

Concept-Based Interactive Query Expansion

Bruno M. Fonseca^{1,2}
maciel@dcc.ufmg.br

Paulo Golgher²
golgher@google.com

Bruno Pôssas^{1,2}
bavep@google.com

Berthier Ribeiro-Neto^{1,2}
berthier@dcc.ufmg.br

Nivio Ziviani¹
nivio@dcc.ufmg.br

ABSTRACT

Despite the recent advances in search quality, the fast increase in the size of the Web collection has introduced new challenges for Web ranking algorithms. In fact, there are still many situations in which the users are presented with imprecise or very poor results. One of the key difficulties is the fact that users usually submit very short and ambiguous queries, and they do not fully specify their information needs. That is, it is necessary to improve the query formation process if better answers are to be provided. In this work we propose a novel concept-based query expansion technique, which allows disambiguating queries submitted to search engines. The concepts are extracted by analyzing and locating cycles in a special type of query relations graph. This is a directed graph built from query relations mined using association rules. The concepts related to the current query are then shown to the user who selects the one concept that he interprets is most related to his query. This concept is used to expand the original query and the expanded query is processed instead. Using a Web test collection, we show that our approach leads to gains in average precision figures of roughly 32%. Further, if the user also provides information on the *type* of relation between his query and the selected concept, the gains in average precision go up to roughly 52%.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—*Relevance Feedback, Query Formulation.*; H.2.8 [Database Management]: Database Applications—*Data Mining*

General Terms

Theory, algorithms, experimentation, performance.

Keywords

Association rules, interactive query expansion, user feedback, web searching.

¹Federal University of Minas Gerais: 30161-970 Belo Horizonte-MG, Brazil

²Google Brazil: Av. Abraão Caram, 430, 4^o andar - Pampulha, Belo Horizonte-MG, Brazil

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CIKM'05, October 31–November 5, 2005, Bremen, Germany.

Copyright 2005 ACM 1-59593-140-6/05/0010 ...\$5.00.

1. INTRODUCTION

The Web is an innovation that has modified the way we learn, work and live. The novelty lies not only on the freedom to publish, but also in the almost universal communication facilities. It marks the beginning of a new era, of a new society, started by what we may call the *information revolution*.

In these new times, the volume of information that can be accessed at low cost and high convenience is mind boggling. To illustrate, Google¹ advertised indexing more than 8 billion Web pages in 2004. And this volume of information tends to become even larger. As a result, information of critical value appears frequently mixed in with other pieces of information that are not of interest.

Because the volume of data is now much larger and frequently poorly organized, finding useful or relevant information might be rather difficult. In fact, this is the case even with modern search engines that take advantage of link analysis to identify popular sources of relevant information.

It is general consensus that Web users provide poor specifications of their information needs in general. Either because they are in a hurry or because they do not understand the search process well, Web users frequently specify short queries with little or no context information associated with them. Further, they are the sole masters in the moment of deciding what is relevant and what is not. The combination of these two factors suggests that interacting with the user is a key step if the precision of the results is to be improved [2]. In other words, to improve the user information search experience we have to ask more information from him.

In this work we focus our attention on the problem of improving the process of query formation. Our approach is to provide high level suggestions for expanding the original user query with additional context. This is done using information extracted from a log of past queries – an important piece of evidence that is generated in abundance by search engines.

Our approach is composed of four basic steps. First, compute past queries related to the current query using the framework of association rules. Second, build a query relations graph encompassing all these query relations. Third, identify subsets of queries that are strongly related among themselves. Each of these subsets is viewed as a *concept*, an entity of the world that might be related to the current query. Fourth, show these related concepts to the user, collect his *feedback* on the most related concept, expand the original query with the concept selected, and execute the expanded query instead.

Our concepts are labelled with subsets of past queries such as, for instance, “*jaguar, lion, tiger*”. Since these labels were specified by the users themselves, they are short in nature. We say that the concepts we use are of a high level nature. By this, we mean

¹<http://www.google.com>

that they are simple, intuitive, and can be interpreted by quick inspection. Because of this, we claim that our concept-based *feedback* approach is intuitive and likely to be engaged by Web users in general.

To validate our approach we ran experiments using a test collection composed of more than 14 million Web pages (extracted from the “.br” domain). Our query log was divided into two parts: a first part composed of 182,017 sample queries from January of 2004 to December of 2004, and a second part composed of 100 test queries from January of 2005 to February of 2005. The first part was used to compute an all-queries relation graph. The second part was used to test our concept-based expansion method.

The results were compiled for the 100 test queries and were quantified using standard 11-point recall-precision figures. As baseline, we adopted a modern ranking function that combines information on the text of the documents, on the anchor texts, and on link analysis [8]. We investigated whether the expansion of the current query with the feedback provided by the user, through the selection of a single most related concept, leads to improved results. Our measures indicate that average precision improved for all recall levels and that the average gain in precision was of 32.26%.

Following, we also asked the user to specify the *nature* of the relation between the current query and the concept selected. We were interested in obtaining extra information from the user on whether this relation was of a synonym, specialization, generalization, or association type. As before, we claim that this can be done in intuitive and natural fashion (for instance, it is simple to determine that “cars” provide a generalization for “ferrari”). Most important, we were interested in determining whether this extra information could be used to obtain improved results. Indeed this was the case, use of this extra information improved the average gain in precision from 32.26% to 52.99%, with regard to our baseline.

To the best of our knowledge, our approach is novel in the following aspects. First, it proposes the use of association rules to mine query relations in a query log. Second, it introduces the notion of a query relations graph which is central to allow identifying a small number of concepts that are strongly related to the current query and that can be meaningfully labelled. Third, it proposes to collect further feedback from the user on the *nature* of the relation between the current query and the concept the user selected for feedback. Fourth, all feedback required from the user is simple, intuitive, and can be provided by glancing through the options displayed. Fifth, it suggests that considerable gains in average precision can be obtained (over 50% in our experiments, with regard to a ranking function that already takes link analysis into consideration) with little input from the user.

The paper is organized as follows. We first discuss related work. We then present our concept generation algorithm. Next, we describe our method for query expansion using a single related concept selected by the user. Following, we present our experimental setup and how we fine tuned our concept generation algorithm. At the end, we discuss our experimental results followed by our conclusions.

2. RELATED WORK

Query expansion has improved the effectiveness of ranked retrieval by automatically adding additional terms to a query. In [7] an original query is run using conventional information retrieval techniques [4]. Then, related terms are extracted from the top documents that are returned in response to the original query using statistical heuristics. This approach has been shown to be effective on some collections, but results on large collections of web data have been mixed. For example, the work in [6], using the

Okapi approach to ranking, have found that the standard parameters are inappropriate for web data, and even with the best parameters found by tuning to that data and queries, the performance gains are insignificant. The work in [3] presents a approach for query expansion using terms that appears with noun compounds. This approach classifies related terms by categories with recognizable names.

The work in [11] was one of the first that used past user queries to improve automatic query expansion. They use the results of past queries to form affinity pools, from which expansion terms are then selected. The process works as follows: (1) for a query that is to be expanded, (2) up to three past queries that are highly similar to the current query are identified and the top 100 documents that were returned for each of these past queries are then merged, (3) forming the affinity pool. Individual terms are then selected from the top ranked documents using a TF-IDF term scoring algorithm. Their method improves relative average precision for the TREC-5 collection by around 15%.

The work in [5] chooses expansion terms from past user queries directly, rather than using them to construct sets of full text documents from which terms are then selected. The method consists of three phases: ranking the original query against the collection of documents; extracting additional query terms from the highly ranked items; then ranking the new query against the collection. The results show relative improvements over unexpanded full text retrieval of 26%–29%.

In [9] they suggest a method for finding relations between queries and phrases of documents based on query logs. They use the hypothesis that the click through information available on search engine logs represents an evidence of relation between queries and documents chosen to be visited by users. This evidence is called cross-reference of documents. Based on this evidence, the authors establish relationships between queries and phrases that occur in the documents chosen. These relationships are then used to expand the initial query or to give query suggestions. This approach can also be used to cluster queries extracted from log files [19]. Cross-reference of documents are combined with similarity functions based on query content, edit distance and document hierarchy to find better clusters. These clusters are used in question answering systems to find similar queries.

The work in [16] presents another log-based approach based on the fact that the relevant terms suggested for an user query are those that co-occur in similar query sessions from search engine logs, rather than in the retrieved documents. The suggested terms in each interactive search step with the user can be organised according to its relevance to the entire query session, rather than to the most recent single query. They show that their log-based method generates better results than document based methods.

In [17] they propose a method to choose expansion terms by mining anchor text for a large document collection. The similarity between search queries and anchor texts reported in [10] is exploited by the authors to produce high quality refinement suggestions.

The work in [12] uses association rules to extract related queries from search engines log files. Association rules are widely used to develop high quality recommendation systems in e-commerce applications available in the Web [13, 18]. These applications take user sessions stored at system logs to obtain information about the user behavior to recommend services and products. The same idea is applied to find related queries and provide suggestions to Web search engine users. They find previous search patterns that match the current query and use this information to suggest related queries that may be useful to users.

Our work differs from the related work in one or more of the following aspects: the use of association rules to mine query rela-

tions, the building of a query relations graph for identifying *strong* concepts (or entities), the use of information on the type of relation between a query and a concept selected to improve retrieval, and the fact that all feedback the user has to provide is simple and intuitive.

3. CONCEPT GENERATION ALGORITHM

Our method for generating concepts from query logs is composed of three steps: (1) determining query relations in the log, (2) building a query relations graph, and (3) identifying *concepts* in this graph for expanding user queries. While the first step can be performed offline, steps 2 and 3 should be performed during query processing time.

In the following, we discuss each of these 3 steps.

3.1 Determining Query Relations

To uncover relations among the queries in the log, we rely on the framework of association rules. The motivation is to use association rules to quantify patterns of query co-occurrence in a same user session. Stronger are these patterns of co-occurrence more related the queries are considered to be. Related queries are important because they allow establishing semantic associations between the current query and concepts that appeared in (related) past queries.

To quantify the strength of these query relations we use association rules. Thus, before proceeding, let us review basic definitions on association rules.

Association Rules

The problem of mining association rules was introduced in the context of customer commercial transactions [1]. The problem in this case is to determine what is the likelihood that a customer buy a set p_2 of products, given that he has bought a set p_1 of products before. This problem can be stated as follows.

Let p be the set of all products referred to in all transactions. Let p_1 and p_2 be two subsets of p , not necessarily disjoint. Let s_1 be a set of transactions involving the acquisition of products in the set p_1 . Let s_2 be a set of transaction involving products in the set p_2 . Then, how to find association rules of the form $p_1 \rightarrow p_2$?

More formally this problem is defined as follows.

Definition 1. Let $I = \{i_1, i_2, \dots, i_m\}$ be the set of unique literals (called items) and D a database of transactions. Each transaction $T \in D$ is a non-empty set of items, i.e., $T \neq \emptyset$ and $T \subseteq I$.

Definition 2. There is a total ordering among the transaction items, which is based on its lexicographical order, so that $i_j < i_{j+1}$, for $1 \leq j \leq m - 1$.

Definition 3. An n -itemset I_j is an ordered set of n unique items, such that $I_j \subseteq I$. Whenever the number n of items is not important, we refer simply to the itemset I_j .

Notice that the order among items in I_j follows the aforementioned total ordering.

Definition 4. The *support* count $\sigma(I_j)$ of an itemset I_j is defined as the number of transactions in D that contain I_j .

Definition 5. An itemset I_j is a *frequent itemset* if its support $\sigma(I_j)$ is greater than or equal to a given threshold, which is known as minimum support.

Definition 6. An *association rule* is an expression $A \rightarrow B$, where A and B are itemsets. The *support* of the rule is defined as $\sigma(A \cup B)$, and the *confidence* as $\frac{\sigma(A \cup B)}{\sigma(A)}$ (i.e., the conditional probability that a transaction contains B , given that it contains A).

For a given pair of minimum confidence and support thresholds and a set of transactions, the problem of mining association rules is to find out all the rules that have confidence and support greater than the minimum corresponding thresholds.

Mining Query Relations

During an interactive session with a search engine, the user specifies a set of queries. These queries might be related, particularly if they co-occur together in other user sessions (besides the one under consideration). Thus, to identify query relations we need to define a user session, which we do as follows.

Definition 7. A query is represented by the triplet $\langle q_i, id_i, t_i \rangle$, where id_i ² is the IP address of the user who submitted the query, t_i is the time stamp for the instant the query was submitted, and q_i is the set of terms that form the query.

Definition 8. A *user session* is a set of triplets $\{\langle q_1, id_1, t_1 \rangle, \langle q_2, id_2, t_2 \rangle, \dots, \langle q_n, id_n, t_n \rangle\}$ for which $id_1 = id_2 = \dots = id_n$ and $(t_2 - t_1) < T, (t_3 - t_2) < T, \dots, (t_n - t_{n-1}) < T$. The parameter T , set to 10 minutes in all our experiments, defines the maximum time interval allowed between any two consecutive queries in a same session.

If the time interval between two consecutive queries exceeds T , we interpret that a new user session has started.

The problem of finding query relations can be mapped to the problem of mining association rules as follows. A user query is mapped into an item and an user session is mapped into a transaction. A query relation is mapped into an association rule between two items.

We illustrate with an example.

Example 1. Consider a log with 3 user sessions SS_1 , SS_2 , and SS_3 , as follows.

Table 1: Example Query Log

Query Log	
SS_1	Q_a
	Q_b
	Q_c
SS_2	Q_a
	Q_b
	Q_d
SS_3	Q_a
	Q_e

This query log is mapped into 3 transactions, one for each user session, as follows: $T_1 = \{Q_a, Q_b, Q_c\}$, $T_2 = \{Q_a, Q_b, Q_d\}$, and $T_3 = \{Q_a, Q_c, Q_e\}$. The set I of items is given by $I = \{Q_a, Q_b, Q_c, Q_d, Q_e\}$.

Once we have mapped queries into items and user sessions into transactions, we proceed to determine association rules for 2 itemsets, i.e., we are only interested in relations between pairs of queries. We consider that two queries are related if there is an association

²The authors acknowledge the inherent imprecision of using IP addresses as user identifiers since distinct users may share the same IP addresses due, for instance, to a proxy or nat server. Even though, since user sessions are short lived, an IP address reasonably identifies a user for the purposes of our study.

rule between them. For instance, assume that a user query Q_a appears accompanied frequently by a second user query Q_b in various user sessions. Then, the association rule $Q_b \rightarrow Q_a$ captures the semantics that Q_b is a query related to the query Q_a . Further, we can frequently interpret that Q_b provides a more specific alternative formulation of Q_a and thus, can be used to expand the terms of Q_a . Let us illustrate with an example.

Example 2. Consider the query log illustrated in Example 1. With respect to Q_a , considering a minimum confidence of 100% and a minimum support greater than 1 session, two association rules can be computed are as follows:

$$\begin{aligned} Q_b &\rightarrow Q_a \\ Q_c &\rightarrow Q_a \end{aligned}$$

Notice that these two rules are more *popular* than the other rules, because they appear in 2 user sessions. Further, both Q_b and Q_c provide alternative specific formulations for Q_a . That is, using either Q_b or Q_c to expand Q_a will narrow the context of query Q_a . Thus, we define the *relation* operator $r(Q_a, Q_b)$ as the relationship between the query Q_a and its alternative specific formulation Q_b . Notice that the direction of the operator is the opposite of the association rule $Q_b \rightarrow Q_a$.

The rule $Q_a \rightarrow Q_b$ could also have been defined to expand Q_b . However, in this case, Q_a appears at the left hand side of the rule and would likely provide a *generic* alternative formulation of Q_b , which is frequently not as useful for expanding Q_b . It happens because the association $Q_a \rightarrow Q_b$ has a confidence lower than the minimum threshold defined for this example. This is a subtle, and yet critically important, distinction in the context of our work.

Given a log of Web queries, we determine all relations involving any pair of queries. The output of this procedure is the list R_i (possibly empty) of queries related to every query Q_i present in the log.

Simplicity is one key advantage of our approach. Because of that, we can compute the relation between queries very quickly, which means that new association rules can be updated periodically to identify new groups of related queries. This feature is important since the topics queried on the Web are dynamic and new relations may arise every day.

3.2 Building a Query Relations Graph

Using the query log and the set of association rules computed from it, we can find past queries related to the current user query (in fact, we do more than this, we identify related *concepts* as we later discuss). For this, we look for a query in the log that matches the current query exactly, i.e., one that contains exactly the same terms that compose the current query (if there is none, our approach cannot be applied). Let Q_a be this query. Then look for queries related to Q_a following the relations we computed. Besides the queries directly related to Q_a , we also inspect queries that are transitively related to Q_a . That is, if $r(Q_b, Q_c)$ and $r(Q_a, Q_b)$ then Q_c might also be a candidate for expanding Q_a . However, we only follow transitive relations if the queries are *firmly* related to each other, as we later explain in Section 3.3.

To follow transitive relations we build a query relations graph for the current query Q_a (we use Q_a to refer both to the current query and to the past query that matches it exactly), defined as follows. Let R_a be the set of queries in the log related to past query Q_a (and thus, to the current user query). For each pair of queries $Q_i, Q_j \in R_a$, we look for the relations of the form $r(Q_i, Q_j)$. Using these relations, we build a query relations graph for Q_a . This graph is referred to as G_a and is built as follows. A vertice is created in

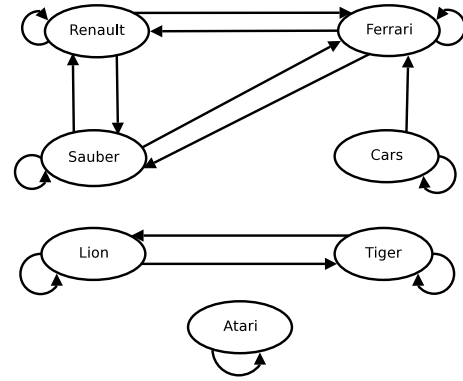


Figure 1: Query relations graph for the query “jaguar”.

G_a for each query in the set R_a . Let $Q_i \in R_a$ and $Q_j \in R_a$ be two vertices of G_a . For each relation $r(Q_i, Q_j)$, we insert in G_a a directed edge from Q_i to Q_j (meaning that Q_j provides a more specific alternative formulation for Q_i).

Figure 1 illustrates the query relations graph for Q_a =“jaguar”. We notice that “sauber” can be used to expand “ferrari”, providing an alternative specific formulation. The converse in this case is also true. We notice further that “ferrari” can be used to expand “cars”, providing a more specific variation. In this case, the converse rule leads to a more generic variation, using “cars” to expand “ferrari” (which might still be useful after all).

3.3 Identifying Concepts

Once a query relations graph G_a for the current user query Q_a has been built, we can use it to identify related *concepts*. Concepts are the minimal cycles in the graph, formally:

Definition 9. A concept C_j is a subset of the nodes in G_a such that starting from any node $Q_i \in C_j$ we can visit all nodes in C_j and return to Q_i , without visiting a node twice. Further, there is no other concept C' such that $C \subset C'$.

Our definition of a concept allows identifying groups of queries that appear together repeatedly. Figure 2 shows the concepts associated with the query Q_a =“jaguar”, according to its corresponding query relations graph illustrated in Figure 1. We notice that for distinct concepts are identified, as follows.

$$\begin{aligned} \text{Concept}_1 &= \{\text{renault}, \text{ferrari}, \text{sauber}\} \\ \text{Concept}_2 &= \{\text{cars}\} \\ \text{Concept}_3 &= \{\text{atari}\} \\ \text{Concept}_4 &= \{\text{lion}, \text{tiger}\} \end{aligned}$$

Further, in this case, the concepts identified clearly refer to *reasonable* alternative formulations for the query “jaguar”.

Our whole motivation in this work is to expand the original query “jaguar” with one of its alternative formulations, providing more context to the original query, as we now discuss.

4. CONCEPT-BASED QUERY EXPANSION

Once we have identified a set of concepts related to the user query, we need to determine which concept better satisfies the user information need. For this, we ask the user to select the concept that better suits his information need at the moment.

While this implies that the user has to provide feedback information, this feedback is of a high level nature and simple to provide.

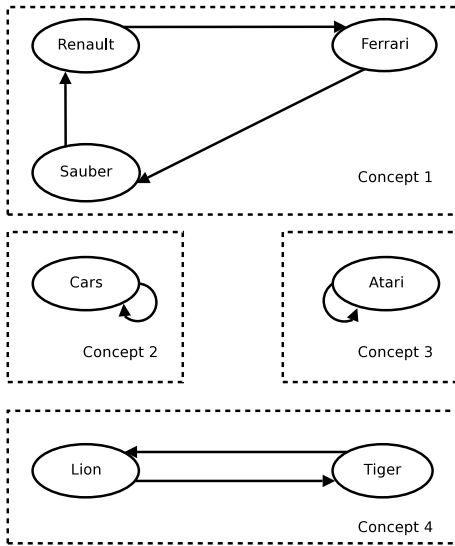


Figure 2: Concepts for the the query Q_α ="jaguar".

In fact, consider the case illustrated in Figure 2, which was obtained from the query log we will use in our experimentation later on. It is simple for the user to determine which of the four available concepts better suites his information need. Compare for instance with the case studied in [15] in which the user selects clusters generated from documents in the answer set. In that case, the clusters were labelled by sets of terms extracted from the documents, making it difficult for the user to interpret the meaning of each cluster. In our approach, the user selects concepts whose labels are subsets composed of past queries. As illustrated in Figure 2, these concepts are high level entities whose meaning is easily identifiable by inspection.

Once the user has selected one concept related to the query, we add this concept to the original user query and the expanded query is processed. To illustrate this expansion process, consider that the user is presented with the four concepts associated with the query "jaguar", illustrated in Figure 2, and that he selects the concept 4 labelled "lion, tiger". Then the original query is expanded to become "jaguar AND (lion OR tiger)".

This approach is simple, intuitive, and yet effective. In fact, in the experiments we later discuss we observed gains in average precision figures of roughly 32%. And these gains were obtained by the simple selection of a concept for feedback by the user.

While our experimentation suggests that our approach is promising, we observed that the adoption of a conjunctive operator (to restrict the semantics of the original query) might be too restrictive. To illustrate this point consider a concept that provides a synonym relation to the user query. Then, our adoption of a conjunctive operator implies that both the synonym and the original query terms need to appear in a document that is to be retrieved. However, in this case of a synonym relation, if just the synonym appears the document might be of interest to the user. This suggests providing some form of flexibility in the use of distinct operators for combining the original query with the concept used to expand it. Fortunately, in our approach there is a simple and intuitive form of changing this operator, as follows.

We classify the concepts to be used for query expansion into four types: (1) synonym, (2) specialization, (3) generalization, and (4) association. When a user selects a concept for query expansion, he might additionally indicate whether the selected concept provides a

synonym, specialization, generalization, or association relation for the current query. This is important because distinct concept types lead to different query processing strategies, as we now discuss.

- **Synonym:** the concept describes a synonym relation. For instance, consider the query "playstation" and the concept "playstation 2, sony playstation". In this case, the expanded query is processed as "playstation OR (playstation 2 OR sony playstation)".
- **Specialization:** the concept describes a more specific semantics for the original query. For instance, consider the query "brazilian president" and the concept "Luiz Inacio Lula da Silva". In this case, the expanded query is also processed as "brazilian president OR Luiz Inacio Lula da Silva".
- **Generalization:** the concept describes a more generic semantics for the original query. For instance, consider the query "jaguar" and the concept "Formula One, Formula one teams". In this case, the expanded query should be processed as "jaguar AND (Formula one OR Formula one teams)".
- **Association:** the concept describes an associative semantics for the original query. For instance, the query "tourism" and the concept "beach, surf". In this case, the expanded query should be processed as "tourism AND (beach OR surf)".

Our option is to use a conjunction with concepts of a more *generic* nature. With concepts more closely related to the user query, we use a disjunction. This is because, for instance, a synonym concept provides an interesting match for a document even if the terms specified in the original query do not appear in that document.

5. EXPERIMENTAL SETUP

In this section we discuss our experimental setup. We also discuss how we fine tuned parameters of our concept-based algorithm such as minimum support and minimum confidence.

5.1 The WBR Reference Collection

In our experimentation we used a collection of Web pages, referred to as the WBR reference collection, collected by the TodoBR³ search engine. This collection is composed of 14,594,308 pages of the Brazilian Web, under the domain ".br". Table 2 presents the main characteristics of this collection.

Table 2: Characteristics of the WBR Reference Collection

Characteristics	
Number of Documents	14,594,308
Number of Distinct Terms	4,149,801
Size (GB)	72

With regard to the user queries, we assembled a query log composed of query samples covering a period of 14 months from January of 2004 to February of 2005. This query log was divided into two parts: a first one covering the 12 months from January of 2004 to December of 2004 and a second one covering the 2 months from January of 2005 to February of 2005. The first part of the query log was used to build a query relations graph for all queries. The second part was used for testing our concept-based query expansion method. Table 3 presents the main characteristics of our query log.

While the first part of the log is composed of 913,076 distinct query samples, the second part is composed of 100 selected test

³<http://www.todobr.com.br>

Table 3: Characteristics of Our Query Log

Query Log		
Jan/2004 to Dec/2004	Number of Distinct Queries	913,076
	Number of Sessions	4,254,460
Jan/2005 to Feb/2005	Number of Distinct Queries	182,017
	Number of Test Queries	100
	Avg. Terms per Query	1.76

queries. These queries were selected according to a 2 steps procedure: (a) select the 50 most frequent queries excluding queries related to sex and (b) select 50 queries randomly excluding the ones previously selected and queries related to sex. The average number of keywords per selected test query is 1.76.

The execution of these 100 test queries in the TodoBR search engine, which implements a modern ranking function that considers link analysis, provided the results that we take as our *baseline*.

For each of the 100 test queries we determined likely meanings (information needs) that can be associated with them. We ended up with 284 information needs associated with the 100 test queries. Table 4 shows some of the information needs that were associated with our test queries.

Table 4: Information Needs Associated with Some of Our Test Queries

Test Query	Information Need
games	games on line
soccer	standings of soccer championships
music	download of musics

Using the first part of our query log, we computed all association rules involving any pair of distinct queries. Following, we applied our concept-based expansion method to the 100 test queries. This was done as follows.

We asked the assistance of 10 volunteer users, all of them familiar with the Web and search engines. We divided the 284 information needs to the 10 users. For each test query we showed a description of a valid information need (i.e., an information need that we associated with a query), and a list of concepts related to the query generated by our approach⁴. For each query, the user then selected the one concept that he believed better described the information need that was presented to him. The original query was expanded with the concept selected and then processed. This strategy is referred to as *concept-based*.

In a second round of experimentation we also asked the user to specify whether the concept selected provided a synonym, specialization, generalization, or association relation with regard to the original query. This was simple to determine and took almost no extra effort from the user, since the nature of the relation was invariably clear. But, it was important because the extra gains in average precision figures were considerable. This second strategy is referred to as *concept-type-based*.

In the feedback process the user had the option of not choosing any of the concepts. In this case, he selected the option *no concepts apply*. For the 284 information needs described, the users found meaningful concepts to 153 (53.9%) of them. That is, in our concept-based approach the users found that the suggestions could be used to complement or describe the intended semantics for the query 53.9% of the time.

⁴Notice that if various distinct information needs were associated with a same query, each of them was treated as a separate query.

We took the 153 information needs for which the user selected a concept and, for each of them, composed a query pool formed by the top 10 ranked documents retrieved by each of the three ranking algorithms we consider (baseline, concept-based, and concept-type-based). Top-10 documents are the most relevant in the web context where users frequently focus only on the first page. This is the same pooling method used for the Web-based collection of TREC [14]. Each query information need contained an average of 23.86 documents. These documents were then manually evaluated for relevance by a pool of specialists with high familiarity in Web searching, resulting in an average number of relevant documents per query information need of 10.62.

5.2 Tuning the Concept Generation Algorithm

To generate the association rules of interest and label them properly, we need to determine values for minimum support, minimum confidence, and minimum inverse concept frequency, as we now discuss. These are the three parameters used for tuning our concept generation algorithm. They are set offline and that do not increase the time performance of our method. On the contrary, since they allow pruning the set of query relations they lead to faster computation.

Minimum Support

We wanted to consider only the query relations that are statistically meaningful (i.e., that occur with a minimum frequency or support). This pruning strategy is used to prevent that statistically non-relevant relations arise from the user sessions.

Figure 3 illustrates the distribution of support values for all the relations that can be generated from our query log. We observe that query relations with support values lower than 3 occur too frequently and are not very discriminative. Thus, we set the minimum support value to 3, which excludes 93% of the query relations from consideration. The remaining 7% provide strong query relations and are maintained.

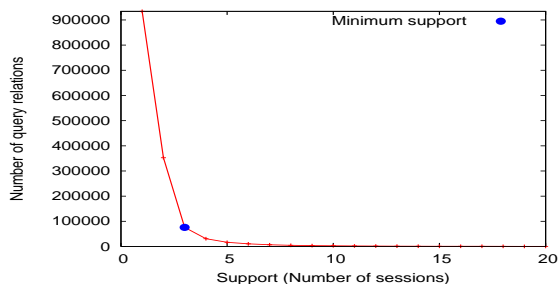


Figure 3: Distribution of support values.

Minimum Confidence

Of the remaining 7% query relations we were left with after the cut done using the minimum support value of 3 sessions, we wanted to keep only the strongest ones. This can be accomplished by varying the minimal confidence, which allows exploring a trade-off between accuracy and the number of query relations that are used. A higher value for the minimum confidence leads to a smaller number of query relations and to faster computation. We aim at reducing the number of query relations without discarding relations that are meaningful for query expansion purposes.

Figure 4 illustrates the distribution of confidence values for the query relations we were left with after the cut by the minimum support. We observe that query relations with confidence values

lower than 20% occur too frequently and are not very discriminative. Thus, we set the minimum confidence threshold to 20% and discarded all query relations with smaller values of confidence (i.e., we were left only with the strongest query relations).

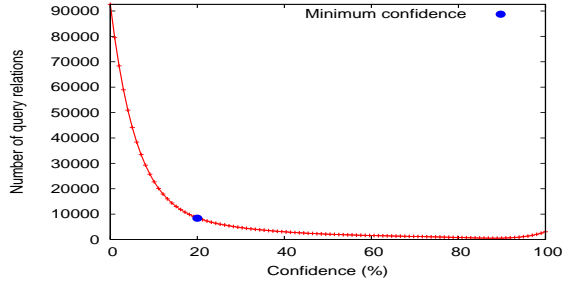


Figure 4: Distribution of confidence values.

Minimum Inverse Concept Frequency

Using the remaining query relations we were left with after the cut done using the minimum confidence of 20%, we generated an all-queries relation graph. Cycles in this graph identify the concepts we work with. The last step is to label each of these concepts. For this, we simply use the list of queries that compose the concept. However, queries that appear too frequently work as stopwords introducing noise in our labelling procedure. For instance, a query such as “mp3” might cover a large number of concepts and so does not provide any discriminative effect. We remove these queries from our query relations graph before labelling the concepts. This is accomplished as follows.

Let ICF be the *inverse concept frequency* of a query. This is defined as follows.

$$ICF(q_i) = \log \frac{N_c}{n_i}$$

where N_c is the total number of concepts in our system and n_i is the number of concepts that contain the query q_i .

Figure 5 shows the distribution of ICFs. We notice that a few queries have relative low values of ICF (to the left of the minimum ICF shown). We remove these queries from the system.

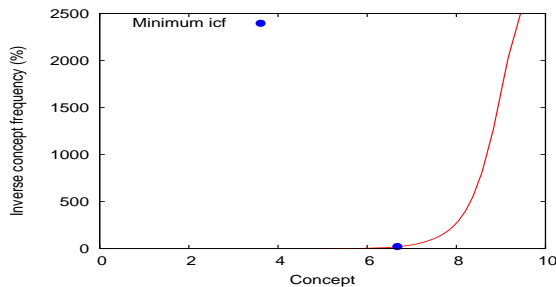


Figure 5: Distribution of inverse concept frequencies.

The minimum value of ICF we used in our experiments was 6.67. Notice that this value can be set slightly different and that this is not important. The important point is to adjust the minimum ICF near the knee of the distribution shown, eliminating queries that appear too frequently in the description of the concepts that are generated. This cleans up the labels associated with the concepts, facilitating the task of the user when it is time to select a concept for feedback.

6. EXPERIMENTAL RESULTS

In this section we describe our experimental results. We quantify retrieval effectiveness through standard measures of average recall and precision. We employed two aggregate metrics: (i) standard 11-point average precision figures and (ii) average precision over the retrieved documents.

Our analysis is based on a comparison to the ranking algorithm of the TodoBR search engine, which combines information extracted from the document text, the anchor text, and hub and authority link values, as described in [8]. That is, our baseline is a modern Web ranking algorithm that combines text related information with link analysis.

We provide results for three distinct ranking algorithms: the baseline, the concept-based expanded query, and the concept-type-based expanded query. In the concept-type-based case, the user specifies not only the concept to expand the query but also the type of the relation between the concept and the query (i.e., synonym, specialization, generalization, association).

Figure 6 shows the 11-point average precision figures for the baseline, the concept-based, and the concept-type-based algorithms. We observe that the concept-based and the concept-type-based algorithms yield better precision than the baseline, regardless of the recall level. The concept-type-based ranking yields the largest gains in average precision, which suggests that obtaining information on the type (nature) of the query relation can be of great value.

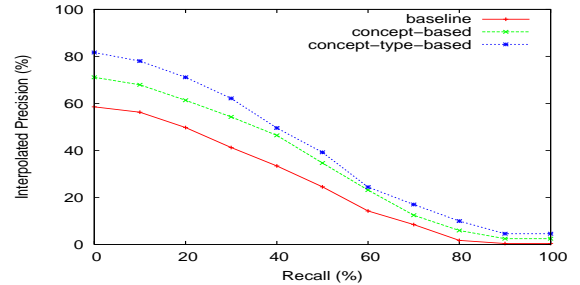


Figure 6: Precision-recall curves for the baseline, the concept-based, and the concept-type-based algorithms, relative to the WBR test collection.

Overall average precision for the baseline, the concept-based, and the concept-type-based algorithms is presented in Table 5. The gains in average precision were of 32.26% for the concept-based and of 52.99% for the concept-type-based with regard to the baseline. Our approach outperforms the baseline because its disambiguation scheme can be used to successfully describe user information needs, capturing the concepts related to the user search experience.

Table 5: Baseline, Concept-based, and Concept-type-based average precision figures for the WBR collection.

Approach	Avg. Precision (%)	Gain (%)
Baseline	26.29	-
Concept-based	34.77	32.26
Concept-type-based	40.22	52.99

In terms of time performance, a simple implementation of our concept generation algorithm, running on a standard PC server powered by a 1GHz Intel Celeron CPU, took 300 milliseconds on average to produce the concepts related to a given query (the average

was computed over the 100 test queries). This is fairly reasonable, since it is similar to the average time to process a Web query. Further, this average processing time was obtained without the benefits of any form of indexing or optimization, which means that much shorter times can be obtained with an optimized version.

7. CONCLUSIONS

We have proposed a concept-based query expansion for Web users. Our motivation was to provide the user with reasonable and meaningful suggestions whenever he specifies a vague (short) query that has also occurred in the past. For that we looked at past user sessions for queries that had co-occurred with the current user query i.e., are related to the current query.

Our approach is distinct because of the framework we proposed for mining, identifying, and labelling query relations. Query relations were modelled as association rules. To capture transitive relations we assembled a query relations graph. To restrict the number of relations to consider, we looked for cycles in the query relations graph. These cycles were interpreted as references to concepts (entities) of the world related to the current query.

Given a test query, the user was faced with concepts related to that query. He then selected a single concept that he interpreted was most strongly related to the query. This concept was used to expand the original query and the expanded query was processed.

Using a Web reference collection, we validated our approach. Our results indicated gains in average precision of roughly 32% with regard to a modern Web ranking function (that takes link analysis into consideration). Further, with additional feedback from the user (on the type of relation between the selected concept and the query) this gain in average precision went up to roughly 52%.

These are interesting results that suggest that considerable gains in precision can be obtained with little extra effort from the user.

8. ACKNOWLEDGMENTS

This work was supported in part by the GERINDO project-grant MCT/CNPq/CT-INFO 552.087/02-5, by CNPq grant 520.916/94-8 (Nivio Ziviani) and by CNPq grant 30.0188/95-1 (Berthier Ribeiro-Neto).

9. REFERENCES

- [1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings Of The ACM Sigmod International Conference Management Of Data*, pages 207–216, Washington, d.c., May 1993.
- [2] P. Anick. Using terminological feedback for web search refinement: a log-based study. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval (SIGIR'03)*, pages 88–95, New York, NY, USA, 2003. ACM Press.
- [3] P. G. Anick and S. Tipirneni. The paraphrase search assistant: terminological feedback for iterative information seeking. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'99)*, pages 153–159, New York, NY, USA, 1999. ACM Press.
- [4] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, 1999.
- [5] B. Billerbeck, F. Scholer, H. E. Williams, and J. Zobel. Query expansion using associated queries. pages 2–9, New Orleans, USA, 2003.
- [6] B. Billerbeck and J. Zobel. When query expansion fails. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval (SIGIR'03)*, pages 387–388, Toronto, Canada, 2003.
- [7] C. Buckley, G. Salton, J. Allan, and A. Singhal. Automatic query expansion using SMART: Trec-3 report. In *Proceedings of the Third Text REtrieval Conference*, pages 69–80, 1995.
- [8] P. Calado, B. Ribeiro-Neto, N. Ziviani, E. Moura, and I. Silva. Local versus global link information in the web. *ACM Transactions on Information Systems (TOIS)*, 21(1):42–63, 2003.
- [9] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *Proceedings of the Eleventh International Conference on the World Wide Web (WWW'02)*, pages 325–332, Hawaii, USA, 2002.
- [10] N. Eiron and K. McCurley. Analysis of anchor text for web search. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval (SIGIR'03)*, pages 387–388, 2003.
- [11] L. Fitzpatrick and M. Dent. Automatic feedback using past queries: social searching? In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval (SIGIR'97)*, pages 387–388, Philadelphia, USA, 1997.
- [12] B. M. Fonseca, P. B. Golgher, E. S. de Moura, B. Pôssas, and N. Ziviani. Discovering search engine related queries using association rules. *Journal of Web Engineering*, 2(4):215–227, 2004.
- [13] A. Geyer-Schulz and M. Hahsler. Evaluation of recommender algorithms for an internet information broker based on simple association rules and on the repeat-buying theory. In B. Masand, M. Spiliopoulou, J. Srivastava, and O. R. Zaiane, editors, *Proceedings of Fourth WebKDD Workshop: Web Mining for Usage Patterns & User Profiles*, pages 100–114, Edmonton, Canada, July 2002.
- [14] D. Hawking, N. Craswell, P. B. Thistlewaite, and D. Harman. Results and challenges in web search evaluation. *Computer Networks*, 31(11–16):1321–1330, May 1999. Also in Proceedings of the 8th International World Wide Web Conference.
- [15] M. A. Hearst and J. O. Pedersen. Reexamining the cluster hypothesis: scatter/gather on retrieval results. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'96)*, pages 76–84, New York, NY, USA, 1996.
- [16] C.-K. Huang, L.-F. Chien, and Y.-J. Oyang. Relevant term suggestion in interactive web search based on contextual information in query session logs. *Journal of the American Society for Information Science and Technology (JASIST)*, 54(7):638–649, 2003.
- [17] R. Kraft and J. Zien. Mining anchor text for query refinement. In *Proceedings of the 13th International Conference on the World Wide Web (WWW'04)*, pages 666–674, New York, USA, 2004.
- [18] W. Lin, S. Alvarez, and C. Ruiz. Efficient adaptive-support association rule mining for recommended systems. *Data Mining and Knowledge Discovery*, 6(1):83–105, 2002.
- [19] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine. In *Proceedings of the Tenth International Conference on the World Wide Web (WWW'01)*, pages 162–168, Hong Kong, Hong Kong, 2001.