END
```

Figure 4.11 Procedure Update_TD_Weight.

```
Procedure Update_TD_Matrix
PROCEDURE Update_TD_Matrix(term, doc)
BEGIN
        //the term belongs to doc and can be of type: R, C or N
        IF (term ∈ doc) THEN
        BEGIN
        //increment by 1
        TD_Matrix[term, doc]→Number_Of_Queries ++;
        END
        //the term does not appear in doc => will be introduced as a C type term
        ELSE BEGIN
        //create entry in index structure
        CREATE ENTRY TD_Matrix[term, doc];
        //give an initial weight
        TD_Matrix[term, doc] := MIN(TD_Matrix[$term_{TYPE\_C}$, doc]→Weight);
        //set No_Of_Queries
        TD_Matrix[term, doc]→No_Of_Queries := 1;
        //set the type flag
        TD_Matrix[term, doc]→Type := C;
        END
END
```

Figure 4.12 Procedure Update_TD_Matrix.

CHAPTER 5

EVALUATION

This chapter is an overview of the experimental settings, and the procedures that were used to evaluate the RFA algorithm. It contains detailed descriptions of the document collection, the queries, the relevance assessments, the independent variables, the test runs, and the evaluation measures.

5.1 Test Document Collection

The document collection used for evaluation consists of a portion of the TIPSTER/TREC document corpus. The TREC collection is the mandatory corpus used in experiments submitted to the Text REtrieval Conference (TREC). The reasons to choose the TREC collection for this evaluation are: (a) it is widely used in information retrieval, (b) it contains a large number of diverse topics (queries) with corresponding relevance assessments, and (c) it is probably the only collection providing enough relevance assessment data-points.

Any version of the TREC document collection contains three items:

- Documents: the actual textual documents stored in SGML format.
- Topics: a topic is a more complex query. (see Section 5.2).
- Relevance Assessments: a list of relevant documents from the collection is associated with each topic. This makes it possible to design automatic evaluation processes.

The TIPSTER collection contains more than 510,000 documents. From these, around 25,000 documents were selected to form the evaluation document collection. The selection process was performed according to the following procedure: for each topic T to be used during evaluation (i.e. topics 51-100), we first selected all relevant documents using the provided relevance assessments. Then, we randomly selected more non-relevant documents, until the total reached 500 for each T. With 50 Ts, the resulting document collection will consist of about 50X500=25,000 documents. The reason to use only a sub-collection is that the RFA algorithm only performs document space transformations when the documents are relevant to the 50 topics. There is no advantage in choosing a larger sub-collection, since in most cases the irrelevant documents do not affect representations. Only the initial effectiveness parameters would be different for a larger collection. The average density of relevant documents per topic for this collection is 0.012.

5.2 Queries and Relevance Assessments

A **query** is defined as the observable representation of a user's information need. This observable representation is in fact *what the user actually types to a retrieval system* (Buckley and Walz, 1999, p. 1).

The standard TIPSTER/TREC collection provides topics rather than queries. A TREC **topic** is a longer query with structure. As defined within the TREC Query Track a *topic represents an information need of a user. It includes a full statement of what information is wanted as well as information the user knows that pertains to the request* (Buckley, 2000, p. 2). An example of a TREC topic is given in Figure 5.1. They are stored in standardized SGML format. For the full version of TREC Topic 051 please refer to Appendix A.

TREC topic no. 051

<top>
<head> Tipster Topic Description
<num> Number: 051
<dom> Domain: International Economics
<title> Topic: Airbus Subsidies
<desc> Description:
Document will discuss government assistance to Airbus Industrie, or mention a trade dispute between Airbus and a U.S. aircraft producer over the issue of subsidies.
<narr> Narrative:
A relevant document will cite or discuss assistance to Airbus Industrie by the French, German, British or Spanish government(s), or will discuss a trade dispute between Airbus or the European governments and a U.S. aircraft producer, most likely Boeing Co. or McDonnell Douglas Corp., or the U.S. government, over federal subsidies to Airbus.
</top>

Figure 5.1 TREC Topic No. 51.

The fact that there are only topics, rather than queries, makes it necessary to generate several queries for each of these topics.

The reasons to do this are:

- TREC topics are not realistic, because they are too long and too well defined. It has been shown that most users do not care to compile such long and complex queries. The number of terms per query averages around two (Toms et al., 2001).

- A retrieval system will never obtain complex information need descriptions such as the topics, from its users. *The **query** is the only information from the user a retrieval system has* (Buckley and Walz, 1999, p. 1).

- The RFA algorithm is based on the probabilistic estimation (support) of the importance of a term to a document. This importance is derived from user relevance assessments. In this case, several data points are needed for each document. This means that each document must be judged relevant to several different queries. In TREC topic relevance sets, each document is usually judged relevant to 1 or 2 topics. The lack of query-to-document assessment data points is actually a characteristic of all test document collections. This is one reason for the little attention received by the document-oriented view technique (Bodoff et al. 2001).

- TREC topics are subjective from the ordinary user's point, because these topics are expert generated. This experiment, tries to emulate as much as possible, a real user situation.

Previous work done within the TREC Query Track, already generated a pool of about 43 queries for each of the 50 TREC topics from 51 to 100 (consult APPENDIX B to see the list of queries generated for TREC Topic 51). Both experts and non-experts generated this corpus.

The queries were grouped in three categories (Buckley, 2000):

- Very short: 2-4 words based on the topic and possibly a few relevant documents from TREC disk 2.
- Sentence: 1-2 sentences using topic and relevant documents.
- Sentence-Feedback only: 1-2 sentences using only relevant documents. The aim is to increase vocabulary variability.

Each of the TREC topics comes with an associated relevant documents set. For this evaluation, all generated queries specific to a TREC topic, were considered to share that TREC topic's relevant documents.

5.3 Experimental Design

The independent variables of the experimental setting were:

(a) Retrieval system type: Table 5.1 lists all retrieval systems evaluated and compared for this study. The systems ST and BR represent the baseline systems. The systems BB and BS are hybrid systems created in order to test two other possible approaches to document representation improvement.

Table 5.1 Retrieval System Types

System name	Notation
Standard	**ST**
RFA	**RFA**
Brauen	**BR**
Brauen-Batch	**BB**
Brauen-Smooth	**BS**

The standard retrieval system (**ST**) is built on the vector space model with LTC as the weighting scheme (Salton and Buckley, 1988; Savoy et al. 1996), and inner product as the similarity measure. The LTC weighting scheme has the following mathematical form:

$$w_{ij} = \frac{[\log(tf_{ij}) + 1] * idf_j}{\sqrt{\sum_{k=1}^{t} ([\log(tf_{ik}) + 1] * idf_k)^2}}$$

In the above formula, w_{ij} is the weight of term j in document i, tf_{ij} is the term frequency of term j in document i, idf_k is the inverse document frequency of term k, and t is the number of terms in document i. This formula is preferred because it accounts for both document length normalization and document collection size normalization.

The RFA retrieval system (**RFA**) consists of a standard system (ST) augmented with the RFA algorithm. The RFA algorithm itself was presented in greater detail in Chapter 4.

The Brauen system (**BR**) consists of the standard system (ST) augmented with the Brauen algorithm (Brauen, 1969), described in Chapter 3. The parameters used for the Brauen algorithm are those reported to be optimal in (Brauen, 1969). They are: beta=30, delta=8 and gamma=0.225.

The Brauen-Batch (**BB**) system represents a hybrid version of the BR system that does not perform document modification for each relevance assessment, but rather batches them for efficiency. The original Brauen algorithm modifies a document vector each time a relevance assessment is available for that document. This, results in low efficiency as well as high weight oscillation. To better illustrate this issue, consider the following situation: a BR retrieval system captures 100 relevance assessments for document D, during a time interval T. Following Brauen's algorithm, the document D's vector is altered 100 times during the time period T. This is obviously not efficient, but more importantly the weights will have 100 different values during T. If the document modification function is rapidly increasing (or decreasing), as the Brauen's function does, the weights oscillation is characterized by high amplitudes. To attenuate this effect, the Brauen-Batch system associates an activation trigger to each document vector. Therefore, each document vector is modified only when the associated trigger becomes active. As described in Section 4.4, such an activation trigger becomes active, whenever a pre-defined number of relevance assessments, is collected for a document.

The Brauen-Smooth (**BS**) system is a hybrid version of the BR system that applies an exponential smoothing after each document modification operation. Exponential smoothing is a forecasting technique that weights past observations with exponentially decreasing weights. The BS system uses the *single exponential smoothing scheme*[1]. The mathematical formula for this scheme, adapted to the retrieval system context, is the following:

$$sw_t^i = \alpha * w_{t-1}^i + (1-\alpha) * sw_{t-1}^i, \quad 0 < \alpha \leq 1, \quad t \geq 3.$$

[1] http://www.itl.nist.gov/div898/handbook/pmc/section4/pmc431.htm

In this equation, sw_t^i is the smoothed weight value of term i at time t, w_{t-1}^i is the non-smoothed weight value of term i at time t-1, sw_{t-1}^i is the smoothed weight value of term i at time t-1, and α is the smoothing constant. The smoothing constant represents the speed at which former smoothed values are dampened. If alpha is close to 1, the dampening is quick, while if alpha is close to 0 the dampening is slow. The initial step of the smoothing process for each term i is to set $sw_2^i = w_1$. There is no sw_1^i. The inconvenience with this system is the fact that it requires double weight document vectors. A double weight document vector has two weights associated with each term from the vector. One of the weights is the BR weight, and the other is its smoothed version. After the training phase of the document vector space is complete, the weight of each term is set to its last forecasted value (last smoothed value). The BS systems were tested for two different smoothing constants: 0.2 and 0.4. For larger values the dampening of past observations is too quick. 0.4 was found to be the best smoothing constant for this document collection.

(b) Stemming: each of the baseline and augmented systems was tested under both stemming and no stemming conditions. Stemming is a technique that reduces classes of words to their common stem (root). For example, “information” and “informational” are both reduced to their common stem “inform”. The reason for introducing stemming as an independent variable is the fact that stemming itself is able to reduce document representation size. It is then interesting to see if stemming together with RFA can yield even better results than RFA alone. For stemming conditions we used the Lovins stemmer (Lovins, 1968).

(c) Browsing batch size: represents the number of documents, out of the total number of retrieved documents, that are examined by the "user" (the evaluation is done automatically) to provide relevance assessments. In order to better simulate the real user experience, for any given training set query, the RFA algorithm does not modify all relevant returned documents. The relevant documents set is restricted to those identified within the first 30 or 50 retrieved documents. All others are ignored. The two browsing batch sizes of 30 and 50 were derived based on the findings reported in (Spink et al. 2001), where the median number of retrieved web pages browsed was found to be 8, while a page usually displays 10 results. At the same time a large percentage of the users (around 48%) only browsed one or two pages. For this study 30 represents the normal user effort load, while 50 represents the maximum user effort load.

(d) ST_N classification threshold (see Figure 4.4): while ST_R was held constant at 0.3, ST_N was tested for 0.03, 0.04 and 0.05. The optimal threshold parameters setting for this collection was found to be: ST_R=0.3 and ST_N=0.05.

Table 5.2 shows a holistic view of the evaluation setting. In this table, the parameter for the BS systems represents the value of the dampening constant α, and the parameter for RFA systems represents the value of the ST_N threshold (see Figure 4.4). The integer values in each cell of the table are used to number the different conditions (test runs).

Table 5.2 The Evaluation Setting

Evaluation set 1	No Stem	Stem
ST	1	15
BR	2	16
BB	3	17
BS(0.2)	4	18
BS(0.4)	29	31
RFA(0.03)	5	19
RFA(0.04)	6	20
RFA(0.05)	7	21
Evaluation set 2		
ST	8	22
BR	9	23
BB	10	24
BS(0.2)	11	25
BS(0.4)	30	32
RFA(0.03)	12	26
RFA(0.04)	13	27
RFA(0.05)	14	28

5.4 Measurements

This section is an overview of the measures that were used in order to evaluate each of the three goals of the RFA algorithm. Section 5.4.1 presents the design of the test runs used to automatically collect data. The actual measures are presented in Section 5.4.2. Because the measurement tools differ from one goal to another, section 5.4.2 is organized according to the three goals (see Section 4.1).

5.4.1 Test Runs

The measurements were observed for two training/evaluation sets, randomly generated from the TREC data. The testing procedure is inspired by (Brauen, 1969), because it is one of the few concerning a document-oriented technique, and because the testing procedure emulates a real-user situation.

The procedure works along the following guidelines:

- The pool of queries is divided into two distinct question sets: the training set and the evaluation set. The training set contains about 80% of the queries from the pool. The remaining 20% of the queries form the evaluation set. The dividing process is repeated twice.
- The training set is used to train the system. Training the system means to modify the document vectors according to the RFA algorithm (or the other algorithms: BR, BB and BS).
- The evaluation set is used to test average retrieval performance measures.

5.4.2 Measures

This section presents the measures used for the RFA algorithm's evaluation. They are grouped according to each of the three goals presented in Section 4.1.

Goal 1: to evaluate size reduction, the average number of indexing terms per document was computed. Under each of the conditions BR, BB, BS and RFA, the average was calculated only for those documents whose representations were altered during the training phase. The reason to select only these documents is to get a clearer estimate of how effective the RFA system is, when compared to the other systems.

Goal 2: the quality of the document representations was evaluated by a set of judges. For this, a number of 121 students were used. There were 103 graduate students and 18 undergraduate students. Their familiarity with the "document keyword" concept was captured using a 5-points Likert scale, as part of the background questionnaire. A score of 1 meant "very familiar", a score of 2 meant "familiar", a score of 3 meant "neutral", a score of 4 meant "not familiar", and a score of 5 meant "not familiar at all." The distribution of familiarity with the "document keyword" concept among the 121 students is presented in Table 5.3.

Table 5.3 Familiarity Distribution Among Students

Scale value	Meaning	# of students
1	Very familiar	19
2	Familiar	36
3	Neutral	23
4	Not familiar	18
5	Not familiar at all	25

The document representation quality was evaluated and compared for the ST, BR and RFA systems. Each subject was presented with 5 evaluation packages. Each package contained one document plus a list of index terms associated with the document. The index terms list associated with each document was built by selecting and combining the top 20 most important (highest support for RFA and highest weights for ST, BR) index terms from all the three retrieval systems ST, BR and RFA. Duplicates were then eliminated and index terms were ordered alphabetically. The subjects did not know the source or the initial ordering of the index terms, in the 3 documents spaces, from which they were selected. Each subject was asked to first carefully read and understand the topic(s) of the document. After that he/she was asked to assign a score, on a scale from 1 to 5, for each of the index terms in the list associated with the document they read. The subjects assigned the scores based on their judgment of how good they considered each term was, in describing the topic(s) of the document. For this evaluation, 25 topics from the 50 available were randomly selected. Then, 2 documents were randomly selected for each topic. The documents were selected out of the ones whose vectors were altered by both the RFA and the BR algorithm, during the training phase. The ST system was not taken into consideration because it does not alter the document vectors. By this, a set of 50 documents was selected for document representation quality evaluation. They were uniformly split among the 121 subjects. Please consult APPENDIX C to see a document representation quality evaluation questionnaire sample.

The quality of the document representation modification function (learning curve) was also evaluated for the BR, BB, BS and RFA systems. The quality of a learning function relies on its ability to rapidly learn the weight of a term-document pair, and to keep the future alterations close to this value. In other words, a poor quality learning function will generate far apart values for the same weight at different points in time. The average mean square error (MSE) for all terms from 50 randomly selected documents was estimated The documents were selected from those that were altered by BR, BB, BS and RFA systems respectively. The reference point for MSE calculation was the last term weight, considered to be the learned term weight.

Goal 3: several measures were computed and compared in order to evaluate the retrieval performance.

They were:

- The 10-Point Average Precision: this is calculated by averaging the precision of the retrieval system at ten different recall levels (0.1, ... , 1.0). A recall level of k is attained when a fraction of k from the total number of relevant documents were retrieved. The average can be calculated for each query individually or averaged over all queries. Brauen (1969) used this measure in evaluating his document modification algorithm. In order to compare the RFA algorithm with the BR algorithm, this measure had to be used.

- The F-measure (van Rijsbergen, 1979) combines precision and recall together. The formula is given by:

$$F_\alpha = \frac{(\alpha^2 + 1)PR}{\alpha^2 P + R}$$

  In this formula, P is precision, R is recall, and α is a variable parameter that allows the control for the relative importance of precision and recall. For low values of α, precision is more important than recall. When α is 1, precision and recall are equally important. If α is greater than 1, recall is more important than precision.

- The average precision fails to take into account the sequentiality effect of how the retrieval system presents the documents to the user. In the best case, the system displays all relevant documents before irrelevant ones. Two measures take this sequentiality effect into consideration: normalized precision and normalized recall. They are calculated for single queries and can also be averaged over all queries. Normalized precision and recall are user-oriented measures related to satisfaction and frustration (Korfhage, 1997). Their definitions from (Brauen, 1969):
  - Normalized Recall: $R_{NORM} = 1 - \frac{1}{R} * \frac{\sum_{i=1}^{R}(r_i - i)}{N - R}$
  - Normalized Precision: $P_{NORM} = 1 - \frac{\sum_{i=1}^{R}\log r_i - \sum_{i=1}^{R} i}{\log\left[\frac{N!}{(N-R)!R!}\right]}$

In the formulas above, N represents the document collection size, R is the total number of relevant documents for a single query, i is the i^{th} relevant document, and r_i is the rank of the i^{th} document. Normalized precision is a measure that takes into consideration the ranking sequence of the retrieved documents. An ideal retrieval system should present all relevant documents first. Intermixing irrelevant documents in between the relevant ones, in the response set, decreases the retrieval quality. For exemplification, consider first that a retrieval system retrieves 100 documents in response to a query. Out of the 100 documents only 50 are relevant to the query. Furthermore, suppose the 50 relevant documents are presented to the user before the other irrelevant 50. The precision in this case is 0.5. Consider now another retrieval system with the exact same characteristics, except it first presents the 50 irrelevant documents to the user, and only afterwards the other 50 relevant ones. The precision of this second retrieval system it is still 0.5. They also have similar recall. Still, they are not the same. The first retrieval system is better, because the way it presents the returned documents: the relevant ones first. Therefore, precision and recall alone cannot pinpoint the effect of the sequence in which retrieved documents are presented to the searcher. Normalized precision is able to emphasize this difference.

CHAPTER 6

RESULTS AND ANALYSIS

This chapter is an overview and analysis of the results. After several RFA parameter configurations were tested, the optimal was found to be: ST_R=0.3 and ST_N=0.05. All RFA systems whose results are reported throughout this section are based on this parameter configuration.

6.1 Size Reduction

This section presents the document representation size reduction results. Table 6.1 and Table 6.2 show the size of the modified document space for all the tested systems. The results are averaged over the two evaluation sets. In the two tables, BSIZE refers to the value of the independent variable "browsing batch size." Each cell in the table displays the average number of index terms per document. Under each of the conditions, BR, BB, BS and RFA, the average is calculated for all documents whose representations were altered during the training phase. The reason to select only these documents is to get a clearer estimate of how effective the RFA algorithm is, when compared to the other systems. From this point on, the formulation "document space," is used to refer only to these document representations. In other words, the "document space" is the set of all document representations altered during the training phase. For the ST systems, the average number of index terms per document is calculated using the set of modified document representations from the BR, BB, BS and RFA systems, to which is compared.

The systems BR, BB and BS generate document spaces with similar or very close sizes. Therefore, they are listed together in the same column in Table 6.1 and 6.2. The bolded percentage values in the "RFA" column show the extent of document space size reduction as compared to the other systems.

Table 6.1 Average Number of Index-Terms per Document (BSIZE=50)

	BSIZE=50		
	ST	**BR, BB, BS**	**RFA**
Stem	150.0	174.3	19.7 **-86.8% < ST** **-88.6% < BR, BB, BS**
No Stem	164.4	190.5	19.6 **-88.0% < ST** **-89.7 < BR, BB, BS**

Table 6.2 Average Number of Index-Terms per Document (BSIZE=30)

	BSIZE=30		
	ST	**BR, BB, BS**	**RFA**
Stem	140.0	162.7	18.5 **-86.7% < ST** **-88.6% < BR, BB, BS**
No Stem	154.7	178.8	18.3 **-88.1% < ST** **-89.7% < BR, BB, BS**

The results clearly indicate that the RFA algorithm reduces the size of the affected document space by at least 86.6%. The size reduction rate goes as high as 89.7%. It is very important at this point to mention that the reduction is obtained while preserving or improving the retrieval effectiveness of the systems augmented with RFA (see results in Sections 6.2 and 6.3). In Section 5.3, it was stated that the reason for introducing stemming as an independent variable was to test if there is any improvement when combining it with the RFA algorithm. The results show no noticeable difference between the "Stem" and "No Stem" conditions.

6.2 Overall Retrieval Effectiveness

This section presents the overall retrieval effectiveness for all the systems that were tested. Tables 6.3 and 6.4 show the comparison of the retrieval effectiveness for all five systems evaluated in this study. In the two tables, P is precision, R is recall, F is the F-measure, P_N is the normalized precision, and R_N is the normalized recall. All systems in Table 6.3, except ST, were evaluated with a value of 50 for the browsing batch size. All systems in Table 6.4, except ST, were evaluated with a value of 30 for the browsing batch size. For the BS system, the smoothing coefficient is 0.4. For the computation of the F-measure, the coefficient α_F was set to 0.5, meaning that precision was considered to be twice as important as recall.

The results are averaged over the two evaluation sets. The relative increase (the percentages in the parenthesis) for systems BR, RFA, BB and BS is relative to the ST system. Parameter values displayed in bold represent the best value for that specific parameter, among all systems in the same table column.

Table 6.3 Retrieval Effectiveness (BSIZE=50)

	BSIZE=50	
	Stem	No Stem
ST	P=0.045 R=0.927 $F_{0.5}$=0.054 P_N=0.767 R_N=0.96	P=0.066 R=0.858 $F_{0.5}$=0.078 P_N=0.762 R_N=0.965
BR	P=0.045 (+0.0%) R=0.94 (+1.4%) $F_{0.5}$=0.054 (+0.0%) P_N=0.785 (+2.3%) R_N=0.96 (+0.0%)	P=0.066 (+0.0%) R=0.882 (+2.7%) $F_{0.5}$=0.078 (+0.0%) P_N=0.786 (+3.1%) R_N=0.964 (-0.1%)
RFA	**P=0.047 (+4.4%)** R=0.892 (-3.7%) **$F_{0.5}$=0.057 (+5.5%)** **P_N=0.79 (+2.9%)** **R_N=0.967 (+0.7%)**	**P=0.069 (+4.5%)** R=0.814 (-5.1%) **$F_{0.5}$=0.082 (+5.1%)** **P_N=0.797 (+4.5%)** **R_N=0.97 (+0.5%)**
BB	P=0.045 (+0.0%) R=0.94 (+1.4%) $F_{0.5}$=0.054 (+0.0%) P_N=0.785 (+2.3%) R_N=0.959 (-0.1%)	P=0.066 (+0.0%) R=0.882 (+2.7%) $F_{0.5}$=0.078 (+0.0%) P_N=0.785 (+3.0%) R_N=0.963 (-0.2%)
BS	P=0.045 (+0.0%) R=0.94 (+1.4%) $F_{0.5}$=0.054 (+0.0%) P_N=0.784 (+2.2%) R_N=0.961 (+0.1%)	P=0.066 (+0.0%) R=0.882 (+2.7%) $F_{0.5}$=0.078 (+0.0%) P_N=0.786 (+3.1%) R_N=0.965 (+0.0%)

Table 6.4 Retrieval Effectiveness (BSIZE=30)

	BSIZE=30	
	Stem	No Stem
ST	P=0.045	P=0.066
	R=0.927	R=0.858
	$F_{0.5}$=0.054	$F_{0.5}$=0.078
	P_N=0.767	P_N=0.762
	R_N=0.96	R_N=0.965
BR	P=0.045 (+0.0%)	P=0.066 (+0.0%)
	R=0.936 (+0.9%)	R=0.873 (+1.7%)
	$F_{0.5}$=0.054 (+0.0%)	$F_{0.5}$=0.078 (+0.0%)
	P_N=0.775 (+1.0%)	P_N=0.775 (+1.7%)
	R_N=0.959 (-0.1%)	R_N=0.964 (-0.1%)
RFA	**P=0.046 (+2.2%)**	**P=0.068 (+3.0%)**
	R=0.894 (-3.5%)	R=0.818 (-4.6%)
	$F_{0.5}$=0.055 (+1.8%)	**$F_{0.5}$=0.08 (+2.5%)**
	P_N=0.779 (+1.5%)	**P_N=0.783 (+2.7%)**
	R_N=0.964 (+0.4%)	**R_N=0.969 (+0.4%)**
BB	P=0.045 (+0.0%)	P=0.066 (+0.0%)
	R=0.935 (+0.8%)	R=0.873 (+1.7%)
	$F_{0.5}$=0.054 (+0.0%)	$F_{0.5}$=0.078 (+0.0%)
	P_N=0.774 (+0.9%)	P_N=0.773 (+1.4%)
	R_N=0.959 (-0.1%)	R_N=0.963 (-0.2%)
BS	P=0.045 (+0.0%)	P=0.066 (+0.0%)
	R=0.936 (+0.9%)	R=0.873 (+1.7%)
	$F_{0.5}$=0.054 (+0.0%)	$F_{0.5}$=0.078 (+0.0%)
	P_N=0.775 (+1.0%)	P_N=0.775 (+1.7%)
	R_N=0.96 (+0.0%)	R_N=0.965 (+0.0%)

6.2.1 Precision – P

RFA systems consistently yield better overall precision P, by more than 4% in the BSIZE=50 case, and by more than 2% in the BSIZE=30 case, than all other systems (ST, BR, BB and BS). This is to say that RFA systems are able to perform better than all other systems, while using a substantially smaller size document representation space.

6.2.2 Recall – R

As expected the size reduction influences the overall recall R. For all non-RFA systems, recall increases with 1.4% in the BSIZE=50/Stem case, with 2.7% in BSIZE=50/No-Stem case, with 0.9% in the BSIZE=30/Stem case, and with 1.7% in the BSIZE=30/No-Stem case. On the other hand RFA's recall performance drops with 3.7% in the BSIZE=50/Stem, with 5.1% in the BSIZE=50/No-Stem case, with 3.5% the BSIZE=30/Stem case, and with 4.6% in the BSIZE=30/No-Stem case. The slightly recall drop does not represent a major drawback, because modern retrieval systems manage large document collections containing large numbers of relevant documents per topic. Furthermore, the recall drop effect can be mitigated by adjusting the ST_N threshold (see Section 7.1).

6.2.3 F-Measure

As mentioned before, in section 5.4.2, the F-measure (van Rijsbergen, 1979) combines precision and recall together (i.e. harmonic mean). It is characterized by the constant α. For low values of α, precision is more important than recall. When α is 1, precision and recall are equally important. When α is greater than 1, recall is more important than precision. In our case α was set to 0.5, meaning precision was higher weighted than recall. A larger value of the F-measure indicates better retrieval. Tables 6.3 and 6.4

clearly indicate that RFA systems are the only ones yielding an improved F-measure when compared to the ST systems. More precisely, the RFA systems improve the F-measure with 5.5% in the BSIZE=50/Stem case, with 5.1% in the BSIZE=50/No-Stem case, with 1.8% in the BSIZE=30/Stem case, and with 2.5% in the BSIZE=30/No-Stem case.

6.2.4 Normalized Precision – P_N

Normalized precision is a measure that takes into consideration the sequence in which retrieved documents are presented to the user. The results indicate that RFA systems have higher normalized precision levels than all other systems in all other conditions. Under "Stem" conditions, when compared to ST systems, the RFA systems improve normalized precision with 5.5% in the BSIZE=50 case and with 1.8% in the BSIZE=30 case. Similarly, under "No Stem" conditions, the RFA systems improve normalized precision with 5.1% in the BSIZE=50 case and with 2.5% in the BSIZE=30 case. This proves the fact that retrieval systems augmented with the RFA algorithm do a better job at pushing relevant documents towards the top of the returned documents list.

6.2.5 Normalized Recall - R_N

Normalized recall is another measure that takes into consideration the sequence in which retrieved documents are presented to the user. The results indicate that RFA systems have slightly higher normalized recall levels than all other systems in all other conditions. More specific, RFA systems improve normalized recall with 0.7% in the BSIZE=50/Stem case, with 0.5% in the BSIZE=50/No-Stem case, and with 0.4% in both BSIZE=30/Stem, and BSIZE=30/No-Stem cases. This fact again, is a clear indication that retrieval systems augmented with the RFA algorithm do a better job (or at least as good) at presenting relevant documents to searchers.

6.3 Retrieval Effectiveness at the First Three Recall Levels

This section presents an in depth analysis of the retrieval effectiveness at the first three recall levels, for all the systems that were tested.

The results presented throughout the previous section only show the overall retrieval performance of the augmented systems. This section details the retrieval improvement that was observed to be taking place within the first few recall levels. The improvements observed for the rest of the recall levels were not noticeable. Tables 6.5 and 6.6 show the results. In these tables, the column header named RL stands for "recall level." The percentage in each cell represents the relative increase in precision with respect to ST systems. The bold values indicate the best performance for a recall level column. The shaded cells indicate the recall level columns where RFA yielded the best results. The results are averaged over the two evaluation sets.

Table 6.5 Precision at the First Three Recall Levels (BSIZE=50)

BSIZE=50	RL-1 (0.1)	RL-2 (0.2)	RL-3 (0.3)
Stem			
ST	0.58	0.55	0.518
BR	0.811 (+**39.8**%)	0.666 (+21.0%)	0.579 (+11.7%)
RFA	0.788 (+35.8%)	**0.683** **(+24.1%)**	**0.584** **(+12.7%)**
BB	0.8 (+37.9%)	0.671 (+22.0%)	0.584 (+12.7%)
BS	0.807 (+39.1%)	0.653 (+18.7%)	0.571 (+10.2%)
No Stem			
ST	0.556	0.523	0.486
BR	0.813 (+**46.2**%)	0.672 (+28.4%)	0.581 (+19.5%)
RFA	0.781 (+40.4%)	**0.689** **(+31.7%)**	**0.61** **(+25.5%)**
BB	0.811 (+45.8%)	0.683 (+30.5%)	0.585 (+20.3%)
BS	0.816 (+46.7%)	0.662 (+26.5%)	0.575 (+18.3%)

Table 6.6 Precision at the First Three Recall Levels (BSIZE=30)

BSIZE=30	RL-1 (0.1)	RL-2 (0.2)	RL-3 (0.3)
Stem			
ST	0.58	0.55	0.518
BR	0.731 (+26.0%)	0.593 (+7.8%)	0.592 (+**14.2**%)
RFA	**0.748** **(+28.9%)**	**0.601** **(+9.2%)**	0.522 (+0.7%)
BB	0.729 (+25.6%)	0.598 (+8.7%)	0.53 (+2.3%)
BS	0.721 (+24.3%)	0.586 (+6.5%)	0.519 (+0.2%)
No Stem			
ST	0.556	0.523	0.486
BR	0.735 (+32.1%)	0.595 (+13.7%)	0.534 (+**9.8**%)
RFA	0.732 (+31.6%)	**0.621** **(+18.7%)**	0.531 (+9.2%)
BB	0.738 (+**32.7**%)	0.603 (+15.2%)	0.529 (+8.8%)
BS	0.738 (+32.7%)	0.589 (+12.6%)	0.522 (+7.4%)

First, a very important observation is the substantial increase in precision for the first three recall levels, obtained when using any of the tested systems (with the exception of ST). The reason why results presented in Section 6.2 reported a smaller increase in overall precision is the fact that the results were averaged over the 10 recall levels.

The second observation of the analysis emphasizes the ability of the RFA systems to perform better or at least the same as the other systems. The grayed cells in Tables 6.5 and 6.6 indicate the situations when the RFA system had the best performance. For all the other cases, the performance of RFA was very close to the best performance. The critical advantage of RFA systems is the fact that they rely on a substantially smaller size document representation space.

6.4 Document Representation Quality

This section presents the evaluation results for the quality of the document representation. Throughout this section the "index-term quality" notion is referred to as the "index-term perceived-quality," because data was collected using human judgments (see Section 5.4.2). Two distinct analysis threads are conducted.

The first analysis thread, detailed in Section 6.4.1, emphasizes the perceived-quality distribution of the index terms generated by the three systems under scrutiny (ST, BR and RFA), according to three predefined index terms categories. The three categories are: "Low perceived-quality", "Medium perceived-quality" and "High perceived-quality." Using this classification, better systems will tend to generate document representations containing a larger number of index-terms belonging to the

"High perceived-quality" category, and a smaller number of index terms belonging to the "Low perceived-quality" category.

The second analysis thread, detailed in Section 6.4.2 presents a statistical analysis to support the results of the perceived-quality distribution analysis.

6.4.1 Index-Terms Perceived-Quality Distribution Analysis

This section presents the perceived-quality distribution of index terms generated by the three systems under scrutiny (ST, BR and RFA), according to three predefined index-terms categories.

The starting point of the analysis is the overall averaged results depicted in Table 6.7. Each cell in the table indicates the average perceived-quality of all index-terms, for all 50 documents, over all 121 subjects. The values in the parentheses represent the standard deviations from the mean values.

Table 6.7 Overall Average Index-Terms Perceived-Quality

	ST	BR	RFA
Average index term quality	2.63 (1.37)	2.35 (1.33)	2.78 (1.4)

The standard deviations for the three systems are in very close range: 1.37 for the ST system, 1.33 for the BR system, and 1.4 for the RFA system. This implies a close to similar distribution of the data points around the mean values for the three systems. At the same time, the means of the data series for the three systems are different. The BR system has the lowest mean 2.35, followed by the ST system 2.63 and finally the RFA system with 2.78. This evidence suggests that: ***the RFA system generates the document representations with the best index-term perceived-quality distribution.*** This means that document representations generated by the RFA system contain a larger number of

"High perceived-quality" index terms and a smaller number of "Medium" and "Low perceived-quality" index terms.

Further analysis to support the formerly stated claim is conducted by looking at two different index-term perceived-quality data series. The first data series is an index term level data series, while the second one is a document level data series. The first data series comprises of all the tuples of the form **(index_term, document_id, subject_id)** and their corresponding scores. The **document_id** field represents a unique document identifier assigned to each of the 50 documents used in this evaluation. Similarly, the **subject_id** field represents a unique identifier assigned to each individual subject that participated in this evaluation. All tuples of this form are considered to be unique. In other words the three fields of this tuple form a primary key together. For example, if ten different subjects assigned a score to a specific term in one document, that term would have been counted ten times. Analysis results for this data series are presented in Table 6.8 and the corresponding bar-chart in Figure 6.1.

Table 6.8 Index-Terms Perceived-Quality Distribution (Index Term Level)

Index-terms perceived-quality distribution				
	Low (<3.0)	**Medium (=3.0)**	**High (>3.0)**	**Total**
ST	5830 (48.27%)	2841 (23.52%)	3405 (28.19%)	**12076**
BR	6981 (57.75%)	2529 (20.92%)	2578 (21.32%)	**12088**
RFA	5256 (44.13%)	2734 (22.95%)	3919 (32.9%)	**11909**

The "Total" column represents the number of index terms that were judged by all subjects. It is actually the total number of distinct tuples of the form (index_term, document_id, subject_id) that exist in the database. Each cell in the columns "Low", "Medium", and "High" displays the number of index terms out of all the judged index terms (the "Total" column) that belong to that category. These cells also display the corresponding percentage value. There are three categories: "Low perceived quality," "Medium perceived quality," and "High perceived quality." An index term is counted as belonging to one of these categories using the following rule: an index term is counted as belonging to the "Low" category, if it was assigned a score less than 3.0 on the evaluation scale; an index term is counted as belonging to the "Medium" category, if it was assigned a score of 3.0 on the evaluation scale; and an index term is counted as belonging to the "High" category, if it was assigned a score greater than 3.0 on the evaluation scale.

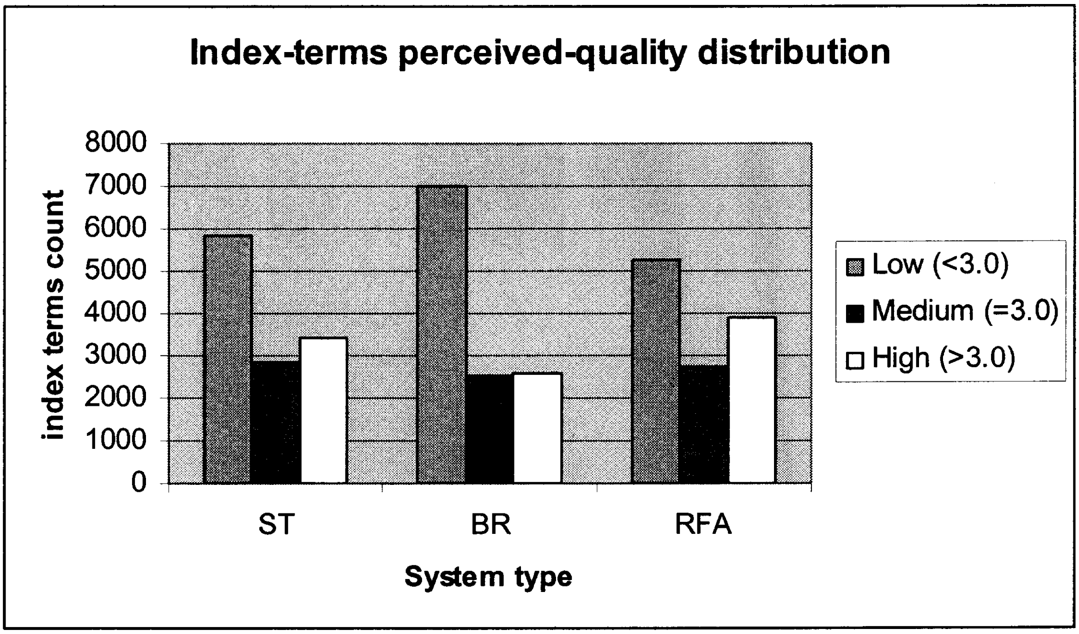


Figure 6.1 Index-Terms Perceived-Quality Distribution (Index Term Level).

It is obvious from Table 6.8 and Figure 6.1 that RFA is able to generate the most number of high perceived-quality index-terms 32.9%, when compared to BR with 21.39% and ST with 28.19%. At the same time, RFA generates less medium and low perceived quality terms than both ST and BR systems. The results support the claim stated above.

The second data series comprises of all the tuples of the form **(index_term, document_id)** and their corresponding scores. All tuples of this form are considered to be unique (together they form a primary key). For example, if ten different subjects assigned a score to a specific term in one document, that term will only be counted once, and its score will be the average of the scores from the ten subjects. As opposed to the first data series, which is an overall view of index-term perceived-quality, this second data series is a document level view of the index-term perceived-quality. Analysis results for this data series are presented in Table 6.9 and the corresponding bar-chart in Figure 6.2.

Table 6.9 Index Terms Perceived-Quality Distribution (Document Level)

Index terms perceived quality distribution				
	Low (<3.0)	**Medium (=3.0)**	**High (>3.0)**	**Total**
ST	680 (68.13%)	46 (4.6%)	272 (27.25%)	**998**
BR	810 (81.08%)	26 (2.6%)	163 (16.31%)	**999**
RFA	573 (58.17%)	29 (2.94%)	383 (38.88%)	**985**

Similarly, all terms from this distribution are assigned to the three categories (“Low,” ”Medium,” and ”High”) in the same manner as for the first distribution. The only difference is that in this case, the index terms are classified based on their average perceived-quality.

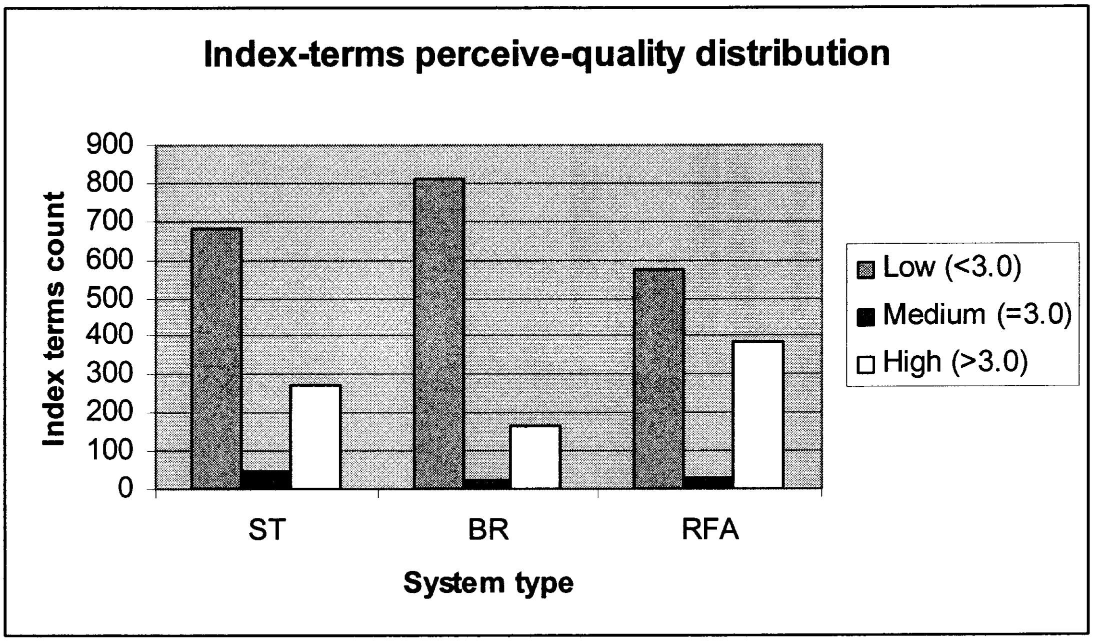


Figure 6.2 Index Terms Perceived-Quality Distribution (Document Level).

Analysis results from the second data series show that the RFA system is able to generate more high perceived-quality index-terms than both ST and BR systems. At the same time, the number of low quality and medium perceived-quality index-terms generated by the RFA system is less than both ST and RFA systems (with the exception of medium quality index terms for BR system (see Table 6.9). The results are consistent and support the claim made at the beginning of the analysis section. It is very interesting to notice that on average, the ST system generates higher perceived-quality index-terms than the BR system. This phenomenon might be attributed to the instability of the BR document representations (i.e. highly oscillating weights), presented in Section 6.5.

The results of the overall analysis, index term level analysis, and document level analysis on the perceived-quality of the index terms generated by the three systems (ST, BR, and RFA) show that: *the document representations generated by the RFA system have the best index-term perceived-quality distribution*. This implies that the document representations generated by the RFA system received the best perceived-quality scores.

6.4.2 Index-Terms Perceived-Quality Statistical Analysis

This section presents a statistical analysis to support the results from the previous section. The dependent variable in the analysis is of ordinal type (see Section 5.4.2). A pair wise system comparison was conducted in order to compare the means of the perceived-quality distributions. The analyzed variable was the index-term perceived-quality score assigned by the judges. Mann-Whitney tests were used in order to investigate if there were statistically significant differences between the mean scores of each pair of the three systems (ST, BR, and RFA). Mann-Whitney is the non-parametric version of the independent samples t-test. This test does not assume that the dependent variable is normally distributed. The only assumption is that the analyzed variable is of ordinal type. The null hypothesis (H0) was that there is no statistically significant difference between the perceived-quality of the index terms generated by any pair of two systems. Results are presented in Table 6.10.

Table 6.10 Mann-Whitney Test Results

Test No.	System 1 (mean score)	System 2 (mean score)	Prob > |z|
1	BR(2.357)	RFA(2.777)	0.0000
2	BR (2.357)	ST (2.639)	0.0000
3	RFA (2.777)	ST (2.639)	0.0000

The results show the following: (a) The mean of the RFA system's scores is significantly higher then the mean of the BR system; (b) The mean of the ST system's scores is significantly higher then the mean of the BR system; (c) The mean of the RFA system's scores is significantly higher then the mean of the ST system. The conclusion of the test is that, overall, the perceived-quality distribution of the index-terms generated by the RFA system is the best among the three systems under scrutiny.

The univariate analysis above did not take into account the possibility that other characteristics may influence the differences between the three systems (e.g. judges' familiarity with the "keyword" concept). Multiple linear regression analyses were performed to investigate if the statistically significant differences remain after controlling for differences in other characteristics. Table 6.11 details the regression specifications and the results. Each cell in the table reports the regression coefficient, and in between parentheses the probability that the coefficient is significantly different from zero (P>|t|). The explanatory variables of main interest are the categorical Sys_BR (taking the value 1 if the score is for the BR system and 0 otherwise), and Sys RFA (taking the value 1 if the score is for the RFA system and 0 otherwise). The reference system is ST, and the two key variables will show whether, on average, the scores for each system are significantly different than the scores from ST, independent of differences in other characteristics. Other factors controlled for in the regressions are female-vs-male (taking the value 1 if the judge was female and 0 otherwise); each level of familiarity compared with the highest; education level (taking the value 1 if undergraduate, and 0 if graduate); categorical variables for each document to control for unobservable individual characteristics of each document (Document_Id in Table 6.11); and categorical variables for each judge to control for unobserved individual characteristics other than sex and education level (Subject_Id in Table 6.11). The results for the control variables "Document_Id" and "Subject_Id" in Regression 3 are reported in APPENDIX E.

Table 6.11 Regression Results

	Regression 1	Regression 2	Regression3
Sys_BR	-.282 (0.000)	-.282 (0.000)	-.282 (0.000)
Sys_RFA	.137 (0.000)	.138 (0.000)	.141 (0.000)
Female	X	.181 (0.000)	.391 (0.000)
Undergrad	X	.341(0.000)	1.726 (0.000)
Familiarity_1	X	.073 (0.003)	-1.800 (0.000)
Familiarity_2	X	.107 (0.000)	-.375 (0.000)
Familiarity_3	X	-.229 (0.000)	-1.405 (0.000)
Familiarity_4	X	.007 (0.751)	-1.070 (0.000)
Document_Id	X	X	APPENDIX E
Subject_Id	X	X	APPENDIX E

The first regression, used as a baseline, shows that not taking into account other factors, RFA scores are on average 14% higher and BR scores are 28% lower when compared with the ST system (p<0.01 for both coefficients). The coefficients of interest remain virtually the same in the richer specifications reported in the last two columns of Table 6.11 (Regression 2 and Regression 3) even if some of the other control variables have a statistically significant effect on the difference in the perceived-quality. T-tests in all three specifications indicate that the coefficients of RFA and BR are statistically different (p<0.01). The key result of the regressions is robust to various specifications (i.e. factors other than the system type). The final conclusion is that: *on average, the document representations generated by the RFA system received the best perceived-quality scores.*

A last analysis was performed to see if there is a correlation between the weights automatically generated by a system and the corresponding average perceived-quality scores. For this the Kolmogorov-Smirnov test (KS) was used to test the similarity of two distributions. KS makes no assumptions about the distribution of the data. This test is investigating if any of the systems (ST, BR and RFA) generate weights that are similar to their average perceived-quality scores assigned by judges. The results indicated no statistically significant results for any of the three systems, meaning that there is no significant correlation between the automatically generated weight of a term and the average perceived-quality of the term. However, the measures compared by this test are not entirely meaningful and/or comparable: on one hand we have a list of automatically generated term weights that are continuous values between 0 and 1; and on the other hand we have a list of normalized (divided by 5) average perceived-quality scores. A more meaningful statistical test would have been possible if the perceived-quality data points would have been rank-ordered by the judges. This kind of test is proposed as a future research objective. Such searcher-centric a study will explore the degree to which an automatic indexing algorithm generates weights that are correlated with the searcher's perceived quality.

6.4.3 Subjects Agreement Analysis

This section presents the subject agreement analysis, a necessary step in any experiment involving human judgment. The scale used during the evaluation is a numerically-coded ordinal scale (5-point Likert scale). The best agreement measure for this type of scale is Kendall Coefficient of Concordance (Siegel and Castellan, 1998), denoted with W. The value of W ranges from 0 to 1. A value of 0 indicates chance agreement. A value of 1 indicates complete agreement between subjects. Our data indicates a mean value of 0.388 ($p<.0001$) for W over all 50 documents. The detailed per-document W values are presented in Table 6.12.

Table 6.12 Agreement Between Raters/Judges (Overall)

Topic No.	Kendall (W)	Topic No.	Kendall (W)
1	0.40511	**26**	0.38051
2	0.50337	**27**	0.33417
3	0.52307	**28**	0.44384
4	0.47613	**29**	0.27689
5	0.36305	**30**	0.35221
6	0.35061	**31**	0.53503
7	0.40136	**32**	0.41691
8	0.37318	**33**	0.46304
9	0.41369	**34**	0.45535
10	0.41854	**35**	0.42148
11	0.40003	**36**	0.51292
12	0.53609	**37**	0.34893
13	0.45516	**38**	0.38919
14	0.38032	**39**	0.52491
15	0.4131	**40**	0.48578
16	0.44401	**41**	0.31459
17	0.38522	**42**	0.30859
18	0.44256	**43**	0.28577
19	0.39617	**44**	0.24308
20	0.37412	**45**	0.29423
21	0.29888	**46**	0.275
22	0.32863	**47**	0.28217
23	0.29328	**48**	0.33326
24	0.35904	**49**	0.35225
25	0.35958	**50**	0.30465
	Mean(W)= 0.389		**(p<.0001)**

Some of the factors that influence the level of agreement between the judges are the following: (a) the perceived-quality of an index term is a highly subjective measure; and (b) the familiarity of the judges with the "keyword concept." While nothing can be done to improve the subjective nature of the measure (perceived quality), it is possible to explore in more detail the agreement between those judges that reported a high familiarity with the "keyword" concept. To do this, all judgments corresponding to judges that in the background questionnaire reported to be "not familiar at all," "not familiar," or "neutral familiar" with the "keyword" concept were removed.

Table 6.13 Agreement Between Raters/Judges (High Familiarity)

Topic No.	Kendall (W)	Topic No.	Kendall (W)
1	0.45672	26	0.47151
2	0.56385	27	0.35733
3	0.61329	28	0.53949
4	0.58155	29	0.33439
5	0.42454	30	0.50174
6	0.37383	31	0.67639
7	0.35679	32	0.64303
8	0.33626	33	0.65694
9	0.42371	34	0.5969
10	0.37024	35	0.6753
11	0.49214	36	0.50517
12	0.61366	37	0.32436
13	0.58585	38	0.45188
14	0.48692	39	0.56187
15	0.51066	40	0.5244
16	0.52039	41	0.43235
17	0.42841	42	0.3424
18	0.50638	43	0.39363
19	0.49875	44	0.35864
20	0.43683	45	0.33511
21	0.34397	46	0.48371 (0.5)
22	0.38272	47	0.61372 (0.06)
23	0.37105	48	0.60212 (0.09)
24	0.42969	49	0.53070 (0.3)
25	0.39881	50	0.51424 (0.4)
	Mean(W)=0.47		(p<.0001)

The results show an improved agreement mean value of 0.47 (p<.0001; or other p that can be seen in table below). The detailed per-document W values are presented in Table 6.13.

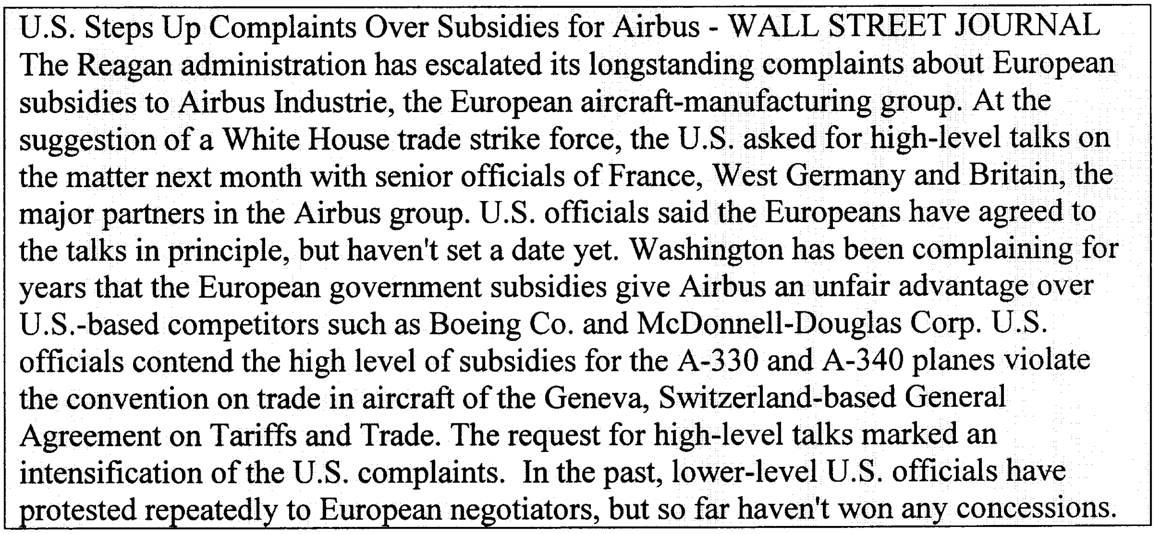
A mean value of agreement between 0.4 and 0.6 usually denotes a moderate agreement. Previous research in human-computer interaction has shown that the probability of two persons having the same understanding of a concept is at most 0.2 (Furnas et al. 1987). Under such circumstances, due to the subjective nature of the measure (perceived-quality), our agreement value represents a good result, having in mind that the RFA algorithm works by generalizing across all users. In other words, the

RFA algorithm is looking exactly for that small number of high quality index terms, on which there is general quality agreement.

6.4.4 Factors Influencing the Perceived-Quality

This section presents three major factors that positively influence the perceived-quality of the document representations generated by the RFA algorithm.

The first major factor contributing to the higher quality of the document representations generated by the RFA systems is the Ordered Terms Pairing heuristic, described throughout Section 4.3. It leverages the discovery of N-worded high quality index terms (in this case N=2). To better illustrate this, consider TREC document "WSJ861222-0013" that was affected by the RFA algorithm.



U.S. Steps Up Complaints Over Subsidies for Airbus - WALL STREET JOURNAL
The Reagan administration has escalated its longstanding complaints about European subsidies to Airbus Industrie, the European aircraft-manufacturing group. At the suggestion of a White House trade strike force, the U.S. asked for high-level talks on the matter next month with senior officials of France, West Germany and Britain, the major partners in the Airbus group. U.S. officials said the Europeans have agreed to the talks in principle, but haven't set a date yet. Washington has been complaining for years that the European government subsidies give Airbus an unfair advantage over U.S.-based competitors such as Boeing Co. and McDonnell-Douglas Corp. U.S. officials contend the high level of subsidies for the A-330 and A-340 planes violate the convention on trade in aircraft of the Geneva, Switzerland-based General Agreement on Tariffs and Trade. The request for high-level talks marked an intensification of the U.S. complaints. In the past, lower-level U.S. officials have protested repeatedly to European negotiators, but so far haven't won any concessions.

Figure 6.3 TREC Document WSJ861222-0013.

Table 6.14 shows the top 20 index terms generated by ST, BR and RFA systems.

Table 6.14 Index-Terms for TREC Document WSJ861222-0013

ST	BR	RFA
airbus	airbus	airbus
subsidies	subsidies	subsidies
complaints	industrie	trade
european	trade	european
intensification	dispute	dispute
talks	boeing	government
level	disputes	**airbus-subsidies**
haven	government	industrie
aircraft	assistance	**trade-dispute**
trade	mcdonnell	aircraft
escalated	governments	issues
complaining	douglas	**united-states**
officials	production	united
protested	gotten	states
longstanding	does	**airbus-trade**
industrie	charge	**subsidy-battle**
europeans	airplanes	subsidy
tariffs	industry	europe
high	subsidies	battle
mcdonnell	trades	**airbus-subsidy**

In the RFA system's case, the two-worded index terms introduced by the OTP heuristic are: "airbus subsidies," "trade dispute," "united states," "airbus trade," "subsidy battle," and "airbus subsidy". With the exception of "airbus trade" all other index terms are of high quality and are more efficient in conveying the topic(s) of the document. At the same time, it should be noted that most of them do not even occur throughout the document's text.

The second major factor is the ability of the RFA system to filter out unwanted terms. An unwanted term is a term that has nothing to do with the topic(s) of the document that it represents. In other words, such a term does not imply anything about the topic(s) described by the document with which it is associated. The index term list presented in Table 6.14 contains several unwanted index terms. In the case of the ST system, some of the unwanted terms are: *talks, level, complaining, longstanding* and *high*. In the case of the BR system, some of the unwanted terms are: *assistance, gotten, does,* and *charge*. In the case of the RFA system, some of the unwanted terms are: *industrie,* and *issues*.

The third major factor is the ability of the RFA algorithm to identify new index terms from queries. By this, the RFA algorithm tackles the word mismatch problem, which was presented as motivation in Section 1.2. In the example presented in Table 6.14, the terms "united" and "states," are high quality index terms that do not occur throughout the document. The document's text contains only "U." (i.e. United) and "S." (i.e. States). Note that none of the other two representations corresponding to ST and BR systems contain the term "united" or "states."

6.5 The Learning Function

This section presents the evaluation results of the document representation modification function, also called the learning function. The quality of a learning function relies on its ability to rapidly learn the weight of a term-document pair, and to keep the future alterations close to this value. In other words, a poor quality learning function will generate far apart values for the same weight of a term-document pair, at different points in time. The average mean square error (MSE) for all terms from 50 randomly selected documents was calculated. The reference point for MSE calculation was the last term's weight value, considered to be the learned term weight. Results averaged over the two evaluation sets are presented in Table 6.15 and 6.16. Overall, the RFA algorithm provides a much smoother learning function. Its MSE is considerably lower (close to 10-fold) than any of the other systems. This means that weights of terms do not oscillate up and down with high amplitudes.

Table 6.15 Learning Function MSE (BSIZE=50)

BSIZE=50	Stem	No Stem
RFA	**0.00065**	**0.00061**
BR	0.00625	0.00572
BB	0.00566	0.00458
BS	0.01928	0.02124

Table 6.16 Learning Function MSE (BSIZE=30)

BSIZE=30	Stem	No Stem
RFA	**0.00073**	**0.0007**
BR	0.00627	0.0058
BB	0.00533	0.00491
BS	0.02172	0.02204

For exemplification, consider Figure 6.4. It shows the learning history for the term "government" with respect to one of the documents in which it occurs (in our collection: docid=3876), for both BR and RFA systems. In the case of the RFA system the "government" is a "Type C" term. BR_ref and RFA_ref in Figure 6.4 represent the actual final learned weights for the term "government". These weights are used as references for calculating the MSE. It is noticeable that in the BR learning function case, weights rapidly evolve in a positive or negative direction. The evolving direction is determined by the appearance pattern of term "government" in the sequence of relevance assessments. If the terms appear in many consecutive relevance assessments, its weight will rapidly increase. If the terms do not appear in many consecutive relevance assessments, its weight will rapidly decrease. For the case in Figure 6.4, MSE(BR)=0.026 while MSE(RFA)=0.009.

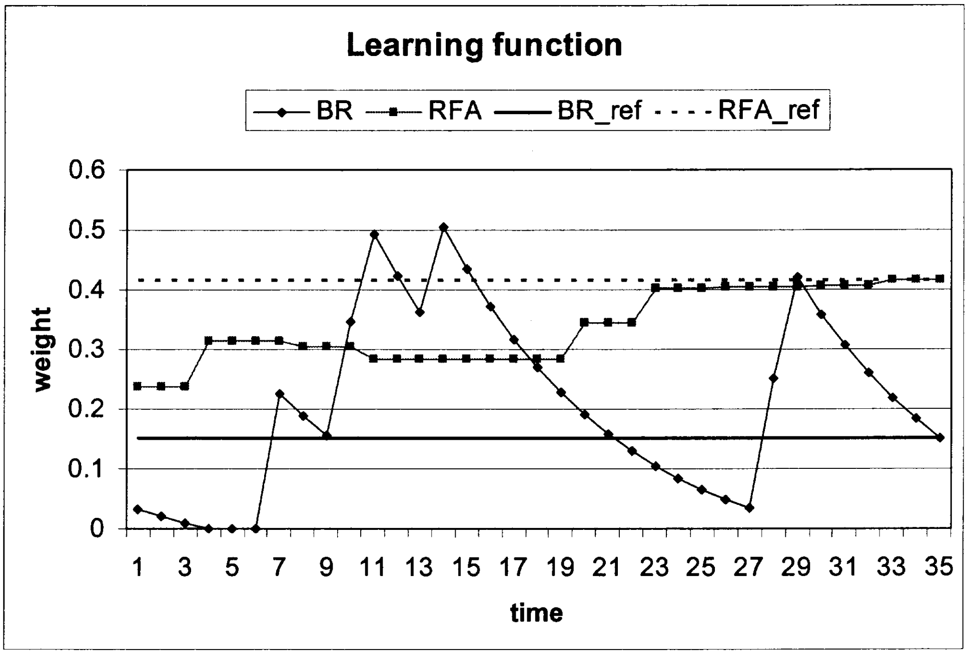


Figure 6.4 Weight Learning History for "government."

Figure 6.5 presents the learning history for the term "wildlife" with respect to one of the documents in which it occurs (in our collection: docid=9278), for both BR and RFA systems. In the case of the RFA system, "wildlife" is a "Type R" term. BR_ref and RFA_ref in Figure 6.5 represent the learned weights for the term "wildlife." These weights are used as references for calculating the MSE. For this case, MSE(BR)=0.04 while MSE(RFA)=0.03. It is noticeable that for terms that appear very often within the relevance feedback history, like "wildlife," the learning function of the BR systems has higher quality than for the terms that do not appear that often (e.g. "government" in Figure 6.4).

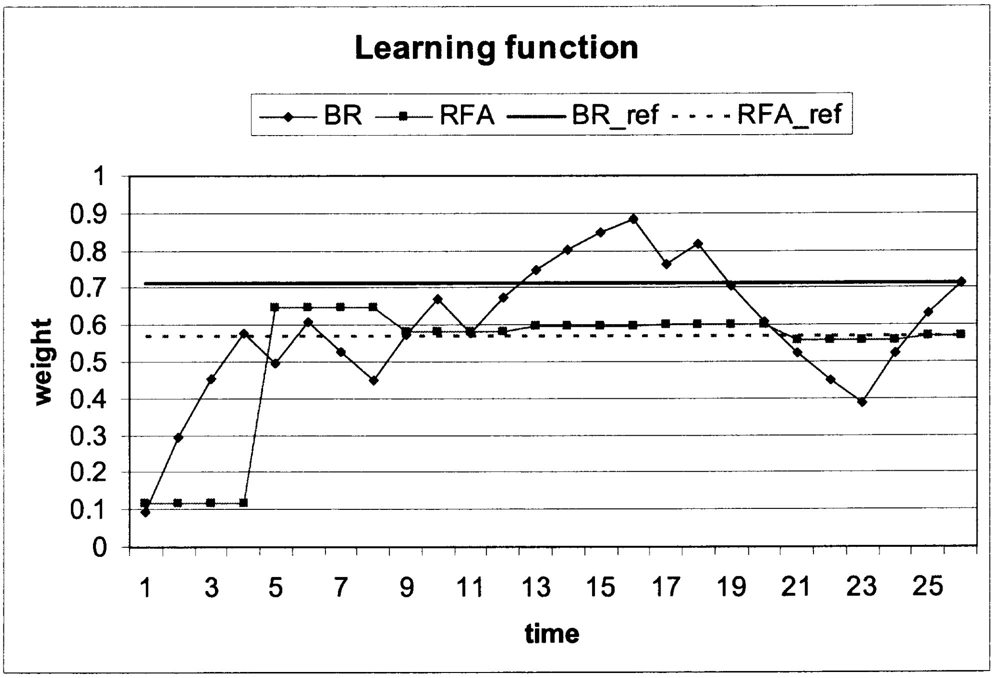


Figure 6.5 Weight Learning History for "wildlife."

A direct effect of the learning functions' characteristics is visible in Tables 6.5 and 6.6. One can notice that RFA systems have better precision for the second and third recall levels, while BR systems have better precision for the first recall level. The first recall level comprises of documents containing terms with the highest support throughout the relevance feedback history. One such example is "wildlife" (See Figure 6.5). For these terms, the BR learning function is smoother, closer to the quality of the RFA learning function. This, combined with the slightly better recall in the case of BR systems, make the BR systems have a slightly better precision for the first recall level. This effect can be mitigated by adjusting the "Type C" threshold (ST_N), so that RFA systems allow more index terms, therefore closing the recall gap between BR and RFA systems. A more detailed discussion on choosing thresholds is presented in Section 7.1.

The second and third recall levels comprise documents containing terms with medium support throughout the relevance support corpus. In this case, the BR learning function generates oscillating term weights (e.g. "government" in Figure 6.4), depending on how terms occur throughout the analyzed relevance assessments. The RFA learning function, driven by term support variance, generates steadier weights that smoothly converge to the final learned value.

6.6 A Comparative Analysis

This section presents a "comparative" analysis with the two systems described in (Piwowarski, 2000) and (Kemp and Ramamohanarao, 2002). These comparisons might not be significant since the evaluations were performed on different test collections, using systems with different algorithms. Also, the RFA algorithm is the only one that uses a size reduction mechanism. Table 6.17 shows the performance of the DIFF$_{10}$ algorithm presented in (Piwowarski, 2000) as compared to RFA(30) and RFA (50) in the no stemming condition. Results for DIFF$_{10}$ were extracted from the formerly cited paper.

Table 6.17 RFA vs. DIFF$_{10}$

	DIFF$_{10}$		**RFA(50)**		**RFA(30)**	
R-level	**Cranfield**	**CISI**	**Set 1**	**Set 2**	**Set 1**	**Set 2**
0.1	+3.36%	-4.34%	+39.33%	**+42.0%**	+29.89%	+33.82%
0.2	-	-	+29.68%	**+33.99%**	+16.51%	+21.14%
0.3	-	-	+22.35%	**+28.72%**	+8.58%	+10.21%
0.4	**+24.6%**	+4.08%	+16.84%	+19.4%	+5.83%	+6.94%
0.7	+37.3%	**+63.63%**	+7.2%	+7.77%	-1.25%	+0.33%
1.0	**+42.5%**	+13.51%	+5.2%	+9.33%	+1.04%	+7.14%

One can notice that the RFA algorithm tends to substantially enhance precision during the first few recall levels. At the same time DIFF$_{10}$ substantially enhances precision during the last few recall levels. As an example, this effect is depicted in Figure 6.6 by fitting the data points for DIFF$_{10}$ on Cranfield and RFA(50) on Set 2.

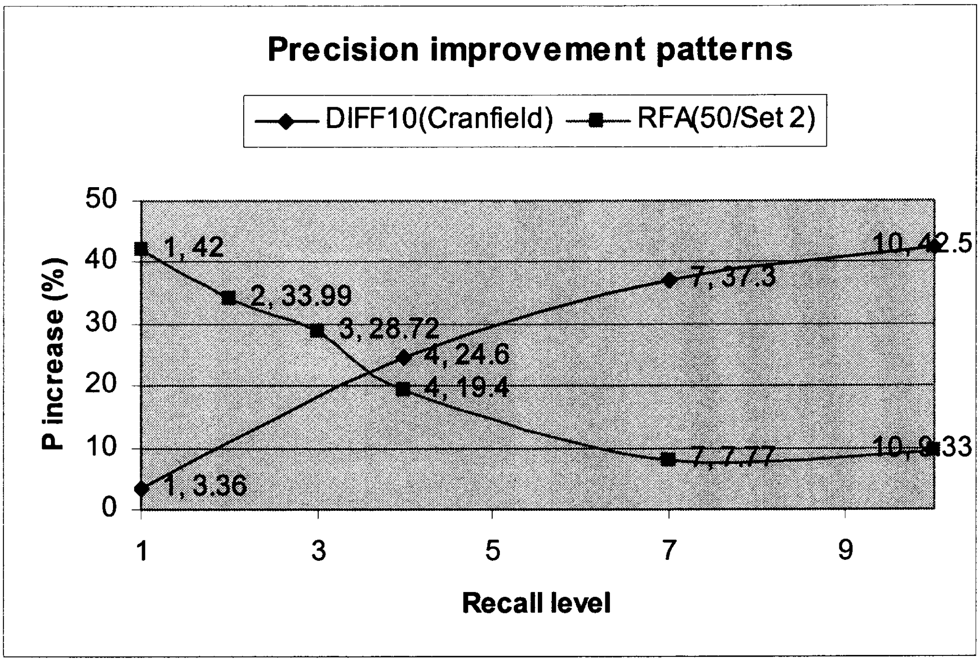


Figure 6.6 Precision Improvement Patterns.

The precision improvement pattern yielded by the RFA algorithm is better because the first few recall levels are more important from a searcher's point of view. They contain the documents that are more probable to be browsed by the searcher.

Table 6.18 shows the performance of the DT3 algorithm presented in (Kemp and Ramamohanarao, 2002) as compared to RFA(30) and RFA (50) in the no stemming condition. Since the only measure available for DT3 is precision(10), defined in Section 3.2, the comparison is presented using the closest available measure from RFA: precision at the first recall level. The results suggest a better performance of the RFA algorithm.

Table 6.18 RFA vs. DT3

DT3 Metric: precision(10)	RFA(50) Metric: avg(P) at R_{level}=0.1	RFA(30) Metric: avg(P) at R_{level}=0.1
+6.0%	+28.8%	+24.0%

CHAPTER 7

DISCUSSION

7.1 Choosing Index Term Thresholds for RFA Algorithm

This section presents a short discussion on choosing the index term thresholds for the RFA algorithm. As formerly presented in Section 4.4 (and Figure 4.4), the RFA algorithm uses two thresholds to classify its index terms into three categories: Type-R, Type-C and Type-N. The two thresholds are ST_N and ST_R. Results presented in this paper correspond to the following threshold settings: ST_R=0.3 and ST_N=0.05. This setting was found "optimal" after several other settings were tested. The most important factor in choosing the optimal setting was the extent of size reduction. In other words, the optimal setting for the RFA algorithm is the one that yields the biggest size reduction, without affecting the overall retrieval effectiveness.

The reduction extent is controlled by the ST_N threshold. Higher values for ST_N correspond to smaller size document representations. As expected, one important effect of maximizing the size reduction is the slight recall drop observed for RFA systems when compared to the other systems (see Section 6.2.2). The slight recall drop due to size reduction maximization settings (i.e. ST_N threshold), causes a lower overall precision for the first recall level (i.e. characterized by documents containing high support index terms). Increasing the value of the ST_N threshold will cause the recall to improve.

Therefore, choosing the right value for ST_N should follow the following guidelines:

- If the concern is to obtain a very small size document representation, ST_N should be set to higher values, without significantly affecting overall retrieval parameters (i.e. precision).
- If the concern is to minimize the recall drop throughout the first three recall levels, ST_N should be set to a lower value. By this, the size of the document representation space will increase, but it will still be smaller than non RFA systems.

The ST_R threshold does not affect the retrieval performance of the RFA system. Its purpose is to separate high quality index terms from medium quality index terms. Still, setting this threshold is very important, because this affects the number of Type-R index terms. Section 7.2 describes in more detail how Type-R index terms can be used as meta-data to complement returned documents. Choosing the right value for ST_N depends on the number of Type-R index terms needed to be returned as complementary meta-data along with retrieved documents.

7.2 Index Terms as Meta-Data

This section presents a discussion about the usefulness of the index terms as meta-data. The three categories are "Type-R," "Type-C," and "Type-N." As previously mentioned, "Type-N" index terms are unwanted index terms. Their influence on retrieval effectiveness is close to zero. "Type-C" index terms are medium quality index terms. They are necessary in order to assure high retrieval effectiveness. Eliminating "Type-C" index terms from consideration would end up in negatively affecting retrieval effectiveness, especially the recall. "Type-R" index terms are high quality index terms. The usefulness of index terms as meta-data comes into play when this data is returned along with the document description itself. More precisely, the retrieval system would be able to return the set of "Type-R" index terms along with the document itself. There are many potential applications for such an added value feature. Since the recent substantial growth of the Internet, considerable research efforts are focused on techniques for hierarchical organization of web returned documents. These techniques use document classification (Dumais and Chen, 2000; Oh et al. 2000; Furnkranz, 1999; Govert et al. 1999; Attardi et al. 1998; Chakrabarti et al. 1998a; Chakrabarti et al. 1998b; Joachims, 1998), concept hierarchy generation (Lawrie et al. 2001; Sanderson and Croft, 1999; Hearst, 1998; Woods, 1997; Forsyth and Rada, 1986), document clustering (Zamir and Etzioni, 1999), or combinations (Bot et al. 2004). Some of them process the whole documents while others only process document snippets returned by the retrieval system. One way or another, all the previously mentioned techniques need to extract quality terms (also called features) from each of the returned documents in order to be able to work. This, in turn, affects efficiency. At the same time the quality of the extracted features is usually not satisfactory. Other possible applications for index terms as meta-data are: rich browsing interfaces, text mining applications, and digital libraries retrieval systems.

For all the RFA systems evaluated in this paper, ST_N was set to 0.3. Tables 7.1 and 7.2 show the index terms broken down with respect to their category. The bolded values in each cell indicate the average number of index terms of that type per document. For example, on average, the number of index terms returned as meta-data along with every document, would be 7.63 in the BSIZE=50/No-Stem condition.

Table 7.1 Term Type Proportions (BSIZE=50)

BSIZE=50	Type R	Type C	Type N
Stem	4.12% **7.81**	6.27% **11.88**	89.6% **169.88**
No Stem	3.65% **7.63**	5.72% **11.96**	90.6% **189.51**

Table 7.2 Term Type Proportions (BSIZE=30)

BSIZE=30	Type R	Type C	Type N
Stem	4.58% **8.03**	5.97% **10.46**	89.44% **156.83**
No Stem	4.08% **7.9**	5.36% **10.39**	90.55% **175.53**

In conclusion, having high quality index terms (features) served by the retrieval system as complementary meta-data associated to each returned document, would be a valuable feature.

CHAPTER 8

LIMITATIONS AND FUTURE RESEARCH

This section presents the limitations of the current study and proposed future research directions to tackle them. Four major issues are analyzed: relevance feedback capture, relevance feedback coverage, an extended OTP heuristic, and multi topic documents.

8.1 Capturing Relevance Feedback

Capturing the relevance feedback is a difficult task. Most searchers would not take the time to provide the necessary explicit feedback. Under these circumstances, an algorithm like RFA would seem not feasible. However, research has shown that it is possible to obtain implicit feedback by analyzing the searcher behavior in real time (Kelly and Belkin, 2001; White et al. 2002). Most searchers follow similar patterns when they find a document relevant to their information need. Such patterns include: saving the document, printing the document, saving a picture from the document, or placing document fragments onto the clipboard (cut and paste). More complex user behavior patterns take into consideration such activities as the time spent reading a document, the scroll pattern or the click-stream sequence. A more complete description of implicit feedback capture and use can be found in (Kelly and Teevan, 2003). Today, web traffic between searchers and retrieval systems is very large. With appropriate tools, it is possible to obtain large amounts of relevance feedback. All these issues make RFA and other algorithms similar to RFA good candidates for building new retrieval models that will exploit to a greater extent the user behavior latent information.

Another feasible alternative for capturing implicit relevance feedback is the web log. A web log explicitly records every action a user performs, as part of a search session. Modern search technologies use web logs as their primary relevance feedback resource. One possible way to approximate a relevance assessment from a web log is to consider the end of the search click stream for certain query as the relevant document for that query. The proposed future research direction, related to the relevance feedback capture issue, consists of the augmentation of the current algorithm with a more realistic RF capture capability. This will include both behavioral and web log based relevance feedback capturing techniques. The augmented version will be tested on top of a real retrieval system.

8.2 Relevance Feedback Coverage

The relevance feedback coverage associated with the query Q is defined as the proportion of Q's returned relevant documents that are actually assessed as being relevant. The bigger the relevance feedback coverage, the more uniform the document space modification is. The effect of limited coverage is that only a small number of document vectors will be affected by the RFA algorithm with respect to the corresponding query. This creates a gap between the returned documents judged relevant more often and other returned documents not judged relevant as often. The documents that generate more relevance assessments are those returned within the first 2 or 3 result pages (top 20-30 hits), because they are most likely to be browsed by the searcher. The other remaining documents are less likely to be browsed. The current version of the RFA algorithm does not implement any technique to improve relevance feedback coverage, but there are ways to tackle this issue.

One technique to tackle limited relevance feedback coverage is the use of web logs. In the case of high traffic retrieval systems, web logs are very large and they cover the entire document space. Therefore, enough relevance assessments can be inferred for every document. As a result the limited coverage effect is minimized.

A second feasible technique is to use document clustering to identify documents that were not involved in the relevance assessment, but are similar to those that were. Suppose that given a query Q the retrieval system returns a set of documents S_{RET}. Furthermore, suppose the following relevance assessment is available: (Q, $\{S_{RF}\}$). Here, Q is the query vector, and S_{RF} is the set of all documents from S_{RET} assessed relevant to Q. The first step is to build a representative document vector (D_R) from all the document vectors in S_{RF}. This can be done by selecting and aggregating the most important index-terms (based on their weights) from each of the document vectors in S_{RF}. Another way is to select the index-terms that are common to all document vectors in S_{RF}. The second step is to cluster new documents from the set (S_{RET}-S_{RF}) around D_R. By this, new documents similar to those belonging to the relevance assessment are identified.

Another way to tackle relevance feedback coverage is to use similarity measures within the retrieved documents set. Suppose that given a query Q the retrieval system returns a set of documents S_{RET}. Furthermore, suppose that the following relevance assessment is available: (Q, $\{S_{RF}\}$). S_{RF} is the set of all documents from S_{RET} assessed relevant to Q. S_{RF} is a subset of S_{RET}. In general, the cardinality of S_{RET} is much larger that the cardinality of S_{RF}. The first step is to compare each of the documents in S_{RF} with all documents in (S_{RET}-S_{RF}). The comparison is done using vector-based similarity measures (e.g. distance, angle, inner product). The most similar documents found in this way are selected and added to a final documents list S_{NEW}. The second step is to refine S_{NEW} by eliminating all duplicates. In the end, S_{NEW} consists of new documents that are similar to the documents in S_{RF}, and can be used to increase relevance feedback coverage.

The proposed future research direction for this limitation consists of augmenting the current algorithm with an efficient technique (or combination of techniques) that mitigates the relevance feedback coverage effect.

8.3 Extended OTP Heuristic

The Ordered Terms Pairing heuristic, introduced in Section 4.3, only selects adjacent pair of query terms in order to generate additional potential high quality index terms. In this case, no term proximity operators are used. The proximity operator for any pair of terms is defined as the distance (number of terms) between the occurrences of the two terms in a query. Basically, the OTP heuristic is set to work with a fixed proximity, which is set to zero (i.e. adjacent terms). Many times, an occurrence proximity which is different from zero can also be a good indication of a strong conceptual relationship between two terms. Consider the following query for example: "text retrieval and mining" Using the current version of the OTP heuristic the possible new terms that are considered are "text retrieval" and "retrieval mining." That is because OTP is only looking at adjacent pairs of terms from the initial query. It is noticeable that the term "text mining," is a high quality index term. Nevertheless, this term is never investigated by the OTP heuristic because it does not currently support proximity operators. In this case, the proximity between the individual terms “text” and “mining” is one. The proposed future research direction for this limitation consists of augmenting the current version of the OTP heuristic with an adjustable proximity operator. Another limitation of the OTP heuristic is the promotion of certain simple and composite terms that are not noun phrases (e.g. “working,” “placed-south”), because no noun-phrase detection is performed by the current version. Consult APPENDIX D for a comprehensive list of low perceived-quality index terms (including non noun-phrase index terms). In order to eliminate this effect another future research direction that is proposed is the augmentation of the current OTP version with a noun-phrase detection capability.

8.4 Multi Topic Documents

One of the characteristics of many documents from the collection used to evaluate the RFA algorithm is the fact that they are topical focused. A topical focused document does usually hold information about a single topic. In other words, the topic of such a document is very narrow. This is not always true for all document collections, especially for web-based documents collections, where documents usually cover multiple topics. The proposed future research direction for this limitation consists of investigating how multi topic documents affect the outcomes of the RFA algorithm, and how can these effects be mitigated.

Finally, this study explored the degree to which the perceived-quality of the document space generated by the RFA algorithm is better than the perceived-quality document spaces generated by other algorithms. The next step is to explore the degree to which the RFA algorithm generates weights that are correlated with the searchers’ perceived-quality. This is a searcher-centric study that will require the construction of a new evaluation tool able to capture this effect.

CHAPTER 9

CONTRIBUTIONS

The research contributions made to the field of Information Retrieval by this thesis are:

- Revitalizes a useful topic, Document Representation Modification, a potentially valuable technique that was given little attention since its inception. The technique was proven to improve retrieval effectiveness by Friedman et al. (1967), Brauen et al. (1968), Ide (1969), Brauen (1969), Piwowarski (2000), Kemp and Ramamohanarao (2002).

- Presents a novel size reduction technique that is implemented by the RFA algorithm. Many other techniques were previously used to reduce the number of terms to index a document. Stemming for example reduces the size of the representation space by grouping words/terms into synonymy-based equivalence classes. Still, many of these equivalence classes are not content bearing with respect to the topicality of documents. Another acclaimed method is the Latent Semantic Indexing (LSI) developed by Deerwester (1990). This technique performs complex mathematical/statistical analysis on the whole document collection in order to come up with a lower dimensional vector for each document. The technique might be useful but it is inefficient. The complexity is too large and it is not suited for indexing large document collections.

- Presents a novel way of identifying the indexing terms for documents. Good indexing term identification has been a problem since the inception of Information Retrieval. It is hard to find those terms/words in a document (or not in the document) that best characterize that document's topic(s). Previously, almost all techniques were based on automatic indexing by lexicographic analysis. In this case, the importance of a word in a document is solely assessed by computing frequency-based measures normalized in different ways. But there is no proven theory stating that the frequency of a word in a document is related to its importance in that document. That is why many of the terms generated by standard lexicographic analysis are of no use. The RFA algorithm identifies the concepts to represent documents from the users' point of view. The words that are used most of the time by users to describe a document are considered to be concept terms in that document, and they are selected as index terms. The source of evidence for this user-view analysis is the relevance feedback history, accumulated by the retrieval system in special data structures.

- Introduces the "perceived-quality" of an index term. This is a non-traditional measure that estimates the quality of index terms from a semantic perspective rather than from a retrieval effectiveness perspective. In other words a high quality index term is one that is able to suggest the topic of the document and not necessarily one that improves retrieval effectiveness. Perceived-quality can be used to evaluate other applications that are concerned with the relationship between index terms and document topics.

- Simple and efficient algorithm. RFA is an algorithm based on a simple idea. Deriving concept terms from users' point of view is straightforward and it does not need complex mathematical models to be formalized. At the same time, the algorithm is suited for retrieval systems governing large document collections. The overhead generated by implementing RFA on top of a standard VSM retrieval system is minimal.

APPENDIX A

TREC TOPIC 51

Figure A.1 shows a full TREC topic description (in this case TREC Topic 51).

TREC topic no. 51

```
<top>
<head> Tipster Topic Description
<num> Number: 051
<dom> Domain: International Economics

<title> Topic: Airbus Subsidies
<desc> Description:
Document will discuss government assistance to Airbus Industrie, or mention a trade
dispute between Airbus and a U.S. aircraft producer over the issue of subsidies.

<smry> Summary:
Document will discuss government assistance to Airbus Industrie, or mention a trade
dispute between Airbus and a U.S. aircraft producer over the issue of subsidies.

<narr> Narrative:
A relevant document will cite or discuss assistance to Airbus Industrie by the French,
German, British or Spanish government(s), or will discuss a trade dispute between
Airbus or the European governments and a U.S. aircraft producer, most likely Boeing
Co. or McDonnell Douglas Corp., or the U.S. government, over federal subsidies to
Airbus.

<con> Concept(s):
1. Airbus Industrie
2. European aircraft consortium, Messerschmitt-Boelkow-Blohm GmbH, British
   Aerospace PLC, Aerospatiale, Construcciones Aeronauticas S.A.
3. federal subsidies, government assistance, aid, loan, financing
4. trade dispute, trade controversy, trade tension
5. General Agreement on Tariffs and Trade (GATT) aircraft code
6. Trade Policy Review Group (TPRG)
7. complaint, objection
8. retaliation, anti-dumping duty petition, countervailing duty petition, sanctions

<fac> Factor(s):
<def> Definition(s):
</top>
```

Figure A.1 TREC Topic 51.

APPENDIX B

QUERY LIST FOR TREC TOPIC 51

This section lists all the queries generated for the TREC Topic 51, as part of the TREC Query Track efforts.

51 01 Support for or protest of Airbus
51 02 How has Airbus been supported by Europe or protested by the United States?
51 03 recent airbus issues
51 04 Airbus subsidies dispute
51 05 Airbus subsidy battle
51 06 Airbus subsidies dispute
51 07 U.S. Airbus subsidies
51 08 Airbus government subsidies
51 09 Airbus trade dispute
51 10 airbus trade dispute
51 11 government airbus industry
51 12 Airbus government trade
51 13 What are the reactions of American companies to the trade dispute and how the dispute progresses?
51 14 What are the issues being debated regarding complaints against Airbus Industrie?
51 15 News related to the Airbus subsidy battle.
51 16 U.S. and Europe dispute over Airbus subsidies
51 17 Is European government risking trade conflicts over issue of Airbus subsidies?
51 18 Subsidies from European governments and unfair trading practices let Airbus consortium successfully compete with Boeing and McDonnel Douglas, american aircraft producers
51 19 How the Airbus industry dispute is affecting global corporate politics
51 20 what is the u.s. reaction to the airbus' government subsidies?
51 21 How is the United States dealing with other nations in the airbus industry
51 22 What events highlight trade and economic issues regarding Airbus Industrie?
51 23 How is the Airbus business in the world ?
51 24 why did the US put duties on airbus?
51 25 What are issues between the US and Airbus?
51 26 What are some problems the U.S. government has regarding European aircraft manufacturers?
51 27 How are US aircraft makers responding to European Airbus subsidies?
51 28 What is the U.S. doing about Airbus Industrie subsidies?
51 29 U.S. trades and negotiation progresses with subsidizes Airbus.
51 30 What are the subsidies of Airbus

51 31 What kinds of negotiations are taking place with Airbus?
51 32 who is involved in the airbus industrie consortium
51 33 Airbus Subsidies
51 34 Airbus subsidies
51 35 airbus industry
51 36 government subsidies for Airbus
51 37 What government subsidies does Boeing charge Airbus has gotten in the production of airplanes?
51 38 Information about European Airbus subsidies
51 39 airbus subsidies industrie
51 40 airbus trade dispute
51 41 assistance to airbus industrie by governments or trade dispute with Boeing or McDonnell Douglas over subsidies
51 42 Airbus subsidies, disputes
51 43 Airbus Industrie Subsidies

APPENDIX C

PERCEIVED-QUALITY EVALUATION QUESTIONNAIRE

This section presents a sample questionnaire used to measure document-representation perceived-quality.

Document Representation Quality - Evaluation Questionnaire

Section 1: Instructions

This questionnaire aims at evaluating the quality of different document indexes. A document index is composed of a set of document index terms, or keywords. A document **index term** is defined as a word or phrase that is significant in describing the topic of the document. It is not necessary that the index term occur throughout the document. For example, some good index terms for a document describing the process of installing and configuring a Microsoft LAN (Local Area Network) would be: ***Microsoft***, ***network***, *LAN*, ***configuration***, and ***installation***.

To complete this questionnaire, please follow the three steps below:

STEP 1: Carefully read and understand the topics discussed in the document in *Section 2*. The topics in a document are equivalent to the answer of the question: "what is the document about?"

STEP 2: Section 3 lists a set of index terms, displayed in alphabetical order. Carefully browse through all these index terms, and assign each a score from 1 to 5. The score indicates how good you think each index term is, in describing the topic of the document from Section 2. The scores 1,2,3,4,5 have the following meanings:

1=Very Bad	**2=Bad**	**3=Average**	**4=Good**	**5=Very Good**

NOTE 1: some index terms consist of only one word (example: "*LAN*"), while some others consist of two words (example: "***local-network***").

Section 2: The document

Computer crime: ugly secret for business.
By Alexander, Michael.

Computer crime is on the rise and costing US industry billions of dollars according to some estimates, but little is being done to stop it. There are two reasons why computer crime is difficult to handle from a legal perspective: computer crime has never been clearly defined, and businesses report only a small percentage of known crimes because they fear bad publicity. Without consensus on what a computer crime actually is, estimating its magnitude is difficult. Computer crimes are also costly and time consuming to investigate and just do not take precedence over murders according to law officials. Public outcry could direct more attention towards the investigation and prosecution of computer crime cases. Businesses need to help combat the occurrence of such crimes by increasing computer security; the incentive to do so is becoming greater. Topic: Computer Crimes Security Theft of Information Law Enforcement Business.

Section 3: Index Terms List

index term	your score
activity	
aid	
aided	
aided-crime	
alexander	
businesses	
cases	
Cases-computer	
committed	
computer	
computer-aided	
computer-crime	
computer-used	
computers	
computerworld	
consuming	
costing	
crime	
crimes	
difficult	
estimating	
extortion	

forgery	
fraud	
graph	
hackers	
hacking	
illegal	
illegal-computer	
impact	
internationally	
legislation	
magnitude	
murders	
occurrence	
outcry	
percentage	
precedence	
publicity	
recent	
sabotage	
theft	
tool	
ugly	
use	
use-computer	
used	
virus	
worldwide	

APPENDIX D

LOW PERCEIVED-QUALITY RFA INDEX TERMS

Figure D.1 lists all the low perceived-quality index terms generated by the RFA algorithm. A low perceived-quality index term has the average perceived quality score less than 2.0.

index-term	perceived_quality_avg	doc-id
activity	1.9	1
aid	1.54	1
aided	1.18	1
tool	2	1
use	1.81	1
used	1.45	1
international	2	5
solve	1.9	5
solving	2	5
used	1.9	5
using	1.81	5
financial	1.61	7
financial-issues	2	7
new	1.84	7
cases	1.38	11
doctors	2	11
help	1.38	11
programs	1.92	11
use	1.23	11
used	1.23	11
working	1.38	11
cases	1.92	13
committed	1.69	13
recent	1.38	13
solve	1.76	13
solving	1.76	13
used	1.23	13
using	1.23	13
cases	1.69	14
computer	2	14
computers	2	14
international	1.46	14
solve	1.61	14

Figure D.1 Low Perceived-Quality RFA Index Terms.

solving	1.53	14
used	1.15	14
using	1.15	14
placed	1.27	17
placed-south	1.63	17
south	1.63	17
world	2	17
developments	1.63	18
did	1.27	18
involved	1.27	18
involvement	1.72	18
role	1.54	18
does	1.5	21
does-fdic	1.64	21
recent	1.85	21
taken	1.5	21
united	2	21
attempts	1.78	23
countries	2	23
involved	1.85	23
occurred	1.92	23
people	1.85	23
recent	1.71	23
world	1.92	23
surrounding	1.85	24
drug	1.57	25
did	1.53	26
role	2	26
predict	1.92	29
does	1.3	32
does-fdic	2	32
recent	1.61	32
states	1.69	32
united	1.61	32
changes	1.69	33
data	1.61	33
estimates	1.76	33
figures	1.92	33
natural	2	33
proven	1.53	33
specific	1.61	33
application	1.27	36
applications	1.36	36
attitudes	1.54	36
benefit	1.72	36
development	1.9	36

Figure D.1 Low Perceived-Quality RFA Index Terms (Continued).

field	1.63	36
new	1.36	36
products	1.36	36
products-attitudes	1.63	36
recent	1.81	36
techniques	1.72	36
used	1.27	36
current	1.45	40
impending	1.63	40
information	1.63	40
ongoing	1.72	40
predict	1.27	40
decrease	1.9	42
inside	1.66	46
involved	1.83	46
bell	1.75	47
bell-breakup	1.58	47
breakup	1.91	47
doing	1.58	47
financial-health	1.66	47
financial-status	1.91	47
future	2	47
health	1.5	47
record-profit	2	47
status	1.75	47

Figure D.1 Low Perceived-Quality RFA Index Terms (Continued).

APPENDIX E

REGRESSION 3 – COMPLETE RESULTS

Figure E.1 shows the complete results of Regression 3 presented in Section 6.4.2.

Source	SS	df	MS	
				Number of obs = 36071
				F(162, 35908) = 107.17
Model	22422.2815	162	138.409145	Prob > F = 0.0000
Residual	46375.8435	35908	1.29151842	R-squared = 0.3259
				Adj R-squared = 0.3229
Total	68798.125	36070	1.90735029	Root MSE = 1.1364

score	Coef.	Std.Err.	t	P>|t|	[95% Conf.Interval]	
sys_br	-0.2827731	0.0146224	-19.34	0	-0.3114335	-0.2541128
sys_rfa	0.141537	0.0146791	9.64	0	0.1127656	0.1703084
female	0.391226	0.0727272	5.38	0	0.2486785	0.5337734
undergrad	1.726876	0.0829214	20.83	0	1.564348	1.889405
fam_numeric1	-1.800655	0.1180705	-15.25	0	-2.032077	-1.569234
fam_numeric2	-0.3757592	0.0797459	-4.71	0	-0.5320637	-0.2194548
fam_numeric3	-1.405215	0.1314459	-10.69	0	-1.662852	-1.147577
fam_numeric4	-1.070209	0.0900332	-11.89	0	-1.246677	-0.8937417
docid1	-1.635034	0.1141853	-14.32	0	-1.85884	-1.411227
docid2	-1.647155	0.1141853	-14.43	0	-1.870962	-1.423348
docid3	-1.415337	0.1141853	-12.4	0	-1.639143	-1.19153
docid4	-1.278961	0.1143306	-11.19	0	-1.503052	-1.054869
docid5	-1.333519	0.1141853	-11.68	0	-1.557325	-1.109712
docid6	-2.633391	0.1528533	-17.23	0	-2.932988	-2.333794
docid7	-2.866918	0.1524648	-18.8	0	-3.165754	-2.568083
docid8	-2.882303	0.1524648	-18.9	0	-3.181139	-2.583467
docid9	-2.836149	0.1524648	-18.6	0	-3.134985	-2.537313
docid10	-2.72187	0.1525569	-17.84	0	-3.020886	-2.422854
docid11	-3.251535	0.2243513	-14.49	0	-3.69127	-2.8118
docid12	-3.141279	0.2243513	-14	0	-3.581014	-2.701543
docid13	-3.214356	0.2243513	-14.33	0	-3.654091	-2.77462
docid14	-3.092561	0.2243513	-13.78	0	-3.532296	-2.652825
docid15	-3.041279	0.2243513	-13.56	0	-3.481014	-2.601543
docid16	0.3535153	0.0977507	3.62	0	0.1619209	0.5451096
docid17	0.0475724	0.0977508	0.49	0.626	-0.144022	0.2391668
docid18	0.1131455	0.0977507	1.16	0.247	-0.0784489	0.3047398
docid19	-0.0324243	0.0975808	-0.33	0.74	-0.2236857	0.1588371
docid20	-0.0884849	0.0975808	-0.91	0.365	-0.2797463	0.1027765
docid21	-1.788228	0.1460934	-12.24	0	-2.074575	-1.501881
docid22	-1.802101	0.1461826	-12.33	0	-2.088624	-1.515579
docid23	-1.822752	0.1460934	-12.48	0	-2.109099	-1.536404
docid24	-1.545116	0.1461866	-10.57	0	-1.831647	-1.258586
docid25	-1.781085	0.1460934	-12.19	0	-2.067433	-1.494738
docid26	0.0806526	0.119669	0.67	0.5	-0.1539023	0.3152076
docid27	0.0679959	0.1199224	0.57	0.571	-0.1670555	0.3030474
docid28	-0.0271075	0.1195517	-0.23	0.821	-0.2614323	0.2072174

Figure E.1 Regression 3 – Complete Results

docid29	0.1882772	0.1195517	1.57	0.115	-0.0460477	0.422602
docid30	0.2319164	0.1197905	1.94	0.053	-0.0028766	0.4667095
docid31	-0.7278589	0.1074975	-6.77	0	-0.9385571	-0.5171607
docid32	-0.8798976	0.1074915	-8.19	0	-1.090584	-0.669211
docid33	-0.8375899	0.1074915	-7.79	0	-1.048276	-0.6269033
docid34	-1.009385	0.1074915	-9.39	0	-1.220071	-0.7986982
docid35	-1.137408	0.1074915	-10.58	0	-1.348095	-0.9267219
docid36	-2.167978	0.1392881	-15.56	0	-2.440987	-1.89497
docid37	-2.052827	0.1392881	-14.74	0	-2.325836	-1.779818
docid38	-1.990706	0.1392881	-14.29	0	-2.263715	-1.717697
docid39	-1.902827	0.1392881	-13.66	0	-2.175836	-1.629818
docid40	-1.802827	0.1392881	-12.94	0	-2.075836	-1.529818
docid41	-0.3673114	0.1365729	-2.69	0.007	-0.6349983	-0.0996245
docid42	-0.4773114	0.1365729	-3.49	0	-0.7449983	-0.2096245
docid43	-0.3502719	0.1365808	-2.56	0.01	-0.6179743	-0.0825695
docid44	-0.6189781	0.1365729	-4.53	0	-0.886665	-0.3512912
docid45	-0.6732271	0.1367064	-4.92	0	-0.9411758	-0.4052784
docid46	0.1625	0.0598962	2.71	0.007	0.0451017	0.2798983
docid47	-0.075	0.0598962	-1.25	0.211	-0.1923983	0.0423983
docid48	0.1861111	0.0598962	3.11	0.002	0.0687128	0.3035094
docid49	0.2958333	0.0598962	4.94	0	0.178435	0.4132316
sid1	1.037381	0.1429797	7.26	0	0.7571361	1.317625
sid2	1.154507	0.1370879	8.42	0	0.8858102	1.423203
sid3	1.897195	0.2082981	9.11	0	1.488924	2.305465
sid4	-1.224898	0.1375151	-8.91	0	-1.494431	-0.9553639
sid5	2.120885	0.0939998	22.56	0	1.936643	2.305127
sid6	0.5953128	0.1548675	3.84	0	0.2917679	0.8988577
sid7	-2.006122	0.1001234	-20.04	0	-2.202367	-1.809877
sid8	1.173333	0.0927907	12.64	0	0.9914607	1.355206
sid10	0.5528875	0.1341175	4.12	0	0.2900132	0.8157618
sid11	0.6968542	0.1062239	6.56	0	0.4886522	0.9050562
sid12	3.712264	0.1832432	20.26	0	3.353101	4.071426
sid13	3.288311	0.2276715	14.44	0	2.842069	3.734554
sid14	-1.215488	0.0932582	-13.03	0	-1.398277	-1.032699
sid15	0.752896	0.181077	4.16	0	0.3979796	1.107812
sid16	-0.5300396	0.1230156	-4.31	0	-0.7711538	-0.2889254
sid17	0.3968761	0.1339631	2.96	0.003	0.1343044	0.6594478
sid18	1.439093	0.1118459	12.87	0	1.219871	1.658314
sid19	-1.37338	0.0938896	-14.63	0	-1.557406	-1.189354
sid20	0.8033333	0.0927907	8.66	0	0.6214607	0.985206
sid22	3.208757	0.2421653	13.25	0	2.734106	3.683408
sid23	-0.0681878	0.1274729	-0.53	0.593	-0.3180385	0.1816628
sid24	-0.9764304	0.0939998	-10.39	0	-1.160673	-0.792188
sid26	0.2271023	0.1272775	1.78	0.074	-0.0223654	0.47657
sid27	1.460655	0.1417418	10.31	0	1.182837	1.738473
sid28	0.8901106	0.1651977	5.39	0	0.5663182	1.213903
sid29	1.238774	0.1010412	12.26	0	1.04073	1.436818
sid30	1.528995	0.1502374	10.18	0	1.234525	1.823465
sid31	3.453467	0.1634458	21.13	0	3.133108	3.773826

Figure E.1 Regression 3 – Complete Results (Continued).

sid32	3.150528	0.2082981	15.13	0	2.742257	3.558799
sid33	-1.175084	0.0932582	-12.6	0	-1.357873	-0.9922953
sid34	1.71086	0.123329	13.87	0	1.469132	1.952589
sid35	-3.330527	0.1588066	-20.97	0	-3.641792	-3.019261
sid36	0.4723168	0.1271801	3.71	0	0.2230399	0.7215937
sid37	3.353989	0.1460163	22.97	0	3.067792	3.640185
sid38	-0.0249872	0.1231186	-0.2	0.839	-0.2663033	0.2163289
sid39	2.506309	0.1391378	18.01	0	2.233594	2.779023
sid40	1.692635	0.1582524	10.7	0	1.382456	2.002815
sid42	-1.339633	0.1299843	-10.31	0	-1.594406	-1.08486
sid43	0.9006451	0.1171627	7.69	0	0.6710027	1.130287
sid44	1.299008	0.1246588	10.42	0	1.054673	1.543343
sid45	0.2045593	0.1010412	2.02	0.043	0.0065156	0.402603
sid46	0.3529392	0.1231186	2.87	0.004	0.1116231	0.5942553
sid47	0.9816253	0.1370879	7.16	0	0.7129289	1.250322
sid48	4.082091	0.2421653	16.86	0	3.607439	4.556742
sid49	0.0631253	0.1274729	0.5	0.62	-0.1867254	0.312976
sid50	1.19977	0.1503064	7.98	0	0.905165	1.494375
sid51	-0.9693878	0.0937328	-10.34	0	-1.153107	-0.7856686
sid52	2.52565	0.1271801	19.86	0	2.276373	2.774927
sid53	0.02	0.0927907	0.22	0.829	-0.1618726	0.2018726
sid54	-1.063333	0.0927907	-11.46	0	-1.245206	-0.8814607
sid55	3.861526	0.1280649	30.15	0	3.610515	4.112537
sid56	4.157385	0.1250281	33.25	0	3.912326	4.402443
sid57	-1.527274	0.1175332	-12.99	0	-1.757643	-1.296906
sid58	1.834179	0.1232697	14.88	0	1.592567	2.075791
sid59	-0.0033333	0.0927907	-0.04	0.971	-0.185206	0.1785393
sid60	-0.350692	0.1651977	-2.12	0.034	-0.6744845	-0.0268996
sid62	3.131795	0.13189	23.75	0	2.873287	3.390304
sid63	2.779983	0.2641597	10.52	0	2.262222	3.297744
sid65	-1.634467	0.1372418	-11.91	0	-1.903465	-1.365469
sid66	-0.3966667	0.0927907	-4.27	0	-0.5785393	-0.214794
sid68	-1.262047	0.1119378	-11.27	0	-1.481448	-1.042645
sid69	-1.93565	0.1001234	-19.33	0	-2.131895	-1.739405
sid72	-2.675026	0.1462992	-18.28	0	-2.961777	-2.388275
sid73	1.030201	0.0931016	11.07	0	0.8477194	1.212683
sid74	-1.226303	0.1372418	-8.94	0	-1.495301	-0.9573055
sid76	-1.490209	0.1086095	-13.72	0	-1.703087	-1.277332
sid78	-1.293989	0.1542102	-8.39	0	-1.596245	-0.991732
sid79	1.064136	0.1429797	7.44	0	0.7838919	1.344381
sid80	2.232902	0.1517382	14.72	0	1.935491	2.530314
sid81	0.7035428	0.1244431	5.65	0	0.4596306	0.9474549
sid82	0.0992796	0.1232697	0.81	0.421	-0.1423327	0.340892
sid83	0.0447208	0.1299843	0.34	0.731	-0.2100524	0.299494
sid84	-0.7095644	0.1272775	-5.57	0	-0.959032	-0.4600967
sid85	1.549429	0.1368273	11.32	0	1.281244	1.817615
sid86	-0.0937239	0.1231353	-0.76	0.447	-0.3350727	0.147625
sid87	-0.728774	0.1010412	-7.21	0	-0.9268177	-0.5307303
sid88	-0.5010025	0.1614574	-3.1	0.002	-0.8174639	-0.1845411
sid89	1.923486	0.1250281	15.38	0	1.678428	2.168545
sid90	0.9471946	0.2082981	4.55	0	0.5389241	1.355465

Figure E.1 Regression 3 – Complete Results (Continued).

sid91	0.3373539	0.1012868	3.33	0.001	0.1388288	0.535879
sid92	1.193239	0.1232697	9.68	0	0.951627	1.434852
sid93	2.298976	0.132223	17.39	0	2.039815	2.558137
sid94	0.0506451	0.1171627	0.43	0.666	-0.1789973	0.2802874
sid96	3.011622	0.1250281	24.09	0	2.766563	3.256681
sid97	3.873317	0.2641597	14.66	0	3.355556	4.391077
sid99	2.48444	0.181077	13.72	0	2.129523	2.839356
sid100	0.0952381	0.0937328	1.02	0.31	-0.088481	0.2789572
sid101	0.6576174	0.1272368	5.17	0	0.4082295	0.9070053
sid103	-0.5071592	0.0938896	-5.4	0	-0.6911856	-0.3231328
sid104	-0.7871125	0.1341175	-5.87	0	-1.049987	-0.5242382
sid105	0.9784357	0.1313356	7.45	0	0.7210139	1.235857
sid106	2.211394	0.1391378	15.89	0	1.938679	2.484108
sid107	2.090528	0.2082981	10.04	0	1.682257	2.498799
sid108	0.016835	0.0932582	0.18	0.857	-0.1659539	0.1996239
sid109	1.448273	0.1232697	11.75	0	1.206661	1.689885
sid110	-1.60211	0.1186385	-13.5	0	-1.834645	-1.369575
sid111	0.9052043	0.1373042	6.59	0	0.6360841	1.174325
sid112	2.297322	0.1460163	15.73	0	2.011126	2.583518
sid113	0.2058851	0.108752	1.89	0.058	-0.0072721	0.4190423
sid114	-1.735215	0.160898	-10.78	0	-2.050579	-1.41985
sid115	0.2358824	0.1231186	1.92	0.055	-0.0054337	0.4771985
sid116	2.434297	0.1832432	13.28	0	2.075135	2.79346
sid117	1.509302	0.1582524	9.54	0	1.199123	1.819481
sid119	0.5020439	0.1087614	4.62	0	0.2888682	0.7152195
sid120	-2.740482	0.1251471	-21.9	0	-2.985774	-2.495189
_cons	3.743845	0.1217855	30.74	0	3.505142	3.982548

Figure E.1 Regression 3 – Complete Results (Continued).

REFERENCES

1. Allan, J., Ballesteros, L., Callan, J. P., Croft, W. B. and Lu, Z. (1995). "Recent Experiments with INQUERY." Proceedings of the Fourth Text Retrieval Conference (TREC-4), NIST Special Publication: pp. 49-63.

2. Andrews, W. (1996). "Search Engines Gain Tools for Shifting Content on the Fly." Web Week 2(11): pp. 41-42.

3. Attardi, G., Marco, S. D. and Salvi, D. (1998). "Categorization by Context." Journal of Universal Computer Science 4(9): pp. 719-736.

4. Bartell, T., Cottrell, G. W. and Belew, R. K. (1998). "Optimizing Similarity Using Multi-Query Relevance Feedback." Journal of the American Society for Information Science 49(8): pp. 742-761.

5. Belew, R. K. (1989). "Adaptive Information Retrieval: Using a Connectionist Representation to Retrieve and Learn About Documents." Proceedings of the 12th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval: pp. 11-20.

6. Belkin, N., Cool, C., Koenemann, J., Ng, K. B. and Park, S. (1996). "Using Relevance Feedback and Ranking in Interactive Searching." Proceedings of the Fourth Text Retrieval Conference (TREC-4), NIST Special Publication.

7. Bodoff, D., Enache, D., Kambil, A., Simon, G. and Yukhimates, A. (2001). "A Unified Maximum Likelihood Approach to Document Retrieval." Journal of the American Society for Information Science and Technology 52(10): pp. 785-796.

8. Bollmann-Sdorra, P. and Raghavan, V. V. (1993). "On the Delusiveness of Adopting a Common Space for Modeling IR Objects: Are Queries documents?" JASIS 44(10): pp. 320-330.

9. Bot, R. S., Wu, Y. B., Chen, X. and Li, Q. (2004). "A Hybrid Classifier Approach for Web Retrieved Documents Classification." Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on Information Technology. Las Vegas, NV. Volume: 1(April 5-7): pp. 326-330.

10. Brauen, T. L., Holt, R. C. and Wilcox, T. R. (1968). "Document Indexing Based on Relevance Feedback." Report ISR-14 to the National Science Foundation, Section XI, Department of Computer Science, Cornell University, Ithaca, NY(June).

11. Brauen, T. L. (1969). "Document Vector Modification." Scientific Report ISR-17 (September).

12. Brauen, T. L. (1971). "Document Vector Modification - in The SMART Retrieval System," by Gerard Salton. Prentice Hall Series in Automatic Computation: pp. 456-485.

13. Buckley, C., Allan, J. and Salton, G. (1994). "Automatic Routing and Ad-hoc Retrieval Using SMART : TREC 2." Text REtrieval Conference - TREC 2.

14. Buckley, C., Salton, G., Allan, J. and Singhal, A. (1995). "Automatic Query Expansion Using SMART: TREC 3." Proceedings of the 3rd Text REtrieval Conference (TREC3) NIST Special Publication(November): pp. 69-80.

15. Buckley, C. and Walz, J. (1999). "The TREC-8 Query Track." TREC 8.

16. Buckley, C. (2000). "The TREC-9 Query Track." TREC 9.

17. Chakrabarti, S., Dom, B. E., Agrawal, R. and Raghavan, P. (1998a). "Scalable Feature Selection, Classification and Signature Generation for Organizing Large Text Databases into Hierarchical Topical Taxonomies." Journal of Very Large Data Bases 7(3): pp. 163-178.

18. Chakrabarti, S., Dom, B. E. and Indyk, P. (1998b). "Enhanced Hypertext Categorization Using Hyperlinks," Proceedings of SIGMOD-98, ACM International Conference on Management of Data (Seattle, WA, 1998): pp. 307-318.

19. Chen, H., Shankaranarayanan, G., She, L. and Iyer, A. (1998). "A Machine Learning Approach to Inductive Query by Examples: An Experiment Using Relevance Feedback, ID3, Genetic Algorithms, and Simulated Annealing." Journal of the American Society for Information Science 49(8): pp. 693-705.

20. Chen, Z., Meng, X., Fowler, R. H. and Zhu, B. (2001). "FEATURES: Real-Time Adaptive Feature and document Learning for Web Search." Journal of the American Society for Information Science and Technology 52(8): pp. 655-665.

21. Clarke, C. L. A. and Cormack, G. V. (2000). "Shortest-Substring Retrieval and Ranking." ACM Transactions on Information Systems (TOIS) 18(1): pp. 44-78.

22. Deerwester, S., Dumais, S. T., Landauer, T. K., Furnas, G. W. and Harshman, R. (1990). "Indexing by Latent Semantic Analysis." Journal of the American Society for Information Science 41(6): pp. 391-407.

23. Dillon, M. and Desper, J. (1980). "Automatic Relevance Feedback in Boolean Retrieval Systems." Journal of Documentation 36: pp. 197-208.

24. Dumais, S. T. and Chen, H. (2000). "Hierarchical Classification of Web Content." Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval (Athens, Greece, 2000): pp. 256-263.

25. Efthimiadis, E. N. (2000). "Interactive Query Expansion: A User-Based Evaluation in a Relevance Feedback Environment." Journal of the American Society for Information Science 51(11): pp. 989-1003.

26. Fidel, R. (1986). "The Possible Effect of Abstracting Guidelines on Retrieval Performance of Free-Text searching." Information Processing and Management 22(4): pp. 1309-1316.

27. Forsyth, R. and Rada, R. (1986). "Adding an Edge in Machine Learning: Applications in Expert Systems and Information Retrieval." Ellis Horwood Ltd.: pp. 198-212.

28. Friedman, S. R., Maceyak, J. A. and Weiss, S. F. (1967). "A Relevance Feedback System Based on Document Transformations." Scientific Report ISR-12(June): Section X.

29. Fuhr, N. and Buckley, C. (1991). "A Probabilistic Learning Approach for Document Indexing." ACM Transactions on Information Systems (TOIS) 9(3): pp. 223-248.

30. Furnas, G. W., Landauer, T. K., Gomez, L. M. and Dumais, S. T. (1987). "The Vocabulary Problem in Human-System Communication." Communications of the ACM 30(11): pp. 964-971.

31. Furnkranz, J. (1999). "Exploiting structural information for text classification on the WWW." Proceedings of IDA-99, 3rd Symposium on Intelligent Data Analysis (Amsterdam, The Netherlands, 1999): pp. 487-497.

32. Garnsey, M. (2002). "Automatic Classification of Financial Accounting Concepts." Proceedings of the 11th Annual Research Workshop on: Artificial Intelligence and Emerging Technologies in Accounting, Auditing and Tax, San Antonio, Texas: pp. 15-26.

33. Gauch, S., Wang, J. and Rachakonda, S. M. (1999). "A Corpus Analysis Approach for Automatic Query Expansion and its Extension to Multiple Databases." ACM Transactions on Information Systems 17(3): pp. 250-269.

34. Gong, Y. and Liu, X. (2001). "Generic Text Summarization Using Relevance Measure and Latent Semantic Indexing." ACM: pp. 19-25.

35. Govert, N., Lalmas, M. and Fuhr, N. (1999). "A Probabilistic Description-Oriented Approach for Categorizing Web Documents." Proceedings of CIKM-99, 8th ACM International Conference on Information and Knowledge Management (Kansas City, MO, 1999): pp. 475-482.

36. Harman, D. K. (1993). "The First Text REtrieval Conference (TREC-1), Washington DC." NIST Special Publication: pp. 500-207.

37. Harman, D. K. (1994). "The Second Text REtrieval Conference, Washington DC." NIST Special Publication: pp. 500-215.

38. Harman, D. K. (1995). "Overview of the Third Text REtrieval Conference (TREC-3), Washington DC." NIST Special Publication: pp. 500-225.

39. Hearst, M. A. and Plaunt, C. (1993). "Subtopic Structuring of Full Length Document Access." Proceedings of SIGIR 1993.

40. Hearst, M. A. (1994). "Multi-Paragraph Segmentation of Expository Text." 32nd Annual Meeting of the Association for Computational Linguistics, New Mexico State University, Las Cruces, New Mexico: pp. 9-16.

41. Hearst, M. A. (1998). "Automated Discovery of WordNet Relations." In "WordNet: an electronic lexical database," Christiane Fellbaum, MIT Press.

42. Hoelscher, C. (1998). "How Internet Experts Search for Information on the Web." World Conference of the World Wide Web, Internet and Intranet.

43. Ide, E. (1969). "Relevance Feedback in Automatic Document Retrieval System." Report ISR-15 to the National Science Foundation, Cornell University, Ithaca, NY(January): pp. 81-85.

44. Ide, E. (1971). "The SMART Retrieval System: New Experiments in Relevance Feedback." Prentice-Hall.

45. Jansen, B. J. (2000). "The Effect of Query Complexity on Web Searching Results." Information Research 6(1).

46. Jansen, B. J., Spink, A. and Saracevic, T. (2000). "Real Life, Real Users and Real Needs: A Study and Analysis of User Queries on the Web." Information Processing and Management 36(2): pp. 207-227.

47. Joachims, T. (1998). "Text Categorization with Support Vector Machines: Learning with Many Relevant Features." Proceedings of European Conference on Machine Learning (ECML '98).

48. Johnson, F., Griffiths, J. R. and Hartley, R. J. (2001). "DEvISE: A Framework for the Evaluation of Internet Search Engines." Resource Report 100.

49. Kanazawa, T. (1999). "R^2D^2 at NTCIR: Using the Relevance-Based Superimposition Model." Proceedings of the first NTCIR Workshop on Research in Japanese Test Retrieval and Term Recognition: pp. 83-88.

50. Kaszkiel, M., Zobel, J. and Sacks-Davis, R. (1999). "Efficient Passage Ranking for Document Databases." ACM Transactions of Information Systems 17(4): pp. 406-439.

51. Kekalainen, J. and Jarvelin, K. (2000). "The Co-Effects of Query Structure and Expansion on Retrieval Performance in Probabilistic Text Retrieval." Information Retrieval 1: pp. 329-344.

52. Kelly, D. and Belkin, N. J. (2001). "Reading Time, Scrolling and Interaction: Exploring Implicit Sources of User preferences for Relevance Feedback." Proceedings of the 24th annual international ACM SIGIR Conference on Research and Development in Information Retrieval: pp. 408-409.

53. Kelly, D. and Teevan, J. (2003). "Implicit Feedback for Inferring User Preference: A Bibliography." ACM SIGIR Forum 37(2): pp. 18-28.

54. Kelly, D. and Belkin, N. J. (2004). "Display Time as Implicit Feedback: Understanding Task Effects." In Proceedings of the 27th Annual ACM International Conference on Research and Development in Information Retrieval (SIGIR '04), Sheffield, UK: pp. 377-384.

55. Kemp, C. and Ramamohanarao, K. (2002). "Long-Term Learning for Web Search Engines." Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2002).

56. Koenemann, J. and Belkin, N. J. (1996). "A Case for Interaction: A Study of Interactive Information Retrieval Behavior and Effectiveness." Conference Proceedings on Human Factors in Computing Systems, Vancouver, British Columbia, Canada: pp. 205-212.

57. Korfhage, R. R. (1997). "Information Storage and Retrieval." Wiley Computer Publishing: pp. 221-232.

58. Lawrie, D., Croft, W. B. and Rosenberg, A. (2001). "Finding Topic Words for Hierarchical Summarization." Proceedings of the 2001 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New Orleans, LA: pp. 349-357.

59. Lovins, J. B. (1968). "Development of a Stemming Algorithm." Mechanical Translation and Computational Linguistics 11: pp. 22-31.

60. Marchionini, G. (1992). "Interfaces for End Users Information Seeking." Journal of the American Society for Information Science 43(2): pp. 156-163.

61. Martin, P., Macleod, I. A. and Nordin, B. (1986). "A Design of a Distributed Full Text Retrieval System." ACM Conference on Research and Development in Information Retrieval.

62. Miike, S., Itoh, E., Ono, K. and Sumita, K. (1994). "A Full Text Retrieval System with Dynamic Abstract Generation Function." Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval: pp. 152-161.

63. Mitra, M., Singhal, A. and Buckley, C. (1998). "Improving Automatic Query Expansion." Proceedings of the 21st annual international ACM SIGIR Conference on Research and Development in Information Retrieval, August: pp. 206-214.

64. Mittendorf, E. and Schauble, P. (1994). "Document and Passage Retrieval Based on Hidden Markov Models." Proceedings of 17th Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval: pp. 318-327.

65. Mizzaro, S. (1997). "Relevance: The Whole History." Journal of the American Society for Information Science 48(9): pp. 810-832.

66. Molina, M. P. (1995). "Documentary Abstracting: Toward a Methodological Model." Journal of the American Society for Information Science 46(3): pp. 225-234.

67. Oh, H., Myaeng, S. H. and Lee, M. (2000). "A Practical Hypertext Categorization Method Using Links and Incrementally Available Class Information." Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval (Athens, Greece, 2000): pp. 264-271.

68. Olsen, K. A., Korfhage, R. R., Sochats, K. M., Spring, M. B. and Williams, J. G. (1993). "Visualization of a Document Collection with Implicit and Explicit Links: The VIBE System." Scandinavian Journal of Information Systems 5: pp. 79-95.

69. Parker, L. M. P. (1983). "Towards a Theory of Document Learning." Journal of the American Society for Information Science 34(1): pp. 16-21.

70. Piwowarski, B. (2000). "Learning in Information Retrieval: A Probabilistic Differential Approach." Proceedings of the BCS-IRSG, 22nd Annual Colloquium on Information Retrieval Research, Cambridge, England.

71. Rocchio, J. J. (1971). "Relevance Feedback in Information Retrieval." In G. Salton, “The SMART Retrieval System: Experiments in Automatic Document Processing,” Englewood Cliffs, NJ: Prentice Hall: pp. 313-323.

72. Salton, G. (1971). "The SMART Retrieval System: Experiments in Automatic Document Processing." Prentice-Hall.

73. Salton, G. (1971b). "The SMART Retrieval System: Relevance Feedback and the Optimization of Retrieval Effectiveness." Prentice-Hall.

74. Salton, G., Wong, A. and Yang, C. S. (1975). "A Vector Space Model for Information Retrieval." Journal of the American Society for Information Science 18(11): pp. 613-620.

75. Salton, G., Fox, E. A. and Wu, H. (1983). "Extended Boolean information retrieval." Communications of the ACM 26(11): pp. 1022-1036.

76. Salton, G., Fox, E. A. and Voorhes (1984). "A comparison of two methods for Boolean query relevance feedback." Information Processing and Management 20: pp. 637-651.

77. Salton, G., Fox, E. A. and Voorhees (1985). "Advanced Feedback Methods in Information Retrieval." Journal of the American Society for Information Science 36(3): pp. 200-210.

78. Salton, G. (1989). "Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer." Addison Wesley, 1989.

79. Salton, G. and Buckley, C. (1990). "Improving Retrieval Performance by Relevance Feedback." Journal of the American Society for Information Science 41(4): pp. 288-297.

80. Salton, G., Allan, J. and Buckley, C. (1993). "Approaches to Passage Retrieval in Full Text Information Retrieval." ACM/SIGIR 6: pp. 49-58.

81. Salton, G., Singhal, A., Buckley, C. and Mitra, M. (1996). "Automatic Text Decomposition Using Text Segments and Text Themes." ACM: pp. 53-65.

82. Sanderson, M. and Croft, B. (1999). "Deriving concept hierarchies from text." Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Berkely, CA: pp. 206-213.

83. Savoy, J., Ndarugendamwo, M. and Vrajitoru, D. (1996). "Report on the TREC-4 Experiment: Combining Probabilistic and Vector-Space Schemes." TREC 4 Proceedings(October): pp. 537-548.

84. Sebastiani, F. (2002). "Machine Learning in Automated Text Categorization." ACM Computing Surveys 34(1): pp. 1-47.

85. Siegel, S. and Castellan, N. J. (1998). "Nonparametric Statistics for the Behavioral Sciences (2nd Ed.)." McGraw Hill College Div, New York, 1988.

86. Silverstein, C. (1999). "Analysis of a Very Large Web Search Engine Query Log." SIGIR Forum 33(1): pp. 6-12.

87. Singhal, A., Buckley, C. and Mitra, M. (1996). "Pivoted Document Length Normalization." Proceedings of the19th ACM-SIGIR Conference. ACM Press: pp. 412-420.

88. Spink, A. and Saracevic, T. (1997). "Interaction in Information Retrieval: Selection and Effectiveness of Search Terms." Journal of the American Society for Information Science 48(8): pp. 741-761.

89. Spink, A., Wilson, T., Ellis, D. and Ford, N. (1998). "Modeling Users' Successive Searches in Digital Environments." D-Lib Magazine, April 1998.

90. Spink, A., Jansen, B. and Ozmultu, H. C. (2000). "Use of Query Reformulation and Relevance Feedback by Excite Users." Internet Research: Electronic Networking Applications and Policy 10(4): pp. 317-328.

91. Spink, A., Wolfram, D., Jansen, B. J. and Saracevic, T. (2001). "Searching the Web: The public and their queries." Journal of the American Society for Information Science 52(3): pp. 226-234.

92. Toms, E. G., Kopak, R. W., Bartlett, J. and Freund, L. (2001). "Selecting Versus Describing: A Preliminary Analysis of the Efficacy of Categories in Exploring the Web." The Tenth Text REtrieval Conference (TREC 2001).

93. Van-Rijsbergen, C. J. (1979). "Information Retrieval." Second Edition, London: Butterworths.

94. White, R. W., Jose, J. M. and Ruthven, I. (2002). "A System Using Implicit Feedback and Top Ranking Sentences to Help Users Find Relevant Web Documents." Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval: pp. 446-446.

95. Woods, W. A. (1997). "Conceptual Indexing: A Better Way to Organize Knowledge." Sun Labs Technical Report: TR-97-61, Technical Reports, 901 San Antonio Road, Palo Alto, California 94303, USA.

96. Zamir, O. and Etzioni, O. (1999). "Grouper: a Dynamic Clustering Interface to Web Search Results." Proceedings of the Eighth International World Wide Web Conference. Toronto, Canada.