World Wide Web (2017) 20:729–747
DOI 10.1007/s11280-016-0413-1

An effective suggestion method for keyword search of databases

Hai Huang[1] · Zonghai Chen[1] · Chengfei Liu[2] · He Huang[3] · Xiangliang Zhang[4]

Received: 14 April 2016 / Revised: 30 June 2016 /
Accepted: 29 August 2016 / Published online: 9 September 2016

Abstract This paper solves the problem of providing high-quality suggestions for user keyword queries over databases. With the assumption that the returned suggestions are independent, existing query suggestion methods over databases score candidate suggestions individually and return the top-k best of them. However, the top-k suggestions have high redundancy with respect to the topics. To provide informative suggestions, the returned k suggestions are expected to be diverse, i.e., maximizing the relevance to the user query and the diversity with respect to topics that the user might be interested in simultaneously. In this paper, an objective function considering both factors is defined for evaluating a suggestion set. We show that maximizing the objective function is a *submodular function maximization problem subject to n matroid constraints*, which is an NP-hard problem.

✉ Hai Huang
hhuang03@ustc.edu.cn

Zonghai Chen
chenzh@ustc.edu.cn

Chengfei Liu
cliu@swin.edu.au

He Huang
hhuang@iim.ac.cn

Xiangliang Zhang
xiangliang.zhang@kaust.edu.sa

[1] Department of Automation, University of Science and Technology of China, Hefei 230027, China

[2] Faculty of ICT, Swinburne University of Technology, Melbourne Vic 3122, Australia

[3] Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei 230031, China

[4] CEMSE Division, King Abdullah University of Science and Technology, Thuwal, Saudi Arabia

An greedy approximate algorithm with an approximation ratio $O(\frac{1}{1+n})$ is also proposed. Experimental results show that our suggestion outperforms other methods on providing relevant and diverse suggestions.

Keywords Query suggestion · Query reformulation and keyword recommendation

1 Introduction

Keyword query enables inexperienced users to easily search databases, and the quality of results returned is actually dependent on the keywords used. For database users who have little knowledge of the database content or domain information, it is often hard to formulate effective keyword queries with "good" terms. Consequently queries are often *underspecified* or *overspecified*. There exist large gaps between the query terms and the appropriate terms that can obtain desired results in the database. To fill in such gaps, query suggestion is used to help users express their information need more precisely so that they can access the required information.

For semantic query suggestion there are two major types of query reformulation [3, 14]. The first type is *specialization*, in which a general or vague concept in the user query is modified to narrow down the search result. For example, suppose that a user has little knowledge about mobile phones. And he poses a keyword query [`Mobile phone`, `Price`] on a mobile database. There are some semantically related terms of `Mobile Phone` such as `iPhone 6s` and `Samsung galaxy S6` in the database, we could replace the term `Mobile phone` by `iPhone 6s` or `Samsung galaxy S6`, and generate the query suggestion [`iPhone6s, Price`] or [`Samsung galaxy S6`, `Price`] for user query. The other type of query reformulation is `parallel movement`, in which the user's topic of interest drifts to others with similar aspects. For example, assume that the user query [`Canon camera, Price`] is issued. If we know that `Nikon camera` is related to `Canon camera`, query suggestion [`Nikon camera, price`] could be suggested to explore price information of other camera brands. In this paper, both types of semantic query suggestions are studied with the usage of the knowledge of domain taxonomy.

In response to a user query, existing query suggestion methods [17, 18] evaluate the relevance scores of candidate suggestions and return the top-k best query suggestions with highest scores. Since the relevance scores are independently measured on candidates, the returned suggestions are semantically relevant not only to the query and also to each other. The redundancy in the returned top-k suggestion set leads to uniform topics that can be found in most of the suggestions. Given a user query from a random user, the set of suggestions with diverse topics is more likely to match user intent than the of set of suggestions belonging to one topic. For example, suppose that a user poses a query [`Mobile phone`, `Price`] on a mobile phone database. Since it is unknown to the system which mobile operating system the user is interested in, the best strategy is to suggest mobile phones covering diverse operating systems such as `IOS system` and `Android system`.

In this paper, we aim to diversify the query suggestion set with respect to topics while keeping the query suggestions relevant to the user query.

The challenging issues in providing diverse query suggestion are: First, how do we measure the relevance, and especially the diversity for a suggestion set? Second, given the overall function over suggestion sets which lineally combines the diversity and relevance

function, how do we find the optimal suggestion set that maximizes the overall function? This paper solves these problems and has the following novel study:

1. In this paper, to measure the quality of suggestion sets, the relevance function and the diversity function over suggestion sets are defined. The diversity function takes the factors of coverage and redundancy into account.
2. The overall function linearly combining the relevance function and diversity function is defined. It is then shown that finding the optimal suggestion set which maximizes the objective function is a submodular function maximization problem subject to n matroid constraints. And a greedy $O(\frac{1}{1+n})$ -approximation algorithm is also proposed.
3. New metrics for evaluating the quality of query suggestion sets are proposed, which takes the diversity into account. The experimental results validate the effectiveness of our method.

In the remainder of the paper, Section 2 introduces preliminary knowledge. From Sections 3 to 6, we present the framework of query suggestion and analyze the nature of our problem. Section 7 describes the algorithm for efficiently generating the query suggestion set. Section 8 illustrates our experiments. Section 9 reports on the related work. Finally Section 10 derives our conclusions.

2 Preliminary

Since it is often that users have little domain knowledge, user queries are underspecified or overspecified. In this paper domain taxonomies are used to replace the terms in user queries by more precise ones for obtaining high quality results in databases. Now we first introduce the concept of taxonomy.

Taxonomy information The taxonomy is a DAG graph. A node in the taxonomy is either a concept node or an instance node (without any successor in the DAG graph). An arc (t_1, t_2) indicates that t_1 is a super concept of t_2 and we use $t_2 \prec t_1$ to denote concept t_1 directly subsumes t_2. And $t_2 \prec^* t_1$ indicates that t_1 subsumes t_2 (directly or indirectly). A concept contains a set of instances and possibly a set of sub concepts. Note that a concept or instance node may have multiple parent nodes (super concepts).

Obviously, concepts in a taxonomy are semantically related. Given two concepts t_1, t_2, we compute the similarity of two concepts by the method proposed in [20] with some modification to bound the similarity value within [0, 1]:

$$sim(t_1, t_2) = \frac{2 \cdot \max_{t \in S(t_1,t_2)}[-\log p(t)]}{-(\log p(t_1) + \log p(t_2))} \tag{1}$$

where $p(t)$ is the probability of encountering an instance of concept t; $S(t_1, t_2)$ is the set of concepts that subsume both t_1 and t_2 in the taxonomy. The similarity of two concepts is computed as the information content they share (times 2) divided by the sum of the information content of the two concepts. It is easy to verify that $sim(t_1, t_2) \in [0, 1]$.

Recently, more and more taxonomy knowledge bases are developed such as Yago [11], CYC [15]. They cover a lot of domain taxonomies such as taxonomies about movies and sciences.

3 Problem definition

A user (keyword) query $q[w_1, w_2, \cdots, w_l]$ consists of l keyword terms. With respect to query q, we aim to find the *best* suggestion set of k query suggestions. The "best" here is motivated by two considerations. First, the k query suggestions in the suggestion set should be highly relevant to user query q. Second, to maximize user satisfaction, the suggestion set should be diverse with respect to the topics that the user might be interested in. To achieve the diversification of the suggestion set while keeping it relevant to the user query, we introduce an objective function f over suggestion sets. For any suggestion set $\mathcal{H}$ with k query suggestions, the objective function f on $\mathcal{H}$ is:

$$f(\mathcal{H}) = \lambda \cdot diversity(\mathcal{H}) + (1 - \lambda) \cdot relevance(\mathcal{H}, q) \tag{2}$$

The objective function f is a linear combination of relevance and diversity function by a tuning parameter $\lambda \in [0, 1]$. The diversity function $diversity(\mathcal{H})$ and the relevance function $relevance(\mathcal{H}, q)$ will be defined in Section 5. The method to produce query suggestions will be presented in the next section.

Our aim is to find the suggestion set $\mathcal{H}^*$ with k query suggestions, which has the highest f score.

4 Candidate query suggestions and topics

In this section, we present the method to generate candidate query suggestions by domain taxonomies, and to identify the potential topics that the query suggestions belongs to.

4.1 Candidate query suggestions

Assume that there exists a domain taxonomy $\mathcal{T}$, which is related to the content of database. Given taxonomy $\mathcal{T}$ and user query $q[w_1, w_2, w_3, \cdots, w_l]$, we reformulate the query by specialization and parallel movement. The specialization is to replace a keyword term w_i (as a general or vague concept in taxonomy $\mathcal{T}$) in user query q to its instances in taxonomy $\mathcal{T}$.

Definition 1 (Specialization movement): Given a keyword term w_i (as a concept in $\mathcal{T}$) in query q, specialization is to replace w_i by sw_i, if sw_i is an instance and $sw_i \prec^* w_i$ in taxonomy $\mathcal{T}$.

For example, given a user query [`mobile phone, price`] and the taxonomy shown in Figure 1, term `Mobile phone` can be replaced by terms `iPhone 6s` and `iPhone 6s plus` which are instances of `Mobile phone` in the taxonomy.

The other type of query reformulation is parallel movement, in which the terms (as instances in $\mathcal{T}$) in a user query are replaced by their sibling instances in $\mathcal{T}$.

Definition 2 (Parallel movement): Given a keyword term w_i (as an instance in $\mathcal{T}$) of query q, parallel movement is to replace w_i by instance sw_i, if there exists a concept t s.t. $w_i \prec t$ and $sw_i \prec t$ in taxonomy $\mathcal{T}$.

For example, term `iPhone 6s` can be replaced by term `iPhone 6s plus` which is one of siblings of instance `iPhone 6s` in the taxonomy (i.e., they have a common direct super concept `IOS system`).

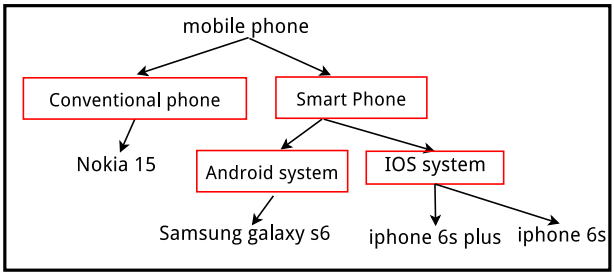

Figure 1 Taxonomy of mobile phones

Definition 3 (Term suggestion set): For each keyword term w_i in user query q, we define $sugg(w_i)$ to be the set of all terms that can be obtained from w_i through the specialization or parallel movement.

Definition 4 (Query suggestion): Given user query $q[w_1, w_2, \cdots, w_l]$, for each keyword term w_i, $sugg(w_i)$ is the term suggestion set of w_i. We say that $s[sw_1, sw_2, \cdots, sw_l]$ is a query suggestion of user query q if $\forall i, sw_i \in sugg(w_i)$ or $sw_i = w_i$.

For example, given the taxonomy shown in Figure 1, [`Samsung galaxy S6, Price`] and [`iPhone 6s, Price`] are candidate query suggestions of user query [`Mobile phone, Price`].

We use $\mathcal{S}$ to denote the set of all possible candidate query suggestions with respect to user query q. Table 1 describes the notations used throughout this paper.

4.2 Topics of query suggestions

A query suggestion could belong to some potential topics that the user is interested in. These topics can be identified according to the concepts in the taxonomy.

Table 1 Notation used throughout this paper

Notation	Description
$q[w_1, w_2, \cdots, w_l]$	User query
$\mathcal{T}$	Taxonomy
$t_2 \prec t_1$	t_1 subsumes t_2 directly
$t_2 \prec^* t_1$	t_1 subsumes t_2 (directly or indirectly)
$\mathcal{S} = \{s_1, s_2, \cdots, s_m\}$	The set of possible query suggestions with respect to user query q
$\mathcal{C}_s$	The set of topics fully supported by query suggestion s
$\mathcal{C} = \{c_1, c_2, \cdots, c_n\}$	Topic set with respect to $\mathcal{S}$
$\mu(s, c)$	The belief that query suggestion s supports topic c
$rel(s, q)$	Relevance score of query suggestion s w.r.t. q
$sup(c)$	The set of query suggestions fully support topic c
$\widetilde{sup}(c)$	The set of query suggestions highly support topic c
$\mathcal{H}$	Candidate suggestion set

In our running example, term `iPhone 6s` replaces the term `Mobile phone` in user query [`Mobile phone, Price`]. In the taxonomy, we can see that `IOS system` and `Smart phones` are two direct super concepts of `iPhone 6s`. It indicates that `iPhone 6s` belongs to the topics `IOS system` and `Smart phones`. For the more complicated case where a candidate suggestion consists of multiple keyword terms replaced, the topics it belongs to can be represented by the combinations of concepts.

Definition 5 (Topic of a query suggestion) Given a suggestion $s[sw_1, sw_2, \cdots, sw_l]$, for each term sw_i of s, $\{T_i\}$ is the set of concepts (in taxonomy $\mathcal{T}$) that directly subsume sw_i. We define $c = (t_1, t_2, \cdots, t_l)$, $t_i \in \{T_i\}$ to be a **topic** that suggestion s belongs to.

If query suggestion s belongs to topic c, we also say that topic c is **fully supported** by suggestion s.

For instance, query suggestion [`iPhone 6s, Price`] fully supports topics (`IOS system, Price`) and (`Smart phone, Price`).

Multiple topics could be fully supported by one query suggestion, and multiple suggestions could support one common topic. We use $\mathcal{C}_s$ to denote the set of topics supported by suggestion s, and $\boldsymbol{sup(c)}$ to denote the set of query suggestions that fully support topic c.

Given user query q and the set of all quey suggestions $\mathcal{S}$ for q, the set of topics associated with user query q denoted by $\mathcal{C}$ is defined as:

$$\mathcal{C} = \bigcup_{s \in \mathcal{S}} \mathcal{C}_s$$

For any topic $c \in \mathcal{C} = \{c_1, c_2, \cdots, c_n\}$, we have $sup(c) \subseteq \mathcal{S}$ and $sup(c_1) \cup sup(c_2) \cup \cdots \cup sup(c_n) = \mathcal{S}$. Note that $sup(c_i) \cap sup(c_j)$ is not necessarily $\emptyset$, i.e., they could have overlappings. Hence $sup(c_1) \cup sup(c_2) \cup \cdots \cup sup(c_n)$ is a cover instead of a partition of query suggestion set $\mathcal{S}$.

Topics are the combinations of concepts in the domain taxonomy, and they are semantically related with each other. A query suggestion could support a certain topic to some extent even though it does not fully support such the topic. Here we use $\mu(s, c)$ to denote the belief that query suggestion s supports topic c. The belief function $\mu(s, c)$ is defined as:

$$\mu(s, c) = \begin{cases} 1, & s \in sup(c) \\ \max_{c_k \in \mathcal{C}_s} rel(c_k, c), & s \notin sup(c) \end{cases} \tag{3}$$

where $rel(c_k, c)$ denotes the relevance between topic c_k and topic c. Given two topics $c_1 = (t_1^1, t_2^1, \cdots, t_l^1)$ and $c_2 = (t_1^2, t_2^2, \cdots, t_l^2)$, we compute the relevance between them as:

$$rel(c_1, c_2) = \prod_{i=1}^{l} sim(t_i^1, t_i^2) \tag{4}$$

where $sim(t_i^1, t_i^2)$ is the similarity value between the concepts t_i^1 and t_i^2 which can be computed by (1), and the independence between concepts is assumed here.

For topic c, if $s \in sup(c)$, $\mu(s, c) = 1$; otherwise, it is computed as the maximal relevance value between topic c and the topics in $\mathcal{C}_s$.

Definition 6 Given a query suggestion s, we say that s **highly supports** topic c if $\mu(s, c) \geq \delta$ where δ is a threshold (which can be set by the systems or users). And we define $\widetilde{sup}(c)$ as the set of query suggestions that highly support topic c.

Obviously, if $\delta = 1$ then we have $\widetilde{sup}(c) = sup(c)$.

5 Relevance and diversity function on suggestion sets

In this section we define the relevance and diversity function on suggestion sets.

5.1 Relevance function on suggestion set

Given a suggestion set $\mathcal{H}$ with k query suggestions, if $\mathcal{H}$ is relevant to user query q, the k query suggestions in $\mathcal{H}$ should be semantically relevant to user query q. We measure the relevance function on suggestion set $\mathcal{H}$ as:

$$relevance(\mathcal{H}, q) = \frac{1}{k} \cdot \sum_{s_i \in \mathcal{H}} rel(s_i, q) \tag{5}$$

where $rel(s_i, q)$ denotes the relevance score of query suggestion s_i with respect to user query q. Given user query $q[w_1, w_2, \cdots, w_l]$ with l keywords and query suggestion $s[sw_1, sw_2, \cdots, sw_l]$, we compute $rel(s, q)$ as:

$$rel(s, q) = \prod_{i=1}^{l} sim(sw_i, w_i) \tag{6}$$

where $sim(w_i, sw_i)$ is the similarity score between concepts or instances w_i and sw_i in the taxonomy.

5.2 Diversity function on suggestion sets

Now we discuss an important issue: how do we measure the diversity over a query suggestion set? Intuitively, the diversity of a query suggestion set depends upon how suggestions in this set are different from each other in terms of their associated topics. Inspired by coverage-based approaches [1, 5, 19], we first consider the coverage of a suggestion set, which measures the extent to which topics are covered by the suggestion set.

Given user query q, we have a query suggestion set $\mathcal{S} = \{s_1, s_2, \cdots, s_m\}$ and a topic set $\mathcal{C} = \{c_1, c_2, \cdots, c_n\}$. Since each c_i in $\mathcal{C}$ could be a potential topic that user is interested in, we maximize the average belief that topics are supported by at least one query suggestion in the suggestion set. For a query suggestion set $\mathcal{H}$, $|\mathcal{H}| = k$, we define the coverage function as:

$$coverage(\mathcal{H}) = \frac{1}{|\mathcal{C}|} \sum_{c_j \in \mathcal{C}} (1 - \prod_{s_i \in \mathcal{H}} \mu(\overline{s_i}, c_j)) \tag{7}$$

where $\mu(\overline{s_i}, c_j)$ denotes the belief that suggestion s_i does not support topic c_j and $\mu(\overline{s_i}, c_j) = 1 - \mu(s_i, c_j)$. For topic c and query suggestions $s_1, s_2 \in \mathcal{S}$, we assume that $\mu(\overline{s_1}, c)$ is independent of $\mu(\overline{s_2}, c)$. Thus $\prod_{s_i \in \mathcal{H}} \mu(\overline{s_i}, c_j)$ indicates the belief that topic c_j is not supported by the queries in $\mathcal{H}$. And $1 - \prod_{s_i \in \mathcal{H}} \mu(\overline{s_i}, c_j)$ indicates the belief that topic c_j is supported by at least one query suggestion in $\mathcal{H}$.

However, coverage is insufficient for evaluating diversity. The redundancy of a suggestion set is another key factor for measuring the diversity. If one topic is *over-supported* by the suggestion set, the diversity would decrease.

In our example, user query [`Mobile phone, Price`] is issued to a mobile phone database. Suppose that one of its 3-element suggestion set is {[`iPhone 6s, Price`], [`iPhone 6s plus, Price`], [`Samsung galaxy S6, Price`]}. These query suggestions support topics such as `IOS system`, `Android system` and `Smart phone`, but they are still not diverse enough. Because they all support the topic `Smart phone`. With this suggestion set, users would have no chance to see the conventional mobiles (not smart phones) with their prices. Since conventional mobiles are normally much cheaper than smart ones, the suggestion set is more likely to dissatisfy the users on a tight budget, thus narrowing down the users' horizons and hurting the diversity of suggestion.

With the aim of maximizing the diversity of the suggestion set and minimizing the dissatisfaction of users, the suggestion set should *not* be dominated by a certain topic. We say that a suggestion set $\mathcal{H}$ is dominated by topic c if all queries in $\mathcal{H}$ highly support c. The formal definition is:

Definition 7 (Domination): Given a suggestion set $\mathcal{H}$, $|\mathcal{H}| = k$, $\mathcal{H}$ is dominated by topic c if $|\mathcal{H} \cap \widetilde{sup}(c)| = k$.

To avoid such the case, we set a budget $k-1$ for each topic. It means that for each topic, the number of query suggestions in suggestion set $\mathcal{H}$ that highly support the topic is at most $k-1$. Namely, $\forall c_i \in \mathcal{C}, |\mathcal{H} \cap \widetilde{sup}(c_i)| \leq k-1$.

Thus, different from the former work [1, 5, 19] we consider the diversity for suggestion set on two aspects. On the one hand, we maximize the belief that topics are supported by at least one query suggestion in return set $\mathcal{H}$, namely the "at least" aspect. On the other hand, we constraint over-supported topics for $\mathcal{H}$, which we call the "at most" aspect. We then define the diversity function as:

$$\begin{aligned} diversity(\mathcal{H}) &= \frac{1}{|\mathcal{C}|} \sum_{c_j \in \mathcal{C}} \left(1 - \prod_{s_i \in \mathcal{H}} \mu(\overline{s_i}, c_j)\right), \\ &\quad \text{if } \forall c_i \in \mathcal{C}, |\mathcal{H} \cap \widetilde{sup}(c_i)| \leq k-1; \\ diversity(\mathcal{H}) &= -\infty, \text{ otherwise} \end{aligned} \tag{8}$$

If suggestion set $\mathcal{H}$ satisfies the constraints, the diversity score of $\mathcal{H}$ is the same as in formula (7). Otherwise, we define the diversity score of $\mathcal{H}$ as $-\infty$, which actually indicates that $\mathcal{H}$ is an unfavorable suggestion set since it does not satisfy the constraints. In formula (8), we set a budget $k-1$ for each topic. Without loss of generality, we can set a budget b_i for each topic c_i more flexibly. For instance, we could set b_i as an integer ranging from 1 to $k-1$ in proportion to the size of $\widetilde{sup}(c_i)$. It means that the more query suggestions support c_i, the more elements in $\widetilde{sup}(c_i)$ are allowed to appear in suggestion set $\mathcal{H}$. Of course, the upper bound is still $k-1$.[1]

[1]For the extreme case where a certain topic is highly supported by all candidate suggestions, we can simply discard this common topic, or set the budget as k for this topic.

6 Problem reformulation and finding the suggestion

We have defined the diversity and relevance functions on suggestion sets. The formula (2) can be reformulated as:

$$\mathcal{H}^* = \underset{\mathcal{H}\subseteq S,|\mathcal{H}|=k}{\arg\max} \left\{ \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j\in\mathcal{C}} (1 - \prod_{s_i\in\mathcal{H}} \mu(\overline{s_i}, c_j) + \frac{(1-\lambda)}{k} \cdot \sum_{s_i\in\mathcal{H}} rel(s_i, q) \right\}$$
$$\text{subject to: for } \forall c_i \in \mathcal{C}, |\mathcal{H} \cap \widetilde{sup}(c_i)| \leq k-1 \tag{9}$$

The desired solution $\mathcal{H}^*$ is the set of k query suggestions which satisfies the constraints and maximizes the objective function.

To achieve the solution, we look deep into the nature of this problem. Actually it is the *submodular function maximization problem subject to n matroids*, which is an NP-Hard problem. We first introduce the concept of submodular function.

Definition 8 (Submodular): Let X be a finite ground set and f : $2^X \rightarrow \mathbb{R}$. Then f is submodular if for all A, B $\subseteq$ X,

$$f(A) + f(B) \geq f(A \cup B) + f(A \cap B).$$

Another equivalent definition of submodularity is as follows: We denote by $f_A(i) = f(A + i) - f(A)$ the marginal value of i with respect to A. Then f is submodular if for all $A \subseteq$ B $\subseteq X$ and $i \in X \setminus B$, $f_A(i) \geq f_B(i)$.

The Submodular set functions have the property of a decreasing marginal gain as the size of the set increases. And they have been well-studied in economics and combinatorial optimization. For our problem, we have the following theorem:

Theorem 1 *The objective function in formula (9) is submodular and monotone (non-decreasing).*

Proof We first prove that the objective function f is monotone, namely given two sets $B \subseteq A$, $f(B) \geq f(A)$. We have:

$$f(A) = \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j\in\mathcal{C}} (1 - \prod_{s_i\in A} \mu(\overline{s_i}, c_j)) + \frac{(1-\lambda)}{k} \cdot \sum_{s_i\in A} rel(s_i, q)$$

$$f(B) = \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j\in\mathcal{C}} (1 - \prod_{s_i\in B} \mu(\overline{s_i}, c_j) + \frac{(1-\lambda)}{k} \cdot \sum_{s_i\in B} rel(s_i, q)$$

Since $A \subseteq B$,

$$\sum_{c_j\in\mathcal{C}} (1 - \prod_{s_i\in B} \mu(\overline{s_i}, c_j)) \geq \sum_{c_j\in\mathcal{C}} (1 - \prod_{s_i\in A} \mu(\overline{s_i}, c_j)),$$

and

$$\sum_{s_i\in B} rel(s_i, q) \geq \sum_{s_i\in A} rel(s_i, q)$$

Thus, we have $f(B) \geq f(A)$.

Now we focus on proving that the objective function f is submodular. We denote $f_A(s) = f(A+s) - f(A)$, the marginal increasing of s w.r.t. set A. Given two sets $A \subseteq B$, we need to prove: $f_A(s) \geq f_B(s)$. We have:

$$f(A) = \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j \in \mathcal{C}} (1 - \prod_{s_i \in A} \mu(\overline{s_i}, c_j)) + \frac{(1-\lambda)}{k} \cdot \sum_{s_i \in A} rel(s_i, q)$$

$$f(A+s) = \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j \in \mathcal{C}} (1 - \prod_{s_i \in A+s} \mu(\overline{s_i}, c_j) + \frac{(1-\lambda)}{k} \cdot \sum_{s_i \in A+s} rel(s_i, q)$$

$$\begin{aligned} f_A(s) &= f(A+s) - f(A) \\ &= \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j \in \mathcal{C}} \prod_{s_i \in A} \mu(\overline{s_i}, c_j) \cdot (1 - \mu(\overline{s}, c_j)) \\ &\quad + \frac{1-\lambda}{k} \cdot rel(s, q) \end{aligned}$$

Similarly, we have:

$$\begin{aligned} f_B(s) &= \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j \in \mathcal{C}} \prod_{s_i \in B} \mu(\overline{s_i}, c_j) \cdot (1 - \mu(\overline{s}, c_j)) \\ &\quad + \frac{1-\lambda}{k} \cdot rel(s, q) \end{aligned}$$

And since $A \subseteq B$, it holds

$$\begin{aligned} &\frac{\lambda}{|\mathcal{C}|} \sum_{c_j \in \mathcal{C}} \prod_{s_i \in B} \mu(\overline{s_i}, c_j) \cdot (1 - \mu(\overline{s}, c_j) \\ &\leq \frac{\lambda}{|\mathcal{C}|} \sum_{c_j \in \mathcal{C}} \prod_{s_i \in A} \mu(\overline{s_i}, c_j) \cdot (1 - \mu(\overline{s}, c_j) \end{aligned}$$

Eventually, we have $f_A(s) \geq f_B(s)$. □

Matroid constraints In our problem, to obtain the desired suggestion set it requires maximizing the objective function and satisfying the constraints. Now we show that it is problem of submodular function maximization problem subject to n matroid constraints.

The matroids are combinatorial structures that generalize the notion of linear independence in matrices.

Definition 9 (Matroid): A matroid M=(N, $\mathcal{I}$) is a finite set N with a collection of sets $\mathcal{I} \subseteq 2^N$, known as the independent sets, satisfying the following properties:

- property 1: if A $\in \mathcal{I}$ and $B \subseteq A$ then B $\in \mathcal{I}$.
- property 2: if A,B $\in \mathcal{I}$ and $|B| > |A|$ then there exists an element b $\in$ B \ A such that $A \cup \{b\} \in \mathcal{I}$.

The sets in M are typically called independent sets; for example, we would say that any subset of an independent set is independent. The union of all sets in M is called the *ground set*. An independent set is called a *basis* if it is not a proper subset of another independent set.

Given user query q with its query suggestion set $\mathcal{S} = \{s_1, s_2, \cdots, s_m\}$ and a set of topics $\mathcal{C} = \{c_1, c_2, \cdots, c_n\}$ associated. For each topic c_i, $\widetilde{sup}(c_i)$ is the set of query suggestions that highly support c_i. Now we define the set $\mathcal{I}_i$ as:

$$\mathcal{I}_i = \{A \subseteq \mathcal{S}, |A| \leq k : |A \cap \widetilde{sup}(c_i)| \leq k - 1\}$$

$\mathcal{I}_i$ is a family of independent query suggestion sets that satisfy the constraint $|A \cap \widetilde{sup}(c_i)| \leq k - 1$. We have the following theorem which can be easily proved.

Theorem 2 *$M_i = (\mathcal{S}, \mathcal{I}_i)$ is a matroid.*

Since for $\widetilde{sup}(c_1), \widetilde{sup}(c_2), \cdots, \widetilde{sup}(c_n)$ there are n constraints, we define n matroids as follows:

$$\begin{aligned}
M_1 &= (\mathcal{S}, \mathcal{I}_1), \\
&\quad \mathcal{I}_1 = \{A \subseteq \mathcal{S}, |A| \leq k : |A \cap \widetilde{sup}(c_1)| \leq k - 1\} \\
M_2 &= (\mathcal{S}, \mathcal{I}_2), \\
&\quad \mathcal{I}_2 = \{A \subseteq \mathcal{S}, |A| \leq k : |A \cap \widetilde{sup}(c_2)| \leq k - 1\} \\
&\quad \cdots \\
M_n &= (\mathcal{S}, \mathcal{I}_n), \\
&\quad \mathcal{I}_n = \{A \subseteq \mathcal{S}, |A| \leq k : |A \cap \widetilde{sup}(c_n)| \leq k - 1\}
\end{aligned}$$

$M_1, M_2, \cdots, M_n$ are matroids on the same ground set $\mathcal{S}$. If $\mathcal{H}$ is a legal set, it indicates that $\mathcal{H}$ should be a common independent set in each $\mathcal{I}_i$, namely $\mathcal{H} \in \bigcap_{i=1}^{n} \mathcal{I}_i$.

Thus, our problem can be formulated as:

$$\mathcal{H}^* = \arg\max\{f(\mathcal{H}) : \mathcal{H} \in \bigcap_{i=1}^{n} \mathcal{I}_i, \} \tag{10}$$

where function f is submodular and monotone.

Our aim is to find the independent[2] query suggestion set from $\bigcap_{i=1}^{n} \mathcal{I}_i$ that maximizes the submodular and monotone function f. This is the submodular function maximization problem subject to n matroids. For such problem, Fisher et al. in [9] show that finding the optimal suggestion H^* is an NP-hard problem and a greedy heuristic always produces a suggestion with an approximation ratio $O(\frac{1}{1+n})$. We will employ greedy strategy to design our approximate algorithm, which is described in Section 7.

7 Implementation of query suggestion algorithm

In this section, we present the algorithm of efficiently obtaining the suggestion set of k query suggestions. It adopts greedy strategy to produce a near-optimal suggestion set with an approximation ratio $O(\frac{1}{1+n})$. The algorithm is described in Algorithm 1.

[2] Actually the suggestion set $\mathcal{H}^*$ is a common basis in $\bigcap_{i=1}^{n} \mathcal{I}_i$ since we require $|\mathcal{H}^*|=k$.

Algorithm 1 Finding the suggestion set of k suggestions

Data: User query q;
Result: suggestion set $\mathcal{H}$;
Initialize $\mathcal{H} = \phi$;
Generate candidate suggestion set $\mathcal{S}$;
while $|\mathcal{H}| < k$ **do**
  for $s \in \mathcal{S}$ **do**
    if $\mathcal{H} \cup \{s\} \in \bigcap_{i=1}^{n} \mathcal{I}_i$ **then**
      $f(\mathcal{H} \cup \{s\}) = \frac{\lambda}{|\mathcal{C}|} \cdot \sum_{c_j \in \mathcal{C}} (1 - \prod_{s_i \in \mathcal{H} \cup \{s\}} \mu(\overline{s_i}, c_j)) + \frac{(1-\lambda)}{k} \cdot \sum_{s_i \in \mathcal{H} \cup \{s\}} rel(s_i, q)$;
    $s^* = \arg\max f(\mathcal{H} \cup \{s\})$;
    $\mathcal{S} = \mathcal{S} \setminus s^*$;
    $\mathcal{H} = \mathcal{H} \cup s^*$;
Return $\mathcal{H}$;

In Algorithm 1, by greedy strategy the suggestion set of k query suggestions is generated iteratively. In each iteration, it selects the top query suggestion s^* that maximizes the objective function and $\{\mathcal{H} \cup s^*\}$ is a common independent set in $\bigcap_{i=1}^{n} \mathcal{I}_i$. Then such query suggestion is added to the suggestion set and deleted from the candidate query suggestion set. This process runs iteratively until k query suggestions are obtained. The k query suggestions in the suggestion set can be ranked by their relevance score. Thus, a ranked list of k query suggestions will be returned to the user ultimately.

For time complexity, the complexity for relevance computation is $O(|\mathcal{S}|)$. The complexity for computing diversity scores is $O(|\mathcal{S}| \cdot |\mathcal{C}|)$. Thus, the overall complexity is $O(|\mathcal{S}| \cdot (|\mathcal{C}| + 1))$.

8 Evaluation

The experiments are run on a Dell T7500 workstation which has 6 Intel Xeon processor X5650 2.67GHz with 12 cores, 24GB of RAM, and a 1.4 TB hard disk. The algorithm described in the paper is implemented with Java 1.6.0.

Data sets Two real-world datasets are used in our experiments: (1) IMDB (Internet Movie Database), which contains information about movies, actors and directors; (2) DBLP, a dataset describing computer science bibliography information.

Taxonomy knowledge We extracted two taxonomies for the fields of movie and computer science from YAGO (http://www.mpi-inf.mpg.de/yago-naga/yago) which consists of facts extracted from Wikipedia and integrated with the Wordnet thesaurus. The details of the two taxonomies are shown in Table 2.

Table 2 Statistics of two taxonomies

#	Movie taxonomy	CS taxonomy
# classes	5,599	2,010
# instances	66,077	8,196
# IS-A relations	292,407	84,186

Query load For each dataset, we select abstract concepts or instances in the two taxonomies as keywords and randomly generate 30 queries. These queries are grouped into three groups based on the number of suggestion candidates they have. Each query group has 10 queries. Generally, the queries with more suggestion candidates are more abstract. The query load information of the two datasets is shown in Figure 2. Queries in query group QG1 and QG2 have a relatively smaller number of candidate suggestions, and queries in group QG3 have a much larger number of candidate suggestions.

Methods We implemented our method **DivQSA** (diverse query suggestion algorithm). For comparison, we also implemented the method **MMR** proposed in [4] as the baseline. MMR strives to obtain diverse results by reducing redundancy among results. It selects the query suggestion which is relevant to the query and contains minimal similarity to previously selected queries. In our setting, we compute the similarity between two query suggestions s_1, s_2 as $Sim(s_1, s_2) = \frac{|topics(s_1) \cap topics(s_2)|}{|topics(s_1) \cup topics(s_2)|}$, where $topics(s_i)$ is the set of topics highly supported by query s_i.

8.1 Varying parameter λ

The objective function f linearly combines the relevance and diversity functions by a tuning parameter $\lambda \in [0, 1]$. In this experiment, we investigate the changes of diversity scores and relevance scores of the suggestion sets when λ varies from 0 to 1 with $k = 10$.

When $\lambda = 1$, the objective function relies only on the diversity factor. For the case of $\lambda = 0$, the objective function relies only on the relevance factor. To clearly show the changes of diversity scores and relevance scores when λ varying, we use relative diversity scores and relative relevance scores to denote the changes. The diversity score in the case of $\lambda = 1$ and the relevance score in the case of $\lambda = 0$ are used as reference scores. The relative score functions for query group QG_i with respect to λ are defined as:

$$Relative\_DIV(QG_i)_\lambda = \frac{avg.Diversity(QG_i)_\lambda}{avg.Diversity(QG_i)_{\lambda=1}}$$

$$Relative\_REL(QG_i)_\lambda = \frac{avg.Relevance(QG_i)_\lambda}{avg.Relevance(QG_i)_{\lambda=0}}$$

where $Relative\_DIV(QG_i)_\lambda$ is the average relative diversity score of the suggestion sets for query group QG_i with respect to λ, and $Relative\_REL(QG_i)_\lambda$ is the average relative relevance score of the suggestion sets for query group QG_i with respect to λ.

	IMDB			DBLP		
	QG1	QG2	QG3	QG1	QG2	QG3
# avg. suggestion candidates	40	238	2639	171	1531	17526

Figure 2 Query groups information for two datasets

The *Relative_DIV* scores and the *Relative_REL* scores for three query groups with λ varying from 0 to 1 are shown in Figure 3. We are interested in the "*sweet range*" of λ. When λ falls in such a range, the *Relative_DIV* score and the *Relative_REL* score are both high, which indicates that the suggestion set is of high quality. For query groups QG1 and QG2 (Figure 3a, b, d and e), we can see that when λ is in range [0.3, 0.7], the *Relative_DIV* scores are high (> 0.9) while the *Relative_REL* scores keep high (> 0.9) too. Thus, the sweet range for QG1 and QG2 is [0.3, 0.7]. For query group QG3, Figure 3c and f show that the sweet range is [0.5, 0.7] on both datasets. Queries in QG3 have much more candidate suggestions and topics. As we know, the diversity is related to the topic coverage of the suggestion set. With the fixed λ and the suggestion set size k, generally the topic coverage of the query suggestion set tends to be lower for the user query with more topics. To make the suggestion sets highly diverse for such queries, it generally needs a larger λ. Thus, the start point of sweet range of λ for QG3 (whose queries have more topics) is larger.

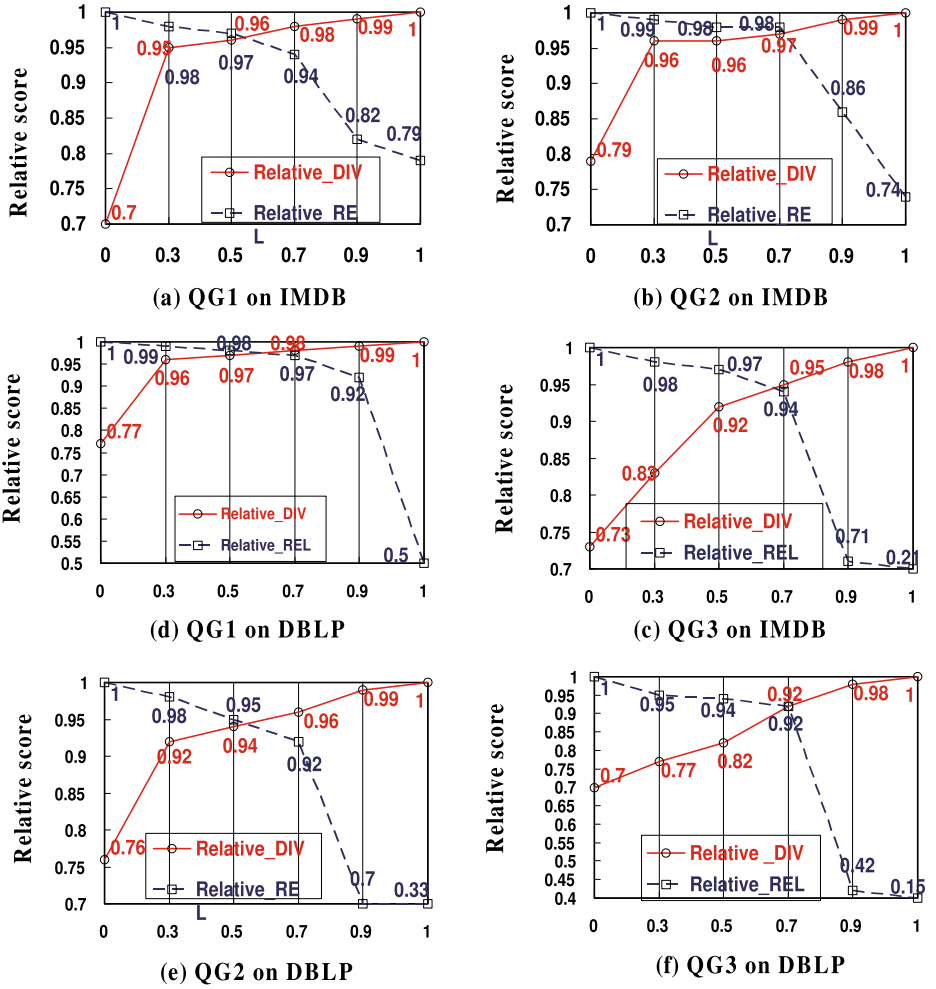

Figure 3 Relative scores of diversity and relevance of suggestions when varying λ from 0 to 1 with $k = 10$

8.2 Performance metrics

In order to properly measure the quality of suggestion sets generated by our method, we need to discuss the metrics for measuring. Classic metrics such as *NDCG* (discounted cumulative gain), *MRR* (mean reciprocal rank) have been widely used in the field of information retrieval for measuring search quality. However, these metrics focus only on the relevance of results and are not appropriate when diversity is taken into account. We modify the classic metrics and propose the new metrics. We first introduce the classic *NDCG*, *MRR* metrics and then discuss the new metrics we propose.

– **Classic metrics *NDCG* and *MRR***

Given query Q and one of its ranked set of query suggestions $\mathcal{S}_Q$, the premise of discounted cumulative gain (*DCG*) is that highly relevant results appearing lower in a result list should be penalized. The relevance value is reduced logarithmically proportional to the position of the result. The discounted cumulative gain at a particular rank threshold k is defined as:

$$DCG(\mathcal{S}_Q, k) = \sum_{j=1}^{k} \frac{2^{r(j)} - 1}{log(1+j)}$$

where $r(j)$ is the human judgment (0 = Bad, 1 = Fair, 2 = Good, 3 = Excellent) at rank position j. Assume that $\mathcal{S}_R$ is an ideal order list of all query suggestions of query Q descending by their human judgments. The normalized discounted cumulative gain (*NDCG*) is computed as:

$$NDCG(\mathcal{S}_Q, k) = \frac{DCG(\mathcal{S}_Q, k)}{DCG(\mathcal{S}_R, k)}$$

Given query Q and its ranked set of query suggestions $\mathcal{S}_Q$, the reciprocal rank (*RR*) of a query response is the inverse of the position of the first relevant result in $\mathcal{S}_Q$. The mean reciprocal rank (*MRR*) is the average of the reciprocal ranks of results for a sample of queries.

– **Our proposed metrics**, ***Div_NDCG* and *Div_MRR***.

The ideal query suggestion set is not only relevant to the user query but also highly diverse with respect to topics. We modify the classic metrics by taking into account the evaluation of diversity, specifically, coverage and redundancy.

For each possible topic c, we label any query suggestion in $\mathcal{S}_Q$ as "Bad" if it does not support topic c and compute its topic-dependent $NDCG(\mathcal{S}_Q, k|c)$. Meanwhile, we compute its redundancy factor as $RD(\mathcal{S}_Q, k|c) = \frac{1}{log(1+m)}$, where m is the number of suggestions in $\mathcal{S}_Q$ that support topic c. The value of $RD(\mathcal{S}_Q, k|c)$ is reduced logarithmically proportional to m. Then, we obtain the average across all topics as:

$$Div\_NDCG(\mathcal{S}_Q, k) = \frac{1}{|C|} \sum_{c \in C} NDCG(\mathcal{S}_Q, k|c) \cdot RD(\mathcal{S}_Q, k|c)$$

The new metric $Div\_NDCG(\mathcal{S}_Q, k)$ given above not only keeps the properties of *NDCG* but also takes the diversity into account. Thus, it is applicable to our evaluation tasks. Similarly, we have the other new metric *Div_MRR* defined as:

$$Div\_MRR(\mathcal{S}_Q, k) = \frac{1}{|C|} \sum_{c \in C} MRR(\mathcal{S}_Q, k|c) \cdot RD(\mathcal{S}_Q, k|c)$$

8.3 Evaluation results of *Div_NDCG* and *Div_MRR*

In this experiment, we evaluate our methods DivQSA$^-$ and DivQSA against the baseline method MMR. MMR focuses on obtaining diverse results by reducing redundancy among results. DivQSA$^-$ considers only the coverage factor, it does *not* takes redundancy factor into account. Namely for each topic, there is no constraint for the maximal number of query suggestions supporting the topic. DivQSA takes both of the coverage factor and redundancy factor into account. We acquire the relevance of query suggestions from human subjects on two datasets DBLP and IMDB. The relevance of query suggestions is judged not only based on their relevance to the original query but also their results in the database. The rank threshold k is set as 5, 10 and 15 separately. Figure 4a and b show the *Div_NDCG* results on two datasets. We can see that our methods outperform method MMR significantly in both datasets. Since method MMR focuses only on reducing redundancy among results, it can not guarantee the coverage of topics. For many topics, the topic-dependent *NDCG* scores are zero, thus lowering the overall scores of MMR. On the contrary, DivQSA$^-$ and DivQSA take the coverage factor into account and obtain higher scores. Compared with DivQSA$^-$, DivQSA considers the redundancy factor, thus obtaining more diverse results and higher scores. The results of *Div_MRR* are shown in Figure 4c and d. Similarly, DivQSA$^-$ and DivQSA also obtain significantly higher scores than method MMR on both datasets.

8.4 Case study

Table 3 shows a query [`Database researchers, Database techniques`] and its top-5 query suggestions generated by MMR and our method DivQSA (with $\lambda = 0.5$). It is shown that the diversity of the suggestion set generated by method MMR is lower. The database index techniques (`R-tree` and `B-tree`) appear three times, and there are only three distinct database techniques in the top-5 query suggestions. And the same database

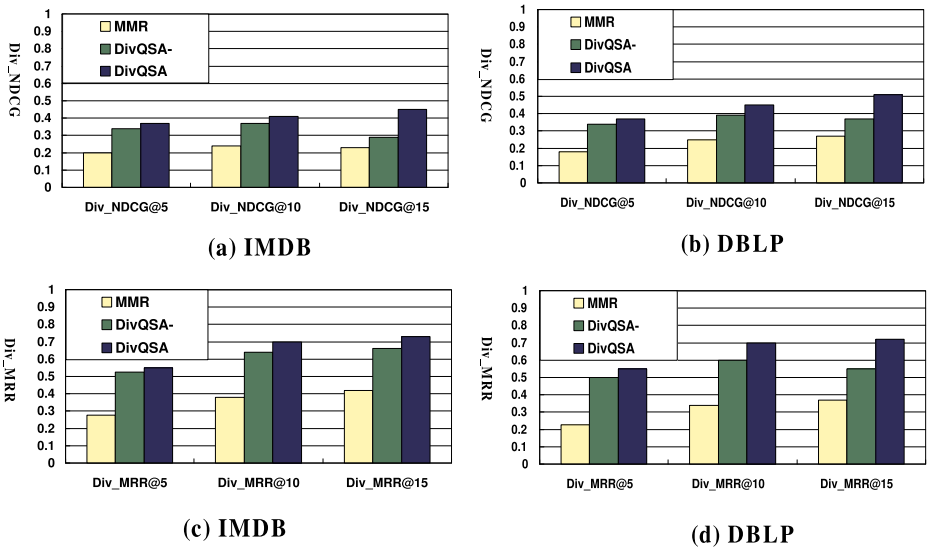

Figure 4 Comparison of our proposed approaches DivQSA$^-$, DivQSA and the baseline method MMR on metrics *Div_NDCG* and *Div_MRR*

Table 3 A user query and its top-5 query suggestions by the baseline method MMR and our method DivQSA

Query:
[Database researchers, Database techniques]

MMR, $\lambda = 0.5, k = 5$	DivQSA, $\lambda = 0.5, k = 5$
1.Georg Gottlob, R-tree	1.Georg Gottlob, R-tree
2.David Maier, Query optimization	2.Surajit Chaudhuri, Query optimizer
3.Rudolf Bayer, B-tree	3.Dan Suciu, Query evaluation
4.Rudolf Bayer R-tree	4.Hector Garcia Molina, Data warehousing
5.Hector Garcia Molina, Data warehousing	5.Jim Gray, Transaction processing

researcher Rudolf Bayer also appears twice. Compared with it, the suggestion set generated by DivQSA are much more diverse. There are five distinct database techniques and database researchers appearing in the suggestion set. Thus, the suggestion set is higher in quality.

8.5 Online running time

In this experiment, we study the online performance of our suggested algorithm. Table 4 shows the results on two datasets. The size of solution set k is set to 10. Queries in group QG1 have relatively fewer candidate queries, and queries in group QG3 have more candidate queries. The running time is within 2 seconds for each query group and it is satisfactory.

9 Related work

Keyword query suggestion over databases The conventional query suggestion methods over relational and XML databases databases [17, 18] suggest queries only based on maximizing relevance to the user query and the content of the database. In [18], the user input query is rewritten to a more relevant query by token expansion. They propose a score function and combine the spelling error penalty and TF/IDF scores of query segments in a heuristics. However, the score function considers finding the best correction to each keyword individually while ignoring the connectivity of these keywords. Furthermore, only the relevance of suggestion is considered. Similarly, in [17] Lu et al. propose a probabilistic framework for query suggestion over XML documents. They consider the connectivity of these keywords when scoring suggestions without taking into account the diversity of suggestions. Unlike the study in [17, 18], in this paper our framework suggests queries that maximize both the relevance and diversity.

Table 4 On-line performance

	Avg running time(s) for IMDB	Avg. running time(s) for DBLP
QG1	0.007	0.02
QG2	0.13	0.48
QG3	0.76	1.7

Query suggestion over Web search Query suggestion over Web search engines often employs query logs and click-through data to suggest queries [7, 12]. The query log based techniques are suitable for systems with a large number of users such as Web search engines with millions of users. However, for query suggestion over databases, the query logs are much smaller. Thus, in this paper we utilize the external taxonomy knowledge to suggest queries.

Results diversification Result diversification has been intensively studied in the fields of information retrieval and Web search. The main goal is to obtain relevant and different documents as search results. Existing approaches can be categorized as either the novelty-based method or the coverage-based method. The novelty-based approaches [4, 6, 21] compare documents to one another and select those documents that convey novel information. The coverage-based approaches [1, 5, 19] directly model the query aspects and seek to find the set of documents to maximally cover these aspects.

In the context of query suggestions over databases, the expected results are query suggestions instead of documents. Our approach represents the possible topics for user query in an explicit way and seeks to maximize the coverage of the possible topics, which can be classified into the coverage-based method. However, the purely coverage-based approaches ignore the redundancy among the covered topics, which could lead to the problem that a certain topic is over-supported by query suggestions and degrades the diversity. Thus, unlike purely coverage-based approaches [1, 5, 19], we seek to avoid the cases that a certain topic is over-supported by query suggestions.

Compared with the data driven approaches, we use external taxonomies which provide domain knowledge for clustering or classifying the results. Thus, it has better performance.

Keyword search over databases There are many approaches about keyword search over databases [2, 8, 10, 13, 16]. Among them, [2, 10, 13] focus on computing rooted trees as keyword search answers. BANKS [2] uses a backward search algorithm searching backwards from the nodes that contain keywords. BANKS-II [13] is proposed to overcome the drawbacks of BANKS. It is able to make forward search from potential roots. BLINKS [10] is proposed as a bi-level index to speed up BANKS-II, as no index is used in BANKS-II.

10 Conclusion

In this paper, we present a method that suggests relevant and diverse queries for a user query over databases. The objective function we defined considers both the relevance factor and diversity factor. It is shown that finding an optimal suggestion set is a submodular function maximization problem subject to n matroid constraints. Our evaluation results demonstrate that the query suggestions obtained by our suggestion algorithm are better than results obtained by the the suggestion method purely based on relevance ranking. In the future work, we will extend the work to split the long keywords to meaningful segments and obtain high-quality suggestions for them.

References

1. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying Search Results. In: WSDM, pp. 5–14 (2009)

2. Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., Sudarshan, S.: Keyword Searching and Browsing in Databases Using Banks. In: ICDE, pp. 431–440 (2002)
3. Boldi, P., Bonchi, F., Castillo, C., Vigna, S.: From Dango to Japanese Cakes: Query Reformulation Models and Patterns. In: Web Intelligence, pp. 183–190 (2009)
4. Carbonell, J.G., Goldstein, J.: The use of Mmr, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In: SIGIR, pp. 335–336 (1998)
5. Carterette, B., Chandar, P.: Probabilistic Models of Ranking Novel Documents for Faceted Topic Retrieval. In: CIKM, pp. 1287–1296 (2009)
6. Chen, H., Karger, D.R.: Less is More: Probabilistic Models for Retrieving Fewer Relevant Documents. In: SIGIR, pp. 429–436 (2006)
7. Cucerzan, S., White, R.W.: Query Suggestion Based on User Landing Pages. In: SIGIR, pp. 875–876 (2007)
8. Ding, B., Yu, J.X., Wang, S., Qin, L., Zhang, X., Lin, X.: Finding Top-K Min-Cost Connected Trees in Databases. In: ICDE, pp. 836–845 (2007)
9. Fisher, M., Nemhauser, G., Wolsey, L.: An analysis of approximations for maximizing submodular set functions ii. Math. Prog. Study **8**, 73–87 (1978)
10. He, H., Wang, H., Yang, J., Yu, P.S.: Blinks: Ranked Keyword Searches on Graphs. In: SIGMOD Conference, pp. 305–316 (2007)
11. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: Yago2: a spatially and temporally enhanced knowledge base from wikipedia. Artif. Intell. **194**, 28–61 (2013)
12. Jones, R., Rey, B., Madani, O., Greiner, W.: Generating Query Substitutions. In: WWW, pp. 387–396 (2006)
13. Kacholia, V., Pandit, S., Chakrabarti, S., Sudarshan, S., Desai, R., Karambelkar, H.: Bidirectional Expansion for Keyword Search on Graph Databases. In: VLDB, pp. 505–516 (2005)
14. Kato, M.P., Sakai, T., Tanaka, K.: Structured Query Suggestion for Specialization and Parallel Movement: Effect on Search Behaviors. In: WWW, pp. 389–398 (2012)
15. Lenat, D.B.: Cyc: a large-scale investment in knowledge infrastructure. Commun. ACM **38**(11), 32–38 (1995)
16. Li, G., Ooi, B.C., Feng, J., Wang, J., Zhou, L.: Ease: an Effective 3-In-1 Keyword Search Method for Unstructured, Semi-Structured and Structured Data. In: SIGMOD Conference, pp. 903–914 (2008)
17. Lu, Y., Wang, W., Li, J., Liu, C.: Xclean: Providing Valid Spelling Suggestions for Xml Keyword Queries. In: ICDE, pp. 661–672 (2011)
18. Pu, K.Q., Yu, X.: Keyword query cleaning. PVLDB **1**(1), 909–920 (2008)
19. Radlinski, F., Dumais, S.T.: Improving Personalized Web Search Using Result Diversification. In: SIGIR, pp. 691–692 (2006)
20. Resnik, P.: Using Information Content to Evaluate Semantic Similarity in a Taxonomy. In: IJCAI, pp. 448–453 (1995)
21. Zhai, C., Cohen, W.W., Lafferty, J.D.: Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In: SIGIR, pp. 10–17 (2003)