When Recommendation Meets Mobile: Contextual and Personalized Recommendation On The Go[*]

Jinfeng Zhuang [†], Tao Mei [‡], Steven C. H. Hoi [†], Ying-Qing Xu [‡], Shipeng Li [‡]
[†] Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798
[‡] Microsoft Research Asia, Beijing 100080, P. R. China
{zhua0016, chhoi}@ntu.edu.sg, {tmei, yqxu, spli}@microsoft.com

ABSTRACT

Mobile devices are becoming ubiquitous. People use their phones as a personal concierge discovering and making decisions anywhere and anytime. Understanding user intent on the go therefore becomes important for task completion on the phone. While existing efforts have predominantly focused on understanding the explicit user intent expressed by a textual or voice query, this paper presents an approach to context-aware and personalized entity recommendation which understands the *implicit* intent without any explicit user input on the phone. The approach, highly motivated from a large-scale mobile click-through analysis, is able to rank both the entity types and the entities within each type (here an entity is a local business, e.g., "I love sushi," while an entity type is a category, e.g., "restaurant"). The recommended entity types and entities are relevant to both user context (past behaviors) and sensor context (time and geo-location). Specifically, it estimates the generation probability of an entity by a given user conditioned on the current context in a probabilistic framework. A random-walk propagation is then employed to refine the estimated probability by mining the temporal patterns among entities. We deploy a recommendation application based on the proposed approach on Window Phone 7 devices. We evaluate recommendation performance on 10 thousand mobile clicks, as well as user experience through subjective user studies. We show that the application is effective to facilitate the exploration and discovery of surroundings for mobile users.

ACM Classification Keywords

H.3.5 Online Information Services: Web-based services

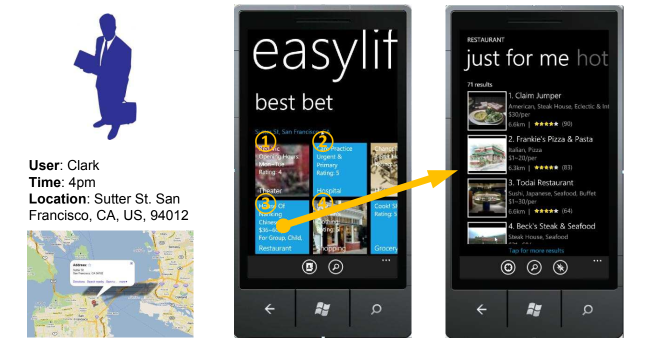
General Terms

Algorithms, Experimentation, Performance

Author Keywords

Entity recommendation, local search, mobile application

[*] This work was performed at Microsoft Research Asia.



(a) user and sensory context (b) rank of entity types (c) rank of restaurant entities

Figure 1. The interface of mobile recommendation application *easylife* developed based on the proposed approach. A user named "clark" gets the recommendation of entity types in (b) when he was in the context of (a), and then can check the ranked entities in (c) by selecting a specific entity type (e.g., nearby restaurants that favor his taste in the entity type of restaurant). The entity type and the entities within each type are ranked according to the relevance to user and sensor context. The proposed contextual and personalized recommendation facilitates the exploration of and decision making on the go.

INTRODUCTION

Mobile devices are becoming ubiquitous and playing vital roles in our daily life. While on the go, people are using their phones as a personal concierge discovering what is around and deciding what to do. Therefore, mobile phone has become a recommendation terminal customized for the individuals (i.e., capable to recommend contextually relevant and personalized entities and simplify the accomplishment of tasks). As a result, it is important to understand user intent through the rich context (both user and sensory context) available on the phone.

Existing research has predominantly focused on recommendation by understanding the intent expressed by text (or the text recognized from voice). For example, previous research tries to estimate user's search intent by detecting meaningful entities from a textual query [7, 13, 15, 25]. However, typing is always a tedious job on the phone and thus intrusive to be used to express intent. An alternative is to leverage speech recognition techniques to support voice as a way of expressing intent. For example, the popular mobile search engines enable a voice-to-search mode [1] [2]. Siri is one of the most popular applications that further structurizes a piece of speech to a set of entities [24]. However, the voice as an expression of user intent has some limitations: 1) it relies on a good recognition engine and usually works well only in a relatively quiet environment, and 2) understanding a long textual query still remains a challenge. Although there exists extensive research on general recommendation [3], most of them do not consider the rich context (e.g, time and location) on the mobile devices. From this perspective, the recommendation is not context-aware and personalized.

Based on our analysis of a large-scale real-world mobile search click-through data, we found that the intent on the mobile is typically context-aware, personalized, and local (i.e., driven by exploration of local business). Therefore, we present in this paper a probabilistic approach to support mobile recommendation without requiring any user input. The approach leverages the rich context signals on the mobile device (i.e., user and sensory context, such as user click-through, geo-location, and time) to ranks the entities (i.e., local business) tailored to user's interest anywhere and anytime. The approach consists of three key components on the cloud: 1) entity crawler which collects entities with attributes (e.g., cuisine for restaurant) from the Web, 2) entity extraction which detects and recognizes entities from a query or click-through, and 3) entity ranking which ranks the entities in a context-sensitive and personalized way without requiring any input. Specifically, we propose a probabilistic entity ranking algorithm in spirit of hybrid recommendation algorithm [3], which models the generating probability of an entity by the user conditioned on the mobile context, followed by a random-walk refining process. To summarize, this paper makes the following contributions:

- We conduct an analysis on a real-world large-scale click-through data collected from a commercial mobile search engine, which motivates the proposed approach. The analysis provides a good knowledge for motivating related research on mobile platforms.
- We propose a probabilistic entity recommendation approach to understand user's implicit intent on the phone. The approach is able to rank both entity types (e.g., categories such as restaurant, hotel, etc.) and entities (i.e., specific local businesses within each type) which are relevant to user and sensory context. This approach actually builds a personal model for each user on the cloud.
- We develop a real application based on this approach on a Windows Phone 7 device and conduct both objective and subjective evaluations to validate its effectiveness.

Figure 1 shows the user interface of the application (called *easylife*) developed based the proposed recommendation.

RELATED WORK

The related research to mobile recommendation includes query suggestion, vertical/local search, and recommendation.

Query Suggestion

The most basic motivation of our approach is to reduce the effort for typing textual queries with a mobile keyboard. Query suggestion and auto-completion are devised to achieve this goal [5, 11, 12, 14, 20]. It completes the query automatically such that a user does not need to type the whole query. However, the user still has to view the returned list of links and find his desired information.

Our approach shares the same goal of reducing user's input effort. However, the scenario and target function are quite different. First, we do not require any user input at all but return a list of sorted entities directly. The mobile user can click these entities to view details in a push model. Therefore, it will reduce user's effort for typing queries and looking up the returned links. We can implement this because the search purpose on mobile platform is often entity-oriented, while the limitation is that we cannot handle general query purpose that does not relate to entities directly. The research in this paper also relates to *intent prediction* [4, 18]. However, it is well known that user intent on mobile devices is quite different from that on desktop PC. The rich context is regarded as a key to solve user intent on the phone.

Object-Level Vertical/Local Search

Instead of general search, our approach extracts predefined categories of entities and recommends them to mobile users. This function connects to object-level vertical search [16, 21] and local search [15]. A vertical search engine focuses on a specific segment of online content which distinguishes from a general search engine. Typical examples include travel, academy, and product search engines. Lane *et. al.* propose to use context signals to improve local search performance [15]. Our approach is also vertical in the sense that it provides entities and their attributes only involving the local business facilities. It does not intend to improve general search function. Moreover, it works at the object level as it extracts entities automatically without requiring users to browse the original web pages related to the entities. The entity extraction plays a fundamental role for the system. However, it is different from vertical search engines because it is not designed to handle explicit user input. Instead, it returns the ranked entities directly according to the user's implicit intent. In this way, the user's input effort is minimized.

Recommendation System

Our goal is to generate a ranked list of entities to satisfy user intent on the go, which is essentially related to a recommendation system [3, 10]. The approach is inspired by an investigation of the recommendation methodologies: content-based, collaborative, and hybrid [3]. The proposed entity/type ranking approach is in spirit to the hybrid recommendation method. That is, we make use of not only user's own query history, but also the history data from other users.

Compared with traditional recommendation systems [3], the proposed approach is also personalized and more context sensitive (sensitive to location and time of the day). Researchers have proposed various recommendation techniques to handle such systems [6, 17, 19]. However, due to the unique characteristics of our entity ranking task, the conventional hybrid recommendation algorithm summarized in [3] cannot be directly applied here. First, the attributes of an entity do not make much sense when determining its position in the ranked list. The query history of a user is not long enough to detect a meaningful pattern to reflect the user's

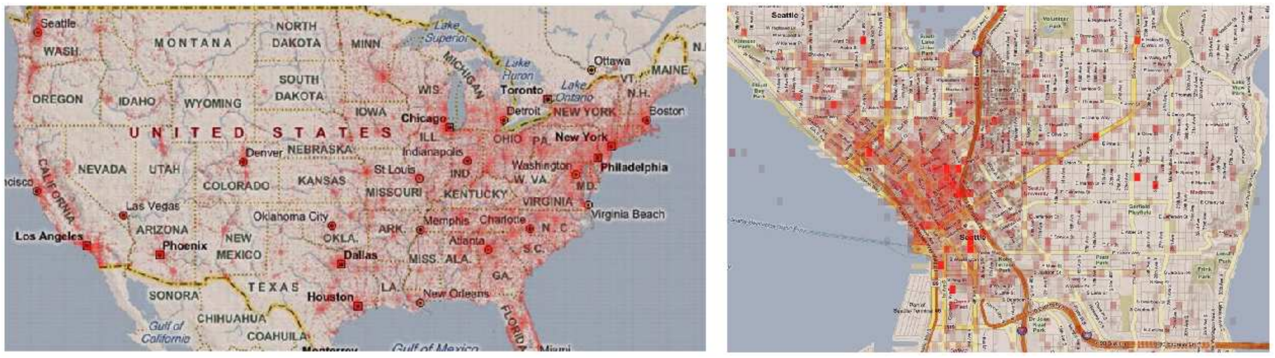


Figure 2. The query distribution with different locations. The depth of the red color is proportional to the number of queries conducted at locations. Left: query map of United States. We observe that big cities are "hot" (red color) areas of mobile search. Right: query map of Seattle city. One can see that mobile search is more popular in downtown areas.

#raw query	#user	#entity query	#entity user
75,221,037	13,711,497	11,492,382	4,012,030

Table 1. The statistics of the queries. #raw query **is the number of collected mobile queries,** #user **is the number users who conducted these raw queries,** #entity query **is the number of queries related to entities, and** #entity user **is the number of users involved in entity queries.**

preference. Moreover, as the recommendation is performed in real time and the database is large-scale (could be more than 10 million query records), complex machine learning algorithms with heavy computational costs may not work here (e.g., [26]). As a result, we consider a simple yet efficient probabilistic approach for modeling the conditional probability of generating some entities for a user. We further refine the ranking by a random walk procedure.

ANALYSIS OF MOBILE CLICK-THROUGH DATA

We collected a large-scale query log data from a commercial mobile search engine. We conduct an analysis to investigate the characteristics of mobile search query, which motivates our recommendation approach and *easylife* application.

The time range of our query database is from 2009-09-30 to 2010-03-28. All these queries were conducted in United States. During the six months, the number of *raw queries* is up to 75,221,037, which were issued by a total of 13,711,497 users, corresponding to 417,895 queries per day. Table 1 lists the query statistics. Each query log consists of "user id" (anonymous without any user identity information), "time," "GPS location," "query," and "URL."

Search via Mobile Phone is Popular

Figure 2 shows the distribution of mobile queries in US. The depth of the red color on the map is proportional to the number of queries conducted in this area. We can intuitively find the hot areas where mobile search is popular. For example, queries in Seattle and New York are significantly more than those in Nontana and North Dakota. Figure 3(a) shows the detailed #query in eight cities. We can conclude that *mobile search is becoming pervasive, especially in big cities.*

Mobile Search Query Is Short

Figure 3(b) shows the number of queries (#word) with different lengths. We observe that #query decreases sharply with #word. About 62.3% of the queries contain less than three words (three excluded), and 76.8% queries contain less than four words. On average, each query contains 2.52 words. Figure 3(c) shows the distribution of queries with different length measured by #letter. Each query contains 18.76 letters on average. These results confirm the assumption that *search queries on mobile platform are usually short*. Users are not willing to type long sentences on the keyboard of very limited size. Therefore, it would reduce user's effort significantly if we can design systems to avoid time consuming user-phone interaction.

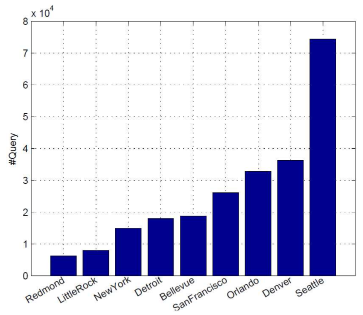
Mobile Search Is Local and Context-Sensitive

The left map in Figure 2 shows the query distribution in US. We can see that #query is diverse in different cities. The right map in Figure 2 shows the query distribution in the city of *Seattle*. We can observe that the mobile search activity is very sensitive to location. For commercial areas, since there are a lot of local businesses like shopping centers, restaurants, and so on, people in these locations are more likely to conduct entity-related search. Figure 3(d) shows #query in different time slot (of the day). The highest peak occurs near to 5–6 pm, when the search on mobiles tends to be active. One possible reason is that people are about to get off work. The lowest point occurs at about 2–3 am. This is reasonable as few people are active in the midnight. The above observation implies that *mobile search is usually context-aware* [15]. Using entity extraction technique, we can detect the *entity query*, i.e., the queries target at searching entities. The ratio of #entity query to #raw query is 15.28% (the ratio depends on the recall of an entity extraction algorithm). This verifies that search on mobile is *local and entity-oriented.*

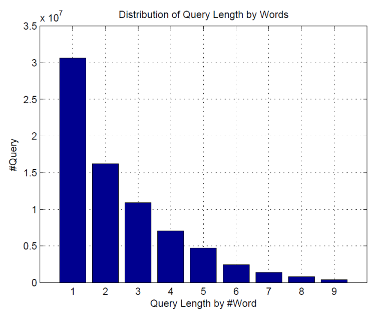
The above characteristics of mobile query motivates the design of a recommendation system that is *context-aware, personalized,* and *without requiring any typing of queries*. Due to the abundant attributes of extracted entities (as shown and Figure 1), a user can get a quick glance of his information need without typing a query explicitly.

APPROACH

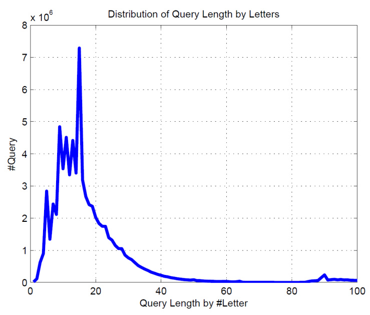
In this section, we introduce the proposed recommendation approach, which consists of two key components: 1) entity extraction which detects and recognizes entities from a tex-



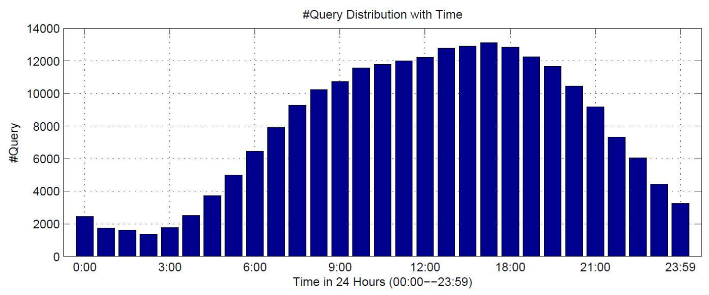
(a) distribution of queries in eight cities.



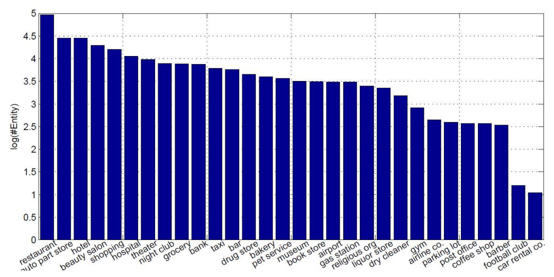
(b) distribution of query length.



(c) distribution of number of query terms.



(d) distribution of query time (time of the day).



(e) distribution of entities within 30 entity types.

Figure 3. The statistics of user click distribution in our database.

Table 2. The common attributes of extracted entities from three examples. Each entity type has its unique attributes. The right picture in Figure 1 shows an example of a restaurant *spotlight*.

Name	Country	Region	Locality	Address	EntityType	Latitude	Longtitude	Phone
Starbucks	U.S.	CA	Irivne	115 Fortune Dr	Coffee Shop	33.6504	-117.7461	(617)277-0087
FrenchDress	U.S.	MA	Boston	49 River St	shopping	42.3572	-71.0703	(617)723-4968
Walmart	U.S.	UT	HeberCity	435 Airport Rd	shopping	40.4827	-111.4207	(435)654-6436

tual query log, and 2) entity ranking which ranks a candidate set of entities and the corresponding entity types to the user. We pay more attention to the entity ranking algorithm.

Entity Extraction

We build a database of extracted entities from several major local business sites on the Web. The extraction algorithms are similar to the previous entity extraction system *Know-ItAll* [8], where the key techniques are rule-based. That is, the first step of entity extractor is to automatically create a collection of extraction rules for each kind of entity types. Beyond of the extraction rule of entities, we make a further step to extract the attributes of entities as well. Please refer to [8] for detailed components of rules. This step aims to get a high recall of entities and the corresponding attributes. After obtaining the initial extracted ones, we can pass them to a search engine to retrieve more entities. Then a pattern learner is employed to filter out high-quality entities for expansion. In summary, we extract 5,457,192 entities under 37 entity types.

Table 2 lists three entities with some extracted common features. Besides the common features, each entity type has a unique set of attributes. For example, a restaurant has the price level and cuisine type as the major features. Due to the limited space, we do not introduce the detailed features of each type here. After obtaining the results by entity extractor, we can design the query parser which maps a query to a specific entity. This can be implemented by supervised classification techniques or topic models [9].

Probabilistic Entity Ranking

In this section, we present the algorithm for the probabilistic entity ranking. Table 3 lists the key notations used in this paper. Figure 4 shows the framework of building a personal user model based on the user click-through data. The key component is an entity ranker that can estimate the conditional probabilities of entities and entity types for a given user under certain context. Given a user $u \in \mathcal{U}$ and his context information (location and time) $\langle l, t \rangle$, the basic idea is to first find the queries issued in this context, and then weight these queries according to the user similarity graph $\mathbf{S}$. To consider the co-occurrence of entities in a query session (e.g., a bar is usually queried after restaurant in a short period of time), the generating probability of entity will be smoothed and propagated on an entity similarity graph $\mathbf{W}$. The two similarity matrices $\mathbf{S}$ and $\mathbf{W}$ can be computed offline, while the other steps can be processed on the fly.

Problem Formulation

By using entity extractor and query parser, we can obtain a query database $\mathcal{Q} = \{q_1, \ldots, q_m\}$, where m is the number of mobile queries containing entities in our database. Each *mobile query* is a 5-dimensional tuple:

$$Q := \langle E, Z, U, L, T \rangle,$$

Table 3. List of key notations.

Symbol	Description
$\mathbb{N}_u$	$\{1,\ldots,u\}$, a set of integers up to u
N_u	$\lvert\mathbb{N}_u\rvert$, the cardinality of $\mathbb{N}_u$
$\mathcal{U}$	$\{\mathbf{u}_i : i \in \mathbb{N}_u\}$, a collection of N_u users
$\mathcal{Q}$	the collection of all the queries
U, E, Z, T, L	represents user, entity, entity type, time, location, respectively
$a.B$	the attribute B of object a

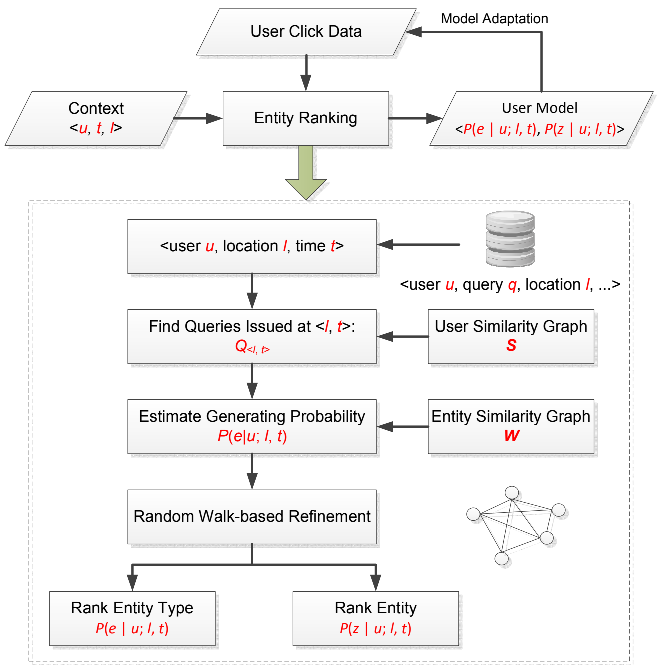


Figure 4. The pipeline of the probabilistic entity ranking algorithm. The components in the rectangle shows the details for "entity ranking."

where $E \in \mathcal{E}$ is the entity searched in Q, Z indicates the entity category of E, $U \in \mathcal{U}$ is the user who conducted the query , L and T are the context, i.e., the location and time when Q was generated, respectively. We use $\mathcal{E}$ to denote the whole collection of extracted entities.

Each *mobile user* U consists of a history of queries:

$$U := \mathcal{Q}_u = \{q_1, \ldots, q_u\}.$$

Given a user $u \in \mathcal{U}$ in the context $\langle l, t\rangle$, the task *entity ranking* is to rank the entities in $\mathcal{E}$ such that the higher ranked entity has a larger probability of being queried by u.

Probabilistic Ranking Algorithm

We first consider the probability of generating the entity e_j by user u_i with the context $\langle l, t\rangle$. The entities in $\mathcal{E}$ are ranked by the conditional probability $P(E|U; L, T)$, i.e., we consider E is generated by the user U conditioning on the context $\langle l, t\rangle$. We assume the final ranking score of a specific entity is proportional to this conditional probability. To this end, we introduce a variable Z to indicate the user's intent. For example, if Z indicates "eat" for the time being, probably the user intends to go to a restaurant. With this intuition, the possible choice of Z is determined by the number of entity categories N_z. We deem $P(E|Z)$ as the popularity of some E under the category Z, and $P(Z|U)$ proportional to the frequency of querying Z by U. We have the conditional probability

$$P(e_j|u_i; l, t) = \sum_{k=1}^{N_z} P(e_j|z_k; l, t) P(z_k|u_i; l, t).$$

Due to the limited query history of U, it is common that some entity types have never been queried by U. Assigning a zero conditional probability of these entity types are apparently not proper. To overcome this problem, we leverage the idea of collaborative recommendation techniques [3]. That is, we introduce the query record of other users to help estimate the probability. If other users similar to u_i have queried an entity before, u_i also has the possibility of searching for the entity. Thus we have

$$\begin{aligned} P(e_j|u_i; l,t) &= \sum_{k=1}^{N_z} P(e_j|z_k, u_n; l,t) \sum_{n=1}^{N_u} P(z_k|u_n; l,t) P(u_n|u_i) \\ &\propto \sum_{k=1}^{N_z} P(e_j|z_k; l,t) \sum_{n=1}^{N_u} P(z_k|u_n; l,t) s(u_n, u_i), \quad (1) \end{aligned}$$

where $s(\cdot,\cdot) : \mathcal{U} \times \mathcal{U} \rightarrow \mathbb{R}_+$ is a function measuring the similarity between two users. The intuition here is the transition probability $P(u_n|u_i)$ is proportional to the similarity between them.

We now instantiate the components in (1). We first query $\mathcal{Q}$ with the context $\langle l, t\rangle$. Denote the returned query set as $\mathcal{Q}_{\langle l,t\rangle}$. Then for $\mathcal{Q}_e, \mathcal{Q}_z \subset \mathcal{Q}_{\langle l,t\rangle}$ we have

$$P(e_j|z_k; l,t) := \frac{|\mathcal{Q}_e : \forall q \in \mathcal{Q}_e, q.E = e_j, q.Z = z_k, q.L = l, q.T = t|}{|\mathcal{Q}_z : \forall q \in \mathcal{Q}_z, q.Z = z_k, q.L = l, q.T = t|}$$

where $|\mathcal{Q}|$ is the number of queries in the query set $\mathcal{Q}$ belonging to category z_t. Similarly, we have

$$P(z_k|u_n; l,t) := \frac{|\mathcal{Q}_z : \forall q \in \mathcal{Q}_z, q.Z = z_k, q.U = u_n, q.L = l, q.T = t|}{|\mathcal{Q}_u : \forall q \in \mathcal{Q}_u, q.U = u_n, q.L = l, q.T = t|}$$

It is intuitive to model the probability of each search intent using their frequency.

User Similarity

Now we focus on the similarity $s(\cdot,\cdot)$ defined on the user space, in which each user can be represented by a query history record. Depending on how to represent users with the query records, a three-level similarity function can be adopted here: *entity-based*, *entity-type-based*, and *entity-attribute-based* similarity.

We first convert the query history into a fixed-length vector akin to the term frequency-inverse document frequency (*tf-idf*) scheme in text processing [23]. Here each "document" implies a single user U, while each term refers to either a specific entity or the entity type. In the former case, the representation is strict in the sense that two users are similar only if they used to query the same entity. The term frequency of E with U is the frequency E has been searched by U. Specifically,

$$tf(e, u) = |\mathcal{Q}_u : \forall q \in \mathcal{Q}_u, q.E = e \text{ and } q.U = u|.$$

The inverse document frequency is

$$idf(e) = |\mathcal{U} : \forall u \in \mathcal{U}, \exists q \in \mathcal{Q}_u, q.E = e|.$$

Thus, we construct the feature vector of u_i by $\mathbf{x}_i \in \mathbb{R}^d$ using tf-idf weighting, i.e., the k-th component is computed by

$$[\mathbf{x}_i]_k = tf(e_k, u_i) \log \frac{1}{idf(e_k)}.$$

Then, we have

$$s_1(u_i, u_j) := \frac{\mathbf{x}_i^\top \mathbf{x}_j}{\sqrt{(\mathbf{x}_i^\top \mathbf{x}_i)(\mathbf{x}_j^\top \mathbf{x}_j)}}.$$

The measure s_1 may not be always optimal in real case. For example, considering two users, one of them is interested in *McDonalds*, the other always prefers *KFC*. When they come to a district where only *BurgerKing* is available, they may both like it since they both need *fast food*. Therefore, entity-type based tf-idf weighting can be complementary to entity based representation because of the coarser granularity. We just need to use the constraint $q.Z = z$ (now Z is the term instead of E) instead of $q.E = e$ when computing the term frequency of E in $\mathcal{Q}_u$. The inverse document frequency becomes the number of users who used to search that entity type. We denote the cosine similarity with entity-type based representation by s_2. We evaluated the effectiveness of both s_1 and s_2 in the experiment.

Remark. In the above methods, the attributes of entities are ignored. One might expect that such attributes can indicate the preference of users. For example, if two users both like Chinese cuisine, they are similar to some extent even if they never search the same Chinese restaurant before. However, each user could have searched different kinds of entities, it is difficult to measure entities of different types directly. Due to the defection of entity crawling process, the entity attributes are often missing. Moreover, the online entity ranking must be effective. Similarity between sets of entities cannot meet this requirement.

Ranking Refinement by Random Walk

The similarity computation in last section ignores the sequential information between queries. Considering the meeting of some friends, they probably query the restaurants before dinner. After having dinner, they may be interested in the bars nearby for night life. In this situation, temporal patterns (restaurant→bar here) exist and make sense in characterizing the relation among entities. Note that sequential co-occurrence depends on the length of time interval. Here we name the time interval by *session*. The longer the session, the more sequential co-occurrence would be detected, and vice versa. We use the number of users that exhibit temporal patterns to measure the transition probability between two entities, i.e.,

$$\begin{aligned} w(e_i, e_j) \;=\; & |\mathcal{U} : \forall u \in \mathcal{U}, \exists q, q' \in \mathcal{Q}_u, q.E = e_i \text{ and} \\ & q'.E = e_j \text{ and } q'.T - q.T < \epsilon|, \end{aligned}$$

where ϵ is the upper bound of the length of the session. The constraint of $\mathcal{U}$ here is there exists at least one session in each user $u \in \mathcal{U}$ such that e_i and e_j are queried sequentially. Thus we can construct the matrix $\mathbf{W} \in \mathbb{R}_+^{N_e \times N_e}$ with $[\mathbf{W}]_{ij} := w(e_i, e_j)$ to measure the possibility of transferring from one entity to another. We can further to normalize $\mathbf{W}$ such that $\forall i \in \mathbb{N}_e \sum_{j \in \mathbb{N}_e} W_{ij} = 1$.

Suppose the ranking score obtained from above step is $\mathbf{p}_0$. Then we refine it by random walk with transition probability matrix $\mathbf{W}$ iteratively:

$$\mathbf{p} = \alpha \mathbf{W}^\top \mathbf{p} + (1 - \alpha)\mathbf{p}_0.$$

One can prove the iteration converges to a fixed point $\mathbf{p}^* = (1 - \alpha)(\mathbf{I} - \alpha \mathbf{W}^\top)^{-1}\mathbf{p}_0$. We use the score $\mathbf{p}^*$ to rank the entities.

Ranking Entity Type

As shown in Figure 1, the returned entities are organized by entity types. Besides ranking entities, we also need to rank the types as the left diagram in Figure 1 to facilitate user's browsing purpose. To this end, we consider the scheme akin to the probabilistic framework of entity ranking:

$$P(z|u; l, t) \propto \sum_{n=1}^{N_u} P(z|u_n; l, t) s(u_n, u).$$

We organize the entities by ranked types with ranking score $P(Z|U; L, T)$. The entities under each category are sorted by $P(E|U; L, T)$.

Remark. We can collect the user click-through data to help rectify the ranking score. The more frequently clicked entities have larger probability to be clicked again. The relevance feedback techniques can be employed to automatically adjust the distributions $P(E|U; L, T)$ and $P(Z|U; L, T)$ [22].

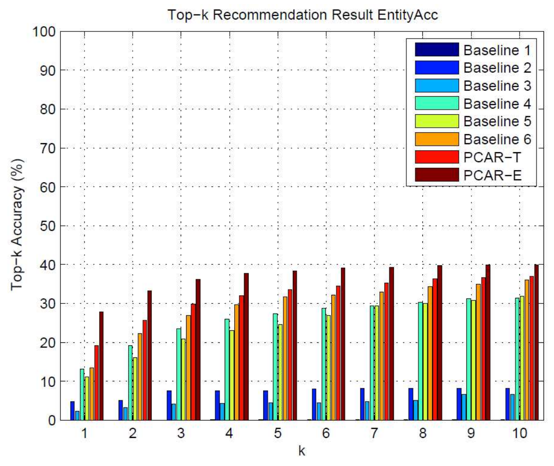
EXPERIMENTS

Data and Setting

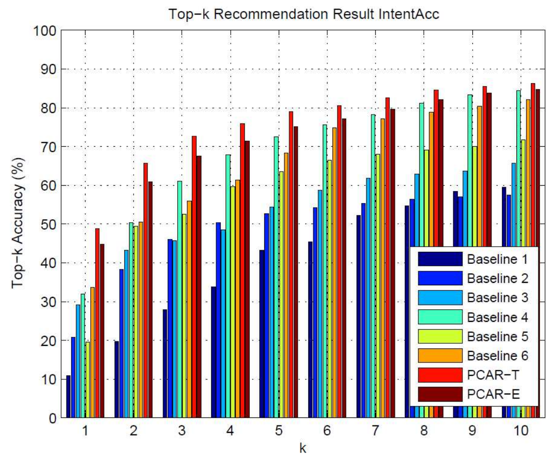
We build the user similarity graph and entity similarity graph based on the mobile click-through data from the first five months, while adopting the data in the last one month for test. We randomly selected 2,000 users and use their queries in the March of 2010 as test data, which contains a total of 58,111 query records. We denote this test set by $\mathcal{Q}_t$.

For each test query record $q \in \mathcal{Q}_t$, we use its location l and time t as the search context and call the system to recommend the entities to its user u. We split the time into 7 intervals: 0:00∼5:00, 6:00∼7:00, 8:00∼11:00, 12:00, 13:00∼17:00, 18:00∼19:00, 20:00∼23:00. We extract a set of queries from the query database with the context $\langle l, t\rangle$. Then we sort the entities contained in these queries by entity ranker. To this end, we extract the queries conducted within five kilometers to u and fallen into the same time interval as t.

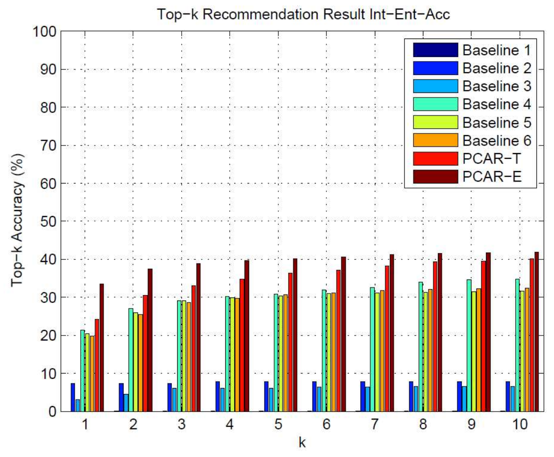
Let $\mathcal{E}_t$ denote the set of the ranked entities, we can measure the top-k accuracy of $\mathcal{E}_t$ by counting the position $\pi(q.E)$ of $q.E$ in $\mathcal{E}_t$. For each $q \in \mathcal{Q}_t$, if $\pi(q.E)$ is less than k, we



(a) Accuracy of entity ranking



(b) Accuracy of intent ranking



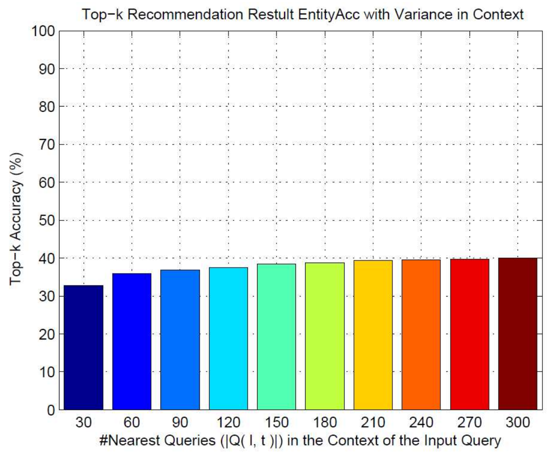
(c) Accuracy of intent-entity ranking with $t = 3$.

Figure 5. The accuracy of the proposed entity ranking.

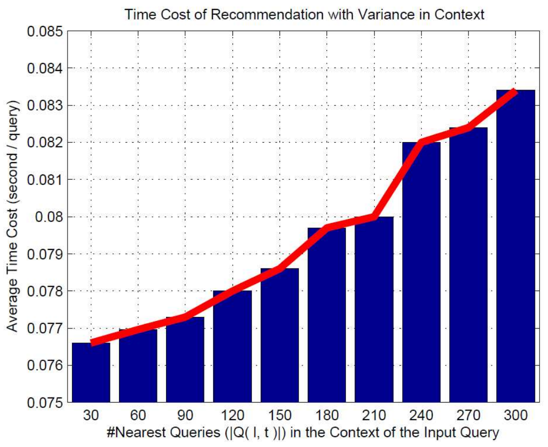
have correctly recommended entity E. Formally the top-k accuracy of the whole test set $\mathcal{Q}_t$ is computed by

$$Accuracy(\mathcal{Q}_t, k) = \frac{\sum_{q \in \mathcal{Q}_t} \mathbb{I}(\pi(q.E) \leq k)}{|\mathcal{Q}_t|},$$

where $\mathbb{I}(c)$ returns 1 if c is true and 0 otherwise.



(a) Top-10 EntityAcc with different $|\mathcal{Q}_{\langle l,t \rangle}|$



(b) Average time cost of each query with different $|\mathcal{Q}_{\langle l,t \rangle}|$.

Figure 6. Sensitivity of entity ranking to the context.

We measure accuracy of three kinds of recommendation:

- **EntityAcc**: the top-k accuracy of the recommended entities;
- **IntentAcc**: the top-k accuracy of the recommended entity type. It reflects the intent of the users at the query location;
- **IntEntAcc**: the accuracy of the top-k entities in each of the top-t entity types, i.e., $t \times k$ entities are contained in $\mathcal{E}_t$.

The recommendation results have a two-level organization (refer to Figure 1). According to the definition, EntityAcc should always be less than IntentAcc. The abbreviation of the examined schemes are summarized as follows [1]:

- **Baseline 1** $\langle l \rangle$: recommend entities according to their distance to the user's current location, and rank them by distance to the user;

[1] The result of traditional content-based recommendation (CBR) is not presented here. In most cases, a user has not searched an entity before a new context. Thus the accuracy of CBR approaches to 0.

Table 4. The recommended entity type (RcmndtnInt in the table) of 4 users.

Name	Jennifer	Peter	Peter	Pok	Miranda	Miranda
Latitude	40.7441	40.7441	40.7441	37.7896	37.7896	37.7896
Longitude	-74.0135	-74.0135	-74.0135	-122.4061	-122.4061	-122.4061
Time	18:26	18:26	15:26	22:15	22:15	13:36
Location	NewYork	NewYork	NewYork	SanFrancisco	SanFrancisco	SanFrancisco
RcmndtnInt	Bank(2.32)	Bank(2.33)	PetService(0.66)	Restaurant(4.01)	Bank(1.03)	Restaurant(0.51)
	Shopping(0.17)	Restaurant(0.23)	Restaurant(0.54)	Hotel(1.01)	Restaurant(0.69)	Bank(0.18)
	Clubs(0.17)	Shopping(0.22)	Hotel(0.19)	Theater(0.80)	Theater(0.53)	Shopping(0.13)

- **Baseline 2** $\langle l, p\rangle$: recommend entities as **Baseline 1**, but rank them by the popularity p [2];
- **Baseline 3** $\langle l, p, t\rangle$: recommend entities as **Baseline 1**, but rank them by their popularity in current time slot;
- **Baseline 4** $\langle l\rangle$: recommended entities that have been queried in current location, and rank them by their distance to the user;
- **Baseline 5** $\langle l, p\rangle$: recommend entities as **Baseline 4**, but rank by popularity;
- **Baseline 6** $\langle l, p, t\rangle$: recommend entities that have been queried in both current location and time slot, then rank by popularity;
- **PCAR-T** $\langle u, l, t\rangle$: the Personalized Context-Aware entity Ranking (PCAR) algorithm proposed in this paper. It is essentially a hybrid recommendation algorithm. The tf-idf representation of users for building user similarity graph is entity-type based;
- **PCAR-E** $\langle u, l, t\rangle$: the same as PCAR-T except the tf-idf representation of users are entity-based.

It is worth noticing that **Baseline 1–3** and **Baseline 4–6** can be categorized into two groups: the first group only considers the entities whose *physical locations* are close to user's current location, while the second only considers the entities which are *queried* in user's current location.

Experiment I: Top-k Recommendation Accuracy

Figure 5 shows the recommendation accuracy of the evaluated schemes. We can draw several observations from this figure. First, among all the baseline methods 1-6, baseline 4-6 reports significantly better results than baseline 1-3. Note that baseline 1-3 deem all the existing entities as candidates when recommending to users, while baseline 4-6 only use the entities that have been queried in current context as candidates. Therefore, we conclude entities used to be searched are more likely to be searched again. Baseline 5-6 are slightly better than baseline 4, which implies: 1) popularity is more important than geometric distance, and 2) time is a effective context for recommendation.

Second, PCAR produces significantly higher accuracy than all the Baseline methods. PCAR not only considers the popularity of entities through $P(E|Z)$, but only leverages the information of each individual user through the generating probability $P(Z|U)$ and the transition probability $P(Z|Z')$.

[2] The *popularity p* of an entity is the number of times it has been queried.

Besides the context, the entity-oriented search depends on the user in nature. For example, when searching a restaurant, different users may have different cuisine preference. People who like Chinese food may conduct totally different queries comparing with people who favor Western food. Therefore, it is important to make the preference personalized. The proposed PCAR system yields the accuracy of 10% higher than that of baselines. Thus we conclude that personalization exists in entity-oriented search.

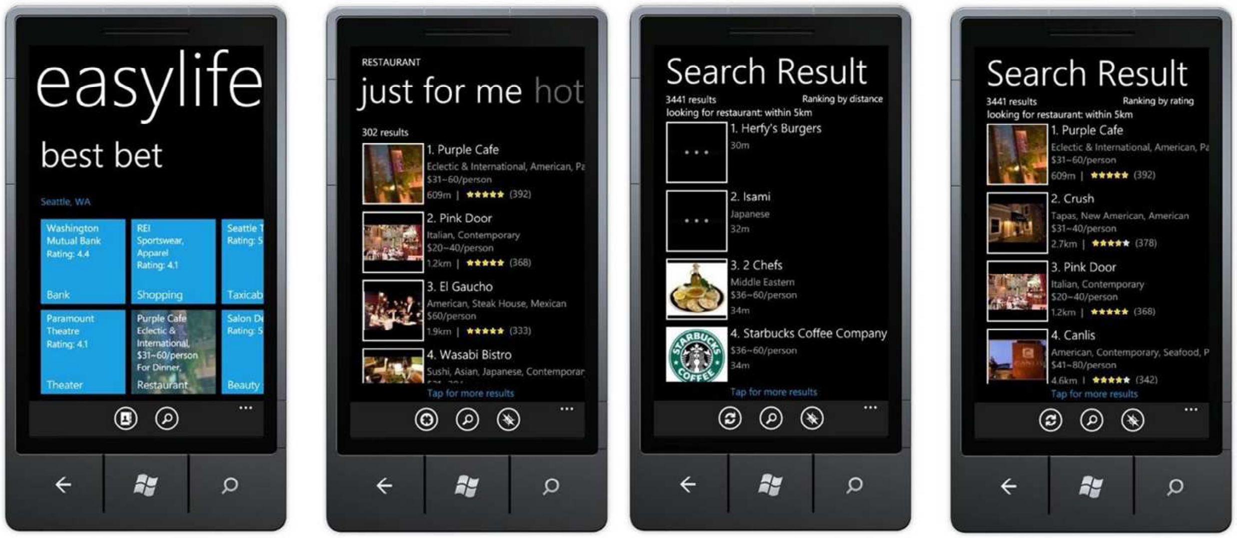
Third, comparing the two user representation schemes, PCAR-E is better than PCAR-T for entity recommendation, while PCAR-T is better than PCAR-E for type recommendation. It implies that the tf-idf representation for users when computing user similarity graph is task-dependent. The top-10 EntityAcc is 40% and IntentAcc approaches to 90%. This high accuracy confirms the efficacy of the proposed probabilistic entity ranking algorithm. It also verifies the importance of taking context and personalization into account. It is observed that the accuracy of intent-entity ranking is around 30%, which leaves much room for improvement in our future work.

Experiment II: Sensitivity to Context

The first step of entity ranker retrieves a set of candidate queries $\mathcal{Q}_{\langle l,t\rangle}$ at the current context $\langle l, t\rangle$. Then it essentially ranks the entities contained in $\mathcal{Q}_{\langle l,t\rangle}$ by a probabilistic estimation of $P(E|U; L, T)$. Therefore, the performance of entity ranker relies on the quality of $\mathcal{Q}_{\langle l,t\rangle}$. In the implementation, we sort each candidate query $q \in \mathcal{Q}$ according its distance to the user. We extract the nearest-n queries (i.e., $|\mathcal{Q}_{\langle l,t\rangle}| = n$) as candidates for recommendation. We evaluate how sensitive of entity ranker to the number n. The results are shown in Figure 6.

First, we observe that the EntityAcc increases with n. When n is a small value (e.g., 30), the accuracy is merely about 32%. This is because a large portion of valuable candidate queries are ignored. The most expected entity may not be selected as candidate at all. In this case, no matter how good the ranking algorithm, it never leads to the good performance. As n increases, more entities are involved into the ranking process. When n is larger than 200, the further gain of accuracy seems marginal. When $n = 300$, the accuracy approaches to 40%. We believe that the performance would be improved with the ongoing collection of the queries.

We also emphasize that, even the accuracy is not too high, the recommended entity list is still meaningful in the sense that it provides a suggestion about the nearby entities. By re-



(a) entity type ranked by user model (b) entity ranked by user model (c) entity ranked by distance (d) entity ranked by user rating

Figure 7. The different recommended entity lists shown in user study.

viewing the attributes of the top ranked entities, the activity of users may be affected positively or guided by the recommendation results.

The average time cost of each test query is in the interval $[0.075, 0.085]$ sec with n ranging from 30 to 300. We conclude it is efficient enough for real-time application. From the flowchart in Figure 4, the computational costs in the perpendicular direction include: find candidate queries, estimate generating probability, and random walk-based refinement. As n is typically less than 300, these steps can be done very quickly. The cost on building user similarity graph and entity similarity graph can be very heavy. However, these two steps can be finished in an offline phase.

To further show the context-awareness and personalization, the two key characteristics of our approach, we sample four users (*Jennifer, Peter, Pok, and Miranda*) to show their recommendation results in different contexts in Table 4. The recommendation results exhibit the personalized characteristics clearly. At the same context, for example, NewYork at 18:26PM, the top recommended entity types of Jennifer are different from the ones of Peter. So is the case of Pok and Miranda. This verifies the personalization target of *easylife*, that is, the recommendation results should be diverse for different users even if the context is the same.

For the same user Miranda, the recommendation results at 22:15 are different from the ones at 13:36 even if the location is the same. Thus our proposed recommendation approach is flexible to meet the user's demand. This fact shows it works in a context-aware manner. Moreover, the three most frequently searched entity types of Miranda are bank, shopping, and restaurant. Depending upon the context, the system can rank them in a proper order. The result in Figure 5(b) has shown the disadvantages of the simple frequency based ranking.

Experiment III: User Study

We developed a real application called *easylife* on a Windows Phone 7 device, based on the proposed contextual recommendation approach. The application is able to automatically recommend both entity types (as shown in Figure 1 (b)) and entities within each type (Figure 1 (c)). The entities come from a dataset with 700 million local businesses and 30 entity types in U.S. We evaluate *easylife* by a series of user studies.

As our approach is built on a large-scale mobile query log data, we randomly selected 12 users frequently conducted mobile searches from our database. As "user id" is anonymous without any identity information (for privacy consideration), we invited 12 subjects to participate in the role play study. Each subject was assigned a user id and asked to play as the role of that user. The subjects were provided the search history (each search record consists of "time," "location," and "query"), and then asked to fill a form to estimate the profile of this user (e.g., age range, interested entity events, residence, etc.). Then, each subject was asked to select two time slots and two locations (one is residence, the other is somewhere else). Based on the selected context (time and location, as well as user id), we provide four ranked recommendation results to this subject: 1) entity types ranked by the built user model, 2) entities ranked by user model which belongs to the top ranked entity type (supposing user will click the "best bet" entity type in *easylife*), 3) entities ranked by distance to current location (without considering user model), and 4) entities ranked by user rating (collected from users without user model). Figure 7 shows the example of four recommendation results. Each subject was asked to give a satisfaction score of each list in a scale of 1—5: the higher the score, the more satisfied the subject is toward the recommended list. As a result, there are 192 scores ($12 \times 4 \times 2 \times 2$) in total, with each recommendation list associated with 48 scores. We computed the average satisfaction scores for each ranked list across different subjects.

Table 5. Subjective evaluation of four recommendation schemes.

Recommendation	Entity type	User model	By distance	By rating
Satisfaction score	4.31	4.00	2.79	2.54

Table 5 lists the evaluation results. The satisfaction score of entity type recommendation is the highest. Among the three entity recommendation schemes, user model achieves a better performance than the recommendations by distance and by rating (the rating data were collected from a popular commercial local business site). This shows that our recommendation technique by user model (i.e., the proposed contextual and personalized approach) is more user-friendly. Moreover, it is a little surprising that user rating is even worse than recommendation by distance. It is partially because user rating may ignore the location context for recommendation, which is critical for daily life. This observation is also confirmed by our analysis on mobile click-through data. The recommendation by user model is significantly better than that by distance. It shows that personalization is important for recommendation task on the mobile devices.

CONCLUSION

In this paper, we first conduct an analysis on a large-scale mobile click-through data collected from a commercial mobile search engine. Our observations indicate that the queries conducted on mobile devices are typically short, context-aware, and local. Motivated by these observations, we propose a query-free entity recommendation approach to understand implicit user intent on the mobile devices. The proposed approach is capable to rank relevant entity types (e.g., restaurant, hotel, bar, etc.) and entities within each type (e.g., "I love sushi" and "MacDonalds" in the type of "restaurant"). We developed a real recommendation application based on the proposed approach on Windows Phone 7 devices and evaluated it through both objective and subjective experiments. The evaluations shows our proposed approach achieves the best user experience than the traditional location and rating-based recommendation schemes.

Our future works include: 1) exploring other machine learning techniques for a better recommendation, such as learning to rank, matrix factorization, and so on; 2) leveraging social signals for improving user similarity (e.g., using the friendship in social media community such as Facebook); and 3) collecting real-world click-through data through the developed mobile application and performing the real-world evaluation on these data.

ACKNOWLEDGEMENTS

We would like to thank Peng Bai, Zheng Chen, Xuedong Huang, Xin Lin, Xiaochuan Ni, Jian-Tao Sun, Peng Xu, Bo Zhang, and Zhimin Zhang, for their insightful discussions. We also thank the subjects involved in our user studies.

REFERENCES

1. http://www.discoverbing.com/mobile/index.html.
2. http://www.google.com/mobile.
3. G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.*, 17(6):734–749, 2005.
4. H. Cao, D. H. Hu, D. Shen, D. Jiang, J.-T. Sun, E. Chen, and Q. Yang. Context-aware query classification. In *SIGIR*, pages 3–10, 2009.
5. 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 *KDD*, pages 875–883, 2008.
6. W. Chu and S.-T. Park. Personalized recommendation on dynamic content using predictive bilinear models. In *WWW*, pages 691–700, 2009.
7. K. Church, B. Smyth, K. Bradley, and P. Cotter. A large scale study of european mobile search behaviour. In *Mobile HCI*, pages 13–22, 2008.
8. O. Etzioni, M. J. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artif. Intell.*, 165(1):91–134, 2005.
9. J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *SIGIR*, pages 267–274, 2009.
10. I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel. Social media recommendation based on people and tags. In *SIGIR*, pages 194–201, 2010.
11. M. Kamvar and S. Baluja. The role of context in query input: using contextual signals to complete queries on mobile devices. In *Mobile HCI*, pages 405–412, 2007.
12. M. Kamvar and S. Baluja. Query suggestions for mobile search: understanding usage patterns. In *CHI*, pages 1013–1016, 2008.
13. M. Kamvar, M. Kellar, R. Patel, and Y. Xu. Computers and iphones and mobile phones, oh my!: a logs-based comparison of search users on different devices. In *WWW*, pages 801–810, 2009.
14. D. Kelly, K. Gyllstrom, and E. W. Bailey. A comparison of query and term suggestion features for interactive searching. In *SIGIR*, pages 371–378, 2009.
15. N. Lane, D. Lymberopoulos, F. Zhao, and A. Campbell. Hapori: Context-based local search for mobile phones using community behavioral modeling and similarity. In *Proc. of ACM International Conference on Ubiquitous Computing*, pages 109–118, 2010.
16. J. Lee, S. won Hwang, Z. Nie, and J.-R. Wen. Query result clustering for object-level search. In *KDD*, pages 1205–1214, 2009.
17. L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *WWW*, pages 661–670, 2010.
18. X. Li, Y.-Y. Wang, D. Shen, and A. Acero. Learning with click graph for query intent classification. *ACM Trans. Inf. Syst.*, 28(3), 2010.
19. Z. Lu, D. Agarwal, and I. S. Dhillon. A spatio-temporal approach to collaborative filtering. In *RecSys*, pages 13–20, 2009.
20. E. Meij, P. Mika, and H. Zaragoza. An evaluation of entity and frequency based query completion methods. In *SIGIR*, pages 678–679, 2009.
21. Z. Nie, J.-R. Wen, and W.-Y. Ma. Object-level vertical search. In *CIDR*, pages 235–246, 2007.
22. Y. Rui, T. S. Huang, M. Ortega, and S. Mehrotra. Relevance feedback: A power tool in interactive content-based image retrieval. *IEEE Trans. CSVT*, 8(5):644–655, Sept. 1998.
23. F. Sebastiani. Machine learning in automated text categorization. *CoRR*, 2001.
24. Siri. http://siri.com/.
25. X. Yin and S. Shah. Building taxonomy of web search intents for name entity queries. In *Proceedings of WWW*, pages 1001–1010, 2010.
26. K. Yu, J. D. Lafferty, S. Zhu, and Y. Gong. Large-scale collaborative prediction using a nonparametric random effects model. In *ICML*, 2009.