Diversifying Search Results Using Query Logs

Gabriele Capannini, Franco Maria Nardini, Raffaele Perego, Fabrizio Silvestri

ISTI - CNR, Pisa, Italy
{name.surname}@isti.cnr.it

ABSTRACT

We study the problem of diversifying search results by exploiting the precious knowledge stored in query logs. When an ambiguous or faceted query is submitted to a search engine, the user has often to reformulate it in order to better satisfy her information need by improving the perceived precision of results returned. Our proposal exploits the presence of different "specializations" of queries in query logs to detect the submission of ambiguous/faceted queries, and manage them by diversifying the search results returned to users, in a way that better covers the different possible interpretations of the query. We present an original formulation of the results diversification problem in terms of an objective function to be maximized that admits the finding of an optimal solution in linear time. We evaluate our approach by using both the TREC Web diversification track methodology, and a novel evaluation methodology based on query log analysis.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Query formulation, Search process, Selection process*

General Terms

Algorithms, Design, Experimentation

Keywords

Query log analysis, search results diversification, diversification

1. INTRODUCTION

Web search engines are nowadays the most popular mean of interaction with the Web. Users interact with web search engines by typing a few keywords representing their information need. These keywords, however, are often ambiguous and have more than one possible interpretation [2, 19].

Consider, for example, the popular single-term query "apple". It might refer to Apple Corp., to the fruit, or to a tour operator which is very popular in the US. Without any further information that may help to disambiguate user intent, the search engine should produce a set of results possibly *covering* all (the majority of) the different interpretations of the query. To help users to find the right information they are looking for, many different UI (i.e. User Interface) solutions have been proposed to present results in the best and helpful way possible. The first, and naïve, solution has been paging the ranked result list. Instead of presenting the whole list of results, the search engine presents results divided in pages (also known with the term SERP, i.e. Search Engine Result Page) containing ten result each. Obviously, this does not enhance the satisfaction of users but serves as a way to help users to immediately recognize if their query formulation has given the desired outcome. When a search engine provides millions of results for a particular query, the searcher can either sift through the endless pages of results or depend on the search engine's judgment as to the most relevant results. Another possibility offered by some search engines is the so called *results clustering*. Search result clustering works by organizing search results into folders that group similar items together [26]. For a nice survey on search results clustering see Carpineto *et al.* [9]. Stemming from the idea that users typically do not go further the first page of results, *results diversification* [1] aims at "packing" the highest possible number diverse results within the first page. Results diversification is a hot research topics of these last years.

To see diversification in action we submitted the query "*rare*" to a popular search engine and we noticed that the first two results are about a company called rare, the third one is a non-governative organization, the following three are about the word definition, and so on and so forth. Figure 1 show the result page for the query "*rare*".

Other examples of diversification, similar to the one just presented, can be found in other queries. Now, an important question opens, then: *Is diversification really needed?* To answer, one should ask wether queries are *ambiguous* or not. We shall give a formal definition of ambiguity in user intent specification in the Section 3. For the moment, let us consider ambiguous a query with multiple meanings and, to be pragmatic, we consider a query ambiguous if it is followed immediately in a user's session by a query consisting of a superset of the terms of the first one. As example, consider the pair of queries "*da vinci*" and "*da vinci code*". The latter quite clearly indicates that user is not satisfied with

Submitted for confidential review.
June 2010.

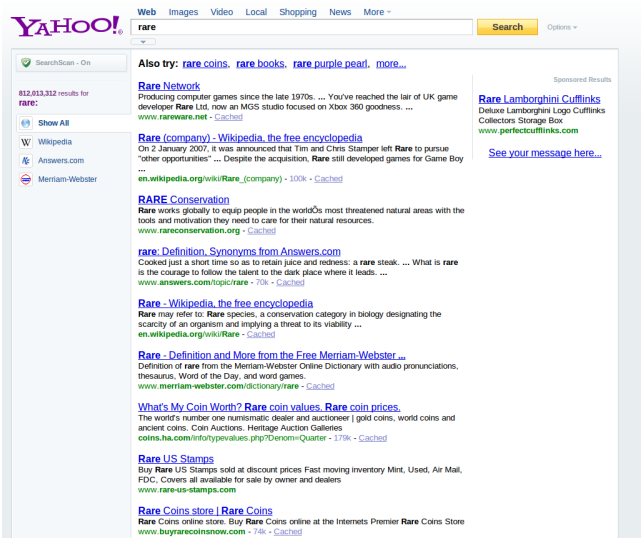

Figure 1: Results from the Yahoo! search engine for the query "rare".

the results of "*da vinci*" because she is not looking for the scientist but, instead, for the popular Dan Brown's book.

The question "*Are there many ambiguous queries?*" is partially answered by interpreting the data plotted in Figure 2.

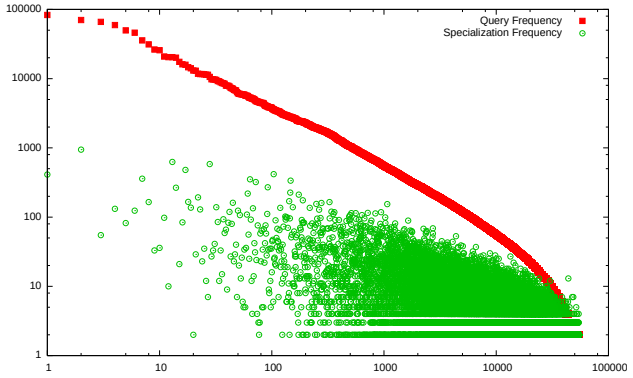

Figure 2: Popularity of "ambiguous" queries and of their "specializations" from the AOL query log.

The figure shows the popularity of ambiguous queries, and of their relative specializations taken from a large query log (in this case the America Online query log). From the plot we can see that the presence of such specialization chains is common in query logs. In the absence of result diversification, users might be unsatisfied from the results returned by the search engine for generic queries. In order to get some relevant results, they had in fact to better specify their information need by means of a new refined query. The possibility of diversifying search results to cover at least the most popular interpretations of these queries would avoid the submission of most of these second-step queries with a twofold advantage: minimizing the risk of dissatisfaction of the average user of the web search engine, and decreasing the computational load over it.

It is known that search services collect detailed information about the queries submitted in the past along with a lot of additional information such as the unique identifier of the user submitting the query, the pages viewed and clicked in the result set, the rank of each result, the exact time at which a particular action was done, etc. These data are extremely valuable for a number of different tasks [24] ranging from the refinement of the ranking function through the analysis of the implicit feedback about the relevance of clicked results, to the optimization of efficiency in several aspects of the search process such as query processing, result caching, snippets generation, etc. We claim that the richness and completeness of knowledge present in query log can be profitably exploited also to address the challenging problem of result diversification.

In this paper we propose a novel approach based on query log analysis to address results diversification. We aim to *detect* and *manage* ambiguous queries by selecting a set of results:

- *covering* the most common interpretations of these queries given in the past by the multitude of search engine users, and
- *maximizing the utility* of the page of results returned to the user for ambiguous queries.

Indeed, through query log analysis we are introducing the following novel contributions to result diversification:

- a methodology to detect the ambiguous queries that would benefit from diversification based on exploiting behaviors of past users;
- a methodology to efficiently and effectively devise the possible topics to include in the diversified list of results along with their probability distribution. Our solution differently from most competitor solutions does not require any taxonomy, or classification model, to be built in advance on queries, but is based on the use of state-of-the-art query recommendation algorithms: namely Query-Flow Graph [5], and Search Shortcuts [4];
- a linear time diversification algorithm re-ranking the result list on the basis of the set and probabilities of query refinements mined from the log. Our re-ranking method outperforms the state-of-the-art technique by Agrawal *et al.* [1] and accounts for the real proportion of users requesting a given ambiguous query meaning. For example if 50% of users asks for *Apple corp.*, we reflect this ratio by forcing the result list to have at least 50% of the results relevant for apple corp.; We also adapt our measures to be used in the original algorithm by Agrawal *et al.* [1] therefore showing that our specialization detection methodology can be used instead of a query categorization model as used by Agrawal *et al.*;
- an objective and meaningful diversification usefulness measure to assess how valuable a diversified result list is.

In our formulation, result diversification is defined as the problem of maximizing the quality of the results returned for ambiguous queries by taking into account the past specializations of such queries present in the query log. We

show that our formulation of the problem is solvable in linear time as opposed to other known methods. The approach proposed is evaluated by using the dataset made available for the 2009 TREC Web Diversity Track. Results computed show that our approach remarkably outperforms others previously proposed. Furthermore, we applied our linear time method to the ambiguous queries extracted from two large query logs from the MSN and AOL commercial search engines. The results of the experiments conducted showed that we are able to improve **from 5 up to 10 times** the "*usefulness*" of results lists returned for these ambiguous queries.

The paper is organized as follow: Section 2 discusses related works. Section 3 presents a formalization of the problem, the specialization extraction methods proposed, and the algorithms we propose. Section 4 discusses some experimental results. In Section 5 we present our conclusions and we outline possible future work.

2. RELATED WORK

The result diversification problem recently attracted a lot of interest as showed by the increasing number of papers published on this topic in major conferences. A very important pioneering work on diversification is [8]. In this paper the authors present the Maximal Marginal Relevance (MMR) problem, and they show how a trade-off between novelty and relevance of search results can be made explicit through the use of two different functions, the first measuring the similarity among documents, and the other the similarity between documents and the query. The trade-off is controlled by means of a parameter that allows the weight given to the two similarity metrics to be tuned. Also Zhai *et al.* [29] stated that in general it is not sufficient to return a set of relevant results as the correlation among the returned results returned is also very important. In a later work, Zhai *et al.* [28] formalize and propose a risk minimization approach that allow an arbitrary *loss* function over a set of returned documents to be defined. *Loss* functions aim at determining the *dissatisfaction* of the user with respect to a particular set of results. Such *loss* function depend on the language models rather than on categorical information about two documents [27].

Diversification has also been studied for purposes different from search engine result diversification. Ziegler *et al.* [30] study the diversification problem from a "recommendation" point of view. Radlinski *et al.* [20] propose a learning algorithm to compute an ordering of search results from a diverse set of orderings. Vee *et al.* [25] study the diversification problem in the context of structured databases with applications to online shopping. Clarke *et al.* [12] study diversification in question answering.

Agrawal *et al.* [1] present a systematic approach to diversify results that aims to minimize the risk of dissatisfaction of the web search engine users. Furthermore, authors generalize some classical IR metrics, including NDCG, MRR, and MAP, to explicitly account for the value of diversification. They show empirically that their algorithm scores higher in these generalized metrics compared to results produced by commercial search engines.

Gollapudi *et al.* [14] use the axiomatic approach to characterize and design diversification systems. They develop a set of axioms that a diversification system is expected to satisfy, and show that no diversification function can satisfy all these axioms simultaneously. They provide three example diversification objectives satisfying different subsets of the axioms. Finally, they propose an evaluation methodology to characterize the objectives and the underlying axioms. They conduct a large scale evaluation based on data derived from Wikipedia, and a product database.

Rafiei *et al.* [21] model the diversity problem as expectation maximization and study the challenges of estimating the model parameters and reaching an equilibrium. One model parameter, for example, is the correlation between pages which authors estimate using textual contents of pages and click data (when available). They conduct experiments on diversifying randomly selected queries from a query log and the queries chosen from the disambiguation topics of Wikipedia.

Cheng *et al.* [10] propose a series of technologies to fulfill the task of actively predict search intent from user browsing behavior data. First, they extract all the queries that users issued after reading a given page. Second, they learn a model to effectively rank these queries according to their likelihoods of being triggered by the page. Third, since search intents can be quite diverse even if triggered by the same page, they propose an optimization algorithm to diversify the ranked list of queries obtained in the second step, and then suggest the list to users. They tested the approach on large-scale user browsing behavior data obtained from a commercial search engine.

Clough *et al.* [13] examine user queries with respect to diversity. They analyze 14.9 million queries from the MSN query log by using two query log analysis techniques (click entropy and reformulated queries). Authors found that a broad range of query types may benefit from diversification. They also found that, although there is a correlation between word ambiguity and the need for diversity, the range of results users may wish to see for an ambiguous query stretches well beyond traditional notions of word sense.

Santos *et al.* [23, 22] introduce a novel probabilistic framework (xQuAD) for Web search result diversification, which explicitly accounts for the various aspects associated to an underspecified query. In particular, they diversify a document ranking by estimating how well a given document satisfies each uncovered aspect and the extent to which different aspects are satisfied by the ranking as a whole. Authors evaluate the xQuAD framework in the context of the diversity task of the TREC 2009 Web track. They exploit query reformulations provided by three major Web search engines (WSEs) as a means to uncover different query aspects.

Similarly to [22], we exploit related queries as a means of achieving diversification of query results. Nevertheless, our approach is very different from the above two. In [22], the authors exploit query reformulations provided by commercial Web search engines as a means to uncover different query aspects of ambiguous queries. All these aspects are then taken in consideration during query processing in order to achieve a result set covering all these aspects. In our case the different meanings and facets of queries are instead disclosed by analyzing user behaviors recorded in query logs. During the analysis also the popularity of the different specializations is derived that is used to maximize the "usefulness" of the final set of documents returned. An approach orthogonal to our is instead investigated by Radlinski and Dumais in [19] where the problem of generating queries that can yield to a more diverse set of retrieved documents is studied starting from the observation that the top-k results

retrieved for a query might not contain representative documents for all of its interpretations.

3. QUERY-LOG BASED DIVERSIFICATION

We assume that a query log Q is composed by a set of records $\langle q_i, u_i, t_i, V_i, G_i \rangle$ registering, for each submitted query q_i: (i) the anonymized user u_i; (ii) the timestamp t_i at which u_i issued q_i; (iii) the set V_i of URLs of documents returned as top-k results of the query, and, (iv), the set C_i of URLs corresponding to results clicked by u_i.

Users generally query a search engine by submitting a sequence of requests. Splitting the chronologically ordered sequence of queries submitted by a given user into *sessions*, is a challenging research topic [18, 3, 16]. Three different hierarchy in the concept of user session can be devised: (i) *Supersessions*, (ii) *Physical Sessions*, and (iii) *Logical Sessions*.

Supersession. A supersession is the sequence of all the queries of a given user recorded in the query log, ordered by timestamp.

Physical Sessions. A physical session is defined as the sequence of queries issued by the same user within a given time interval. A typical timeout threshold used in web log analysis is $t_0 = 30$ minutes [18].

Logical Sessions. A logical session [3] is a topically coherent sequence of queries. A logical session is not strictly related to timeout constraints but collects all the queries a user submitted being motivated by the same information need (i.e. planning an holiday in a foreign country, gathering information about a car to buy, and so on).

Obviously a physical session can contain one or more logical sessions. Jones et al. [16] introduced the concepts of *mission* and *goal* to consider coherent information needs at different levels of granularity, being a goal a sub-task of a mission (i.e., booking the flight is one of the goal in the more general mission of organizing an holiday). Since session splitting methodologies are out of the scope of this paper, we resort to adopt a state-of-the-art-technique to devise user logical sessions whose discovering is very important for our result diversification framework.

We thus adopted the session splitting solution based on Query-Flow Graph, which consists in building a Markov Chain model of the query log and subsequently finding paths in the graph which are more likely to be followed by random surfers. For a complete coverage of the method we refer to the original papers [5], and [6]. Thus, by processing the query log Q we obtain the set of logical user sessions exploited by our result diversification solution which is entirely based on information mined from query logs. Both the query topics possibly benefiting from diversification, and the probability of each distinct specialization among the spectrum of possibilities, are mined from logs storing historical information about the interaction of users with the WSE.

As an example, let us assume that in a given query log the queries *leopard mac OS X*, *leopard tank*, and *leopard pictures*, are three specializations of query *leopard* that commonly occur in logical query sessions. The presence of the same query refinements in several sessions issued by different users gives us evidence that a query is ambiguous, while the relative popularity of its specializations allow us to compute the probabilities of the different meanings. On the basis of these information learned from historical data, once a query q is encountered by the WSE, we: (a) check if q is ambiguous or faceted, and if so, (b) exploit the knowledge about the different specializations of q submitted in the past to retrieve documents relevant for all of them. Finally, (c) use the relative frequencies of these specializations to build a final result set that maximize the probability of satisfying the average user. In the following we describe more precisely how ambiguous/faceted queries are detected and managed.

Algorithm AmbiguousQueryDetect$(q, \mathcal{A}, f(), s)$
INPUT:
q: the currently submitted query;
$\mathcal{A}$: a query recommendation algorithm trained with Q;
$f() : Q -> \mathcal{N}$: a function returning query popularity in Q;
s: a parameter determining the sensitiveness of the algorithm.

OUTPUT:
if q is ambiguous, the set of popular specializations along with their probabilities is returned, $\emptyset$ otherwise.

1. $\overline{S_q} \leftarrow \mathcal{A}(q)$; *compute the set of specializations of q present in Q and returned as related queries by $\mathcal{A}$.*
2. $S_q \leftarrow \{q_i \in \overline{S_q} | f(q_i) \geq \frac{f(q)}{s}\}$; *select the popular specializations of $\overline{S_q}$*
3. if $(|S_q| \geq 2)$ then $RETURN(S_q)$ else $RETURN(\emptyset)$;

Figure 3: Algorithm to determine if the current query can benefit from diversification.

3.1 Mining Specializations from Query Logs

Let q and q' be two queries submitted by the same user during the same logical session recorded in Q. We adopt the terminology proposed in [6], and we say that a query q' is a "specialization" of q if the user information need is stated more precisely in q' than in q. Given the above generic definition, any algorithm that exploit the knowledge present in query log sessions to provide users with useful suggestions of related queries, can be easily adapted to the purpose of devising specializations of submitted queries. Given the frequency $f(q)$ of a query topic q in Q, and a query recommendation algorithm $\mathcal{A}$ trained with query log Q, the *Ambiguous Query Detect* algorithm depicted in Figure 3 can be used to detect efficiently and effectively queries that can benefit from result diversification.

In this work we experimented the use of two different query recommendation algorithms for computing the possible specializations of queries. The algorithms tested are the one based on the query flow graph model presented in [6] (Query-Flow Graph), and the one derived from [4] and presented in a submitted paper [7] (Search-Shortcuts). In both cases we acknowledge the authors for kindly providing us their code. Both algorithms learn their suggestion model from the query log, and return as related specializations, only queries that are present in Q, and for which related probabilities can be thus easily computed.

DEFINITION 1 (PROBABILITY OF SPECIALIZATION). *Let $\widehat{Q} = \{q \in Q,\ s.t.\ |S_q| > 1\}$ be the set of ambiguous queries in Q devised with the above algorithm, and $P(q'|q)$ the probability for $q \in \widehat{Q}$ to be specialized from $q' \in S_q$.*

We assume that the distribution underlying the possible specialization of an ambiguous query is known and complete, i.e., $\sum_{q' \in S_q} P(q'|q) = 1$, and $P(q'|q) = 0, \forall q' \notin S_q,\ \forall q \in \widehat{Q}$. To our purposes these probability distributions are simply

estimated by dividing the frequency returned by algorithm in Figure 3 using the following formula:

$$P(q'|q) = f(q')/\sum_{q_i \in S_q} f(q_i)$$

Obviously, query logs can not give the complete knowledge about all the possible specializations for a given ambiguous query, but we can expect that the most popular interpretations are present in a large query log covering a long time period.

Now, let us give some additional assumptions and notations.

$\mathcal{D}$ is the collection of documents indexed by the search engine which returns, for each submitted query q, an ordered list R_q of r documents $R_q =< d_1, d_2, \ldots, d_r >$. The rank of document $d \in \mathcal{D}$ within result list R is indicated with $rank(d, R)$.

Moreover, let d_1 and d_2 be two documents of $\mathcal{D}$, and $\delta : \mathcal{D} \times \mathcal{D} \rightarrow [0,1]$ a distance function having the discriminative and symmetric properties (we do not require the distance function δ to be metric), i.e. (i) $d_1 = d_2 \Rightarrow \delta(d_1, d_2) = 0$, and (ii) $\delta(d_1, d_2) = \delta(d_2, d_1)$.

DEFINITION 2 (RESULTS' UTILITY). *The* utility *of a result $d \in R_q$ for a specialization q' is defined as:*

$$U(d|R_{q'}) = \sum_{d' \in R_{q'}} \frac{1 - \delta(d, d')}{rank(d', R_{q'})}. \quad (1)$$

where $R_{q'}$ is the list of results that the search engine returned for specialized query q'.

Such utility represents the goodness of $d \in R_q$ in satisfying a user intent that is better represented by specialization q'. The intuition for U is that a result $d \in R_q$ is more useful for specialization q' if it is very similar to an highly ranked item present in result list $R_{q'}$.

The utility function specified in Equation (1) uses a function to measure the distance between documents. In all our experiments we use a $\delta(d_1, d_2)$ function defined as follow:

$$\delta(d_1, d_2) = \begin{cases} 1 - cosine(d_1, d_2), & cosine(d_1, d_2) \geq c \\ 0, & otherwise \end{cases} \quad (2)$$

where $cosine(d_1, d_2)$ is the cosine similarity between the two snippets returned by the underlying search engine for a given query. We threshold the value of distance to be 0 when the similarity is below a given constant c. In our experiments we tested various values for c, i.e. $c = 0.4, 0.5, 0.6$. It is easy to prove that the function in Equation (2) is discriminative, i.e. for any two documents $d_1, d_2 \in D$, we have $\delta(d_1, d_2) = 0$ iff $d_1 = d_2$, and symmetric, i.e. $\delta(d_1, d_2) = \delta(d_2, d_1)$.

Using the above definitions, we are able to define two different approaches to diversification based on query logs. The first one is an instantiation of the method by Agrawal *et al.* [1], while the second one is a novel formulation proposed by us whose solution can be computed in polynomial, i.e. linear, time.

The QL_DIVERSIFY(K) *Problem.*

In a recent paper, Agrawal *et al.* [1] have defined a new problem, namely the DIVERSIFY(K) problem that is covering-like problem aimed to include the maximum number possible of "categories" into the list of k results that are selected to be returned in response to a user's query.

We briefly recall the definition of the problem as stated in the original paper [1].

> DIVERSIFY(K): Given query q, a set of documents D, a probability distribution of categories for the query $P(c|q)$, the quality values of the documents $V(d|q, c)$, and an integer k. Find a set of documents $S \subseteq D$ with $|S| = k$ that maximizes
>
> $$P(S|q) = \sum_c P(c|q) \left(1 - \prod_{d \in S} (1 - V(d|q, c))\right) \quad (3)$$

First of all, let us point out that Equation (3) uses two concepts similar to those we have defined above. The probability of a query to be part of a category is similar to our concept of probability of specialization (Definition 1) and the quality value $V(d|q, c)$ is analogous to our Results' Utility $U(r|R_{q'})$ (Definition 2).

We can see, in fact, the set of possible specializations S_q as the set of possible categories for q. The utility, then, can be seen as the utility of picking up result r for the category/specialization q'. We, then, seek a subset S of R with $|S| = k$ maximizing:

$$P(S|q) = \sum_{q' \in S_q} P(q'|q) \left(1 - \prod_{d \in S} \left(1 - \widetilde{U}(d|R_{q'})\right)\right) \quad (4)$$

In Equation (4) we use $\widetilde{U}(d|R_{q'})$, which is simply a normalization of $U(d|R_{q'})$ that makes the definition become the *probability* of being useful, instead of defining the marginal utility of a result d. The normalization factor is computed by assuming that, in the optimal case, result d is at distance $\delta(d, \cdot) = 0$. In this case, the utility function is equal to $\sum_{d' \in R_{q'}} \frac{1}{rank(d', R_{q'})} = \sum_{i=1}^{|R_{q'}|} 1/i = H_{|R_{q'}|}$, where $H_{|R_{q'}|}$ is the $|R_{q'}|$-th harmonic number. Therefore, $\widetilde{U}(d|R_{q'}) = \frac{U(d|R_{q'})}{H_{|R_{q'}|}}$.

We call this problem QL_DIVERSIFY(K) to differentiate it from the original Agrawal *et al.* [1] formulation. As shown by Agrawal *et al.* [1] the problem is intractable, yet the function defined is *submodular* and, thus, a greedy algorithm consisting in adding to the result set the documents giving the largest marginal increase to the objective function yields to a solution whose value is greater than $(1 - 1/e)$ the optimal [17].

We have implemented the algorithm IA-SELECT defined by Agrawal *et al.* [1] using our definitions of specialization probability and results' utility. We report results in Section 4.

The MAXUTILIY_DIVERSIFY(K) *Problem.*

In the original Agrawal *et al.* paper, the problem they define, actually maximize the *weighed* coverage of the categories with pertinent results. As also authors noted, this implies that the function does not actually seek to maximize the diversity but rather the probability of not having taken any useful result. This, in turn, means that the opti-

mal solution can be found when results useful for only one category/specialization are considered.

We are not satisfied by this analysis. We observe, instead, that it is possible to maximize the sum of the various utilities for the chosen subset S of documents.

Motivated by the above observation we define the following problem.

MAXUTILITY(K): Given query q, a set of results for q R_q, a probability distribution of specializations for the query $P(q_i|q)$, the utility values of the documents $U(d|R_{q'})$, and an integer k. Find a set of documents $S \subseteq R_q$ with $|S| = k$ that maximizes

$$U(S|q) = \sum_{q' \in S_q} P(q'|q) \sum_{d \in S} U(d|R_{q'}) \quad (5)$$

with the constraints that every specialization is covered proportionally to its probability. Formally, let $R \bowtie q' = \{d \in R_q | U(d|R_{q'}) > 0\}$. We require that for each $q' \in S_q$, $|R \bowtie q'| \geq \lfloor kP(q'|q) \rfloor$.

Roughly speaking, we aim at selecting from R_q, k results that maximize the overall utility of the result list. The idea of having the result list divided into $|S_q|$ subsets is that we want to satisfy all the possible specializations. Obviously, the more likely a specialization the greater the number of results we devote to it[1].

Since $U(d|R_{q'})$ measures the utility of a result r when the submitted query was q and the intended specialization q', for a given specialization of query q, the total utility to satisfy the user equals the sum of the utilities of all the selected results. Finally, the sum over all possible specializations of the query q weighted by $p(q'|q)$, gives the possibility to maximize the global utility of the result set R over all the known specializations of the queries.

In QL_DIVERSIFY(K) instead of maximizing the probability of covering useful results (as it is done in MAXUTILITY(K)) we seek to maximize directly the overall utility. This simple relaxation allows the problem to be simplified and solved in a very simple and efficient way. Furthermore, the constraints bounding the minimum number of results tied to a given specialization, boost the quality of the final diversified result list in a way that also ensure the number of specialization covered reflects the preferences of past users. In this way, if 40% of users query for "*leopard*" meaning the animal, at least 40% of results will be related to pages on this topic.

The main difference between Equation (5) and Equation (4) is that the latter needs to select, in advance, the subset S of documents before computing the final score. In our case, instead, a simple arithmetic argument shows that:

$$\begin{aligned} U(S|q) &= \sum_{q' \in S_q} P(q'|q) \sum_{d \in S} U(d|R_{q'}) \\ &= \sum_{d \in S} U(d|q) \end{aligned}$$

where $U(d|q)$ is the overall utility of document d for query q and it is computed according to the following equation:

[1]if $k < |S_q|$ we select the top-k S_q specializations, i.e. those with the k greatest $p(q'|q)$, $q' \in S_q$.

Algorithm OptSelect(q, D, k)
INPUT:
q: the currently submitted query;
D: the set of document from which to extract the diversified result set;
k: the number of documents to return in the diversified list;
OUTPUT:
S: the set of diversified documents with $|S| = k$.

1. Compute the value of $U(d_i|q)$ for all $d_i \in D$ according to Equation (6);
2. Create a min-heap M_i of size $|M_i| = \lfloor kP(q_i|q) \rfloor + 1$ for each possible specialization $q_i \in S_q$;
3. For each document $d \in D$ if $U(d|R_{q'}) > 0$ insert d into M_i; If d is not inserted in any M_i's insert it in a global min-heap M of size k.
4. Sort the min-heaps;
5. $S = \emptyset$;
6. *WHILE* $(|S| < k)$
7. *IF* all the M_i's are empty *THEN*
8. let d_p be the document not in S with greatest utility score from M;
9. *ELSE*
10. Pop the d_p with the greatest utility score from the collection of sorted M_i's;
11. $S = S \cup d_p$;
12. *ENDWHILE*
13. sort S by overall utility scores of its members;
14. RETURN(S);

Figure 4: The algorithm to find the subset of size k of D maximizing the value of $U(S|q)$.

$$U(d|q) = \sum_{q' \in S_q} P(q'|q) U(d|R_{q'}) \quad (6)$$

Therefore, to maximize $U(S|q)$ we can simply resort to compute for each $d \in D$ the utility of d for specialization $q' \in S_q$, and then to *select* the top-k highest scoring documents of D. Obviously, to have admissible solutions we have to carefully select results to include in the final list in order to not pick results only similar to a single specialization. For this reason we use a collection of $|S_q|$ min-heaps each of those keeps the top $\lfloor kP(q_i|q) \rfloor + 1$ useful documents also for that specialization. It is trivial to prove that the algorithm in Figure 3.1 returns the set S maximizing the objective function in Equation (5). Also, given that k is a positive constant, the running time of the algorithm is linear in the size of the set D. All the heap insertion and sort operations are, in fact, carried out on data structures of constant sizes $\leq k$.

4. EXPERIMENTS

In this section, we describe the experimental settings used for testing our diversification solution. We firstly discuss the feasibility of our approach, and then we present two different evaluations. While the first one is based on the official TREC Web Track Diversity Challenge guidelines, the second one aims at showing the importance of having a good diversification method based on real users' interests. In the test conducted we used two large scale query logs from which specialization were mined as discussed in the paper.

The AOL data-set contains about 20 million queries issued by about 650,000 different users, submitted to the AOL

search portal over a period of three months from 1st March, 2006 to 31st May, 2006.

The MSN Search query log contains instead 15 million queries submitted to the MSN US search portal over a period of one month in 2006. Queries are mostly in english. Both query logs comes with all the information needed to address the diversification problem according to our approach. We train the two query recommendation methods used for derive query specializations (Search-Shortcuts and Query-Flow Graph) on those logs.

4.1 Feasibility of the Diversification Solution

Something worth to be discussed is the feasibility of our approach based on query logs. First of all, it is important to point out that our method differently from others approaches such as the one by Agrawal *et al.* [1], does not require any pre-existing taxonomy, or classification model, to be built in advance. The only information we need is, for each ambiguous query, the list of possible specializations (and their probability) extracted from the query log, and the relative list of relevant documents to be re-ranked on the basis of the specializations. A *back-of-the-envelope* computation highlights the small footprint of the data structures needed to actually implement our method. For each ambiguous query in $\overline{Q}$ an average number of $|\overline{S_q}|$ specializations need to be stored. Given the ambiguous query with the largest number $\widehat{S_q}$ of specializations, we have then to store for each one of these specialization an average number $\overline{R}$ of documents requiring $\overline{D}$ Bytes each in the average. Resuming, storing N_a ambiguous query along with the data needed to assess the similarity among result lists will incur in a maximal memory occupancy of $\widehat{S_q}(N_a + \overline{R} \cdot \overline{D})$ Bytes, which is a reasonable amount of memory also for large values of N_a if we consider today computers memory capacity. Moreover, given the low complexity of the algorithms presented in the previous Section to actually determine if diversification is needed for the current query, and to build when necessary the diversified result list, the approach results to be feasible also in time. As a final remark, we point out that in practice (i.e. in our implementation) since we are using a search shortcuts approach, we can use an index-based structure to store specializations. This further reduces the memory occupancy of the data structure used to store them.

4.2 Evaluation based on TREC

We conduct our experiments in the context of the diversity task of the TREC 2009 Web track [11]. The goal of this task is to produce a ranking of documents for a given query that maximizes the coverage of the possible aspects underlying this query, while reducing its overall redundancy with respect to the covered aspects.

In our experiments, we consider a subset of the TREC ClueWeb09 dataset[2] collection, as used in the TREC 2009 Web track, comprising a total of 50 million English Web documents. A total of 50 topics were available for this task. Each topic includes from 3 to 8 sub-topics, as identified by TREC assessors, with relevance judgements provided at subtopic level. As an example the first TREC topic is identified by the query *obama family tree*, and three subtopics are provided: i) *Find the TIME magazine photo essay "Barack Obama's Family Tree"*, ii) *Where did Barack Obama's parents and grandparents come from?*, and iii) *Find biographical information on Barack Obama's mother.* In our experiments, the query associated to each topic is used as the initial query.

The evaluation results in the diversity task of the TREC 2009 Web track are reported according to two official metrics: α-NDCG and IA-P. The α-normalized discounted cumulative gain (α-NDCG [12]) metric balances relevance and diversity through the tuning parameter α. The larger the value of α, the more diversity is rewarded. In contrast, when $\alpha = 0$, only relevance is rewarded, and this metric is equivalent to the traditional NDCG [15].

Furthermore, our evaluation is also based on a generalization of standard IR metrics that rewards the diversity of a ranking. In particular, we use the intent-aware precision (IA-P [1]) metric, which extends the traditional notion of precision in order to account for the possible aspects underlying a query and their relative importance. In our evaluation, both α-NDCG and IA-P are reported at four different rank cutoffs: 5, 10, 20, and 100. These cutoffs focus on the evaluation at early ranks, which are particularly important in a Web search context. Both α-NDCG and IA-P are computed following the standard practice in the TREC 2009 Web track [11]. In particular, α-NDCG is computed with $\alpha = 0.5$, in order to give equal weights to both relevance and diversity.

An ad-hoc modified version of the Terrier[3] IR platform is used in the TREC evaluation for both indexing and retrieval. We extend Terrier in order to obtain summaries of any retrieved document. We use Porter's stemmer and standard English stopwords removal during for producing the ClueWeb-B index.

Table 1 shows the values of α-NDCG and IA-P for our *OptSelect* and Agrawal's *IA-Select* by using the query specializations, and their probabilities obtained from Search-Shortcuts. We test four different instances of the two algorithms by varying the number of results retrieved for each specialized query ($|R_{q'}|$). Best results are highlighted in bold. *OptSelect* noticeably outperforms *IA-Select* for all the four values shown. For both α-NDCG, and IA-P the best values are obtained by setting $|R_{q'}|$ equal to 50 or 100. Referring to α-NDCG, we highlight the gain of about 30% of its value obtained by increasing $|R_{q'}|$ from 10 to 20. Furthermore, both the two values for α-NDCG and IA-P still increase for $|R_{q'}| = 50$, and $|R_{q'}| = 100$.

The same behavior can be highlighted for IA-P. By setting $|R_{q'}|$ to 50 or 100, IA-P presents the highest scores, and it gains more than 50% of its value moving from $|R_{q'}| = 10$ to $|R_{q'}| = 20$.

Table 2 shows the values of α-NDCG and IA-P for our *OptSelect* and Agrawal's *IA-Select* by using query specializations, and their probabilities generated by Query-Flow Graph [6]. Also for this case *OptSelect* noticeably outperforms *IA-Select*. However, by using Query-Flow Graph, α-NDCG, and IA-P do not outperform the values obtained with the Search-Shortcuts technique. This is mainly due to the method used for producing query specializations. To be effective, our diversification method needs an effective method for producing query specializations. While Search-Shortcuts is always able to produce specializations for any given TREC query, Query-Flow Graph is not always able to

[2]http://boston.lti.cs.cmu.edu/Data/clueweb09/

[3]http://www.terrier.org

	α-NDCG			
	@5	@10	@20	@100
OptSelect-10	0.087	0.104	0.121	0.169
OptSelect-20	0.114	0.125	0.143	0.187
OptSelect-50	0.115	0.124	0.143	0.187
OptSelect-100	**0.117**	**0.127**	**0.148**	**0.190**
IASelect-10	0.042	0.042	0.046	0.050
IASelect-20	0.065	0.065	0.066	0.071
IASelect-50	0.080	0.076	0.077	0.081
IASelect-100	**0.082**	**0.078**	**0.080**	**0.085**
	IA-P			
	@5	@10	@20	@100
OptSelect-10	0.051	0.050	0.045	0.027
OptSelect-20	0.061	0.054	0.048	0.028
OptSelect-50	**0.064**	**0.056**	**0.050**	**0.028**
OptSelect-100	0.063	**0.056**	**0.050**	**0.028**
IASelect-10	0.018	0.015	0.013	0.004
IASelect-20	0.023	0.019	0.014	0.005
IASelect-50	**0.035**	**0.033**	0.023	0.007
IASelect-100	**0.035**	0.029	**0.024**	**0.008**

Table 1: α-NDCG, and IA-P values for *OptSelect* and IA-*Select* ($c = 0.5$) by varying the number of results returned for specialized queries ($|R_{q'}|$). Query specializations, and their probabilities come from Search-Shortcuts.

	α-NDCG			
	@5	@10	@20	@100
OptSelect-10	0.072	0.94	0.101	0.139
OptSelect-20	0.101	0.113	0.132	0.164
OptSelect-50	0.103	0.119	0.138	0.179
OptSelect-100	**0.109**	**0.121**	**0.138**	**0.183**
IASelect-10	0.026	0.027	0.027	0.03
IASelect-20	0.038	0.039	0.042	0.048
IASelect-50	0.043	0.045	0.048	0.055
IASelect-100	**0.045**	**0.047**	**0.049**	**0.058**
	IA-P			
	@5	@10	@20	@100
OptSelect-10	0.045	0.043	0.043	0.022
OptSelect-20	0.056	0.051	0.045	0.022
OptSelect-50	0.059	0.054	0.049	**0.023**
OptSelect-100	**0.06**	**0.055**	**0.050**	**0.023**
IASelect-10	0.013	0.012	0.012	0.004
IASelect-20	0.018	0.016	0.016	0.005
IASelect-50	0.019	0.016	0.016	0.005
IASelect-100	**0.021**	**0.017**	**0.018**	**0.006**

Table 2: α-NDCG, and IA-P values for *OptSelect* and IA-*Select* ($c = 0.5$) by varying the number of results returned for the set of specialized queries ($|R_{q'}|$). Query specializations and their probabilities come from Query-Flow Graph.

do it. It provide us query specializations only for 28 TREC queries.

Table 3 shows another interesting result. We want to study how the threshold c affects the performances of *OptSelect*, and *IA-Select*. By assigning high values to c (0.5, or 0.6) both the two algorithms provide the highest values for α-NDCG, and IA-P. For a lack of space we omit values greater than 0.6 or less than 0.4. In both cases (c less than 0.4, or greater than 0.6) the performances of the two methods decrease.

A similar evaluation is presented in [22]. By exploiting the

	α-NDCG			
	@5	@10	@20	@100
OptSelect-20 (0.4)	0.076	0.097	0.114	0.160
OptSelect-20 (0.5)	**0.114**	**0.125**	**0.143**	**0.187**
OptSelect-20 (0.6)	0.112	0.121	0.140	0.182
IASelect-20 (0.4)	0.041	0.040	0.044	0.047
IASelect-20 (0.5)	**0.065**	**0.065**	**0.066**	**0.071**
IASelect-20 (0.6)	0.061	0.062	0.064	0.069
	IA-P			
	@5	@10	@20	@100
OptSelect-20 (0.4)	0.041	0.047	0.045	0.026
OptSelect-20 (0.5)	**0.061**	**0.054**	0.048	**0.027**
OptSelect-20 (0.6)	**0.061**	**0.054**	**0.049**	0.026
IASelect-20 (0.4)	0.017	0.013	0.011	0.004
IASelect-20 (0.5)	**0.023**	**0.019**	**0.014**	**0.005**
IASelect-20 (0.6)	0.021	**0.019**	0.013	**0.005**

Table 3: α-NDCG, and IA-P values for *OptSelect* and *IA-Select* by varying the threshold c. Query specializations, and their probabilities comes from Search-Shortcuts.

TREC 2009 web track, Santos *et al.* propose an evaluation of their xQuAD framework by using a simulation of *IA-Select*, together with other methods. They simulate a best-case scenario, by considering the official sub-topics provided by the TREC collection as input to their diversification models. By doing so, they can isolate the impact of the query generation component and focus on comparing the diversification strategies provided by the considered approaches. Due to our formulation of the problem, our method strictly depends on an external source of information (like query logs) to diversify results for on-line query processing, and the quality of the diversification process is deeply affected by the quality of this information. For these reason we can not compare our results with theirs.

4.3 Evaluation based on Query Log Data

To assess the validity of our method we conducted several experiments in order to evaluate the impact of diversification on "ambiguous" queries result lists. We firstly apply both Search-Shortcuts and Query-Flow Graph to the two query logs obtaining a set of "ambiguous" queries along with their specializations, and their distribution of probability. Secondly we apply our method to the search results obtained from the Yahoo! BOSS infrastructure. For a given "ambiguous" query, we submit the query to the Yahoo! engine, we re-rank the resulting list by means of the Algorithm (4), and finally we compare the two lists. We aims at showing that our diversification technique can provide a user with a result list of k documents having a utility greater than the original set which is composed by the top-k results returned from Yahoo!.

To do so, our datasets containing "ambiguous" queries and the relative "specializations" have been split into two different subsets. The first one (containing approximatively the 70% of the total of the queries) has been used for training (i.e. to build the data structure describe in the previous section) and the last one as a repository of test queries. For any "ambiguous" query q in the test query set, our method obtains from the model the query "specializations" with the related distribution of probability. We then apply Algorithm (4) to obtain the diversified list of results. To assess

the impact of our diversification method on the utility of the diversified result list, we compute the ratio among the utilities of the two lists, i.e. the original, and the diversified one. More formally, we compute $\frac{\sum_1^k(U(d_i \in D))}{\sum_1^k(U(d_i \in N))}$ where D is the diversified list produced by our algorithm, whereas N is the original results obtained from Yahoo! BOSS. It is clear that if two lists share all the results, the ratio is equal to 1.

In all of our tests we fixed the number of results fetched from Yahoo! BOSS to $|R_q| = 200$, and the number of diversified results to $k = 20$.

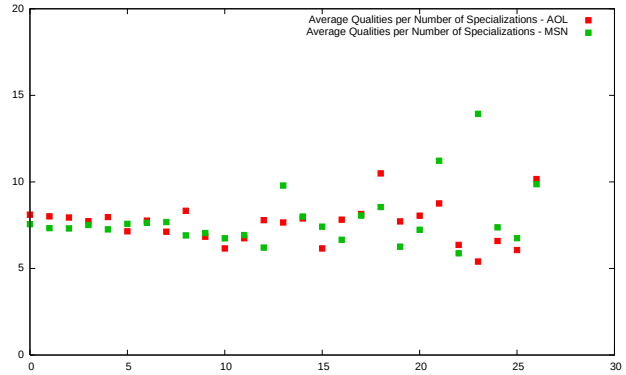

Figure 5: Average utilities per number of specializations referring to the AOL and MSN query logs.

Figure 5 shows the average utility per number of "specializations" for the two query logs considered in our experiments. In all the cases taken into account our method diversifies the final list by improving the usefulness measure for a factor ranging from 5 to 10 with respect to the usefulness of the original result set.

We point out that this ratio measure the increase in usefulness, not the absolute value of the usefulness. Therefore it is a method that incontrovertibly point out the benefits in using our diversification algorithm.

In Table 5 we resume the computed "specializations" for the query "rock and roll" for both query logs according to the actual specialization probabilities shown in Table 4. We report on results only for the diversified list of results for the query in the AOL query log. In the example we set $k = 5$. First, the list contains highly heterogeneous results. The original results for the query "rock and roll" are all related to what is "rock and roll" (i.e. with the first and the third results from wikipedia, and answers.com respectively), to a museum of rock and roll, and to a popular led zeppelin song from YouTube. In our list, instead, we take into account what users, according to the specialization of the query "rock and roll", expect to see in the result list. They frequently refine "rock and roll" with two queries "rock and roll lyrics", and "rock and roll sites". The results in the diversified list are highly correlated to this two topics. Indeed, the first three results are directly related to the two dominant specializations. Interestingly, the last two results keep the "informational" content of the list at an acceptable level. This is the effect of our algorithm that tries to maximize the coverage of all the specializations.

Furthermore, we measured the number of times our method is able to provide diversified results when they are actually needed, i.e. a sort of *recall* measure. In both cases we are able to provide diversified results in a large fraction of the ambiguous queries. More precisely, in the case of AOL, we are able to diversify results for the 61% of the "ambiguous" queries, whereas in MSN this *recall* measure raises up to 65%.

"ambiguous" query	"specialized" queries	$P(q'\|q)$
rock and roll	rock and roll lyrics	0.37
	rock and roll sites	0.41
	rock and roll vendors	0.18
rock and roll	rock and roll accordion	0.24
	rock and roll instruments	0.46
	rock and roll songs	0.28

Table 4: An example of "ambiguous" query, its most frequent "specializations", and the associated probabilities from the two query logs examined. The first one refers to AOL, while the second one to MSN.

5. CONCLUSIONS AND FUTURE WORK

We studied the problem of diversifying search results by exploiting the knowledge derived from query logs. Users, tend to interact with search engines in order to satisfy their information needs. If they submit an ambiguous/faceted query they often have tend to reformulate the query to find more easily the information they are looking for. We exploit the presence of different "specializations" of ambiguous queries in query logs as a way to diversify the search results returned to users and better cover the different possible interpretations of a query. We presented a general framework for query result diversification comprising: an efficient and effective methodology, based on state-of-the-art query recommendation algorithms, to detect ambiguous queries that would benefit from diversification, and to devise all the possible common specializations to include in the diversified list of results along with their probability distribution; a linear time diversification algorithm which re-ranks the original result list on the basis of the mined specializations by remarkably outperforming the state-of-the-art technique proposed by Agrawal *et al.* in [1]. A novel formulation of the problem based on the concept of maximizing the sum of results "*utilities*" is proposed and justified that allows the diversification problem to be modeled as a maximization problem which admits a linear time solution. The approach is evaluated by using both the metrics and the datasets provided for the TREC 2009 Web Track Diversity Challenge, and by using a utility-aware evaluation metrics here proposed. Our experimental results shows that our approach is both feasible and effective. In particular, we are able to improve the "*utility*" of diversified results from a factor of 5 to a factor of 10, and we remarkably outperform the Agrawal solution on the tests conducted by measuring both the α-normalized discounted cumulative gain [12] and the intent-aware precision [1] metrics. Future work will regard the exploitation of contextual information for optimizing results diversification for the current user, and the refinement of our approach for the partecipation to the 2010 Web Track Diversity Challenge.

6. REFERENCES

[1] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *Proc. WSDM'09*. ACM, 2009.

Original Result Set		Diversified Result Set	
1	**Rock and roll - Wikipedia, the free encyclopedia** rock and roll often written as rock roll or rock n roll is a genre of popular music that originated and evolved in the united states during	1	**Rock Charts - The First Internet Charts** free mp3 music charts lots of songs and mp3 downloads
2	**Rock and Roll Hall of Fame and Museum** official site of the rock and roll hall of fame and museum in cleveland ohio site includes information on past and future inductees and museum exhibitions	2	**album: Rock N Roll Music on IMEEM** imeem is the world's largest social music service with millions of free streaming songs and videos make playlists to share with friends and embed anywhere
3	**rock n roll: Definition from Answers.com** rock n roll or rockandroll n a form of popular music arising from and incorporating a variety of musical styles especially rhythm and blues	3	**Rock 'n Roll Music** rock n roll music this project for rock n roll music includes a collection of links to biographies of rock n roll musicians
4	**YouTube - Led Zeppelin Rock n Roll 1973** copyright 1976the song remains the same led zeppelins wan song records you need to quit being a fag and listen to some real rock and roll	4	**The Rise Of Rock N Roll** rock n roll in the popular mind was born fully realized in the mid1950s a product of the fusion of parts of rhythm and blues country and pop music
5	**Rock and Roll Music 1969: Album review and songs from Answers.com** rock and roll music similar albums best of the frost detroit kick out the jams ramblin' gamblin' man migration milestones artist the frost	5	**rock n roll: Definition from Answers.com** rock n roll or rockandroll n a form of popular music arising from and incorporating a variety of musical styles especially rhythm and blues

Table 5: Original and diversified list of results for the query "rock and roll" using specializations from the AOL query log.

[2] A. Anagnostopoulos, A. Z. Broder, and D. Carmel. Sampling search-engine results. In *Proc. WWW'05*. ACM, 2005.

[3] R. Baeza-Yates. Graphs from search engine queries. In *Proc. SOFSEM'07*, pages 1–8, Harrachov, CZ, 2007.

[4] R. Baraglia, F. Cacheda, V. Carneiro, D. Fernandez, V. Formoso, R. Perego, and F. Silvestri. Search shortcuts: a new approach to the recommendation of queries. In *Proc. RecSys'09*. ACM, 2009.

[5] P. Boldi, F. Bonchi, C. Castillo, D. Donato, A. Gionis, and S. Vigna. The query-flow graph: model and applications. In *In Proc. CIKM'08*. ACM, 2008.

[6] P. Boldi, F. Bonchi, C. Castillo, and S. Vigna. From 'dango' to 'japanese cakes': Query reformulation models and patterns. In *Proc. WI'09*. IEEE CS Press, 2009.

[7] D. Broccolo, L. Marcon, F. Nardini, R. Perego, and F. Silvestri. An efficient algorithm to generate search shortcuts. In *to appear*. ACM Press.

[8] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proc. SIGIR'98*. ACM, 1998.

[9] C. Carpineto, S. Osiński, G. Romano, and D. Weiss. A survey of web clustering engines. *ACM Comput. Surv.*, 41(3):1–38, 2009.

[10] Z. Cheng, B. Gao, and T.-Y. Liu. Actively predicting diverse search intent from user browsing behaviors. In *Proc. WWW'10*. ACM Press, 2010.

[11] C. Clarke, N. Craswell, and I. Soboroff. Preliminary report on the trec 2009 web track. In *Proc. TREC'09*. ACM, 2009.

[12] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon. Novelty and diversity in information retrieval evaluation. In *Proc. SIGIR'08*. ACM, 2008.

[13] P. Clough, M. Sanderson, M. Abouammoh, S. Navarro, and M. Paramita. Multiple approaches to analysing query diversity. In *Proc. SIGIR'09*. ACM, 2009.

[14] S. Gollapudi and A. Sharma. An axiomatic approach for result diversification. In *Proc. WWW'09*. ACM, 2009.

[15] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, 2002.

[16] R. Jones and K. L. Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proc. CIKM'08*. ACM, 2008.

[17] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

[18] B. Piwowarski and H. Zaragoza. Predictive user click models based on click-through history. In *Proc. CIKM'07*. ACM, 2007.

[19] F. Radlinski and S. Dumais. Improving personalized web search using result diversification. In *Proc. SIGIR'06*. ACM, 2006.

[20] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *Proc. ICML'08*. ACM, 2008.

[21] D. Rafiei, K. Bharat, and A. Shukla. Diversifying web search results. In *Proc. WWW'10*. ACM Press, 2010.

[22] R. Santos, C. Macdonald, and I. Ounis. Exploiting query reformulations for web search result diversification. In *Proc. WWW'10*. ACM Press, 2010.

[23] R. Santos, J. Peng, C. Macdonald, and I. Ounis. Explicit search result diversification through sub-queries. In *Proc. ECIR'10*. ACM Press, 2010.

[24] F. Silvestri. Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval*, 1(1-2):1–174, 2010.

[25] E. Vee, U. Srivastava, J. Shanmugasundaram, P. Bhat, and S. A. Yahia. Efficient computation of diverse query results. In *Proc. ICDE'08*. IEEE CS, 2008.

[26] O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to web search results. In *Proc. WWW'99*. Elsevier North-Holland, Inc., 1999.

[27] C. Zhai. *Risk minimization and language modeling in Information Retrieval*. PhD thesis, CMU, 2002.

[28] C. Zhai and J. Lafferty. A risk minimization framework for information retrieval. *IP&M*, 42(1):31–55, 2006.

[29] C. X. Zhai, W. W. Cohen, and J. Lafferty. Beyond independent relevance: methods and evaluation metrics for subtopic retrieval. In *Proc. SIGIR'03*. ACM, 2003.

[30] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proc. WWW'05*. ACM, 2005.