A Unified Framework for Recommending Diverse and Relevant Queries

Xiaofei Zhu Jiafeng Guo Xueqi Cheng Pan Du Hua-Wei Shen
Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
{zhuxiaofei, guojiafeng, dupan,shenhuawei}@software.ict.ac.cn, cxq@ict.ac.cn

ABSTRACT

Query recommendation has been considered as an effective way to help search users in their information seeking activities. Traditional approaches mainly focused on recommending alternative queries with close search intent to the original query. However, to only take relevance into account may generate redundant recommendations to users. It is better to provide diverse as well as relevant query recommendations, so that we can cover multiple potential search intents of users and minimize the risk that users will not be satisfied. Besides, previous query recommendation approaches mostly relied on measuring the relevance or similarity between queries in the Euclidean space. However, there is no convincing evidence that the query space is Euclidean. It is more natural and reasonable to assume that the query space is a manifold. In this paper, therefore, we aim to recommend diverse and relevant queries based on the intrinsic query manifold. We propose a unified model, named manifold ranking with stop points, for query recommendation. By turning ranked queries into stop points on the query manifold, our approach can generate query recommendations by simultaneously considering both diversity and relevance in a unified way. Empirical experimental results on a large scale query log of a commercial search engine show that our approach can effectively generate highly diverse as well as closely related query recommendations.

Categories and Subject Descriptors

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

General Terms

Algorithm, Experimentation, Performance, Theory

Keywords

Query Recommendation, Diversity, Manifold Ranking with Stop Points

1. INTRODUCTION

With the exponential growth of information on the Web, search engine has become an indispensable tool for Web users to seek their desired information. However, it is never easy for users to formulate a proper query to search because query is usually very short [7] and words are ambiguous [23]. Furthermore, users sometimes cannot express their search intent precisely due to the lack of domain-specific knowledge. Therefore, how to help users formulate a suitable query has been recognized as a challenging problem. To overcome this problem, a valuable technique, query recommendation, has been employed by most commercial search engines, such as *Google* [1], *Yahoo!*[2], and *Bing*[3] to improve usability. Query recommendation aims to suggest queries that may better reflect users' information needs to help them find what they need more quickly.

Traditional query recommendation approaches mainly focused on recommending alternative queries with close search intent to the original query. Query logs are widely used in these approaches [2, 15, 23], where similar queries are identified based on users' historical behavior and used as recommendations for each other. However, to only take relevance/similarity into account may generate redundant recommendations. For example, when a user issues a query 'abc', the system may recommend him/her 'abc television' and 'abc tv', which are both very relevant to 'abc' but of the equivalent meaning. Recommending such queries at the same time will decrease the recommendation quality since they provide almost the same information to users. Therefore, it is important to provide diverse as well as relevant query recommendations. By reducing the redundancy, we are able to cover multiple potential search intents of users and minimize the risk that users will not be satisfied.

In addition, previous query recommendation approaches mostly relied on measuring the similarity between queries in the Euclidean space, either based on query terms or click-through data. However, there is no convincing evidence that the query space is Euclidean. Inspired by the research work on document modeling [26, 27], it is more natural and reasonable to assume that the query space is a manifold, either linear or non-linear. The local geometric structure is essential to reveal the relationship between queries. In our study, we find that ranking queries in terms of the intrinsic global manifold structure [26, 27] is superior to the pairwise distance in the Euclidean space.

In this paper, therefore, we propose to recommend diverse and relevant queries based on the intrinsic query manifold. We propose a novel unified model, named manifold ranking with stop points, for query recommendation. Specif-

[1]http://www.google.com/
[2]http://www.yahoo.com/
[3]http://www.bing.com/

ically, our approach leverages a manifold ranking process over query manifold, which can naturally make full use of the relationships among queries to find relevant and salient queries. Meanwhile, we introduce the stop points into query manifold to capture the diversity during the ranking process. The stop points are points that stop spreading their ranking scores to their nearby neighbors during the manifold ranking process. By turning ranked queries into stop points, the ranking scores of other queries close to these stop points (i.e., queries which share similar search intent with the ranked queries) will be naturally penalized during the ranking process based on the intrinsic query manifold. Therefore, our approach can generate query recommendations by simultaneously considering both diversity and relevance between queries in a unified way. Like traditional manifold ranking algorithm, the new proposed ranking approach also shows a nice convergence property.

We conducted extensive experiments to evaluate the proposed approach based on a large collection of query logs from a commercial search engine. Empirical experimental results show that our approach can effectively generate highly diverse as well as closely related query recommendations.

The main contributions of our approach can be summarized as follows: (1) we first exploit the intrinsic global query manifold structure to measure the similarity between queries; (2) we propose a novel ranking approach, i.e., manifold ranking with stop points, for query recommendation which addresses relevance and diversity simultaneously in a unified way; (3) the proposed ranking approach has a nice convergence property; (4) we show that our query recommendation approach is superior to other baseline methods in producing diverse and relevant query recommendations.

The rest of the paper is organized as follows. Section 2 introduces related work. Section 3 describes our proposed approach in detail. Experimental results are discussed in Section 4 and conclusion is made in Section 5.

2. RELATED WORK

Query recommendation. Query recommendation has been employed as a core utility by many industrial search engines, which focuses on improving queries raised by users. The intention of query recommendation is closely related to query expansion [24, 7, 21], query substitution [12] and query refinement[14, 9]. The main difference is that query recommendation aims to recommend full queries submitted by previous users. Most of the work on query recommendation is focused on measures of query similarity, where query log data has been widely used in these approaches.

Click-through information conveyed in query log is often leveraged to measure the similarity of queries. The basic assumption is that queries sharing more clicked URLs are considered more similar [4]. A query-URL bipartite graph can be constructed from the click-through data with the vertices on one side corresponding to queries and on the other side to URLs. Beeferman et al. [2] applied agglomerative clustering algorithm to the click-through bipartite graph to identify related queries for recommendation. Wen et al. [23] proposed to combine both user click-through data and query content information to determine the query similarity. Li et al. [15] recommended related queries by computing the similarity between queries based on query-URL vector model and leveraging a hierarchical agglomerative clustering method to rank similar queries. Ma et al. [16] developed a two-level query recommendation method based on two bipartite graphs (user-query and query-URL bipartite graphs) extracted from the click-through data.

However, most previous work only focused on recommendation relevance, while not explicitly addressed the problem of diversity. Mei et al. [18] tackled this problem using a hitting time approach based on the query-URL bipartite graph. Their approach can recommend more diverse queries by boosting long tail queries. However, the weakness of their approach is that it would sacrifice the relevance considerably when improving the diversity, and many long tail queries recommended to users may not be familiar to them.

Different from existing approaches to query recommendation, our approach exploits the intrinsic global manifold structure to measure the similarity between queries and employs a novel ranking approach to address relevance and diversity simultaneously.

Diversity rank. Beyond relevance, diversity has also been recognized as a crucial criteria in ranking[5, 28, 17, 13, 25]. Top ranked results are expected to convey as little redundant information as possible, and cover as many aspects as possible.

Among the existing research work, Maximal Marginal Relevance (MMR) [5] is the most well-known method used for result set diversification. It has been widely used in the text summarization community. MMR utilizes a greedy strategy that iteratively selects the best scoring object, and then updates remaining object scores by computing a penalty based on the similarity of each object with the selected object. The closest work to ours is Grasshopper proposed by Zhu et al. [28], which applies an absorbing random walk on the graph. In order to achieve diversity, it turns the selected object into an absorbing state and then selects the next object based on the expected number of visits to each node before absorption. Although our work share similar idea with this approach in handling ranked objects, they are largely different in ranking strategy. Grasshopper uses two different measures, i.e., stationary distribution and expected number of visits, to select the top ranked object and the remaining objects. In contrast, all the objects are ranked with a consistent strategy (i.e., using their ranking scores) in our approach. A recent improvement on diversity ranking using random walk based approach is DivRank [17]. DivRank employs a time-variant random walk process, which uses the rich-gets-richer mechanism in ranking. However, the main drawback is that the computation of the expected number of visits is intractable and has to resort to some approximation strategies.

Different from these above approaches, we introduce the notion of stop points in manifold ranking and propose a unified model which simultaneously consider both relevance and diversity for query recommendation.

Manifold ranking. The manifold ranking algorithm first constructs a weighted network on the data, and assigns a positive ranking score to the input query and zero to the remaining points which are to be ranked with respect to the input query. Then all points spread their ranking scores to their nearby neighbors via the network until a global stable state is reached. Points without the input one are ranked according to their final ranking scores.

Manifold ranking was used to rank data with respect to the intrinsic global manifold structure collectively revealed by a huge amount of data [26, 27]. It has been applied in many research fields [10, 22, 19] recently where a ranking is needed in essentials. For example, He et al. [10] leveraged manifold ranking to measure relevance between the query and database images for image retrieval. Wan et al. [22] applied the manifold ranking process to utilize the relationships between the topic and the sentences for text summarization. However, so far there is no related work on applying manifold ranking for query recommendation.

To the best of our knowledge, this paper is the first article attempt to utilize manifold ranking for query recommendation.

3. OUR APPROACH

3.1 Preliminaries and Notations

Given a set of data points (i.e. queries) $\mathcal{X} = \{q_0, q_1, \ldots, q_n\} \subset \mathbb{R}^m$, the first point q_0 is the input query and the rest of the points q_i $(1 \le i \le n)$ are the candidate queries. Hereafter, query and point will not be discriminated unless otherwise specified. Let $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ denote a metric on $\mathcal{X}$ (e.g. Euclidean distance), where $d(q_i, q_j)$ is the distance between q_i and q_j. Let $f : \mathcal{X} \to \mathbb{R}$ denote a ranking function which assigns to each point q_i $(0 \le i \le n)$ a ranking value f_i. We can view f as a vector $f = [f_0, \ldots, f_n]^T$. We also define a vector $y = [y_0, \ldots, y_n]^T$, in which $y_0 = 1$ for the input query q_0 and $y_i = 0$ $(1 \le i \le n)$ for all the candidate queries.

3.2 Query Manifold

In our work, queries are assumed to be sampled from a low-dimensional manifold which is embedded in the high-dimensional ambient space. Our recommendation approach is then based on such a query manifold structure. Here we build a *k-nearest-neighbor query graph* using the click-through information in query logs to model the local query manifold structure.

The click-through data can help us find similar queries. The basic idea is that if two queries share many clicked URLs, they have similar search intent to each other [15]. Therefore, we model queries in terms of query-URL vectors, instead of query-term vectors. We represent each query q_i as a L_2-normalized vector, where each dimension corresponds to one unique URL in the click-through data. Specifically, given a query q_i $(0 \le i \le n)$, the j-th element of the feature vector of q_i is

$$q_i^j = \begin{cases} \frac{e_{ij}}{\sqrt{\sum_{k=1}^m e_{ik}^2}} & \text{if } q_i \text{ clicked } u_j; \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where m denotes the total number of unique URLs in the click-through data and e_{ij} denotes the weight for the pair of query q_i and its clicked URL u_j. Here we follow the CF-IQF weighting scheme [8] and define the weight $e_{ij} = cf_{ij} \times \log(n/qf_j)$, where cf_{ij} denotes the total click frequency on u_j given q_i, qf_j denotes the total number of unique queries which have clicked u_j, and n denotes the total number of unique queries in the query log. The distance between two queries q_i and q_j is then measured by the Euclidean distance between their normalized feature vectors

$$d(q_i, q_j) = \sqrt{\sum_{k=1}^m (q_i^k - q_j^k)^2}. \quad (2)$$

With the definitions above, we construct the k-nearest-neighbor query graph as follows. Firstly, each query is represented as a data point on the manifold. We then connect any two points with an edge if they are among the k nearest neighbors to each other ($k = 50$ in our case). In this way, we are able to preserve the sparse property of the query manifold. We define an affinity matrix W for the query manifold, where

$$w_{ij} = e^{\frac{-d(q_i, q_j)^2}{2\sigma^2}}, \quad (3)$$

if there is an edge linking q_i and q_j, and $w_{ii} = 0$ as there are no loops in the graph. Here σ is empirically set to 1.25.

3.3 Manifold Ranking with Stop Points

A traditional manifold ranking process [26] over the query manifold can be described as follows:

1. Symmetrically normalize W by $S = D^{-1/2} W D^{-1/2}$ in which D is the diagonal matrix with (i,i)-element equal to the sum of the i-th row of W.
2. Iterate $f^{(t+1)} = \alpha S f^{(t)} + (1-\alpha) y$ until convergence, where α is a parameter in $[0, 1)$.
3. Let f_i^* denote the limit of the sequence of $\{f_i^{(t)}\}$. Rank each point q_i according its ranking scores f_i^* (largest ranked first).

In the above ranking process, all the points spread their ranking scores to their neighbors via the weighted graph. The spread process is repeated until a global stable state is achieved, and all the points are ranked according to their final ranking scores. With the traditional manifold ranking process, we can obtain relevant and salient queries for recommendation given the input query.

To explicitly address the diversity of query recommendation, we introduce stop points into query manifold and propose a novel ranking approach, named manifold ranking with stop points. The stop points are a special type of points on query manifold, which stop spreading their ranking scores to their neighbors during the manifold ranking process. Intuitively, we can imagine the stop points as the "black holes" on the manifold, where no ranking scores would be able to "escape" from them. By turning ranked queries into stop points, the ranking scores of other queries close to the stop points (i.e., queries which share similar search intent with the ranked queries) will be naturally penalized during the ranking process based on the intrinsic query manifold.

Here we derive the new iteration algorithm for manifold ranking with stop points. Let T denote the set of stop points, and R denote the set of free points (all the data points excluding the stop points). The normalized matrix S in traditional manifold ranking can then be reorganized as a block matrix $\begin{pmatrix} S_{RR} & S_{RT} \\ S_{TR} & S_{TT} \end{pmatrix}$, and the original iteration equation in step 2 can be written as

$$\begin{pmatrix} f_R \\ f_T \end{pmatrix}^{(t+1)} = \alpha \begin{pmatrix} S_{RR} & S_{RT} \\ S_{TR} & S_{TT} \end{pmatrix} \begin{pmatrix} f_R \\ f_T \end{pmatrix}^{(t)} + (1-\alpha) \begin{pmatrix} y_R \\ y_T \end{pmatrix}, \quad (4)$$

where f_R and f_T denote the ranking scores of points in set R and T respectively, and y_R and y_T denote the prior on the points in set R and T respectively.

Since stop points never spread their scores to their nearby points, we set $S_{RT} = S_{TT} = 0$, then we get the new iteration equation for manifold ranking with stop points:

$$\begin{pmatrix} f_R \\ f_T \end{pmatrix}^{(t+1)} = \alpha \begin{pmatrix} S_{RR} & 0 \\ S_{TR} & 0 \end{pmatrix} \begin{pmatrix} f_R \\ f_T \end{pmatrix}^{(t)} + (1-\alpha) \begin{pmatrix} y_R \\ y_T \end{pmatrix}. \quad (5)$$

As we turn the queries already selected for recommendation into stop points, the ranking scores of stop points are no longer useful for us since the stop points would not be selected later again. All we care about is the ranking scores of the free points in set R. Therefore, we only need to compute f_R with the iteration equation

$$f_R^{(t+1)} = \alpha S_{RR} f_R^{(t)} + (1-\alpha) y_R, \quad (6)$$

where the parameter α specifies the relative contributions to the ranking scores from neighbors and the initial ranking scores. It is important to know that, with the stop points introduced, the new iteration algorithm still has a nice convergence property, which means it can achieve a global stable state. This convergence property is shown in Theorem 1.

THEOREM 1. *The sequence* $\{f_R^{(t)}\}$ *converges to*

$$f_R^* = (1-\alpha)(I - \alpha S_{RR})^{-1} y_R.$$

PROOF. Without loss of generality, suppose $f_R^{(0)} = y_R$. By iteration equation (6), we have

$$f_R^{(t)} = (\alpha S_{RR})^t y_R + (1-\alpha) \textstyle\sum_{i=0}^{t-1} (\alpha S_{RR})^i y_R.$$

Let $P = D_{RR}^{-1} W_{RR}$, P is the similarity transformation of S_{RR} as follows:

$$\begin{aligned} S_{RR} &= D_{RR}^{-1/2} W_{RR} D_{RR}^{-1/2} \\ &= D_{RR}^{1/2} D_{RR}^{-1} W_{RR} D_{RR}^{-1/2} \\ &= D_{RR}^{1/2} P D_{RR}^{-1/2}, \end{aligned}$$

hence P and S_{RR} have the same eigenvalues.

Let λ be an eigenvalue of P, according to the Gershgorin circle theorem, we have

$$|\lambda - P_{ii}| \le \textstyle\sum_{j=1, j\neq i}^{|R|} |P_{ij}|,$$

where $|R|$ is the size of the free point set. Note that $P_{ii} = 0$ and $\sum_{j=1,j\neq i}^{|R|} |P_{ij}| \le 1$, so we have $|\lambda| \le 1$. Since $0 \le \alpha < 1$ and $|\lambda| \le 1$, then $\lim_{t\to\infty} (\alpha S_{RR})^t = 0$, $\lim_{t\to\infty} \sum_{i=0}^{t-1} (\alpha S_{RR})^i = (I - \alpha S_{RR})^{-1}$. Hence, we have

$$f_R^* = \lim_{t\to\infty} f_R^{(t)} = (1-\alpha)(I - \alpha S_{RR})^{-1} y_R.$$

□

We can use this closed form to compute the ranking scores f_R^* for all the free points directly. In large scale real-world problems, however, an iterative algorithm is preferable due to computational efficiency.

3.4 Recommendation Approach

Based on the ranking algorithm above, we finally obtain our query recommendation approach. We first construct a k-nearest-neighbor query graph to model the query manifold structure based on query logs and set all the query points as free points. Giving an input query, we apply the proposed ranking algorithm, i.e., manifold ranking with stop points, over the query manifold until a global stable state is achieved, and rank the queries according to their ranking scores. The free point with the largest ranking score (except the input query) will be selected as a recommendation, and set as a stop point in the following iteration. The process iterates until a pre-specified number of recommendations acquired. The recommendation algorithm using manifold ranking with stop points is shown in Algorithm 1. Note that it would be time consuming if we directly apply the ranking algorithm on the whole query manifold. Since most queries are irrelevant to the input query, we can use a width first search strategy to construct a sub-manifold to save the computational cost.

Algorithm 1 Query Recommendation using Manifold Ranking with Stop Points

```
Input:
$q$ - the input query
$\chi$ - all the other queries
$K$ - recommendation size
$S$ - normalized affinity matrix of the query manifold
$T$ - stop point set
$R$ - free point set
Output: Top $K$ recommendation query set $U$
Initialization: $U = \phi, T = \phi, R = \chi$
1: for $k = 1 \dots K$ do
2:     obtain $S_{RR}$ based on $S$, $T$ and $R$.
3:     iterate $f_R^{(t+1)} = \alpha S_{RR} f_R^{(t)} + (1-\alpha) y_R$ until convergence
       started with $f_R^{(0)} = 0$, where $\alpha$ is a parameter in [0,1).
4:     select the query $q_k$ with the largest ranking score (except
       the input query) as a recommendation, $U = U \cup \{q_k\}$.
5:     turn query $q_k$ from free point into stop point, $T = T \cup \{q_k\}$
       and $R = R - \{q_k\}$.
6: end for
```

4. EXPERIMENTS

4.1 Data Set

Our experiments are based on the Microsoft 2006 RFP dataset[4] which contains about 15 million queries (from US users) that were sampled over one month in May, 2006. For each query, the following details are available: a query ID, the query itself, the user session ID, a time-stamp, the clicked URL, the rank of that URL and the number of results. We cleaned the raw data by ignoring non-English queries, converting letters into lower case, and replacing all non-alphanumeric characters in each query with whitespace. To further reduce the noise in clicks, the click-through between a query and a URL with a frequency less than 3 was removed. After cleaning, we obtained the click-through data with totally 191,585 queries, 251,427 URLs and 318,947 edges. On average, each query clicks 1.66 distinct URLs, and each URL is clicked by 1.27 distinct queries.

[4]http://research.microsoft.com/users/nickcr/wscd09/

We randomly sampled 150 queries with frequencies between 700 and 15,000 for evaluation. This restriction is to avoid the navigational queries (for which the query recommendations may not be so useful) or very specific queries (for which there are no recommendations) [3].

4.2 Baselines

To evaluate the performance of our approach, called Mani_stop for short, for query recommendation, we adopt four baselines for comparison:

- *Naive*: It represents each query as a URL vector shown in Equation (1) and directly measures the Euclidean distance between queries. For a given query q, queries with smallest distance scores are ranked higher and selected as recommendations. Different from other baseline, this model only considers relevance for recommendation without emphasizing diversity.

- *Hitting_time*[18]: It recommends queries by using the hitting time from candidate queries q_s to the test query q as a measure for ranking. The hitting time from node i to node j in a random walk is the expected number of steps before node j is visited starting from node i, and it decreases when the number of paths from i to j increases and the lengths of the paths decrease. The basic idea of *Hitting_time* is to boost long tail queries for recommendation.

- *MMR* (Maximal Marginal Relevance) [5]: MMR measures the relevance and diversity independently and provides a linear combination, called "marginal relevance", as the metric. Formally,

$$\begin{aligned} MMR \overset{def}{=} Arg \max_{q_i \in R \setminus S} [\lambda Sim_1(q_i, q) \\ - (1-\lambda) \max_{q_j \in S} Sim_2(q_i, q_j)], \end{aligned} \tag{7}$$

  where R is a set of candidate queries, S is the subset of queries in R which is already recommended, Sim_1 and Sim_2 are both similarity metrics between queries and λ is a parameter for linear combination. When λ=1 it computes the standard relevance-ranked list, while when λ=0 it computes a maximal diversity ranking. For a given query q, MMR will iteratively recommends queries with the largest "marginal relevance".

- *Grasshopper*(Graph Random-walk with Absorbing StateS that HOPs among PEaks for Ranking)[28]: The Grasshopper model leverages an absorbing random walk over the query graph. The model starts with a teleporting random walk P:

$$P = \lambda \widetilde{P} + (1-\lambda) 1 r^T, \tag{8}$$

  where $\widetilde{P}$ is the raw transition matrix, r is the user-supplied initial distribution, and λ is a parameter to control the tradeoff. When $\lambda = 1$ it ignores the user-supplied prior ranking r, while when $\lambda = 0$ it returns to the ranking specified by r. The query with the largest weight is selected as the first recommendation, which is then set as an absorbing state. The model then reruns the random walk with absorbing states, and selects the next query based on the expected number of visits to each node before absorption.

4.3 Parameter Setting

To make a fair comparison, we need to tune the parameters for baseline approaches. Since both Naive and Hitting_time involve no parameter tuning, we only need to tune parameter λ for two approaches (MMR and Grasshopper). Based on one held-out data with respect to the metrics introduced in the latter part of this section, we tested the two approaches using 11 λ values (i.e. $0, 0.1, 0.2, \cdots, 1$) and selected the best λ value for MMR ($\lambda = 0.6$) and Grasshopper ($\lambda = 0.9$). In our experiments, we fixed the parameter α in our method (Mani_stop) at 0.99, consistent with the experiments performed in [26, 27].

4.4 Examples of Recommendations

Here we first present the comparison of recommendations generated by our approach and baseline methods. Table 1 shows two samples from our test queries including their top 10 recommendations generated by five methods.

From the results we clearly see that Naive approach tends to recommend closely related but somewhat redundant queries. For example, for the test query 'abc', we can find equivalent recommendations like 'abc tv' and 'abc television', or recommendations sharing very close meaning like 'abc news', 'abc breaking news' and 'abc world news'. We can also find redundant examples in the recommendations for the test query 'yamaha', e.g., 'yamaha motor', 'yamaha motorcycle', and 'yamaha motorcycles'. Since Naive method only considers relevance, it will inevitably produce many redundant recommendations.

Meanwhile, we can easily find that the three other baseline approaches (Hitting_time, MMR, Grasshopper) recommended queries with better diversity. However, there is still some redundancy in these approaches. For example, for the test query 'abc', 'abc tv' and 'abc television' are both recommended by Hitting_time and MMR, while for test query 'yamaha', 'yamaha motor' and 'yamaha motorcycles' are both recommended by Grasshopper. Moreover, the recommendation provided by Hitting_time may not so closely related to the original query, e.g. recommendation of 'espn sports' with respect to 'abc', and recommendation of 'bluebook motorcycle' with respect to 'yamaha'. It may also bring up noisy long tail queries which hurt user experience, like recommendation 'yahama' for query 'yamaha'.

Among all these approaches, we observe that our Mani_stop approach obtains best performance, where more diverse as well as closely related queries can be found in its recommendation results.

4.5 Automatic Evaluation

Evaluating the quality of query recommendation is difficult, since there is usually no ground truth of recommendations. Therefore, we first conduct automatic evaluation over query recommendation for more objective comparison between different approaches. The Open Directory Project (ODP)[5] and a commercial search engine (i.e., Google) are leveraged in this automatic evaluation. Besides, there is no evaluation metric that seems to be universally accepted as the best for measuring the performance of algorithms that aim to obtain diverse rankings [20]. Therefore, we adopt the following three metrics (*Relevance*, *Diversity* and *Q-*

[5]http://www.dmoz.org/

Table 1: Examples of query recommendations provided by different approaches (top 10 results)

query	Naive	Hitting_time	MMR	Grasshopper	Mani_sink
abc	abc shows	abc shows	abc shows	abc tv	abc tv
	abc television	abc television	abc breaking news	abc news	abc news
	abc tv	associated builders and contractors	associated builders and contractors	abc family	abc nightline
	abc news	abc tv	abc nightline	abc shows	abc family
	abc breaking news	news stories	abc tv	abc breaking news	associated builders and contractors
	abc family	abc news	abc television	nightline	abc shows
	abc sports	abc world news tonight	abc family	goodmorning america	abc daytime
	abc world news	abc family channel	abc sports	abc sports	goodmorning america
	world news tonight	espn sports	abc daytime	abc daytime	abc sports
	abc soap operas	abc nightline	goodmorning america	national news	abc soap operas
yamaha	yamaha america	yahama	yamaha america	yamaha motor	yamaha motor
	yamaha motor corp	yamaha america	yamaha atv parts	yamaha america	yamaha motor corp
	yamaha motor	yamaha motor corp	yamaha boat motors	yamaha motor corp	yamaha america
	yamaha motor co	yamaha motor co	yamaha motor corp	yamaha motorcycles	yamaha marine
	yamaha motorcycle	yamaha motor	yamaha snowmobiles	motorcycles	yamaha atv
	yamaha motors	yamaha motorcycle	yamaha motor	yamaha marine	yamaha snowmobiles
	yamaha motorcycles	yamaha snowmobiles	yamaha drums	yamaha atv	yamaha drums
	yamaha quads	yamaha quads	yamaha guitars	yamaha motorcycle parts	yamaha guitars
	yamaha snowmobiles	yamaha outboard motors	yamaha motorcycles	yamaha snowmobiles	yamaha quads
	yamaha scooters	bluebook motorcycles	yamaha atvs	yamaha quads	yamaha boat motors

measure) to help evaluate the relevance and diversity in recommendation.

Relevance. We adopt the same method used in [1] to evaluate the relevance of recommended queries. Specifically, we measure the relevance of two queries based on the similarity between their corresponding categories provided by ODP.

Given two queries q and q', let C and C' denote the corresponding set of top k ($k = 10$ in our case) ODP categories from Google Directory[6], respectively. We define the similarity between two categories $c \in C$ and $c' \in C'$ as the length of their longest common prefix $l(c, c')$ divided by the length of the longest category of c and c'. More concisely, denoting the length of a category c with $|c|$, the similarity between two categories c and c' is

$$Sim(c, c') = \frac{|l(c, c')|}{max\{|c|, |c'|\}}. \tag{9}$$

For instance, the similarity between the two categories 'Arts/Television/News' and 'Arts/Television/Stations/North _America /United_States' is 2/5, since they share the common prefix 'Arts/Television' and the length of the longest category is five. We then evaluate the relevance between two queries by measuring the simialrity between the most similar categories of the two queries among C and C'. Specifically, the relevance between query q and q' is then defined as

$$r(q, q') = \max_{c \in C, c' \in C'} Sim(c, c'). \tag{10}$$

For an input query q, the relevance of its recommendations is then defined as

$$rel(q) = \frac{1}{|U|} \sum_{q' \in U} r(q, q'), \tag{11}$$

where U denotes the recommendation set and $|U|$ is the number of queries in U.

Diversity. We measure the diversity of recommended queries based on the differences between their top ranked search results provided by Google. Specifically, given two queries q and q', we compute the proportion of different URLs among their top k ($k = 10$ in our case) search results by

$$d(q, q') = \begin{cases} 1 - \frac{|o(q,q')|}{k} & \text{if}(q \neq q'); \\ 0 & \text{otherwise}, \end{cases} \tag{12}$$

where $o(q, q')$ is the number of overlapped URLs among the top k search results of query q and q'. Then for an input query q, the diversity of its recommendations is defined as

$$div(q) = \sqrt{\frac{\sum_{q \in U} \sum_{q' \in U} d(q, q')}{|U|(|U| - 1)}}, \tag{13}$$

where $|U|$ is the number of queries in the recommended query set U.

Note that we do not adopt an opposite measure, e.g., the overlap of the top ranked search results, to evaluate the relevance between two queries since that kind of relevance is not desired by query recommendation. Obviously, it is not reasonable to assign high relevance scores to queries with similar search results as the input query and recommend them to users. Therefore, we evaluate relevance with a topical similarity defined above.

Q-measure. The two metrics above measure the relevance and diversity of recommendations respectively. It is better to have a measure which can combine these two aspects to have a comprehensive evaluation of the effectiveness of recommendation methods. Here we borrow the F-measure scheme and introduce a new metric to assess relevance and diversity tradeoff, referred as Q-measure. Formally, it is defined as the weighted harmonic mean of relevance and diversity:

$$\begin{aligned} Q(q) &= \frac{(1 + \beta^2) \cdot rel(q) \cdot div(q)}{\beta^2 \cdot rel(q) + div(q)} \\ &= \frac{(1 + \beta^2)}{\frac{\beta^2}{div(q)} + \frac{1}{rel(q)}}, \end{aligned} \tag{14}$$

where β is a parameter which can be used to control the tradeoff between relevance and diversity. If the value of

[6]http://www.google.com/dirhp

$\beta > 1$, Q-measure then emphasizes the diversity. In our experiment settings, we have $\beta = 1$ which equally emphasize importance of relevance and diversity.

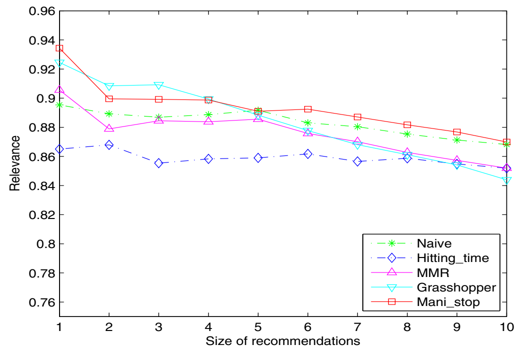


Figure 1: Average Relevance of Query Recommendation over Different Recommendation Size under Five Approaches.

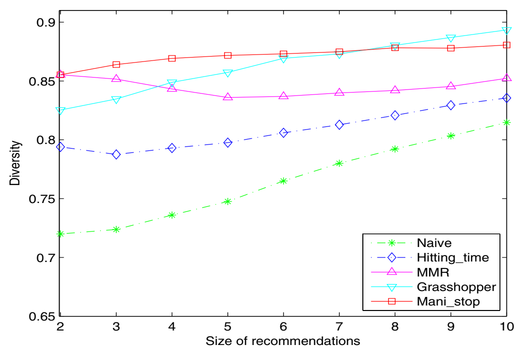


Figure 2: Average Diversity of Query Recommendation over Different Recommendation Size under Five Approaches.

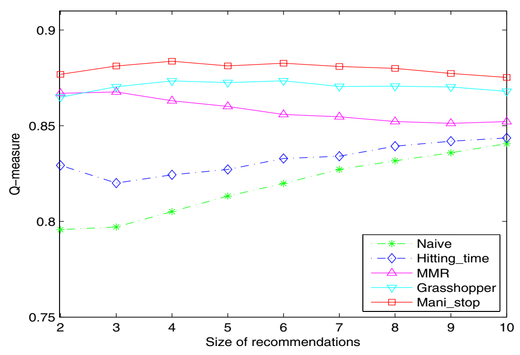


Figure 3: Average Q-measure of Query Recommendation over Different Recommendation Size under Five Approaches.

4.5.1 *Results and Discussion*

The automatic evaluations were conducted over a variety of recommendation size (up to top 10) and the results are presented in Figure (1-3).

Figure 1 shows the average relevance values of query recommendations under five different methods. As expected, for all recommendation methods, the average relevance value gradually decreases when the recommendation size increases. We notice that Mani_stop can achieve better performance in relevance as compared with MMR and Naive methods, which directly measure the relevance between queries in a Euclidean space. It demonstrates the effectiveness of the intrinsic query manifold in capturing the relevance between queries. We also note that the relevance of Grasshopper drops very quickly as the recommendation number increases, which makes its relevance performance not very stable. Besides, the relevance value of Hitting_time is lower than that of all the other four approaches on average. The major reason is that Hitting_time employs the expected hitting time from other queries to the give query to rank recommendations, so that it can boost long tail queries for recommendation. However, the recommendations with low hitting time to the given query may not be necessarily closely related, and thus hurt the relevance performance.

The average diversity values of query recommendations under the five different approaches are shown in Figure 2[7]. Not surprisingly, the diversity of Naive is the lowest one in the five approaches, since Naive only focuses on recommending queries according to their relevance with the input query. Hitting_time obtains better diversity by boosting the long tail queries for recommendation, but the improvement is limited. Both MMR, Grasshopper and Mani_stop can receive higher diversity value by explicitly address the diversity in ranking. Among these three approaches, Mani_stop obtains the highest diversity on average (0.872) by introducing the stop points into query manifold.

Figure 1 and Figure 2 show the relevance and diversity of different approaches, respectively. To better evaluate the overall effectiveness of different approaches, we show the Q-measure (a weighted harmonic mean of relevance and diversity) in Figure 3. From the results, we can see that Hitting_time works better than the Naive approach. Both MMR and Grasshopper can further outperform Hitting_time, while Grasshopper obtains better results than MMR by leveraging a "soft" penalization. Among the five approaches, the proposed Mani_stop approach consistently outperforms the other four baseline approaches in terms of Q-measure. We conduct t-test ($p - value <= 0.05$) over the results and find that the performance improvement is significant as compared with the baselines.

These above results clearly confirm that our Mani_stop approach can effectively generate highly diverse as well as closely related query recommendations.

4.5.2 *Impact of Parameter k*

As our approach is based on the query manifold structure (i.e. k-nearest-neighbor query graph), we also study the impact of parameter k, which defines the number of nearest neighbors when constructing the graph and plays an important role in terms of both effectiveness and efficiency.

[7]The diversity at one recommendation is not shown here due to that it is meaningless with the diversity metric.

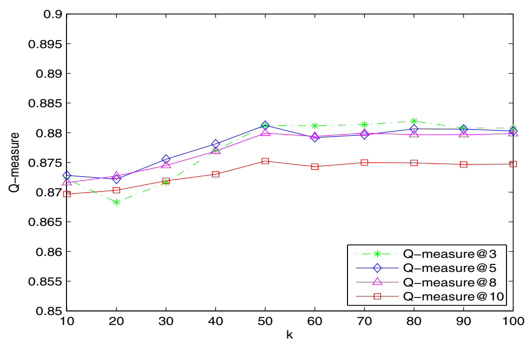


Figure 4: Q-measure over Different Recommendation Size (3,5,8,10) with k varying from 10 to 100.

Figure 4 shows the performance of our approach in terms of Q-measure under different recommendation sizes when varying k from 10 to 100 . The X-axis is parameter k, while the Y-axis is the Q-measure value measured by the automatic evaluation. We can see that as increasing the number of k, the Q-measure value exhibits a rise, until $k = 50$. It indicates that, if value of k is too small, the relationship between queries may be not be well revealed by the manifold structure. If $k > 50$, there is a slight decrease of the performance. The reason is that when $k > 50$, the possibility that noisy queries are becoming added into queries' neighborhood is increased, which will potentially affect the quality of query recommendation results.

4.6 Manual Evaluation

We further conduct manual evaluation for comparing different recommendation methods. For each query, we create a recommendation pool by merging the topmost (e.g., 10 in our work) recommendations from all the methods. Then we invite 3 human judges, with or without computer science background, to label the recommendations in the pool manually. For each test query, the human judges are required to identify the relevant recommendations and further group them into clusters according to their search intent.

We create a label tool as shown in Figure 5 to help ease the labeling process. For each test query, the human judges are presented with the recommendation pool (the left panel), and the search results of the query from a commercial search engine (the right panel). When labeling a recommendation, the human judge first takes a relevance assessment on the recommendation. If the recommendation is irrelevant to the test query at all, he just skips it. If relevant, he then needs to compare this recommendation with previous labeled recommendations and mark it in the same column as the one with the same search intent, or mark it in a new column (as a new intent), otherwise. The human judges are allowed to use a search engine of their choice for better understanding the meaning of the query and the recommendations. Note here there are three major cases in which two queries are considered as sharing the same search intent: (1) they are equivalent expressions, e.g., 'post code' and 'zip code'; (2) one query is the subconcept of the other, e.g. 'abc breaking news' and 'abc news'; (3) they are closely related with each other and share many similar search results, e.g. 'travel directions' and 'travel maps'. Since the labeling task is costly, we just randomly pick 50 queries for manual evaluation.

4.6.1 Evaluation Metrics

With the human labeled data, here we evaluate the quality of the recommendations produced by different approaches using the following two measures: (1) *α-normalized Discounted Cumulative Gain (α-nDCG)* [6] which has been widely used in the TREC Web track [8] diversity task; and (2) *Intent-Coverage.*

α-nDCG. The α-nDCG, which rewards diversity in ranking, is a new version of the nDCG [11], the normalized Discounted Cumulative Gain measure. When $\alpha = 0$, the α-nDCG measure corresponds to the standard nDCG, and when α is closer to 1, the diversity is rewarded more in the metric. The key difference between α-nDCG and nDCG is that they use different gain value. For each recommendation, the gain value $G(k)$ of α-nDCG is defined as follows:

$$G(k) = \sum_{i=1}^{I} J_i(k)(1-\alpha)^{C_i(k-1)}, \tag{15}$$

where $C_i(k-1)$ is the number of relevant recommendations found within the top $k-1$ recommendations for intent i, $J_i(k)$ is a binary variable indicating whether the recommendation at rank k belongs to intent i or not, and I is the total number of unique intents for each test query. The computation of α-nDCG exactly follows the procedure described in [6] with $\alpha = 0.5$.

Intent-Coverage. The Intent-Coverage measures the proportion of unique intents covered by the top k recommended queries for each test query. Since each intent represents a specified user information need, higher Intent-Coverage indicates larger probability to satisfy different users. Note that the Intent-Coverage is different from the diversity measure used in automatic evaluation, since only relevant recommendations to the test query will be considered in Intent-Coverage. Therefore, Intent-Coverage can better reflect the diversity quality of recommendations than the diversity measure in automatic evaluation. The Intent-Coverage is formally defined as:

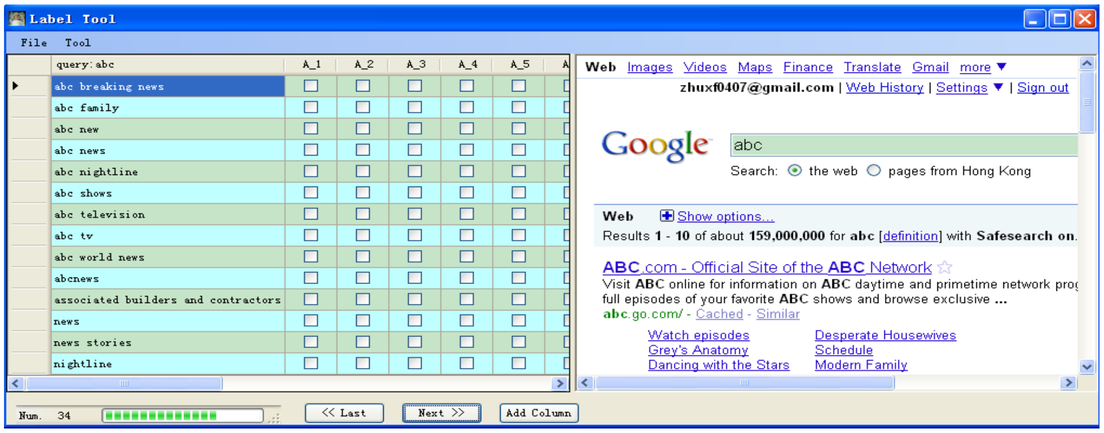
$$Intent - Coverage(k) = \frac{1}{I}\sum_{i}^{I} B_i(k), \tag{16}$$

where $B_i(k)$ is a binary variable indicating whether the intent i found within the top k recommendations or not, and I is the total number unique intents for each test query.

In our experiments, we compare the performance of different methods in terms of α-nDCG@5, α-nDCG@10, Intent-Coverage@5, and Intent-Coverage@10. Table 2 reports the performance of different recommendation approaches under manual evaluation. The numbers in the parentheses are the relative improvements compared with baseline methods.

From Table 2, we can see that Naive obtains the lowest Intent-Coverage and also shows a poor overall performance as measured by α-nDCG, since it only consider the relevance in recommendation. Hitting_time approach obtains better performance than Naive approach as it implicitly addresses the diversity in query recommendation by boosting long tail queries in top ranked results. Both MMR and Grasshopper approaches can further outperform Hitting_time approach

[8]TREC Web Track: http://plg.uwaterloo.ca/ trecweb



query: abc	A_1	A_2	A_3	A_4	A_5
abc breaking news	☐	☐	☐	☐	☐
abc family	☐	☐	☐	☐	☐
abc new	☐	☐	☐	☐	☐
abc news	☐	☐	☐	☐	☐
abc nightline	☐	☐	☐	☐	☐
abc shows	☐	☐	☐	☐	☐
abc television	☐	☐	☐	☐	☐
abc tv	☐	☐	☐	☐	☐
abc world news	☐	☐	☐	☐	☐
abcnews	☐	☐	☐	☐	☐
associated builders and contractors	☐	☐	☐	☐	☐
news	☐	☐	☐	☐	☐
news stories	☐	☐	☐	☐	☐
nightline	☐	☐	☐	☐	☐

Num. 34

Web Images Videos Maps Finance Translate Gmail more ▼

zhuxf0407@gmail.com | Web History | Settings ▼ | Sign out

Google abc

Search: ● the web ○ pages from Hong Kong

Web Show options...

Results 1 - 10 of about 159,000,000 for abc [definition] with Safesearch on.

ABC.com - Official Site of the ABC Network

Visit ABC online for information on ABC daytime and primetime network prog
full episodes of your favorite ABC shows and browse exclusive ...

abc.go.com/ - Cached - Similar

Watch episodes Desperate Housewives
Grey's Anatomy Schedule
Dancing with the Stars Modern Family

Figure 5: The label tool for query recommendation. The left panel shows the recommendation pool of a given input query and the right panel shows the the search results of the query from a commercial search engine. The columns A_i in the left panel denotes the i-th intent of the test query. A recommended query belonging to the i-th intent of the test query will be marked in the corresponding column.

Table 2: Performance of recommendation results over a sample of queries under five different approaches. Performance metrics α-nDCG@5, α-nDCG@10, Intent-Coverage@5 and Intent-Coverage@10 are shown. Numbers in parentheses indicates relative % improvement over Naive/Hitting_time/MMR/Grasshopper. Paired t-tests are performed, and results which show significant improvements (p-value $<$ 0.05) are marked ‡.

	α-nDCG@5	α-nDCG@10	Intent-Coverage@5	Intent-Coverage@10
Naive	0.717	0.689	0.300	0.536
Hitting_time	0.770 (7.4‡/*/*/*)	0.738 (7.1‡/*/*/*)	0.348 (16‡/*/*/*)	0.585 (9.3‡/*/*/*)
MMR	0.799 (11.4‡/3.8/*/*)	0.742 (7.7‡/0.5/*/*)	0.384 (28‡/10.3‡/*/*)	0.585 (9.1‡/0/*/*)
Grasshoppper	0.794 (10.7‡/3.1/-0.6/*)	0.768 (11.5‡/4.1/3.5‡/*)	0.373 (24.3‡/7.2/-2.9/*)	0.616 (14.9‡/5.1/5.3/*)
Mani_stop	0.838 (16.9‡/8.8‡/4.9‡/5.5‡)	0.806 (17‡/9.2‡/8.6‡/4.9‡)	0.436 (45.3‡/25.3‡/13.5‡/16.9‡)	0.665 (24.1‡/13.5‡/13.7‡/8‡)

on α-nDCG by explicitly addressing recommendation diversity. It seems that as a "soft" version of MMR, Grasshopper can achieve better performance than MMR when recommendation size is large. Compared with the four baseline methods, our Mani_stop approach achieves the best performance in terms of all measures, which is consistent with results reported in the automatic evaluation. We also conduct the t-test ($p - value < 0.05$) and find that the improvements over all baseline methods are significant. It shows that by exploiting the intrinsic global query manifold structure and employing manifold ranking with stop points, we can recommend highly diverse as well as closely related queries.

5. CONCLUSIONS

In this paper, we address the problem of recommending relevant and diverse queries. We propose a novel unified model, named manifold ranking with stop points, to solve this problem. Our approach leverages a manifold ranking process over query manifold, which can naturally make full use of the relationships among queries to find relevant and salient queries. Meanwhile, we introduce the stop points into query manifold to capture the diversity during the ranking process. In this way, our approach can generate query recommendations by simultaneously considering both diversity and relevance between queries in a unified way. Like traditional manifold ranking algorithm, the new proposed ranking approach also shows a nice convergence property.

Both automatic and manual evaluations are conducted to demonstrate the effectiveness of our approach. The extensive empirical results clearly show that our approach can outperforms all the four baseline methods (Naive, Hitting_time, MMR and Grasshopper) in recommending highly diverse as well as closely related query recommendations.

For the future work, we would like to explore different ways for modeling the query manifold structure and investigate how it may affect the recommendation performance. It would also be interesting to apply the proposed approach to a variety of applications, e.g., image retrieval, expert finding, and product recommendation, where both diversity and relevance are demanded in ranking.

6. ACKNOWLEDGMENTS

This research work was funded by the National Natural Science Foundation of China under Grant No. 61003166 and Grant No. 60933005.

7. REFERENCES

[1] R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 76–85, 2007.

[2] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 407–416, 2000.

[3] P. Boldi, F. Bonchi, C. Castillo, D. Donato, and S. Vigna. Query suggestions using query-flow graphs. In *Proceedings of the 2009 workshop on Web Search Click Data*, pages 56–63, 2009.

[4] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 875–883, 2008.

[5] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336, 1998.

[6] 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 *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 659–666, 2008.

[7] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *Proceedings of the 11th international conference on World Wide Web*, pages 325–332, 2002.

[8] H. Deng, I. King, and M. R. Lyu. Entropy-biased models for query representation on the click graph. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 339–346, 2009.

[9] J. Guo, G. Xu, H. Li, and X. Cheng. A unified and discriminative model for query refinement. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 379–386, 2008.

[10] J. He, M. Li, H.-J. Zhang, H. Tong, and C. Zhang. Manifold-ranking based image retrieval. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 9–16, 2004.

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

[12] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*, pages 387–396, 2006.

[13] J. P. Kelly and D. Bridge. Enhancing the diversity of conversational collaborative recommendations: a comparison. *Artif. Intell. Rev.*, 25:79–95, April 2006.

[14] R. Kraft and J. Zien. Mining anchor text for query refinement. In *Proceedings of the 13th international conference on World Wide Web*, pages 666–674, 2004.

[15] L. Li, Z. Yang, L. Liu, and M. Kitsuregawa. Query-url bipartite based approach to personalized query recommendation. In *Proceedings of the 23rd national conference on Artificial intelligence*, pages 1189–1194, 2008.

[16] H. Ma, H. Yang, I. King, and M. R. Lyu. Learning latent semantic relations from clickthrough data for query suggestion. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 709–718, 2008.

[17] Q. Mei, J. Guo, and D. Radev. Divrank: the interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1009–1018, 2010.

[18] Q. Mei, D. Zhou, and K. Church. Query suggestion using hitting time. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 469–477, 2008.

[19] R. Ohbuchi and T. Shimizu. Ranking on semantic manifold for shape-based 3d model retrieval. In *Proceeding of the 1st ACM international conference on Multimedia information retrieval*, pages 411–418, 2008.

[20] F. Radlinski, P. N. Bennett, B. Carterette, and T. Joachims. Redundancy, diversity and interdependent document relevance. *SIGIR Forum*, 43(2):46–52, 2009.

[21] M. Theobald, R. Schenkel, and G. Weikum. Efficient and self-tuning incremental query expansion for top-k query processing. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 242–249, 2005.

[22] X. Wan, J. Yang, and J. Xiao. Manifold-ranking based topic-focused multi-document summarization. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 2903–2908, 2007.

[23] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Clustering user queries of a search engine. In *Proceedings of the 10th international conference on World Wide Web*, pages 162–168, 2001.

[24] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 4–11, 1996.

[25] M. Zhang. Enhancing diversity in top-n recommendation. In *Proceedings of the third ACM conference on Recommender systems*, pages 397–400, 2009.

[26] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *Proceedings of the 17th Annual Conference on Neural Information Processing Systems*, 2003.

[27] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *Proceedings of the 17th Annual Conference on Neural Information Processing Systems*, 2003.

[28] X. Zhu, A. B. Goldberg, J. V. Gael, and D. Andrzejewski. Improving diversity in ranking using absorbing random walks. In *Proceedings North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT)*, pages 97–104, 2007.