Mining Web Graphs for Recommendations

Hao Ma, Irwin King, *Senior Member*, *IEEE*, and Michael Rung-Tsong Lyu, *Fellow*, *IEEE*

Abstract—As the exponential explosion of various contents generated on the Web, *Recommendation* techniques have become increasingly indispensable. Innumerable different kinds of recommendations are made on the Web every day, including movies, music, images, books recommendations, query suggestions, tags recommendations, etc. No matter what types of data sources are used for the recommendations, essentially these data sources can be modeled in the form of various types of graphs. In this paper, aiming at providing a general framework on mining Web graphs for recommendations, 1) we first propose a novel diffusion method which propagates similarities between different nodes and generates recommendations; 2) then we illustrate how to generalize different recommendation problems into our graph diffusion framework. The proposed framework can be utilized in many recommendation tasks on the World Wide Web, including query suggestions, tag recommendations, expert finding, image recommendations, image annotations, etc. The experimental analysis on large data sets shows the promising future of our work.

Index Terms—Recommendation, diffusion, query suggestion, image recommendation.

1 Introduction

With the diverse and explosive growth of Web information, how to organize and utilize the information effectively and efficiently has become more and more critical. This is especially important for Web 2.0 related applications since user-generated information is more freestyle and less structured, which increases the difficulties in mining useful information from these data sources. In order to satisfy the information needs of Web users and improve the user experience in many Web applications, *Recommender Systems*, have been well studied in academia and widely deployed in industry.

Typically, recommender systems are based on *Collaborative Filtering* [14], [22], [25], [41], [46], [49], which is a technique that automatically predicts the interest of an active user by collecting rating information from other similar users or items. The underlying assumption of collaborative filtering is that the active user will prefer those items which other similar users prefer [38]. Based on this simple but effective intuition, collaborative filtering has been widely employed in some large, well-known commercial systems, including product recommendation at Amazon,[1] movie recommendation at Netflix,[2] etc. Typical collaborative filtering algorithms require a user-item rating matrix which contains user-specific rating preferences to infer users' characteristics. However, in most of the cases, rating data are always unavailable since information on the Web is less structured and more diverse.

1. http://www.amazon.com.
2. http://www.netflix.com.

- *H. Ma is with The Chinese University of Hong Kong, Room 101, HSH Engineering Building, Shatin, N.T., Hong Kong. E-mail: hma@cse.cuhk.edu.hk.*
- *I. King and M.R. Lyu are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong. E-mail: {king, lyu}@cse.cuhk.edu.hk.*

Manuscript received 15 Apr. 2009; revised 6 Apr. 2010; accepted 6 Aug. 2010; published online 23 Dec. 2010.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2009-04-0350. Digital Object Identifier no. 10.1109/TKDE.2011.18.

Fortunately, on the Web, no matter what types of data sources are used for recommendations, in most cases, these data sources can be modeled in the form of various types of graphs. If we can design a general graph recommendation algorithm, we can solve many recommendation problems on the Web. However, when designing such a framework for recommendations on the Web, we still face several challenges that need to be addressed.

The first challenge is that it is not easy to recommend latent semantically relevant results to users. Take *Query Suggestion* as an example, there are several outstanding issues that can potentially degrade the quality of the recommendations, which merit investigation. The first one is the ambiguity which commonly exists in the natural language. Queries containing ambiguous terms may confuse the algorithms which do not satisfy the information needs of users. Another consideration, as reported in [26] and [53], is that users tend to submit short queries consisting of only one or two terms under most circumstances, and short queries are more likely to be ambiguous. Through the analysis of a commercial search engine's query logs recorded over three months in 2006, we observe that 19.4 percent of Web queries are single term queries, and further 30.5 percent of Web queries contain only two terms. Third, in most cases, the reason why users perform a search is because they have little or even no knowledge about the topic they are searching for. In order to find satisfactory answers, users have to rephrase their queries constantly.

The second challenge is how to take into account the personalization feature. Personalization is desirable for many scenarios where different users have different information needs. As an example, Amazon.com has been the early adopter of personalization technology to recommend products to shoppers on its site, based upon their previous purchases. Amazon makes an extensive use of collaborative filtering in its personalization technology. The adoption of personalization will not only filter out irrelevant information to a person, but also provide more specific information that is increasingly relevant to a person's interests.

The last challenge is that it is time consuming and inefficient to design different recommendation algorithms for different recommendation tasks. Actually, most of these recommendation problems have some common features, where a general framework is needed to unify the recommendation tasks on the Web. Moreover, most of existing methods are complicated and require to tune a large number of parameters.

In this paper, aiming at solving the problems analyzed above, we propose a general framework for the recommendations on the Web. This framework is built upon the heat diffusion on both undirected graphs and directed graphs, and has several advantages.

1. It is a general method, which can be utilized to many recommendation tasks on the Web.
2. It can provide latent semantically relevant results to the original information need.
3. This model provides a natural treatment for personalized recommendations.
4. The designed recommendation algorithm is scalable to very large data sets.

The empirical analysis on several large scale data sets (AOL clickthrough data and Flickr image tags data) shows that our proposed framework is effective and efficient for generating high-quality recommendations.

The rest of the paper is organized as follows. We review related work in Section 2. Section 3 presents the diffusion models on both undirected graphs and directed graphs. In Section 4, we demonstrate the empirical analysis of our models and recommendation algorithms on several diversified data sources. Finally, conclusion is given in Section 5.

2 Related Work

Recommendation on the Web is a general term representing a specific type of information filtering technique that attempts to present information items (queries, movies, images, books, Web pages, etc.) that are likely of interest to the users. In this section, we review several work related to recommendation, including collaborative filtering, query suggestion techniques, image recommendation methods, and clickthrough data analysis.

2.1 Collaborative Filtering

Two types of collaborative filtering approaches are widely studied: neighborhood-based and model-based.

The neighborhood-based approaches are the most popular prediction methods and are widely adopted in commercial collaborative filtering systems [37], [47]. The most analyzed examples of neighborhood-based collaborative filtering include user-based approaches [7], [21] and item-based approaches [15], [37], [50]. User-based approaches predict the ratings of active users based on the ratings of their similar users, and item-based approaches predict the ratings of active users based on the computed information of items similar to those chosen by the active user. User-based and item-based approaches often use the Pearson Correlation Coefficient algorithm (PCC) [47] and the Vector Space Similarity algorithm (VSS) [7] as the similarity computation methods. PCC-based collaborative filtering generally can achieve higher performance than the other popular algorithm VSS, since it considers the differences of user rating style.

In the model-based approaches, training data sets are used to train a predefined model. Examples of model-based approaches include the clustering model [33], the aspect models [23], [24], [52] and the latent factor model [9]. Kohrs and Merialdo [33] presented an algorithm for collaborative filtering based on hierarchical clustering, which tried to balance robustness and accuracy of predictions, especially when few data were available. Hofmann [23] proposed an algorithm based on a generalization of probabilistic latent semantic analysis to continuous-valued response variables. Recently, several matrix factorization methods [39], [40], [46], [48], [49], [54] have been proposed for collaborative filtering. These methods all focus on fitting the user-item rating matrix using low-rank approximations, and use it to make further predictions. The premise behind a low-dimensional factor model is that there is only a small number of factors influencing preferences, and that a user's preference vector is determined by how each factor applies to that user.

Although collaborative filtering methods have been extensively studied recently, most of these methods require the user-item rating matrix. However, on the Web, in most of the cases, rating data are always unavailable since information on the Web is less structured and more diverse. Hence, collaborative filtering algorithms cannot be directly applied to most of the recommendation tasks on the Web, like query suggestion and image recommendation.

2.2 Query Suggestion

In order to recommend relevant queries to Web users, a valuable technique, query suggestion, has been employed by some prominent commercial search engines, such as *Yahoo!*,[3] *Live Search*,[4] *Ask*,[5] and *Google*.[6] However, due to commercial reasons, a few public papers have been released to reveal the methods they adopt.

The goal of query suggestion is similar to that of query expansion [11], [13], [56], [61], query substitution [31], and query refinement [35], [57], which all focus on understanding users' search intentions and improving the queries submitted by users. Query suggestion is closely related to query expansion or query substitution, which extends the original query with new search terms to narrow down the scope of the search. But different from query expansion, query suggestion aims to suggest full queries that have been formulated by previous users so that query integrity and coherence are preserved in the suggested queries [18]. Query refinement is another closely related notion, since the objective of query refinement is interactively recommending new queries related to a particular query.

In [61], local (i.e., query-dependent documents) and global (i.e., the whole corpus) documents are employed in query expansion by applying the measure of global analysis to the selection of query terms in local feedback. Although experimental results show that this method is generally

3. http://www.yahoo.com.
4. http://www.live.com.
5. http://www.ask.com.
6. http://www.google.com.

more effective than global analysis, it performs worse than the query expansion method proposed in [13] based on user interactions recorded in user logs. In another approach reported in [35], anchor texts are employed for the purpose of query refinement. This work is based on the observation that Web queries and anchor texts are highly similar. These methods employ different kinds of data sources (documents, anchor texts, query logs, etc.) for suggesting queries. Since most of these methods are only designed for query suggestions, the extensibility of these methods are very limited. In [4] and [16], two query recommendation methods based on clickthrough data are proposed. The main disadvantage of these two algorithms is that they ignore the rich information embedded in the query-click bipartite graph,[7] and consider only queries that appear in the query logs, potentially losing the opportunity to recommend highly semantically related queries to users. Cao et al. [10] developed a context-aware query suggestion method by mining clickthrough and session data. This work first extracts some concepts from the clickthrough data by building clusters. Then, these concepts as well as the query sessions are employed to build a concept sequence suffix tree for query suggestion. Recently, Mei et al. proposed a general query suggestion method using hitting time on the query-click bipartite graph in [42]. This method can generate semantically relevant queries to users' information needs. The main advantage of this work is that it can suggest some long tail queries (infrequent queries) to users. However, this is also the disadvantage of this approach since sometimes it may accidentally rank the infrequent queries highly in the results while potentially downgrades the ranks of the most related queries.

Actually, as reported in [42], several different ranking methods using random walks can also be employed into the query suggestion tasks on a query-URL bipartite graph, including PageRank [8], HITS [32], etc. PageRank is basically computing the stationary distribution of a smoothed Markov chain. Personalized PageRank generalizes PageRank by smoothing the Markov chain with a query-specific jumping probability vector instead of a uniform vector, thus, is often used for query-dependent ranking [19], [20], [28]. HITS is an alternative query-dependent ranking algorithm which computes hub and authority scores in an iterative way. In [12], the query suggestion and the document retrieval problem are interpreted using the Markov random walks, in which the queries or documents with the largest probabilities after t-step random walks are recommended to the users.

2.3 Clickthrough Data Analysis

In the field of clickthrough data analysis, the most common usage is for optimizing Web search results or rankings [1], [29], [30], [55], [59]. In [59], Web search logs are utilized to effectively organize the clusters of search results by 1) learning "interesting aspects" of a topic and 2) generating more meaningful cluster labels. In [30], a ranking function is learned from the implicit feedback extracted from search engine clickthrough data to provide personalized search results for users. Besides ranking, clickthrough data is also well studied in the query clustering problem [5], [60]. Query clustering is a process used to discover frequently asked questions or most popular topics on a search engine. This process is crucial for search engines based on question-answering [60]. Recently, clickthrough data has been analyzed and applied to several interesting research topics, such as Web query hierarchy building [51] and extraction of class attributes [44]. In [51], the proposed method consists of two stages: generating candidate queries and determining "generalization/specialization" relations between these queries in a hierarchy. A typical relationship can be learning from clickthrough data is that "bmw" is a child of "car." The method proposed in [44] can extract attributes such as "capital city" and "President" for the class "Country," or "cost," "manufacturer" and "side effects" for the class "Drug." The method initially relies on a small set of linguistically motivated extraction patterns applied to each entry from the query logs, then employs a series of Web-based precision-enhancement filters to refine and rank the candidate attributes.

2.4 Image Recommendation

Besides query suggestion, another interesting recommendation application on the Web is image recommendation. Image recommendation systems, like Photoree,[8] focus on recommending interesting images to Web users based on users' preference. Normally, these systems first ask users to rate some images as they like or dislike, and then recommend images to the users based on the tastes of the users. In the academia, a few tasks are proposed to solve the image recommendation problems since this is a relatively new field and analyzing the image contents is a challenge job. Recently, in [63], by employing the Flickr data set, Yang et al. proposed a context-based image search and recommendation method to improve the image search quality and recommend related images and tags. However, since it is a context-based method, the computational complexity is very high and it cannot scale to large data sets. While in our framework proposed in this paper, by diffusing on the image-tag bipartite graph with one or more images, we can accurately and efficiently suggest semantically relevant nonpersonalized or personalized images to the users.

In general, comparing with previous work, our work is a general framework which can be effectively, efficiently, and naturally applied to most of the recommendation tasks on the Web.

3 Diffusion on Graphs

In this section, we first introduce a novel graph diffusion model based on heat diffusion. This model can be applied to both undirected graphs and directed graphs. We then present how to infer the parameter based on the graph structure. Last, we analyze the computational complexity of our model.

3.1 Heat Diffusion

Heat diffusion is a physical phenomenon. In a medium, heat always flows from a position with high temperature to a position with low temperature. Recently, heat

7. The query-click graph is a bipartite graph with queries on one side and clicked documents on the other.

8. http://www.photoree.com.

diffusion-based approaches have been successfully applied in various domains such as classification and dimensionality reduction problems [6], [34], [36]. Lafferty and Lebanon [36] approximated the heat kernel for a multinomial family in a closed form, from which great improvements were obtained over the use of Gaussian or linear kernels. In [34], Kondor and Lafferty proposed the use of a discrete diffusion kernel for categorical data, and showed that the simple diffusion kernel on the hypercube can result in good performance for such data. Belkin and Niyogi employed a heat kernel to construct the weight of a neighborhood graph, and apply it to a nonlinear dimensionality reduction algorithm in [6]. In [62], Yang et al. proposed a ranking algorithm known as the DiffusionRank using heat diffusion process; simulations showed that it is very robust to Web spamming.

In this paper, we use heat diffusion to model the similarity information propagation on Web graphs. In Physics, the heat diffusion is always performed on a geometric manifold with initial conditions. However, it is very difficult to represent the Web as a regular geometry with a known dimension. This motivates us to investigate the heat flow on a graph. The graph is considered as an approximation to the underlying manifold, and so the heat flow on the graph is considered as an approximation to the heat flow on the manifold.

3.2 Diffusion on Undirected Graphs

Consider an undirected graph $G = (V, E)$, where V is the vertex set, and $V = \{v_1, v_2, \ldots, v_n\}$. $E = \{(v_i, v_j) \mid \text{there is an edge between } v_i \text{ to } v_j\}$ is the set of all edges. The edge (v_i, v_j) is considered as a pipe that connects nodes v_i and v_j. The value $f_i(t)$ describes the heat at node v_i at time t, beginning from an initial distribution of heat given by $f_i(0)$ at time zero. $\mathbf{f}(t)$ denotes the vector consisting of $f_i(t)$.

We construct our model as follows: suppose, at time t, each node i receives an amount $M(i, j, t, \Delta t)$ of heat from its neighbor j during a time period Δt. The heat $M(i, j, t, \Delta t)$ should be proportional to the time period Δt and the heat difference $f_j(t) - f_i(t)$. Moreover, the heat flows from node j to node i through the pipe that connects nodes i and j. Based on this consideration, we assume that $M(i, j, t, \Delta t) = \alpha(f_j(t) - f_i(t))\Delta t$, where α is the thermal conductivity—the heat diffusion coefficient. As a result, the heat difference at node i between time $t + \Delta t$ and time t will be equal to the sum of the heat that it receives from all its neighbors. This is formulated as

$$\frac{f_i(t + \Delta t) - f_i(t)}{\Delta t} = \alpha \sum_{j:(v_j, v_i) \in E} (f_j(t) - f_i(t)), \tag{1}$$

where E is the set of edges. To find a closed form solution to (1), we express it in a matrix form

$$\frac{\mathbf{f}(t + \Delta t) - \mathbf{f}(t)}{\Delta t} = \alpha(\mathbf{H} - \mathbf{D})\mathbf{f}(t), \tag{2}$$

where

$$H_{ij} = \begin{cases} 1, & (v_i, v_j) \in E \text{ or } (v_j, v_i) \in E, \\ 0, & i = j, \\ 0, & \text{otherwise}, \end{cases} \tag{3}$$

and

$$D_{ij} = \begin{cases} d(v_i), & i = j, \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where $d(v_i)$ is the degree of node v_i. From the definition, the matrix $\mathbf{D}$ is a diagonal matrix.

In order to generate a more generalized representation, we normalize all the entries in matrices $\mathbf{H}$ and $\mathbf{D}$ by the degree of each node. The matrices $\mathbf{H}$ and $\mathbf{D}$ can be modified to

$$H_{ij} = \begin{cases} 1/d(v_i), & (v_i, v_j) \in E, \\ 0, & i = j, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

and

$$D_{ij} = \begin{cases} 1, & i = j, \\ 0, & \text{otherwise}. \end{cases} \tag{6}$$

In the limit $\Delta t \to 0$, this becomes

$$\frac{d}{dt}\mathbf{f}(t) = \alpha t(\mathbf{H} - \mathbf{D})\mathbf{f}(t). \tag{7}$$

Solving this differential equation, we have

$$\mathbf{f}(1) = e^{\alpha(\mathbf{H}-\mathbf{D})}\mathbf{f}(0), \tag{8}$$

where $d(v)$ denotes the degree of the node v, and $e^{\alpha(\mathbf{H}-\mathbf{D})}$ could be extended as

$$\begin{aligned} e^{\alpha(\mathbf{H}-\mathbf{D})} = \mathbf{I} + \alpha(\mathbf{H} - \mathbf{D}) + \frac{\alpha^2}{2!}(\mathbf{H} - \mathbf{D})^2 \\ + \frac{\alpha^3}{3!}(\mathbf{H} - \mathbf{D})^3 + \cdots. \end{aligned} \tag{9}$$

The matrix $e^{\alpha(\mathbf{H}-\mathbf{D})}$ is called the diffusion kernel in the sense that the heat diffusion process continues infinitely many times from the initial heat diffusion.

In order to interpret (8) and the heat diffusion process more intuitively, we construct a small undirected graph with only five nodes as showed in Fig. 1a.

Initially, at time zero, suppose node 1 is given 3 units of heat, and node 2 is given 2 units of heat; then the vector $\mathbf{f}(0)$ equals $[3, 2, 0, 0, 0]^{\mathbf{T}}$. The entries in matrix $\mathbf{H} - \mathbf{D}$ are

$$\mathbf{H} - \mathbf{D} = \begin{pmatrix} -1 & 1 & 1 & 1 & 1 \\ \frac{1}{4} & -1 & 0 & 0 & 0 \\ \frac{1}{4} & 0 & -1 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & -1 & 0 \\ \frac{1}{4} & 0 & 0 & 0 & -1 \end{pmatrix}.$$

Without loss of generality, we set the thermal conductivity $\alpha = 1$, and vary time t from 0 to 1 with a step of 0.05. The curve for the amount of heat at each node with time is shown in Fig. 1b. We can see that, as time passes, the heat sources nodes 1 and 2 will diffuse their heat to nodes 3, 4, and 5. The heat of nodes 3, 4, and 5 will increase respectively, and the trends of their heat curves are the same since these three nodes are symmetric in this graph.

Another example is shown in Fig. 1c. Initially, at time zero, suppose node 1 is given 4 units of heat,then the vector $\mathbf{f}(0)$ equals $[4, 0, 0, 0]^{\mathbf{T}}$. The related heat curve is shown in Fig. 1b. We can see that the node 2, the closest node to the heat source, gains more heat than other nodes. This also

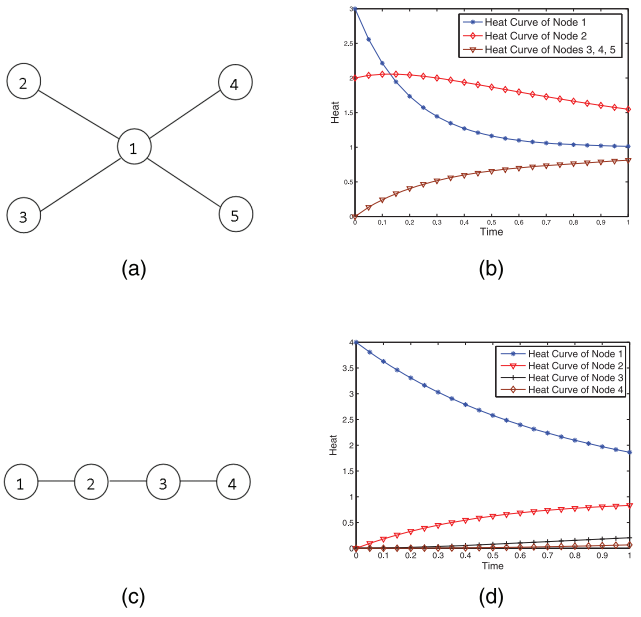


Fig. 1. Two simple heat diffusion examples on an undirected graph. (a) Example 1. (b) Curve of heat change with time. (c) Example 2. (d) Curve of heat change with time.

indicates that if a node has more paths connected to the heat source, it will potentially obtain more heat. This is a perfect property for recommending relevant nodes on a graph.

3.3 Diffusion on Directed Graphs

The above heat diffusion model is designed for undirected graphs, but in many situations, the Web graphs are directed, especially in online recommender systems or knowledge sharing sites. Every user in knowledge sharing sites typically has a trust list. The users in the trust list can influence this user deeply. These relationships are directed since user a is in the trust list of user b, but user b might not be in the trust list of user a. At the same time, the extent of trust relations is different since user u_i may trust user u_j with trust score 1 while trust user u_k only with trust score 0.2. Hence, there are different weights associated with the relations. Based on this consideration, we modify the heat diffusion model for the directed graphs as follows.

Consider a directed graph $G = \{V, E, W\}$, where V is the vertex set, and $V = \{v_1, v_2, \ldots, v_n\}$. $W = \{w_{ij} \mid$ where w_{ij} is the probability that edge (v_i, v_j) exists$\}$ or the weight that is associated with this edge. $E = \{(v_i, v_j) \mid$ there is an edge from v_i to v_j and $w_{ij} > 0\}$ is the set of all edges.

On a directed graph $G(V, E)$, in the pipe (v_i, v_j), heat flows only from v_i to v_j. Suppose at time t, each node v_i receives $RH = RH(i, j, t, \Delta t)$ amount of heat from v_j during a period of Δt. We make three assumptions: 1) RH should be proportional to the time period Δt; 2) RH should be proportional to the heat at node v_j; and 3) RH is zero if there is no link from v_j to v_i. As a result, v_i will receive $\sum_{j:(v_j,v_i)\in E} \sigma_j f_j(t)\Delta t$ amount of heat from all its neighbors that point to it.

At the same time, node v_i diffuses $DH(i, t, \Delta t)$ amount of heat to its subsequent nodes. We assume that

1. The heat $DH(i, t, \Delta t)$ should be proportional to the time period Δt.
2. The heat $DH(i, t, \Delta t)$ should be proportional to the heat at node v_i.
3. Each node has the same ability to diffuse heat.
4. The heat $DH(i, t, \Delta t)$ should be proportional to the weight assigned between node v_i and its subsequent nodes. As a result, node v_i will diffuse $\alpha w_{ij} f_i(t)\Delta t / \sum_{k:(i,k)\in E} w_{ik}$ amount of heat to each of its subsequent nodes v_j, and each v_j should receive $\alpha w_{ij} f_i(t)\Delta t / \sum_{k:(i,k)\in E} w_{ik}$ amount of heat from node v_i.

Therefore, $\sigma_j = \alpha w_{ji} / \sum_{k:(j,k)\in E} w_{jk}$. In the case that the outdegree of node v_i equals zero, we assume that this node will not diffuse heat to others. To sum up, the heat difference at node v_i between time $t + \Delta t$ and t will be equal to the sum of the heat that it receives, deducted by what it diffuses. This is formulated as

$$\frac{f_i(t+\Delta t) - f_i(t)}{\Delta t} = \alpha\left(-\tau_i f_i(t) + \sum_{j:(v_j,v_i)\in E} \frac{w_{ji}}{\sum_{k:(j,k)\in E} w_{jk}} f_j(t)\right), \tag{10}$$

where τ_i is a flag to identify whether node v_i has any outlinks. Solving it, we obtain

$$\mathbf{f}(1) = e^{\alpha(\mathbf{H}-\mathbf{D})}\mathbf{f}(0), \tag{11}$$

where

$$H_{ij} = \begin{cases} w_{ji} \Big/ \sum_{k:(j,k)\in E} w_{jk}, & (v_j, v_i) \in E, \\ 0, & i = j, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

and

$$D_{ij} = \begin{cases} \tau_i, & i = j, \\ 0, & \text{otherwise}. \end{cases} \tag{13}$$

3.4 Random Jump

The heat can only propagate through the links that connect nodes in a given graph, but in fact, there are random relations among different nodes even if these nodes are not connected. For an example, in the clickthrough data, people of different cultures, genders, ages, and environments, may implicitly link queries together, but we do not know these latent relations. Another good example is the trust relations in a social network. On online social network sites, users always explicitly state the trust relations to other users. Actually, there are some other implicit hidden trust relations among these users that cannot be observed. Hence, to capture these relations, we propose to add a uniform random relation among different nodes. More specifically, let γ denote the probability that such phenomena happen, and $(1-\gamma)$ is the probability of taking a "random jump." Without any prior knowledge, we set $\mathbf{g} = \frac{1}{n}\mathbf{1}$, where $\mathbf{g}$ is a uniform stochastic distribution vector, $\mathbf{1}$ is the vector of all ones, and n is the number of nodes. Based on the above consideration, we modify our model to

$$\mathbf{f}(1) = e^{\alpha\mathbf{R}}\mathbf{f}(0), \quad \mathbf{R} = \gamma(\mathbf{H} - \mathbf{D}) + (1-\gamma)\mathbf{g}\mathbf{1}^T. \tag{14}$$

Following the setting of γ in PageRank [17], [43], we set $\gamma = 0.85$ in all of our experiments conducted in Section 4.

TABLE 1
Samples of Search Engine Clickthrough Data

ID	Query	URL	Rank	Calendar Time
358	facebook	http://www.facebook.com	1	2009-01-01 07:17:12
358	facebook	http://en.wikipedia.org/wiki/Facebook	3	2009-01-01 07:19:18
3968	apple iphone	http://www.apple.com/iphone/	1	2009-01-01 07:20:36
1398	Michael Jordan	http://www.youtube.com/watch?v=OFxXSXGd4hs	4	2009-01-01 07:30:18
...	...	...	...	...

3.5 Complexity Analysis

When the graph is very large, a direct computation of $e^{\alpha \mathbf{R}}$ is very time consuming. We adopt its discrete approximation to compute the heat diffusion equation

$$\mathbf{f}(1) = \left(\mathbf{I} + \frac{\alpha}{P}\mathbf{R}\right)^P \mathbf{f}(0), \tag{15}$$

where P is a positive integer. In order to reduce the computational complexity, we introduce two techniques: 1) since $\mathbf{f}(0)$ is a vector, we iteratively calculate $(\mathbf{I} + \frac{\alpha}{P}\mathbf{R})^P \mathbf{f}(0)$ by applying the operator $(\mathbf{I} + \frac{\alpha}{P}\mathbf{R})$ to $\mathbf{f}(0)$; 2) for matrix $\mathbf{R}$, we employ a data structure which only stores the information of nonzero entries, since it is a very sparse matrix. Thus, supposing a graph is connected by M edges (relationships between nodes), the complexity of executing the heat diffusion process is $O(PM)$, which represents the number of iterations P multiplied by the number of edges M in a graph. In most cases, $P = 10$ is enough for approximating the heat diffusion equation. The complexity $O(PM)$ shows that our heat diffusion algorithm enjoys very good performance in scalability since it is linear with respect to the number of edges in the graph.

However, since the size of Web information is very large, the graph built upon the Web information can become extremely large. Then, the complexity $O(PM)$ is also too high, and the algorithm becomes time consuming and inefficient to get a solution. To overcome this difficulty, we first extract a subgraph starting from the heat sources. Given the heat sources, the subgraph is constructed by using depth-first search in the original graph. The search stops when the number of nodes is larger than a predefined number. Then, the diffusion processes will be performed on this subgraph efficiently and effectively. Generally, it will not decrease the qualities of the heat diffusion processes since the nodes too far away from the heat sources are normally not related to the sources.

4 Empirical Analysis

In Section 3, we introduced our graph diffusion models for recommendations. In this section, 1) we show how to convert different Web data sources into correct graphs in our models; and 2) we conduct several experiments on query suggestions, and image recommendations.[9]

4.1 Query Suggestion

Query Suggestion is a technique widely employed by commercial search engines to provide related queries to users' information need. In this section, we demonstrate how our method can benefit the query suggestion, and how to mine latent semantically similar queries based on the users' information need.

4.1.1 Data Collection

We construct our query suggestion graph based on the clickthrough data of the *AOL* search engine [45]. In total, this data set spans 3 months from 01 March, 2006 to 31 May, 2006. There are a total of 19,442,629 lines of clickthrough information, 4,802,520 unique queries, and 1,606,326 unique URLs.

Clickthrough data record the activities of Web users, which reflect their interests and the latent semantic relationships between users and queries as well as queries and clicked Web documents. As shown in Table 1, each line of clickthrough data contains the following information: a user ID (u), a query (q) issued by the user, a URL (l) on which the user clicked, the rank (r) of that URL, and the time (t) at which the query was submitted for search. Thus, the clickthrough data can be represented by a set of quintuples $\langle u, q, l, r, t\rangle$. From a statistical point of view, the query word set corresponding to a number of Web pages contains human knowledge on how the pages are related to their issued queries [55]. Thus, in this paper, we utilize the relationships of queries and Web pages for the construction of the bipartite graph containing two types of vertices $\langle q, l\rangle$. The information regarding user ID, rank and calendar time is ignored.

This data set is the raw data recorded by the search engine, and contains a lot of noise which will potentially affect the effectiveness of our query suggestion algorithm. Hence, we conduct a similar method employed in [59] to clean up the raw data. We filter the data by only keeping those frequent, well formatted, English queries (queries which only contain characters "a," "b," ..., "z," and space). After cleaning and removing duplicates, we get totally 2,019,265 unique queries and 915,771 unique URLs in our data collection. After the construction of the query-URL bipartite graph using this data collection procedure, we observe that a total of 7,633,400 edges exist in the query-URL bipartite graph, which indicates that, on average, each query has 3.78 distinct clicks, and each URL is clicked by 8.34 distinct queries.

4.1.2 Graph Construction

For the query-URL bipartite graph, consider an undirected bipartite graph $B_{ql} = (V_{ql}, E_{ql})$, where $V_{ql} = Q \cup L$, $Q = \{q_1, q_2, \ldots, q_n\}$, and $L = \{l_1, l_2, \ldots, l_p\}$. $E_{ql} = \{(q_i, l_j) |$ there is an edge from q_i to $l_j\}$ is the set of all edges. The edge (q_j, l_k) exists if and only if a user u_i clicked a URL l_k after issuing a query q_j.

9. All the experiments are computed by a workstation consisting of two Intel Xeon CPUs (2.5 GHz, Quad-Core) and 8 Giga memories. The operating system is Windows Server 2003, 32-bit.

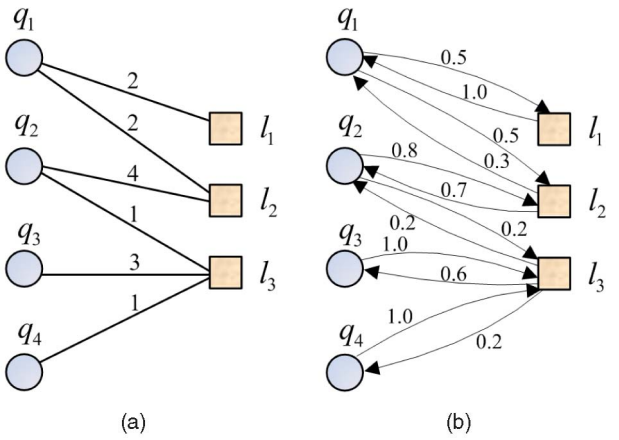


Fig. 2. Graph construction for query suggestion. (a) Query-URL bipartite graph. (b) Converted query-URL bipartite graph.

See Fig. 2a for an example. The values on the edges in Fig. 2a specify how many times a query is clicked on a URL.

We cannot simply employ the bipartite graph extracted from the clickthrough data into the diffusion processes since this bipartite graph is an undirected graph, and cannot accurately interpret the relationships between queries and URLs. Hence, we convert this bipartite graph into Fig. 2b. In this converted graph, every undirected edge in the original bipartite graph is converted into two directed edges. The weight on a directed query-URL edge is normalized by the number of times that the query is issued, while the weight on a directed URL-query edge is normalized by the number of times that the URL is clicked.

4.1.3 Query Suggestion Algorithm

After the conversion of the graph, we can easily design the query suggestion algorithm in Algorithm 1.

Algorithm 1. Query Suggestion Algorithm

1: A converted bipartite graph $G = (V^+ \cup V^*, E)$ consists of query set V^+ and URL set V^*. The two directed edges are weighted using the method introduced in previous section.
2: Given a query q in V^+, a subgraph is constructed by using depth-first search in G. The search stops when the number of queries is larger than a predefined number.
3: As analyzed above, set $\alpha = 1$, and without loss of generality, set the initial heat value of query q $f_q(0) = 1$ (the choice of initial heat value will not affect the suggestion results). Start the diffusion process using
$$\mathbf{f}(1) = e^{\alpha \mathbf{R}} \mathbf{f}(0).$$
4: Output the Top-K queries with the largest values in vector $\mathbf{f}(1)$ as the suggestions.

4.1.4 Query Suggestion Results

We display the suggestion results of our algorithm and those from *Google*, *Yahoo!*, *Live Search*, and *AOL* in Table 2. We call our algorithm DRec which means Recommendations by Diffusion. In our algorithms, the parameter α is set to 1, and the size of the subgraph is set to 5,000. From the suggestions, we can see that the query suggestions generated by our method are generally as good as those from commercial search engines. For some queries, our suggestions are even better.

In order to compare our method with other approaches, we create a set of 200 queries as the testing queries, covering a wide range of topics, such as Computers, Arts, Business, and others. Some of the results generated by our DRec algorithm are shown in Table 3.

From the results, we observe that our recommendation algorithm not only suggests queries which are literally similar to the test queries, but also provides latent semantically relevant recommendations. For instance, if the test query is a technique, such as "java," we recommend "virtual machine" and "sun microsystems." The latter suggestion is the company who owns the *Java Platform*, and the former suggestion is a key feature of the *Java* programming language. They both have high latent semantic relations to the query "java." If the test query is a human name, such as "michael jordan," one of the most successful NBA basketball player, the latent semantic suggestions are "nba," "nike," "jordan xi" (a model of Air Jordan shoes), and "air jordans," All of the results show that our latent semantic query suggestion algorithm has a promising future.

Since the data set we use is different from the data sets that these commercial search engines employ, it is difficult to quantitatively evaluate our results with those from the commercial search engines objectively. Hence, we compare our DRec method with the baseline approaches using SimRank [27], Forward Random Walk (FRW) [12], and Backward Random Walk (BRW) [12].

In the method of SimRank, we use the query-URL bipartite graph to calculate the similarities between queries. Then, based on the similarities, we recommend the top-5 similar queries to users. SimRank is based on the intuition that two queries are very similar if they link to a lot of similar URLs. On the other hand, two URLs are very similar if they are clicked as a result of several similar queries. Based on this intuition, in SimRank, we first calculate the similarities between URLs, then we compute the similarities for queries based on the similarities of URLs. We iteratively update the similarities until they converge.

In FRW and BRW methods, we perform forward random walk and backward random walk starting from the query node on the query-URL graph. After the random walks, we use the top-ranked queries as the suggestions.

Evaluating the quality of semantic relations is difficult, in particular for the contents generated by users, as there are no linguistic resources available. In this paper, we conduct both a manual evaluation by a panel of three human experts, and an automatic evaluation based on the ODP[10] database.

In the evaluation by human experts, the three experts are three PhD students without any overlaps with the authors. They do not know which algorithms are tested. They are also not allowed to communicate with each other during the evaluation process. We ask all the experts to rate the query suggestion results. We define a 6-point scale (0, 0.2, 0.4, 0.6, 0.8, and 1) to measure the relevance between the testing

10. http://www.dmoz.org.

TABLE 2
Query Suggestion Comparisons between DRec and Four Commercial Search Engines

Query = aa					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
american airlines	0.0971	N/A	aa route planner	aa route finder	N/A
alcoholics anonymous	0.0294		aa route finder	aa route planner	
airlines	0.0193		aa autoroute	aa airlines	
airline tickets	0.0107		aa route map	aa route map	
american air	0.0069		aa roadwatch	aa meetings	
Query = travel					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
travelocity	0.0735	travel guide	yahoo travel	travelocity	aol travel
expedia	0.0310	travel magazine	travel agents	orbitz	travel channel
orbitz	0.0180	travelzoo	travel insurance	cheap tickets	liberty travel
airline tickets	0.0122	train travel	travel channel	expedia	american express travel
hotels	0.0036	last minute travel	travel inn	priceline	
Query = disease					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
bird flu	0.0118	N/A	lyme disease	list of diseases	lyme disease
hpv	0.0069		diseases	infectious diseases	parkinsons disease
cdc	0.0057		crohn's disease	rare diseases	celiac disease
scabies	0.0053		celiac disease	liver disease	crohns disease
center for disease control	0.0051		parkinson's disease	heart disease	
Query = pizza					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
pizza hut	0.0398	pizza recipe	pizza hut	pizza hut	pizza hut
dominos pizza	0.0283	pizza history	domino's pizza	pizza recipe	dominos pizza
california pizza kitchen	0.0134	making pizza	california pizza kitchen	pizza hut coupons	papa johns pizza
papa johns	0.0124	pizza games	round table pizza	pizza delivery	dominoes pizza
round table pizza	0.0082	homemade pizza	cici's pizza	pizza history	
Query = free music					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
free music downloads	0.0522	free online music	free music downloads	listen to free music	free music downloads
napster	0.0102	free ipod music	free music videos	free music downloads	free sheet music
limewire	0.0080	free music for myspace	free music lyrics	free radio	free music video downloads
music downloads	0.0062	free country music	legal free music downloads	limewire	
musicmatch	0.0039	free rap music	free music video codes	yahoo music	
Query = windows					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
microsoft	0.0282	house windows	windows media player	windows xp	windows update
windows update	0.0219	anderson windows	windows live messenger	house windows	windows media player
anderson windows	0.0202	pella windows	windows update	windows update	windows live
microsoft updates	0.0104	windows and doors	windows xp	windows live messenger	
internet explorer	0.0094	windows media player	windows vista	windows vista	
Query = job					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
monster	0.0266	job bible	job search	part time jobs	job search
americas job bank	0.0127	book of job	job centre	monster job	job listings
career builder	0.0119	teen job	job bank	job search	usa jobs
monster jobs	0.0077	story of job	job descriptions	government jobs	monster jobs
job search	0.0069	part time jobs	job corps	ups jobs	
Query = pets					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
petfinder	0.0165	neopets	pets at home	pets for sale	pet smart
yahoo pets	0.0144	types of pets	pets for sale	free pets	pet finder
petsmart	0.0122	pictures of pets	virtual pets	pet adoption	pet meds
neopets	0.0113	reptile pets	wonder pets	pets health	pet supplies
avma	0.0063	peta	sims 2 pets	pet suppliers	
Query = president					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
george washington	0.0110	N/A	president bush	list us presidents	barack obama
white house	0.0106		president obama	presidents in order	presidents day
abraham lincoln	0.0094		hillary clinton for president	president facts	us presidents
us presidents	0.0067		president bill clinton	president election	
thomas jefferson	0.0062		alan rosenberg president	presidents for kids	
Query = news					
DRec (with Heat Values)		Google	Yahoo!	Live Search	AOL
msnbc	0.0408	bbc news	false news	fox news	fox news
cnn	0.0129	sports	news yahoo	cnn news	new york daily news
bbc	0.0127	science news	fox news	abc news	cnn news
google news	0.0123	india news	bbc news	local news	buffalo news
fox news	0.0088	entertainment news	cnn news	yahoo news	

TABLE 3
Examples of DRec Query Suggestion Results ($k = 50$)

Testing Queries	Suggestions				
	Top 1	Top 2	Top 3	Top 4	Top 5
michael jordan	nba	nike	jordan xi	air jordans	michael jordan bio
java	sun java	java download	java updates	virtual machine	sun microsystems
apple	itunes	ipod	quicktime	apple ipod	apple stores
fitness	exercise	fitness magazine	muscle and fitness	mens fitness	weight loss
solar system	planets	jupiter	saturn	neptune	pluto
sunglasses	chanel sunglasses	oakley	maui jim sunglasses	designer sunglasses	oakley sunglasses
flower delivery	flowers	florist	gift baskets	cheap flowers	proflowers
wedding	wedding channel	wedding dresses	the knot	wedding plans	wedding poems
astronomy	apod	star charts	planets	solar system	skyandtelescope
real estate	remax	realtor	homes for sale	coldwell banker	houses for sale

queries and the suggested queries, in which 0 means "totally irrelevant" while 1 indicates "entirely relevant." The average values of evaluation results are shown in Fig. 3. We observe that, when measuring the results by human experts, our DRec algorithm increases the accuracy for about 19.81, 13.0, and 7.5 percent comparing with the SimRank, BRW, and FRW algorithm, respectively.

For the automatic evaluation, we utilize the ODP database. ODP, also known as dmoz, is one of the largest, most comprehensive human-edited directories of the Web. In our experiment, we adopt the same method used in [3] to evaluate the quality of the suggested queries. When a user types a query in ODP, besides site matches, we can also find *categories* matches in the form of paths between directories. Moreover, these categories are ordered by relevance. For instance, the query "Java" would provide the hierarchical category "Computers : Programming : Languages : Java," where ":" is used to separate different categories. One of the results for "Virtual Machine" would be "Computers : Programming : Languages : Java : Implementations." Hence, to measure how related two queries are, we can use a notion of similarity between the corresponding categories (as provided by ODP). In particular, we measure the similarity between two categories $\mathcal{D}$ and $\mathcal{D}'$ as the length of their longest common prefix $\mathcal{F}(\mathcal{D}, \mathcal{D}')$ divided by the length of the longest path between $\mathcal{D}$ and $\mathcal{D}'$. More precisely, denoting the length of a path with $|\mathcal{D}|$, this similarity is defined as $Sim(\mathcal{D}, \mathcal{D}') = |\mathcal{F}(\mathcal{D}, \mathcal{D}')|/max\{|\mathcal{D}|, |\mathcal{D}'|\}$. For instance, the similarity between the two queries above is $4/5$ since they share the path "Computers : Programming : Languages : Java" and the longest one is made of five directories. We have evaluated the similarity between two queries by measuring the similarity between the most similar categories of the two queries, among the top five answers provided by ODP.

As shown in Fig. 4, we observe that, when evaluating using ODP database, our proposed DRec algorithm increases the suggestion accuracy for about 22.45, 11.9, and 7.5 percent comparing with the SimRank, BRW, and FRW algorithm, respectively. This indicates again that our proposed query suggestion algorithm is very effective. The major difference between our proposed heat diffusion model and random walk model is that heat diffusion model has the definitions of thermal conductivity and time frame. Heat will diffuse from one node to another node based on thermal conductivity and time frame step by step, which means heat diffusion is a relatively slow diffusion process. However, in the case of random walk, the diffusion from one node to its neighbors is done immediately. Hence, in our model, after the diffusion process, the recommended items will preserve more information of the original node. That is also why the results outperform random walk models.

4.1.5 Impact of Parameter α

The parameter α plays an important role in our method. It controls how fast heat will propagation on the graph. Hence, we also conduct experiments on evaluating the impact of parameter α. The evaluation results are shown in Fig. 5. We can observe that the best α setting is 1. If we

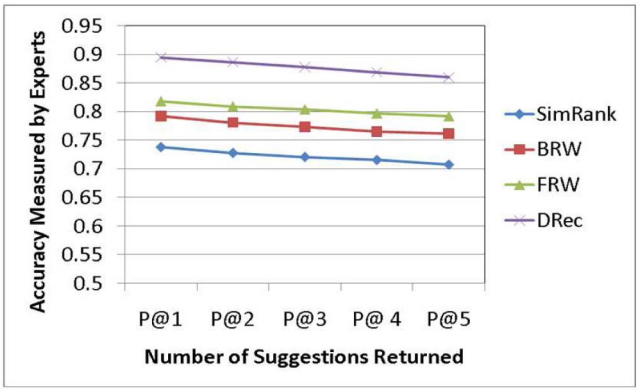


Fig. 3. Accuracy comparisons measured by experts.

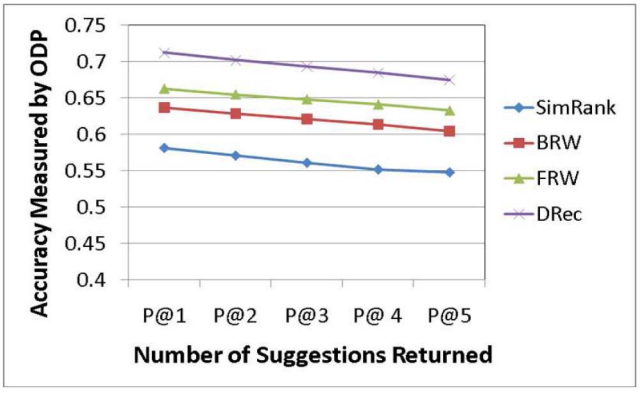


Fig. 4. Accuracy comparisons measured by ODP.

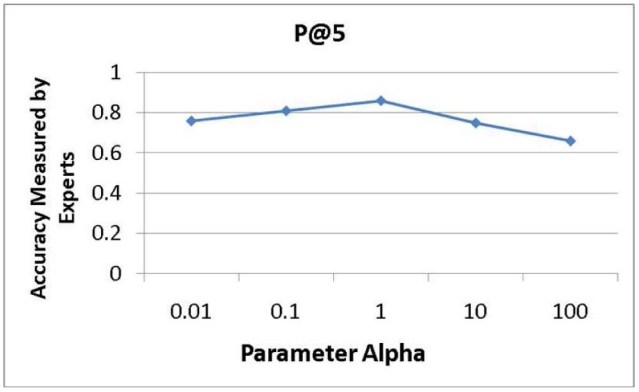


Fig. 5. Impact of α (subgraph size is 5,000).

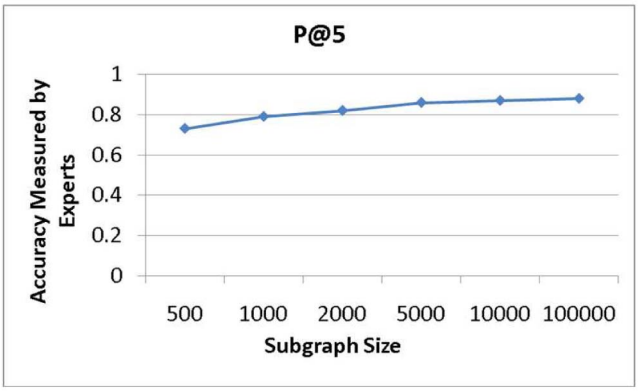


Fig. 6. Impact of the size of subgraph ($\alpha = 1$).

choose relatively smaller thermal conductivity, the performance will drop since some relevant nodes cannot get enough heat. On the other hand, if we choose relatively larger value of α, the performance will also decrease. This is because if the heat transfers very fast, some irrelevant nodes will gain more heat, hence will hurt the performance.

4.1.6 Impact of the Size of Subgraph

As mentioned in Section 3.5, due to the reason that Web graphs are normally very huge, we will perform our algorithm on a subgraph extracted from the original graph. Hence, it is necessary to evaluate how the size of this subgraph affects the recommendation accuracy. Fig. 6 shows the performance changes with different subgraph sizes. We observe that when the size of the graph is very small, like 500, the performance of our algorithm is not very good since this subgraph must ignore some very relevant nodes. When the size of subgraph is increasing, the performance also increases. We also notice that the performance on subgraph with size 5,000 is very close to the performance with size 100,000. This indicates that the nodes that are far away from the query node are normally not relevant with the query node.

4.1.7 Efficiency Analysis

As analyzed in Section 3.5, our algorithm is very efficient, and can be applied to large data sets. Our algorithm has similar complexity with FRW and BRW methods. The computation time for the query suggestion task of these three method (subgraph size is 5,000) is normally around 0.10 seconds. However, SimRank is not very efficient since it has a high computational complexity. It takes more than 15 minutes to compute a query suggestion task in our data set.

4.2 Image Recommendation

Finding effective and efficient methods to search and retrieve images on the Web has been a prevalent line of research for a long time [58]. The situation is even tougher in the research of *Image Recommendation*. In this section, we present how to recommend related images to the given images using *Flickr* data set.

4.2.1 Data Collection

We use Flickr, a popular image hosting Web site and online community for users to share personal photographs, tag photographs, and communicate with other users. As of November 2007, it claims to host more than 2 billion images [2]. Hence, Flickr is an ideal source for the investigation of image-related research.

With this Flickr data set, we can apply our recommendation framework into several application areas, including Image-to-Image Recommendation, Image-to-Tag Suggestion, Tag-to-Image Retrieval, and Tag-to-Tag Suggestion. In this section, we will only show the performance of our model on the image-to-image recommendation application.

For the research of image recommendation, via the Flickr API, we have crawled related information of 36,450,736 images, which spans from January 1 2007 to December 31 2007. For each image, we record the following information: the user ID who submitted the image, the image ID, the time when the image was uploaded, and the tags associated with the image. Totally, we find 807,008 unique users, 5,362,213 unique tags and 322,935,565 edges (tag assignments) between images and tags.

4.2.2 Image Recommendation

Basically, the graph construction for image recommendation is similar to the one introduced in Section 4.1. The only difference is that here the nodes in bipartite graph are images and tags, respectively. By using the similar algorithm which is introduced in Algorithm 1, we can also provide image recommendations. The recommendation results are shown in Fig. 7.

In Fig. 7, we perform three image recommendations. The first recommendation is based on the image shown in Fig. 7a, which is a picture taken from Grand Canyon, a national park in the United States. Figs. 7b to 7f are the five recommendations to this picture. We can observe that these recommendations are all latent semantically related to the original picture. This shows the effectiveness and the promising future of our DRec method. Figs. 7g and 7m are another two examples, with Fig. 7h-Fig. 7i and Fig. 7n-Fig. 7r as the recommendations, respectively. The results also show very good performance of our approach, especially for Fig. 7g. The China Great Wall in this query photo is very dark and we can hardly recognize the object in the photo, while our method still can recommend very clear and related results to users, all of which are scenes from Great Wall.

In order to evaluate the quality of our image recommendation results, we randomly selected 200 images and compare the results generated by our DRec algorithm with

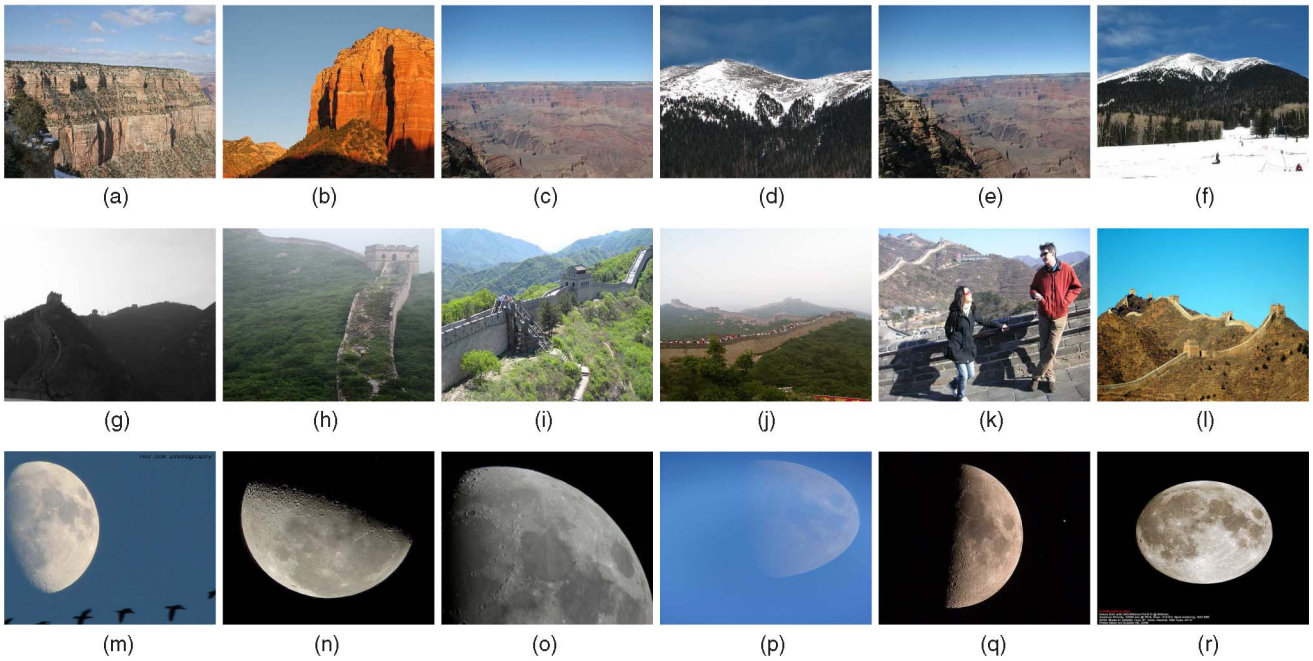


Fig. 7. Examples for image recommendations. (a) Seed image 1. (b) Suggestion 1. (c) Suggestion 2. (d) Suggestion 3. (e) Suggestion 4. (f) Suggestion 5. (g) Seed image 2. (h) Suggestion 1. (i) Suggestion 2. (j) Suggestion 3. (k) Suggestion 4. (l) Suggestion 5. (m) Seed image 3. (n) Suggestion 1. (o) Suggestion 2. (p) Suggestion 3. (q) Suggestion 4. (r) Suggestion 5.

those generated by SimRank, FRW, and BRW approaches. All the results are evaluated by three experts using a 6-point scale (0, 0.2, 0.4, 0.6, 0.8, and 1). From Fig. 8, we can observe that our DRec method consistently performs better than SimRank, FRW, and BRW.

If we use the tags instead of the images as the diffusion sources, then this problem turns to be the problem of tag recommendations. Since the recommendation processes are the same, we do not discuss the results in this paper.

4.2.3 Personalized Image Recommendation

Personalization is becoming more and more important in many applications since it is the best way to understand different information needs from different users.

Actually, our method can be easily extended to the personalized image recommendations. In the query suggestions conducted in Section 4.1 and image suggestions performed in this section, we only employ one node (either a query or an image) as the heat source. In the personalized image recommendations, we can set all the images submitted by a specified user as the heat sources, and then start the diffusion process. This ensures that the suggested images are of interests of this user.

In order to evaluate the quality of our personalized image recommendation method, we create 10 groups: Given1, Given2, ..., and Given10, where Given1 means in this group, all the users only submitted 1 images. We then randomly select 50 users from the user list for each group, hence totally we have 500 users. For each of these users, we start the diffusion processes once with the submitted images as the heat sources. After generating the results, we ask three experts to rate these recommendations. We again define a 6-point scale (0, 0.2, 0.4, 0.6, 0.8, and 1) to measure the relevance between the testing images and the suggested images, in which 0 means "totally irrelevant" while 1 indicates "entirely relevant." The average values of evaluation results for each group are reported in Fig. 9. We

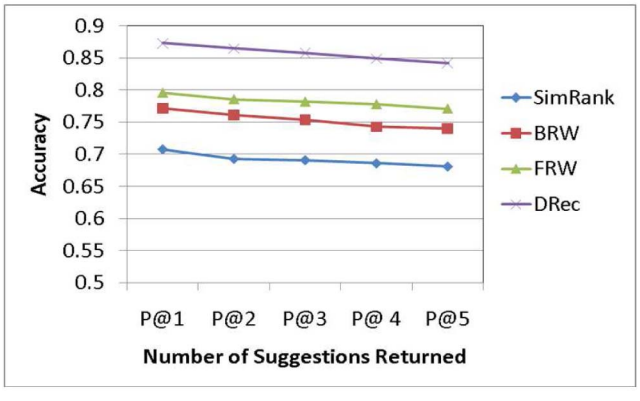


Fig. 8. Accuracy comparisons measured by experts in image recommendation.

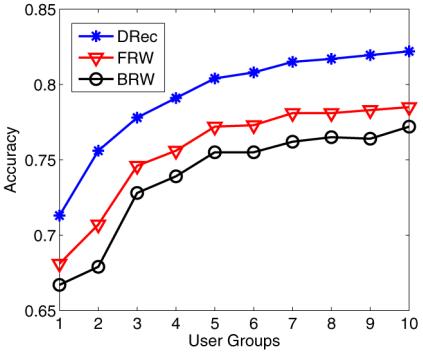


Fig. 9. Accuracy of personalized image recommendations.

TABLE 4
Search Results Improvement

Query	Rank	Web Sites	Heat Values	Query	Rank	Web Sites	Heat Values
sony	1	www.sony.com	0.6237	camera	1	www.dpreview.com	0.2660
	2	www.sonystyle.com	0.1051		2	www.adorama.com	0.2132
	3	www.sony.net	0.0633		3	www.ritzcamera.com	0.2085
	4	www.sonypictures.com	0.0141		4	www.dcresource.com	0.1025
	5	www.sonyericsson.com	0.0139		5	www.digitalcamera-hq.com	0.0131
microsoft	1	www.microsoft.com	0.5337	chocolate	1	www.chocolate.com	0.1749
	2	windowsupdate.microsoft.com	0.2000		2	www.hersheys.com	0.1345
	3	support.microsoft.com	0.0837		3	www.godiva.com	0.1153
	4	office.microsoft.com	0.0313		4	www.ghirardelli.com	0.0971
	5	www.msn.com	0.0132		5	www.scharffenberger.com	0.0876

can observe that our method generally produces high quality results, and as the number of images increases, the recommendation quality also increases.

5 Conclusion

In this paper, we present a novel framework for recommendations on large scale Web graphs using heat diffusion. This is a general framework which can basically be adapted to most of the Web graphs for the recommendation tasks, such as query suggestions, image recommendations, personalized recommendations, etc. The generated suggestions are semantically related to the inputs. The experimental analysis on several large scale Web data sources shows the promising future of this approach.

6 Future Work

6.1 Search Results Improvement

In Table 2, we list the heat values of the suggested queries. These values not only can be used in query suggestions, but also are very informative in the advertisement when customers bid for query terms. Actually, since the diffusions are between all the nodes in the graph (including the nodes representing queries and the nodes representing URLs), all the URLs also have heat values. Hence, it is easy to infer that, for a given query, after the diffusion process, the heat values of URLs represent the relatedness to the original query, which can also be employed as the ranking of these URLs. This ranking actually is the wisdom of the crowd since it is based on the query-URL click data, which reflects the intelligent judgements of the Web users.

The Top-5 Web sites given the queries "sony," "camera," "microsoft," and "chocolate" are shown in Table 4. For example, the ranking order for "sony" is different from all of the results retrieved by those four commercial search engines (which we do not list here due to the space limitation). If this order is incorporated into the original results, the search results can be greatly improved since they are the representations of the implicit votes of all the search users. In the future, we plan to compare this ranking method with other previous Web search results ranking approaches, like [59], [64].

6.2 Social Recommendation

Since our model is quite general, we can apply it to more complicated graphs and applications, such as *Social Recommendation* problem. Recently, as the explosive growth of Web 2.0 applications, social-based applications gain lots of traffics on the Web. Social recommendation, which produces recommendations by incorporating users' social network information, is becoming to be an indispensable feature for the next generation of Web applications.

The social recommendation problem includes two different data sources, which are social network and user-item relation matrices. An example is shown in Fig. 10a. We can see that in the social network graph, there are trust scores between different users, while in the user-item relation matrix, binary relations connect users and items. We can convert these two graphs into a single and consistent one, as shown in Fig. 10b.

With the constructed graph, for each user (heat source), we can start the diffusion process and then recommend the Top-N items to this user. In fact, during the diffusion process on the graph as shown in Fig. 10b, there are two possible ways to diffuse heat from users to items. The first route is within the user-item bipartite graph, which captures the intuition that similar users will purchase (or view) similar items. The second route is passing through the social network graph, which reflects the social interactions and influences between users. Hence, our DRec diffusion method naturally fuses these two data sources together for social recommendations. We plan to conduct this social recommendation research in the future.

Acknowledgments

The work described in this paper was fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Projects Nos. CUHK 412808 and CUHK 415409) and a grant from the

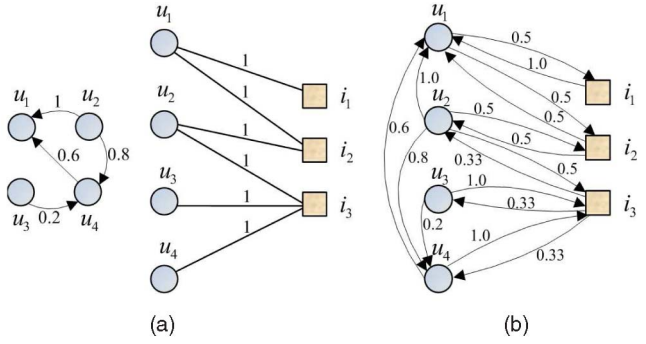


Fig. 10. Example for social recommendation. (a) Social network and user-item relations. (b) Converted graph.

Guangdong Province International Collaboration Scheme (Project No. 2009B050700044). The authors also would like to thank the reviewers and editor for their helpful comments.

REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais, "Improving Web Search Ranking by Incorporating User Behavior Information," *SIGIR '07: Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 19-26, 2006.

[2] E. Auchard, "Flickr to Map the World's Latest Photo Hotspots," *Proc. Reuters,* 2007.

[3] R. TiberiBaeza-Yates and A. Tiberi, "Extracting Semantic Relations from Query Logs," *KDD '07: Proc. 13th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 76-85, 2007.

[4] R.A. Baeza-Yates, C.A. Hurtado, and M. Mendoza, "Query Recommendation Using Query Logs in Search Engines," *Proc. Current Trends in Database Technology (EDBT) Workshops,* pp. 588-596, 2004.

[5] D. Beeferman and A. Berger, "Agglomerative Clustering of a Search Engine Query Log," *KDD '00: Proc. Sixth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 407-416, 2000.

[6] M. Belkin and P. Niyogi, "Laplacian Eigenmaps for Dimensionality Reduction and Data Representation," *Neural Computation,* vol. 15, no. 6, pp. 1373-1396, 2003.

[7] J.S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. 14th Conf. Uncertainty in Artificial Intelligence (UAI),* 1998.

[8] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," *Computer Networks and ISDN Systems,* vol. 30, nos. 1-7, pp. 107-117, 1998.

[9] J. Canny, "Collaborative Filtering with Privacy via Factor Analysis," *SIGIR '07: Proc. 25th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 238-245, 2002.

[10] 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," *KDD '08: Proc. 14th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 875-883, 2008.

[11] P.A. Chirita, C.S. Firan, and W. Nejdl, "Personalized Query Expansion for the Web," *SIGIR '07: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 7-14, 2007.

[12] N. Craswell and M. Szummer, "Random Walks on the Click Graph," *SIGIR '07: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 239-246, 2007.

[13] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma, "Query Expansion by Mining User Logs," *IEEE Trans. Knowledge Data Eng.,* vol. 15, no. 4, pp. 829-839, July/Aug. 2003.

[14] A.S. Das, M. Datar, A. Garg, and S. Rajaram, "Google News Personalization: Scalable Online Collaborative Filtering," *WWW '07: Proc. 16th Int'l Conf. World Wide Web,* pp. 271-280, 2007.

[15] M. Deshpande and G. Karypis, "Item-Based Top-n Recommendation," *ACM Trans. Information Systems,* vol. 22, no. 1, pp. 143-177, 2004.

[16] G. Dupret and M. Mendoza, "Automatic Query Recommendation Using Click-Through Data," *Proc. Int'l Federation for Information Processing, Professional Practice in Artificial Intelligence (IFIP PPAI),* pp. 303-312, 2006.

[17] N. Eiron, K.S. McCurley, and J.A. Tomlin, "Ranking the Web Frontier," *WWW '04: Proc. 13th Int'l Conf. World Wide Web,* pp. 309-318, 2004.

[18] W. Gao, C. Niu, J.-Y. Nie, M. Zhou, J. Hu, K.-F. Wong, and H.-W. Hon, "Cross-Lingual Query Suggestion Using Query Logs of Different Languages," *SIGIR '07: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 463-470, 2007.

[19] T. Haveliwala, A. Kamvar, and G. Jeh, "An Analytical Comparison of Approaches to Personalizing Pagerank," technical report, 2003.

[20] T.H. Haveliwala, "Topic-Sensitive Pagerank," *WWW '04: Proc. 11th Int'l Conf. World Wide Web,* pp. 517-526, 2002.

[21] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl, "An Algorithmic Framework for Performing Collaborative Filtering," *SIGIR '99: Proc. 22nd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 230-237, 1999.

[22] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl, "Evaluating Collaborative Filtering Recommender Systems," *ACM Trans. Information Systems,* vol. 22, no. 1, pp. 5-53, 2004.

[23] T. Hofmann, "Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis," *SIGIR '03: Proc. 26th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 259-266, 2003.

[24] T. Hofmann, "Latent Semantic Models for Collaborative Filtering," *ACM Trans. Information Systems,* vol. 22, no. 1, pp. 89-115, 2004.

[25] Z. Huang, H. Chen, and D. Zeng, "Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering," *ACM Trans. Information Systems,* vol. 22, no. 1, pp. 116-142, 2004.

[26] B.J. Jansen, A. Spink, J. Bateman, and T. Saracevic, "Real Life Information Retrieval: A Study of User Queries on the Web," *ACM SIGIR Forum,* vol. 32, no. 1, pp. 5-17, 1998.

[27] G. Jeh and J. Widom, "Simrank: A Measure of Structural-Context Similarity," *KDD '02: Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 538-543, 2002.

[28] G. Jeh and J. Widom, "Scaling Personalized Web Search," *WWW '04: Proc. 12th Int'l Conf. World Wide Web,* pp. 271-279, 2003.

[29] T. Joachims, "Optimizing Search Engines Using Clickthrough Data," *KDD '02: Proc. Eighth ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* pp. 133-142, 2002.

[30] T. Joachims and F. Radlinski, "Search Engines that Learn from Implicit Feedback," *Computer,* vol. 40, no. 8, pp. 34-40, 2007.

[31] R. Jones, B. Rey, O. Madani, and W. Greiner, "Generating Query Substitutions," *WWW '06: Proc. 15th Int'l Conf. World Wide Web,* pp. 387-396, 2006.

[32] J.M. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," *J. ACM,* vol. 46, no. 5, pp. 604-632, 1999.

[33] A. Kohrs and B. Merialdo, "Clustering for Collaborative Filtering Applications," *Proc. Computational Intelligence for Modelling, Control and Automation (CIMCA),* 1999.

[34] R.I. Kondor and J.D. Lafferty, "Diffusion Kernels on Graphs and Other Discrete Input Spaces," *ICML '02: Proc. 19th Int'l Conf. Machine Learning,* pp. 315-322, 2002.

[35] R. Kraft and J. Zien, "Mining Anchor Text for Query Refinement," *WWW '04: Proc 13th Int'l Conf. World Wide Web,* pp. 666-674, 2004.

[36] J.D. Lafferty and G. Lebanon, "Diffusion Kernels on Statistical Manifolds," *J. Machine Learning Research,* vol. 6, pp. 129-163, 2005.

[37] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," *IEEE Internet Computing,* vol. 7, no. 1, pp. 76-80, Jan./Feb. 2003.

[38] H. Ma, I. King, and M.R. Lyu, "Effective Missing Data Prediction for Collaborative Filtering," *SIGIR '07: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 39-46, 2007.

[39] H. Ma, I. King, and M.R. Lyu, "Learning to Recommend with Social Trust Ensemble," *SIGIR '09: Proc. 32nd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 203-210, 2009.

[40] H. Ma, H. Yang, M.R. Lyu, and I. King, "SoRec: Social Recommendation Using Probabilistic Matrix Factorization," *CIKM '08: Proc. 17th ACM Conf. Information and Knowledge Management,* pp. 931-940, 2008.

[41] B. Marlin, "Modeling User Rating Profiles for Collaborative Filtering," *Advances in Neural Information Processing Systems 16,* S. Thrun, L. Saul, and B. Schölkopf, eds., MIT Press, 2004.

[42] Q. Mei, D. Zhou, and K. Church, "Query Suggestion Using Hitting Time," *CIKM '08: Proc. 17th ACM Conf. Information and Knowledge Management,* pp. 469-477, 2008.

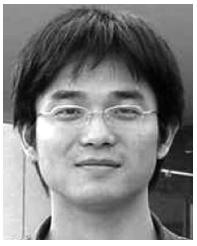
[43] L. Page, S. Brin, R. Motwani, and T. Winograd, "The Pagerank Citation Ranking: Bringing Order to the Web," Technical Report Paper SIDL-WP-1999-0120 (Version of 11/11/1999), 1999.

[44] M. Pasca and B.V. Durme, "What You Seek Is What You Get: Extraction of Class Attributes from Query Logs," *IJCAI '07: Proc. 20th Int'l Joint Conf. Artifical Intelligence,* pp. 2832-2837, 2007.

[45] G. Pass, A. Chowdhury, and C. Torgeson, "A Picture of Search," *Proc. First Int'l Conf. Scalable Information Systems,* June 2006.

[46] J.D.M. Rennie and N. Srebro, "Fast Maximum Margin Matrix Factorization for Collaborative Prediction," *ICML '05: Proc. 22nd Int'l Conf. Machine Learning,* 2005.

[47] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An Open Architecture for Collaborative Filtering of Netnews," *CSCW '94: Proc. ACM Conf. Computer Supported Cooperative Work,* 1994.
[48] R. Salakhutdinov and A. Mnih, "Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo," *ICML '05: Proc. 25th Int'l Conf. Machine Learning,* 2008.
[49] R. Salakhutdinov and A. Mnih, "Probabilistic Matrix Factorization," *Advances in Neural Information Processing Systems,* vol. 20, pp. 1257-1264, 2008.
[50] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-Based Collaborative Filtering Recommendation Algorithms," *WWW '01: Proc. 10th Int'l Conf. World Wide Web,* pp. 285-295, 2001.
[51] D. Shen, M. Qin, W. Chen, Q. Yang, and Z. Chen, "Mining Web Query Hierarchies from Clickthrough Data," *AAAI '07: Proc. 22nd Nat'l Conf. Artificial Intelligence,* pp. 341-346, 2007.
[52] L. Si and R. Jin, "Flexible Mixture Model for Collaborative Filtering," *ICML '03: Proc. 20th Int'l Conf. Machine Learning,* 2003.
[53] C. Silverstein, M.R. Henzinger, H. Marais, and M. Moricz, "Analysis of a Very Large Web Search Engine Query Log," *ACM SIGIR Forum,* vol. 33, no. 1, pp. 6-12, 1999.
[54] N. Srebro and T. Jaakkola, "Weighted Low-Rank Approximations," *ICML '03: Proc. 20th Int'l Conf. Machine Learning,* pp. 720-727, 2003.
[55] J.-T. Sun, D. Shen, H.-J. Zeng, Q. Yang, Y. Lu, and Z. Chen, "Web-Page Summarization Using Clickthrough Data," *SIGIR '05: Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 194-201, 2005.
[56] M. Theobald, R. Schenkel, and G. Weikum, "Efficient and Self-Tuning Incremental Query Expansion for Top-k Query Processing," *SIGIR '05: Proc. 28th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 242-249, 2005.
[57] B. Vélez, R. Weiss, M.A. Sheldon, and D.K. Gifford, "Fast and Effective Query Refinement," *ACM SIGIR Forum,* vol. 31(SI) pp. 6-15, 1997.
[58] L. von Ahn and L. Dabbish, "Labeling Images with a Computer Game," *CHI '04: Proc. SIGCHI Conf. Human Factors in Computing Systems,* pp. 319-326, 2004.
[59] X. Wang and C. Zhai, "Learn from Web Search Logs to Organize Search Results," *SIGIR '07: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 87-94, 2007.
[60] J.-R. Wen, J.-Y. Nie, and H. Zhang, "Query Clustering using User Logs," *ACM Trans. Information Systems,* vol. 20, no. 1, pp. 59-81, 2002.
[61] J. Xu and W.B. Croft, "Query Expansion using Local and Global Document Analysis," *SIGIR '07: Proc. 19th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 4-11, 1996.
[62] H. Yang, I. King, and M.R. Lyu, "DiffusionRank: A Possible Penicillin for Web Spamming," *SIGIR '07: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 431-438, 2007.
[63] Y.-H. Yang, P.-T. Wu, C.-W. Lee, K.-H. Lin, W.H. Hsu, and H. Chen, "ContextSeer: Context Search and Recommendation at Query Time for Shared Consumer Photos," *Proc. 16th ACM Int'l Conf. Multimedia,* pp. 199-208, 2008.
[64] B. Zhang, H. Li, Y. Liu, L. Ji, W. Xi, W. Fan, Z. Chen, and W.-Y. Ma, "Improving Web Search Results Using Affinity Graph," *SIGIR '05: Proc. 30th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 504-511, 2005.



Hao Ma received the BEng and MEng degrees in the School of Information Science and Engineering from Central South University in 2002 and 2005, respectively, and the PhD degree from the Computer Science and Engineering Department, The Chinese University of Hong Kong (CUHK) in 2010. He worked as a system engineer at Intel Shanghai before he joined CUHK as a PhD student in November 2006. His research interests include information retrieval, data mining, machine learning, social network analysis, recommender systems, human computation, and social media analysis.

Irwin King received the BSc degree in engineering and applied science from the California Institute of Technology, Pasadena and the MSc and PhD degrees in computer science from the University of Southern California, Los Angeles. Currently, he is with the Chinese University of Hong Kong. His research interests include machine learning, web intelligence, social computing, data mining, and multimedia information processing. In these research areas, he has more 200 technical publications in journals and conferences. In addition, he has contributed more than 20 book chapters and edited volumes. Moreover, he has more than 30 research and applied grants. One notable patented system he has developed is the VeriGuide System, which detects similar sentences and performs readability analysis of text-based documents in both English and in Chinese to promote academic integrity and honesty. He is an associate editor of the *IEEE Transactions on Neural Networks (TNN)* and *IEEE Computational Intelligence Magazine (CIM)*. He is a member of the editorial boards of the Open Information Systems Journal, *Journal of Nonlinear Analysis and Applied Mathematics*, and *Neural Information ProcessingCLetters and Reviews Journal (NIP-LR)*. He has also served as special issue guest editor for *Neurocomputing*, *International Journal of Intelligent Computing and Cybernetics (IJICC)*, *Journal of Intelligent Information Systems (JIIS)*, and *International Journal of Computational Intelligent Research (IJCIR)*. Currently, he is serving the Neural Network Technical Committee (NNTC) and the Data Mining Technical Committee under the IEEE Computational Intelligence Society (formerly the IEEE Neural Network Society). He is a senior member of the IEEE and a member of the ACM, International Neural Network Society (INNS), and Asian Pacific Neural Network Assembly (APNNA). He is also a member of the Board of Governors of INNS and a vice-president and governing board member of APNNA.

Michael Rung-Tsong Lyu received the BS degree in electrical engineering from National Taiwan University, Taipei, China, in 1981, the MS degree in computer engineering from the University of California, Santa Barbara, in 1985, and the PhD degree in computer science from the University of California, Los Angeles, in 1988. He was with the Jet Propulsion Laboratory as a technical staff member from 1988 to 1990. From 1990 to 1992, he was with the Department of Electrical and Computer Engineering, The University of Iowa, as an assistant professor. From 1992 to 1995, he was a member of the technical staff in the applied research area of Bell Communications Research (Bellcore), Morristown, New Jersey. From 1995 to 1997, he was a research member of the technical staff at Bell Laboratories, Murray Hill, New Jersey. In 1998, he joined The Chinese University of Hong Kong, where he is now a professor in the Department of Computer Science and Engineering. He is also a founder and director of the Video over InternEt and Wireless (VIEW) Technologies Laboratory. He has published 330 refereed journal and conference papers in these areas. He was the editor of two book volumes: *Software Fault Tolerance* (New York: Wiley, 1995) and *The Handbook of Software Reliability Engineering* (Piscataway, N.J.: IEEE and New York: McGraw-Hill, 1996). He initiated the First International Symposium on Software Reliability Engineering (ISSRE) in 1990. He was the program chair for ISSRE 1996 and general chair for ISSRE 2001. He was also PRDC 1999 program cochair, WWW10 program cochair, SRDS 2005 program cochair, PRDC 2005 general cochair, ICEBE 2007 program cochair, and SCC 2010 program cochair. He will be the general chair for DSN 2011 in Hong Kong. He was on the editorial boards of the *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Reliability*, *Journal of Information Science and Engineering,* and *Wiley Software Testing, Verification and Reliability Journal*. His research interests include software reliability engineering, distributed systems, fault-tolerant computing, mobile and sensor networks, Web technologies, multimedia information processing and retrieval, and machine learning. He is an IEEE fellow, an AAAS fellow, and a croucher senior research fellow. He was also named by the IEEE Reliability Society as the 2010 Engineer of the Year.