Personalized Web Search with Location Preferences

Kenneth Wai-Ting Leung #1, Dik Lun Lee #2, Wang-Chien Lee *3

#Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong
[1]kwtleung@cse.ust.hk [2]dlee@cse.ust.hk

#Department of Computer Science and Engineering, The Pennsylvania State University, USA
[3]wlee@cse.psu.edu

***Abstract*— As the amount of Web information grows rapidly, search engines must be able to retrieve information according to the user's preference. In this paper, we propose a new web search personalization approach that captures the user's interests and preferences in the form of concepts by mining search results and their clickthroughs. Due to the important role location information plays in mobile search, we separate concepts into content concepts and location concepts, and organize them into ontologies to create an *ontology-based, multi-facet (OMF)* profile to precisely capture the user's content and location interests and hence improve the search accuracy. Moreover, recognizing the fact that different users and queries may have different emphases on content and location information, we introduce the notion of content and location entropies to measure the amount of content and location information associated with a query, and *click* content and location entropies to measure how much the user is interested in the content and location information in the results. Accordingly, we propose to define personalization effectiveness based on the entropies and use it to balance the weights between the content and location facets. Finally, based on the derived ontologies and personalization effectiveness, we train an SVM to adapt a personalized ranking function for re-ranking of future search. We conduct extensive experiments to compare the precision produced by our OMF profiles and that of a baseline method. Experimental results show that OMF improves the precision significantly compared to the baseline.**

I. Introduction

In mobile search, the interaction between users and mobile devices are constrained by the small form factors of the mobile devices. To reduce the amount of user's interactions with the search interface, an important requirement for mobile search engine is to be able to understand the users' needs, and deliver highly relevant information to the users. Personalized search is one way to resolve the problem. By capturing the users' interests in user profiles, a personalized search middleware is able to adapt the search results obtained from general search engines to the users' preferences through personalized reranking of the search results. In the personalization process, user profiles play a key role in reranking search results and thus need to be trained constantly based on the user's search activities. Several personalization techniques have been proposed to model users' content preferences via analysis of users' clicking and browsing behaviors [5], [9], [12], [14]. In this paper, we recognize the importance of location information in mobile search and proposes to incorporate the user's *location preferences* in addition to content preferences in user profiles.

We propose an *ontology-based*, *multi-facet* (OMF) user profiling strategy to capture both of the users' content and location preferences (i.e., "multi-facets") for building a personalized search engine for mobile users. Figure 1 shows the general process of our approach, which consists of two major activities: 1) Reranking and 2) Profile Updating.

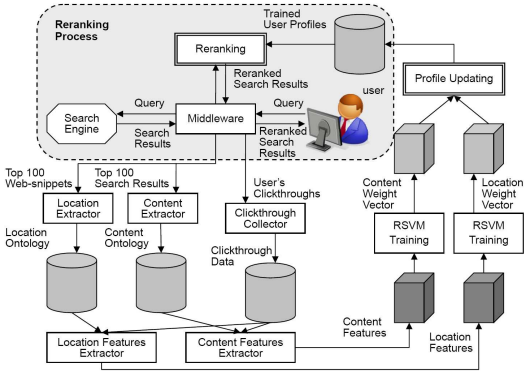


Fig. 1. The general process of proposed personalization approach.

- **Reranking**: When a user submits a query, the search results are obtained from the backend search engines (e.g., Google, MSNSearch, and Yahoo). The search results are combined and reranked according to the user's profile trained from the user's previous search activities.
- **Profile Updating**: After the search results are obtained from the backend search engines, the content and location concepts (i.e. important terms and phrases) and their relationships are mined online from the search results and stored, respectively, as content ontology and location ontology. When the user clicks on a search result, the clicked result together with its associated content and location concepts are stored in the user's clickthrough data. The content and location ontologies, along with the clickthrough data, are then employed in RSVM [9] training to obtain a content weight vector and a location weight vector for reranking the search results for the user.

There are a number of challenging research issues we need to overcome in order to realize the proposed personalization approach. First, we aim at using "concepts" to represent and profile the interests of a user. Therefore, we need to build up and maintain a user's possible *concept space*, which are important concepts extracted from the user's search results. Additionally, we observe that location concepts exhibit different characteristics from content concepts and thus need to be treated differently. Thus, we propose to represent them

in separate content and location ontologies. These ontologies not only keep track of the encountered concepts accumulated through past search activities but also capture the "relationships" among various concepts, which plays an important role in our personalization process.

Second, we recognize that the same content or location concept may have different degrees of importance to different users and different queries. Thus, there is a need to characterize the diversity of the concepts associated with a query and their relevances to the user's need. To address this issue, we introduce the notion of content and location entropies to measure the amount of content and location information a query is associated with. Similarly, we propose *click* content and location entropies to measure how much the user is interested in the content and/or location information in the results. We can then use these entropies to estimate the personalization effectiveness for a given user and a particular query, and use the measure to adapt the personalization mechanism to enhance the accuracy of the search results.

Finally, the extracted content and location concepts from search results and the feedback obtained from clickthroughs need to be transformed into a form of user profile for future reranking. Additionally, it is critical to be able to combine and balance the obtained location and content preferences seamlessly. Our strategy for this issue is to train an SVM to adapt personalized ranking functions for content and location preferences and then employ the derived personalization effectiveness to strike a balanced combination between them.

The main contributions of this paper are five-fold:

- The ontology-based, multi-facet (OMF) framework is an innovative approach for personalizing web search results by mining content and location concepts for user profiling. To the best knowledge of the authors, there is no existing work in the literature that takes into account both types of concepts. This paper studies their unique characteristics and provides a coherent strategy to integrate them into a uniform solution.
- We propose a content ontology and a location ontology to accommodate the extracted content and location concepts as well as the relationships among the concepts.
- We introduce different entropies to indicate the amount of concepts associated with a query and how much a user is interested in these concepts. With the entropies, we are able to estimate the effectiveness of personalization for different users and different queries.
- Based on the proposed ontologies and entropies, we adopt an SVM to learn personalized ranking functions for content and location preferences. We use the personalization effectiveness to integrate the learned ranking functions into a coherent profile for personalized reranking.
- We implement a working prototype to validate the proposed ideas. It consists of a middleware for capturing user clickthroughs, performing personalization, and interfacing with commercial search engines at the backend. Empirical results show that OMF can successfully capture users' content and location preferences and utilize the preferences to produce relevant results for the users. It significantly out-performs strategies which use either content or location preference only.

The rest of the paper is organized as follows. We review the related work in Section II. In Section III, we present our ontology extraction method for building the content and location ontologies. In Section IV, we introduce the notion of content and location entropies and show how they benefit search personalization. We classify the users and queries in our experiments into different classes according to their entropies, and show the effectiveness of personalization for each class in Section VII-D. In Section V, we review the method to extract user preferences from the clickthrough data to create the user profiles. In Section VI, we discuss the RSVM method [9] for learning a linear weight vector (consisting both content and location features) to rank the search results. Performance results of our user profiling and personalization strategies for different classes of users and queries are presented in Section VII. Section VIII concludes the paper.

II. Related Work

Most commercial search engines return roughly the same results to all users. However, different users may have different information needs even for the same query. For example, a user who is looking for a laptop may issue a query "apple" to find products from Apple Computer, while a housewife may use the same query "apple" to find apple recipes. The objective of personalized search is to disambiguate the queries according to the users' interests and to return relevant results to the users.

Clickthrough data is important for tracking user actions on a search engine. Table I is an example clickthrough data for the query "university". It consists of the search results of a user's query and the results that the user has clicked on. c_i's are the content concepts and l_i's are the location concepts extracted from the corresponding results. Many personalized web search systems [5], [9], [12], [14] are based on analyzing users' clickthroughs. Joachims [9] proposed to use document preference mining and machine learning to rank search results according to user's preferences. Later, Agichitein et al. [5] proposed a method to learn users' clicking and browsing behaviors from the clickthrough data using a scalable implementation of neural networks called RankNet [6]. More recently, Ng et al. [12] extended Joachims method by combining a spying technique together with a novel voting procedure to determine user preferences. In [10], Leung et al. introduced an effective approach to predict users' conceptual preferences from clickthrough data for personalized query suggestions.

Gan et. al [8] suggested that search queries can be classified into two types, **content (i.e., non-geo)** and **location (i.e., geo)**. Typical examples of geographic queries are "hotels hong kong", "building codes in seattle" and "virgina historical sites". A classifier was built to classify geo and non-geo queries, and the properties of geo queries were studied in detail. It was found that a significant number of queries were location queries focusing on location information. Hence, a

TABLE I

EXAMPLE CLICKTHROUGHS FOR THE QUERY "UNIVERSITY"

Doc	Search Results	c_i	l_i
d_1	University of Cambridge	student, department	UK
d_2	Libraries, Oxford University	library	UK
d_3	Harvard University	student	United States
$\mathbf{d_4}$	**Education UK, University of Manchester**	**research, library**	**UK, Manchester**
d_5	University of Virginia	student	VA
$\mathbf{d_6}$	**University of Edinburgh**	**research**	**UK**
d_7	Library Services, University of Birmingham	library	UK, Birmingham
$\mathbf{d_8}$	**University of Leeds**	**research affiliation**	**UK**

number of location-based search systems designed for geo queries have been proposed. These include Yokoji et al. [15], who proposed a location-based search system for web documents. A parser was employed to extract location information from web documents, which was converted into latitude-longitude pairs or polygons. When a user submits a query together with the location information specified in a latitude-longitude pair, the system creates a search circle centered at the specified latitude-longitude pair and retrieves documents containing location information within the search circle.

More recently, Zhou et al. [16] proposed a hybrid index structure to handle both content and location-aware queries. The system first detects geographical scopes from web documents and represents the geographical scopes as multiple minimum bounding rectangles (MBRs) based on geographical coordinates. A hybrid index structure is used to index the content and location information of the web documents. A user is required to present their content and location interest in their search queries. A ranker is then employed to rank the search results according to the content and location relevances using the hybrid index.

The differences between our work and existing works are:

- Existing works such as [15], [16] require the users' to manually define their location preferences explicitly (with latitude-longitude pairs or text form). With the automatically generated content and location user profiles, our method does not require users to explicitly define their location interest manually.
- Our method automatically profiles both of the user's content and location preferences, which are automatically learnt from the user's clickthrough data without requiring extra efforts from the user.
- Our method uses different formulations of entropies derived from a query's search results and a user's clickthroughs to estimate the query's content and location ambiguities and the user's interest in content or location information. The entropies allow us to classify queries and users into different classes and effectively combine a user's content and location preferences to rerank the search results.

TABLE II

EXAMPLE CONCEPTS EXTRACTED FOR THE QUERY "SOUTHEAST ASIA"

Content Concept c_i	**Location Concept** l_i
biking	Cambodia
language	Indian Ocean
people	Indonesia
relief effort	Malaysia

III. CONCEPT EXTRACTION

Our personalization approach is based on "concepts" to profile the interests and preferences of a user. Therefore, an issue we have to address is how to *extract* and *represent* concepts from search results of the user. We propose in this paper an *ontology-based*, *multi-facet (OMF)* profiling method in which concepts can be further classified into different types, such as content concepts, location concepts, name entities, dates etc. As an important first step, we focus on two major types of concepts, namely, **content concepts** and **location concepts**. A content concept, like a keyword or key-phrase in a Web page, defines the content of the page, whereas a location concept refers to a physical location related to the page. Table II shows an example query "southeast asia" with the content and location concepts extracted.

We argue that the interests of a search engine user can, in the long run, be effectively represented by concepts extracted from the user's search results. The extracted concepts indicate *a possible concept space* arising from a user's queries, which can be maintained along with the clickthrough data for future preference adaptation. In our personalization framework, we adopt ontologies to model the concept space because they not only can represent concepts but also capture the relationships between concepts. Due to the different characteristics of the content concepts and location concepts, we use different techniques for their concept extraction and ontology formulation. In Section III-A, we first discuss our method to mine and build the content ontology from the search results. In Section III-B, we present our method to derive a location ontology from the search results.

A. Content Ontology

We assume that if a keyword/phrase exists frequently in the web-snippets[1] arising from the query q, it represents an important concept related to the query, as it co-exists in close proximity with the query in the top documents. Thus, our content concept extraction method first extracts all the keywords and phrases from the web-snippets arising from q. After obtaining a set of keywords/phrases (c_i), the following support formula, which is inspired by the well-known problem of finding frequent item sets in data mining [7], is employed to measure the interestingness of a particular keyword/phrase c_i with respect to the query q:

$$support(c_i) = \frac{sf(c_i)}{n} \cdot |c_i| \quad (1)$$

[1]"Web-snippet" denotes the title, summary and URL of a Web page returned by search engines.

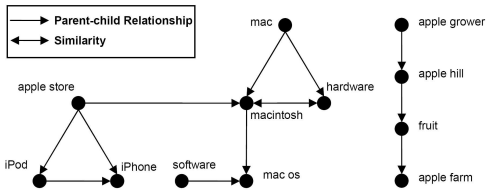


Fig. 2. Example Content Ontology Extracted for the Query "apple".

where $sf(c_i)$ is the snippet frequency of the keyword/phrase c_i (i.e. the number of web-snippets containing c_i), n is the number of web-snippets returned and $|c_i|$ is the number of terms in the keyword/phrase c_i. If the support of a keyword/phrase c_i is higher than the threshold s ($s = 0.03$ in our experiments), we treat c_i as a concept for the query q.

As mentioned, we use ontologies to maintain concepts and their relationships extracted from search results. We capture the following two types of relationships for content concepts:

- **Similarity**: Two concepts which coexist a lot on the search results might represent the same topical interest. If $coexist(ci, cj) > \delta_1$ (δ_1 is a threshold), then c_i and c_j are considered as similar.
- **Parent-Child Relationship**: More specific concepts often appear with general terms, while the reverse is not true. Thus, if $pr(cj|ci) > \delta_2$ (δ_2 is a threshold), we mark c_i as c_j's child. For example, the concept "Manchester United" tends to occur together with "soccer", while the concept "soccer" might also occur with concepts such as "Chelsea", "Real Madrid" or "Juventus", i.e., not only with the concept "Manchester United".

Figure 2 shows an example content ontology created for the query "apple". Content concepts linked with a double-sided arrow ($\leftrightarrow$) are similar concepts, while concepts linked with a one-sided arrow ($\rightarrow$) are parent-child concepts. The ontology shows the possible concept space arising from a user's queries. In general, the ontology covers more than what the user actually wants. For example, when the query "apple" is submitted, the concept space for the query composes of "mac", "software", "fruit", ..., etc. If the user is indeed interested in apple as a fruit and clicks on pages containing the concept "fruit", the clickthrough is captured and the clicked concept "fruit" is favored. The content ontology together with the clickthrough serve as the user profile in the personalization process. They will then be transformed into a linear feature vector to rank the search results according to the user's content information preferences. The details of the transformation will be discussed in Section VI-A.

B. Location Ontology

Our approach for extracting location concepts is different from that for extracting content concepts. First, a document usually embodies only a few location concepts. As a result, very few of them co-occur with the query terms in web-snippets. To alleviate this problem, we extract location concepts from the full documents.

Second, due to the small number of location concepts embodied in documents, the similarity and parent-child relationship cannot be accurately derived statistically. Additionally, the geographical relationships among many locations have already been captured as facts. Thus, we create a predefined location ontology consisting of about 17,000 city, province, region, and country names obtained from [2] and [4]. In the location ontology, we organize all the cities as children under their provinces, all the provinces as children under their regions, and all the regions as children under their countries. The statistics of our location ontology are provided in Table III. Figure 3 shows an example of the predefined hierarchy of geographical locations for the United States.

TABLE III
STATISTICS OF THE LOCATION ONTOLOGY

No. of Countries	7	Total No. of Nodes	16899
No. of Regions	190	Country-Region Edges	190
No. of Provinces	6699	Region-Province Edges	1959
No. of Towns	10003	Province-City Edges	14897

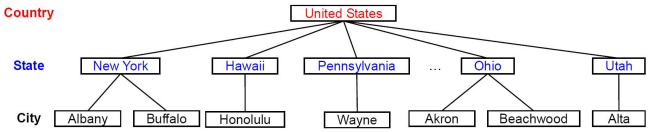


Fig. 3. Example Predefined Hierarchy for "United States".

The location ontology extraction method first extracts all of the keywords and key-phrases from the documents returned for q. If a keyword or key-phrase in a retrieved document d matches a location name in our predefined location ontology, it will be treated as a location concept of d. For example, given the concept "Albany" from document d, we would match it against our location ontology. If a match is found, we would explore the corresponding location hierarchy (i.e. the United States' hierarchy), which would identify "Albany" as a city under the state "New York". Thus, the location "/United States/New York/Albany/" is associated with document d. If a concept matches several nodes in the location ontology, all matched locations will be associated with the document.

Similar to the content ontology, locations are assigned with different weights according the user's clickthroughs. If a user is interested in "New York" and clicks on results about "New York", the clickthroughs would gradually favor the location "New York" by assigning higher weight to it, but the weights of the other locations trees such as "Hawaii" and "Utah" would remain zero. The weighted location ontology is then transformed into a linear feature vector for ranking. The details of the transformation will be discussed in Section VI-A.

IV. CONCEPT AND ENTROPY

A personalization process based on either the content concepts or location concepts may bring different adaptation effects for different queries and users. As shown earlier, our personalization approach consists of two separate preference adaptation processes, tailored for content and location concepts. In order to seamlessly integrate the two processes into one coherent personalization framework, an important issue we have to address is how to weigh the content preference and location preference in the integration step.

An idea to address this issue is to adjust the weights of content preference and location preference based on their

effectiveness in the personalization process. For a given query issued by a particular user, if the personalization based on content concepts is more effective than based on location concepts, more weight should be put on content-based preference; and vice versa. Therefore, the key question to answer in this section is how to measure "personalization effectiveness" for content and location facets of our personalization approach.

As discussed in [8], different queries may induce from the search results different concept spaces which are different in both sizes and diversities. Additionally, different users may have different interests and preferences on the search results. The diversity of content and location information of a query can be derived from the extracted content and location concepts, while the interest/preference of a user can be captured by his clickthrough behavior. In this section, we introduce the *content and location entropies* for measuring the diversity of content and location information from the search results of a query. In addition, we introduce the *click* content and location entropies to determine how much a user is interested in the content and location information associated with a query. Based on these proposed entropies, we derive the notion of *personalization effectiveness* to effectively combine a user's content and location preferences for reranking the search results. Finally, we perform a case study by classifying the users and queries into different classes according to their content/location and *click* content/location entropies.

A. Content and Location Entropies

Different queries may be associated with different amount of content and location information. For example, queries such as "Overseas Study" may have strong associations to a large number of location concepts. However, queries such as "Programming" tend to be content-oriented with only weak association to location concepts (i.e., most concepts, such as "books" and "software tools", related to computer programming are location independent). Meanwhile, some queries (e.g. "Shopping") can be rich in both content and location information. To formally characterize the content and location properties of a query, we use *entropy* to estimate the amount of content and location information retrieved by a query.

In information theory [13], *entropy* indicates the uncertainty associated with the information content of a message from the receiver's point of view. In the context of search engine, entropy can be employed in a similar manner to denote the uncertainty associated with the information content of the search results from the user's point of view. Since we are concerned with content and location information only in this paper, we define two entropies, namely, **content entropy** and **location entropy**, to measure, respectively, the uncertainty associated with the content and location information of the search results. The information entropy of a discrete random variable X is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i) \tag{2}$$

where n is the possible values $\{x_1, x_2, ...x_n\}$ of X and $p(x_i) = Pr(X = x_i)$. We adopt the above formula to compute the content and location entropies of a query q (i.e. $H_C(q)$ and $H_L(q)$) as follows.

$$H_C(q) = -\sum_{i=1}^{k} p(c_i) \log p(c_i) \quad H_L(q) = -\sum_{i=1}^{m} p(l_i) \log p(l_i) \tag{3}$$

where k is the number of content concepts $C = \{c_1, c_2, ..., c_k\}$ extracted, $|c_i|$ is the number of search results containing the content concept c_i, $|C| = |c_1|+|c_2|+...+|c_k|$, $p(c_i) = \frac{|c_i|}{|C|}$, m is the number of location concepts $L = \{l_1, l_2, ..., l_m\}$ extracted, $|l_i|$ is the number of search results containing the location concept l_i, $|L| = |l_1| + |l_2| + ... + |l_m|$, and $p(l_i) = \frac{|l_i|}{|L|}$.

B. Click Content and Location Entropies

As with content and location entropies, we introduce *click content entropy* and *click location entropy* to indicate, respectively, the diversity of a user's interest on the content and location information returned from a query. The entropy equations for click content and location concepts are similar to Equation (3), but only the clicked pages, and hence the clicked concepts, are considered in the formula. Since the click entropies reflects the user's actions in response to the search results, they can be used as an indication of the diversity of the user's interests. Formally, the click content entropy $H_{\overline{C}}(q, u)$ and click location entropy $H_{\overline{L}}(q, u)$ of a query q submitted by the user u are defined as follows.

$$H_{\overline{C}}(q, u) = -\sum_{i=1}^{t} p(\overline{c_i}_u) \log p(\overline{c_i}_u) \tag{4}$$

$$H_{\overline{L}}(q, u) = -\sum_{i=1}^{v} p(\overline{l_i}_u) \log p(\overline{l_i}_u) \tag{5}$$

where t is the number of content concepts clicked by the user u, $\overline{C}_u = \{\overline{c_1}_u, \overline{c_2}_u, ..., \overline{c_t}_u\}$, $|\overline{c_i}_u|$ is the number of times that the content concept c_i has been clicked by user u, $|\overline{C}_u| = |\overline{c_1}_u| + |\overline{c_2}_u| + ... + |\overline{c_t}_u|$, $p(\overline{c_i}, u) = \frac{|\overline{c_i}_u|}{|\overline{C}_u|}$, v is the number of location concepts $\overline{L}_u = \{\overline{l_1}_u, \overline{l_2}_u, ..., \overline{l_v}_u\}$ clicked by u, $|\overline{l_i}_u|$ is the number of times that the location concept l_i is being clicked by the user u, $|\overline{L}_u| = |\overline{l_1}_u| + |\overline{l_2}_u| + ... + |\overline{l_v}_u|$, and $p(\overline{l_i}, u) = \frac{|\overline{l_i}_u|}{|\overline{L}_u|}$.

C. Personalization Effectiveness

As discussed in the last subsection, a query result set with high content/location entropy indicates that it has a high degree of ambiguity. Thus, applying personalization on the search results helps the user to find out the relevant information. On the other hand, when the content/location entropy is low, meaning that the result set is already very focused and should have matched the query quite precisely, personalization can do very little in further improving the precision of the result.

For click entropies, we expect that the higher the click content/location entropies, the worse the personalization effectiveness, because high click content/location entropies indicate that the user is clicking on the search results with high uncertainty, meaning that the user is interested in a diversity

of information in the search results. When the user's interests are very broad (or the clickthroughs could be "noisy" due to irrelevant concepts existing in the clicked documents), it is difficult to (i) find out the user's actual needs and (ii) personalize the search results towards the user's interest. On the other hand, if the click content/location entropies are low, the personalization effectiveness would be high because the user has a focus on certain precise topic in the search results (only a small set of content/location concepts in the search results has been clicked by the user). Hence, the profiling process can identify the user's information needs and the personalization process can personalize the results to meet those needs.

Based on the above reasoning, we propose the following equations to estimate the personalization effectiveness using the extracted content and location concepts with respect to the user u.

$$e_C(q,u) = \frac{H_C(q)}{H_{\overline{C}}(q,u)} \qquad e_L(q,u) = \frac{H_L(q)}{H_{\overline{L}}(q,u)} \tag{6}$$

We expect that queries with high $e_C(q,u)$ and $e_L(q,u)$ would yield better personalization results. An experimental evaluation is presented in Section VII-C, where we compute $e_C(q,u)$ and $e_L(q,u)$ for the content and location concepts to see how location-based and content-based personalization affects web search.

D. Case Study: Query and User Clustering

In this subsection, we perform a case study by using the proposed entropies to cluster and characterize queries and users. The entropies are derived from clickthrough data collected from 50 users who issued and evaluated a total of 250 test queries on our experimental prototype (see Section VII-A for details).

1) Query classes: We compute the content and location entropies for the 250 test queries and display them on a scatter plot with location entropy as x-axis and content entropy as y-axis (see Figure 4(a)). We use K-Means to cluster the test queries into four classes, which are shown in Fig. 4(a). Observing the content and location entropy values of the four query classes, we characterize the four classes as follows.

- **Explicit Queries:** Queries with low degree of ambiguity, i.e., they have small total content and location entropies; specifically $H_C(q) + H_L(q)$ is small.
- **Content Queries:** Queries with $H_C(q) > H_L(q)$.
- **Location Queries:** Queries with $H_L(q) > H_C(q)$.
- **Ambiguous Queries:** Queries with high degree of ambiguity, i.e., they have large total content and location entropies; specifically $H_C(q) + H_L(q)$ is large.

Example queries from the four different query classes are presented in Table IV. It is interesting to note that explicit queries receive both low content and location entropies, because the search results returned for these queries are very focused. For example, the queries "Sony" and "IBM" return pages about the companies and no ambiguity are observed on the search results. Search results from content queries are

TABLE IV
CONTENT AND LOCATION ENTROPIES FOR THE SAMPLE QUERIES

Explicit	$H_C(q)$	$H_L(q)$	**Location**	$H_C(q)$	$H_L(q)$
Canon	6.6921	5.9792	Beijing	6.6492	8.0116
IBM	6.8683	5.3383	Campus Life	6.7888	7.8522
Sony	6.6698	5.7683	Overseas Study	6.8080	7.8934
Content	$H_C(q)$	$H_L(q)$	**Ambiguous**	$H_C(q)$	$H_L(q)$
Disney Movie	8.1204	6.8074	Manchester	8.3160	7.5705
Dual Core	8.1538	6.9552	Apartment	8.2124	7.5031
Programming	8.3827	6.4718	Shopping	8.0739	7.2339

rich in content information but weak in location information. For example, the query "Programming" focuses on tips and skills on program languages, and as such has little association with location information, which is consistent with our understanding of the programming language. On the other hand, location queries are rich in location information but weak in content information. For example, we found that "Beijing", as the capital of China and the host of Olympics 2008, is associated with many other location names, resulting in high location entropy. Finally, ambiguous queries are rich in both content and location information. For example, the query "Manchester" not only retrieves a lot of location information about traveling in Manchester, but also a lot of content information about the football club, "Manchester United".

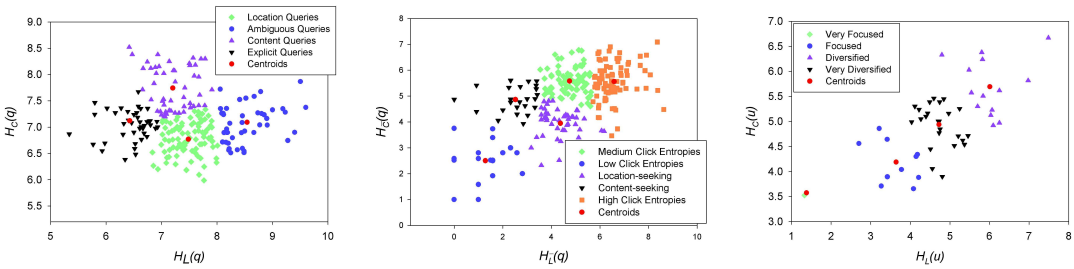
2) User classes: As with query classes, we display the queries on a scatter plot with click location entropy as x-axis and click content entropy as y-axis. The test queries are clustered into five classes with K-Means using five randomly chosen queries as the initial clusters. Again, the classes are stable with different initial clusters. The five query classes, namely, **Low Click Entropies**, **Medium Click Entropies**, **High Click Entropies**, **Content-seeking** (i.e. $H_{\overline{C}}(q,u) > H_{\overline{L}}(q,u)$), and **Location-seeking** (i.e. $H_{\overline{L}}(q,u) > H_{\overline{C}}(q,u)$), are shown in Figure 4(b).

Since clicks are performed by the users after they have read and judged the result snippets with respect to the relevance of the results to their individual needs, the click entropies can be used as an indication of user behaviors. We use the following formula to compute the average content/location click entropy of a user ($H_C(u)$ and $H_L(u)$) to estimate the user's behavior.

$$H_C(u) = \frac{1}{n}\sum_{i=1}^{n} H_{\overline{C}}(q_i,u) \quad H_L(u) = \frac{1}{n}\sum_{i=1}^{n} H_{\overline{L}}(q_i,u) \tag{7}$$

where $\{q_1, q_2, ...q_n\}$ are the queries submitted by user u. We compute $H_C(u)$ and $H_L(u)$ for each of the 50 users and display the users on a scatter plot with $H_L(u)$ as x-axis and $H_C(u)$ as y-axis. The users are clustered into four classes using K-Means (see Figure 4(c)). It is interesting to note that the users are more or less distributed along the diagonal, i.e., a user with diversified/focused location interest also has diversified/focused content interest, and vice versa. Thus, we name and describe the four user classes as follows.

- **Very Focused:** Users with low content and location entropies, i.e., they have very clear topic focuses in the search results and hence only click on a few topics. These users can be considered *careful/knowledgeable search*



(a) $H_L(q)$ vs $H_C(q)$ (b) $H_{\overline{L}}(q,u)$ vs $H_{\overline{C}}(q,u)$ (c) $H_L(u)$ vs $H_C(u)$

Fig. 4. $H_L(q)$ vs $H_C(q)$, $H_{\overline{L}}(q,u)$ vs $H_{\overline{C}}(q,u)$, and $H_L(u)$ vs $H_C(u)$; and the corresponding clusters.

engine users in that they are very selective on the results they click.

- **Focused:** Users with higher content and location entropies and hence less focused than the Very Focused class.
- **Diversified:** Users with even higher content and location entropies and hence more diversified topical interests than the first two user classes.
- **Very Diversified:** Users with high content and location entropies; they click on many topics. These users can be considered *novice search engine users* who tend to trust the search engine and click on many results it returns [5].

Experimental evaluation of the personalization effectiveness for each user class is shown in Section VII-D.

V. User Preferences Extraction

Given that the concepts and clickthrough data are collected from past search activities, user's preference can be learned. In this section, we review two alternative preference mining algorithms, namely, **Joachims Method** and **SpyNB Method**, that we adopt in our personalization framework.

A. Joachims Method

Joachims method [9] assumes that a user would scan the search result list from top to bottom. If a user skips a document d_j at rank j but clicks on document d_i at rank i where $j < i$, he/she must have read d_j's web snippet and decided to skip it. Thus, Joachims method concludes that the user prefers d_i to document d_j (denoted as $d_j <_{r'} d_i$, where r' is the user's preference order of the documents in the search result list).

Applying Joachims method to the example clickthrough data in Table I, we can obtain a set of document preference pairs as shown in Table V. The document preference pairs are then employed in a ranking SVM algorithm [9] to learn a linear feature weight vector, which is composed of either content or location concept features, to rank the search results according to the user's content and location preferences.

B. SpyNB Method

Similar to Joachims method, SpyNB [12] learns user behavior models from preferences extracted from clickthrough data. SpyNB assumes that users would only click on documents that are of interest to them. Thus, it is reasonable to treat the clicked documents as positive samples. However, unclicked documents are treated as unlabeled samples because they could be either

TABLE V
Document Preference Pairs from Joachims Method

Preference Pairs containing d_4	Preference Pairs containing d_6	Preference Pairs containing d_8
$d_4 <_{r'} d_1$	$d_6 <_{r'} d_1$	$d_8 <_{r'} d_1$
$d_4 <_{r'} d_2$	$d_6 <_{r'} d_2$	$d_8 <_{r'} d_2$
$d_4 <_{r'} d_3$	$d_6 <_{r'} d_3$	$d_8 <_{r'} d_3$
	$d_6 <_{r'} d_5$	$d_8 <_{r'} d_5$
		$d_8 <_{r'} d_7$

relevant or irrelevant to the user. Based on this interpretation of clickthroughs, the problem becomes how to predict from the unlabeled set reliable negative documents which are irrelevant to the user. To do this, the "spy" technique incorporates a novel voting procedure into Naïve Bayes classifier [11]. The details of the SpyNB method can be found in [12]. Let P be the positive set, U the unlabeled set and PN the predicted negative set ($PN \subset U$) obtained from the SpyNB method. SpyNB assumes that the user would always prefer the positive set rather than the predicted negative set. Thus, user preference pairs can be obtained as follows.

$$d_i < d_j \ , \quad \forall l_i \in P \ , \quad l_j \in PN \tag{8}$$

Similar to Joachims method, the ranking SVM algorithm is also employed to learn a linear feature weight vector to rank the search results according to the user's content and location preferences.

VI. Personalized Ranking Functions

Ranking SVM [9] is employed in our personalization approach to learn the user's preferences. For a given query, a set of content concepts and a set of location concepts are extracted from the search result as the document features. Since each document can be represented by a feature vector, it can be treated as a point in the feature space. Using clickthrough data as the input, RSVM aims at finding a linear ranking function, which holds for as many document preference pairs as possible. In our experiments, an adaptive implementation, *SVMlight* available at [3], is used for the training. It outputs a **content weight vector** $\overrightarrow{w_{C,q,u}}$ and a **location weight vector** $\overrightarrow{w_{L,q,u}}$, which best describes the user interests based on the user's content and location preferences extracted from the user clickthroughs, respectively. In the following, we discuss two issues in the RSVM training process: 1) how to extract the feature vectors for a document; 2) how to combine the content and location weight vectors into one integrated weight vector.

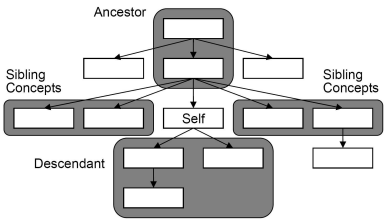


Fig. 5. Parent-child relationships, 1) Ancestors, 2) Descendants, and 3) Sibling Concepts, in a concept ontology.

A. Extracting Features for Training

Two feature vectors, namely, **content feature vector** (denoted by $\phi_C(q,d)$) and **location feature vector** (denoted by $\phi_L(q,d)$) are defined to represent documents. The feature vectors are extracted by taking into account the concepts existing in a documents and other related concepts in the ontology of the query. For example, if a document d_k embodies the content concept c_i and location concept l_i, the weight of component c_i in the content feature vector $\phi_C(q,d_k)$ of document d_k is incremented by one as defined in Equation (9), and the weight of l_i in the location-based feature vector $\phi_L(q,d_k)$ is incremented by one as defined in Equation (11). The similarity and parent-child relationships of the concepts in the extracted concept ontologies are also incorporated in the training based on the following four different types of relationships: **(1) Similarity**, **(2) Ancestor**, **(3) Descendant**, and **(4) Sibling**, in our ontologies. Figure 5 shows an example of the different types of parent-child relationships in our ontologies. Additionally, the ontology captures the similarity relationship. We argue that all of the above relationships may help the users to find more related information in the same class. Therefore, we assign the pre-determined weights to related concepts. The related concepts components in content and location feature vectors are thus incremented by the weights as defined in Equation (10) and Equation (12).

The extraction of content feature vector and location feature vector are defined formally as follows.

1) **Content Feature Vector**
If content concepts c_i is in a web-snippet s_k, their values are incremented in the content feature vector $\phi_C(q,d_k)$ with the following equation:

$$\forall c_i \in s_k, \phi_C(q,d_k)[c_i] = \phi_C(q,d_k)[c_i] + 1 \quad (9)$$

For other content concepts c_j that are related to the content concept c_i (either they are similar or c_j is the ancestor/descendant/sibling of c_i) in the content ontology, they are incremented in the content feature vector $\phi_C(q,d_k)$ according to the following equation:

$$\forall c_i \in s_k;\ \ \phi_C(q,d_k)[c_j] = \phi_C(q,d_k)[c_j] + sim_R(c_i,c_j) + ancestor(c_i,c_j) + descendant(c_i,c_j) + sibling(c_i,c_j) \quad (10)$$

2) **Location Feature Vector**
If location concepts l_i is in a web-snippet d_k, their values are incremented in the location feature vector $\phi_L(q,d_k)$ with the following equation:

$$\forall l_i \in d_k, \phi_L(q,d_k)[l_i] = \phi_L(q,d_k)[l_i] + 1 \quad (11)$$

For other location concepts l_j that are related to the concept l_i (l_j is the ancestor/descendant/sibling of l_i) in the location ontology, they are incremented in the location feature vector $\phi_L(q,d_k)$ according to the following equation.

$$\forall l_i \in d_i;\ \ \phi_L(q,d_k)[l_j] = \phi_L(q,d_k)[c_j] + ancestor(l_i,l_j) + descendant(l_i,l_j) + sibling(l_i,l_j) \quad (12)$$

B. Combining Weight Vectors

The content feature vector $\phi_C(q,d)$ together with the document preferences obtained from Joachims or SpyNB methods are served as input to RSVM training to obtain the **content weight vector** $\overrightarrow{w_{C,q,u}}$. The **location weight vector** $\overrightarrow{w_{L,q,u}}$ is obtained similarly using the location feature vector $\phi_L(q,d)$ and the document preferences. The two weight vectors $\overrightarrow{w_{C,q,u}}$ and $\overrightarrow{w_{L,q,u}}$ represent the content and location user profiles for a user u on a query q in our *ontology-based, multi-facet (OMF)* user profiling method.

As discussed in Section IV, the higher the personalization effectiveness parameters, $e_C(q,u)$ and $e_L(q,u)$, the better the personalization effect. To optimize the personalization effect, we use the following formula to combine the two weight vectors, $\overrightarrow{w_{C,q,u}}$ and $\overrightarrow{w_{L,q,u}}$, linearly according to the values of the personalization effectiveness parameters, $e_C(q,u)$ and $e_L(q)$, to obtain the final weight vector $\overrightarrow{w_{q,u}}$ for user u's ranking. The two weight vectors, $\overrightarrow{w_{C,q,u}}$ and $\overrightarrow{w_{L,q,u}}$, are first normalized before the combination.

$$\overrightarrow{w_{q,u}} = \frac{e_C(q,u)\cdot\overrightarrow{w_{C,q,u}}}{e_C(q,u)+e_L(q,u)} + \frac{e_L(q,u)\cdot\overrightarrow{w_{L,q,u}}}{e_C(q,u)+e_L(q,u)} \quad (13)$$

Let $e(q,u) = \frac{e_C(q,u)}{e_C(q,u)+e_L(q,u)}$, then we get the following equation from Equation (13).

$$\overrightarrow{w_{q,u}} = e(q,u)\cdot\overrightarrow{w_{C,q,u}} + (1-e(q,u))\cdot\overrightarrow{w_{L,q,u}} \quad (14)$$

After the final weight vector, $\overrightarrow{w_{q,u}}$, is computed, a linear ranking function is adopted for rank adaptation of future search results. The documents in the future search will be ranked according to the following formula:

$$f(q,d) = \overrightarrow{w_{q,u}}\cdot\phi(q,d) \quad (15)$$

where q is a query, d is a document in the search results, $\overrightarrow{w_{q,u}}$ is the weight vector defined in Equation (13), and $\phi(q,d)$ is a feature vector representing the match between query q and document d.

VII. Experimental Results

A. Experimental Setup

We developed a metasearch engine which comprises Google, MSNSearch and Yahoo as the backend search engines to ensure a broad topical coverage of the search results. The metasearch engine collects clickthrough data from the users and performs personalized ranking of the search results based on the learnt profiles of the users.

TABLE VI

TOPICAL CATEGORIES OF THE TEST QUERIES

1	Amusement Parks	6	Dining	11	Religion
2	Animation	7	Hotels	12	Sport Games
3	Charity	8	Technologies	13	Travelling
4	Courses	9	Locations	14	Video Games
5	Programming	10	Photography	15	Weather

TABLE VII

STATISTICS OF THE COLLECTED CLICKTHROUGH DATA

No. of users	50
No. of queries assigned to each user	5
No. of URLs retrieved	25,000
No. of unique URLs retrieved	21,257
No. of content concepts retrieved	31,542
No. of unique content concepts retrieved	23,147
No. of location concepts retrieved	173,366
No. of unique location concepts retrieved	5,840

50 users are invited to submit totally 250 test queries (see [1]) to our metasearch engine. For each query submitted, the top 100 search results are returned to the users. Table VI shows the topical categories of the test queries. Each of the 50 users is assigned 5 test queries randomly selected from the 15 different categories in Table VI to avoid any bias. The users are given the tasks to find results that are relevant to their interests. The clicked results are stored in the clickthrough database and are treated as positive samples in RSVM training. The clickthrough data, the extracted content concepts, and the extracted location concepts are used to create OMF profiles. The threshold for content concept is set to 0.03. A small mining thresholds is chosen because we want as many content concepts as possible that can be included in the user profiles. As discussed in Section III-B, the location concepts are prepared from [2] and [4]. They consist of 18,955 cities in 200 countries. Table VII shows the statistics of the clickthrough data collected.

In addition to the clickthrough data, the users are asked to perform *relevance judgment* on the top 100 results for each query by filling in a score for each search result to reflect the relevance of the search result to the query. The score indicates three levels of relevancy ("Good", "Fair" and "Poor"). Documents rated as "Good" are considered relevant (positive samples), while those rated as "Poor" are considered irrelevant (negative samples) to the user's needs. The documents rated as "Fair" are treated as unlabeled. Documents rated as "Good" (relevant documents) are used to compute the *average relevant rank improvements* (i.e., the difference between the average ranks of the relevant documents in the search results before and after personalization) and *top N precisions*, the two primary metrics for our evaluation.

B. Content and Location Weight Vectors

Table VIII shows an example of the content and location weight vectors learned from RSVM Training using SpyNB method. In this example, since user u_1 is interested in topics about "fruit" for the query "apple", content concepts about "fruit" (e.g., "fruit" and "orchard") receive higher weights, while content concepts unrelated to "fruit" (e.g., "apple store" and "support") receive negative weights. Moreover, location concepts (e.g., "California" and "Washington") that are well known for growing apples receive higher weights. On the other hand, since u_2 is interested in products from Apple Computer for the query "apple", the content concepts about "Apple Computer", such as "apple product" and "apple computer", receive higher weights, while unrelated concepts, such as "farm" and "tree", receive negative weights. Further, certain location concepts (i.e., "Hong Kong" and "China") receive higher weights probably due to u_2's interested in local "apple store" in Hong Kong.

TABLE VIII

EXAMPLE WEIGHT VECTORS LEARNED FOR THE QUERY "APPLE"

u_1		u_2	
Content Concept	**Weight**	**Content Concept**	**Weight**
fruit	0.721	apple product	0.314
orchard	0.193	apple computer	0.252
farm market	0.076	apple store	0.171
support	-0.015	farm	-0.021
apple product	-0.019	tree	-0.027
Location Concept	**Weight**	**Location Concept**	**Weight**
United States	0.234	Hong Kong	0.227
Washington	0.206	China	0.210
California	0.203	Singapore	0.114

We observe that u_1 and u_2 are seeking different content and location information on the same query "apple". Our experiments show that the trained concept weights can successfully capture u_1 and u_2's content and location information preferences. When the weight vectors are used in ranking the search results, results which contain the user's preferred content and location concepts (i.e., those with positive weights) will be ranked higher, while results which contain the user's less preferred concepts (i.e., those with negative weights) will be ranked lower.

Location concepts offer an additional dimension for capturing a user's interest. For example, the profiles shown in Table VIII indicate that u_1 and u_2 have quite different interests in locations (i.e., u_1 and u_2 are interested in, respectively, locations in United States and Asia). When they both issue the query "farm", u_1 will see more pages about farms in United States whereas u_2 will see more from Asia. Without the location preferences, there will not be much difference between the results that u_1 and u_2 receive since "farm" appears in both of their content preferences.

C. Entropy and Personalization Effectiveness

In this subsection, we evaluate the effectiveness of our personalization method by comparing the ranking of the relevant documents (i.e., documents rated as "Good" by the users) before and after the personalization process (i.e., the higher the average relevant rank improvement, the better the personalization effectiveness). Figure 6(a) shows the average relevant rank improvements for queries with different content and location entropies (i.e., $H_C(q)$ and $H_L(q)$) using the Joachims(Content), Joachims(Location), SpyNB(Content) and SpyNB(Location) methods. Joachims/SpyNB(Location) is the Joachims/SpyNB method employing only the location-based

features in personalization, while Joachims/SpyNB(Content) is the Joachims/SpyNB method using only the content-based features in personalization. We observe that the higher $H_C(q)$ and $H_L(q)$ are, the better the personalization effectiveness because queries with higher content or location entropy have a higher degree of ambiguity on content or location information, giving a larger room for personalization methods to improve the search results.

Figure 6(b) shows the average rank improvement for queries with different click content and location entropies (i.e., $H_{\overline{C}}(q,u)$ and $H_{\overline{L}}(q,u)$. The result aligns with our expectation that queries with low $H_{\overline{C}}(q,u)$ and $H_{\overline{L}}(q,u)$ benefit more from personalization because users are focused on a few specific topic in the search results, making it possible for the search engine to learn and adapt the search results towards those topics.

Finally, we present the average rank improvement on queries based on sorted personalization effectiveness (i.e., $e_C(q,u)$ and $e_L(q,u)$) in Figure 6(c). As shown, queries with high $e_C(q,u)$ and $e_L(q,u)$ yield better personalization results. This result validates our proposal of $e_C(q,u)$ and $e_L(q,u)$ as an accurate measurement of the personalization effectiveness for a user's query.

D. Evaluation of Ranking Quality

In the evaluation of the ranking quality of the personalization method, we use the ranked results returned by the backend search engines (i.e., Google, MSN and Yahoo) as the baseline. The top N precisions grouped by query classes for the baseline method are shown in Figures 7 and 8. Several observations about the baseline method can be made. First, the baseline method is good for explicit queries, which is expected to have good performance because they are very focused. Second, it has very poor precisions for ambiguous queries, showing that general search engines by design do not handle the ambiguity of the queries well. Finally, its precisions for content and location queries are slightly better than the precisions on ambiguous queries. These observations show that the commercial search engines perform well for explicit queries but suffer in various degrees for vague queries.

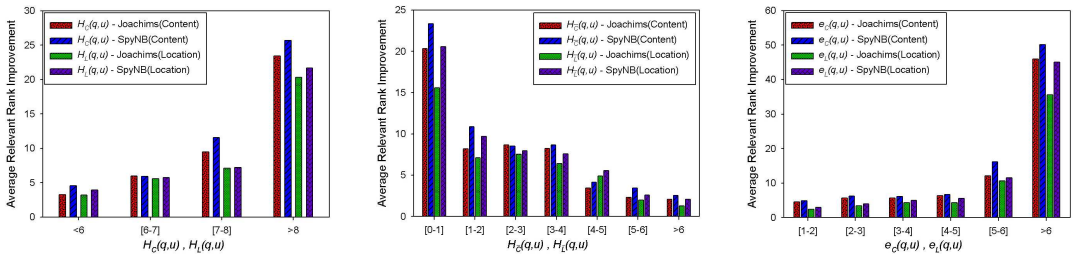
Figure 7 shows the top N precisions grouped by query classes for our personalization approach using Joachims' methods for preference extraction. Joachims(Location) uses only the location-based features in our personalization method, while Joachims(Content) uses only the content-based features. Joachims(e) employs *both* the content-based and location-based features, weighted by their personalization effectiveness (see Equation (14)).

We observe that Joachims(Content) method performs the best on content queries. It boosts the top 1, 10, and 20 precisions of content queries from 0.4583, 0.3563, and 0.3125 to 0.7519, 0.5874, and 0.4176 (64%, 65%, and 34% in percentage gain), comparing to the baseline method. The top 1, 10, and 20 precisions for ambiguous queries are also improved significantly, boosted from 0.3519, 0.2815, and 0.2528 of the baseline method to 0.6583, 0.5396, and 0.4083 (87%, 92%, and 62% in percentage gain). Joachims(Content) method performs fine on location queries, because location queries also contain a certain amount of content information. It improves the top 1, 10, and 20 precisions of location queries from 0.5208, 0.4063, and 0.3563 of the baseline method to 0.6967, 0.5774, and 0.4398 (33%, 42%, and 23% in percentage gain). Finally, as expected, the precisions are the best for explicit queries. However, the improvement is not as significant as in other query classes because the baseline method already performs reasonably well for explicit queries. The top 1, 10, and 20 precision are only improved from 0.6289, 0.4398, and 0.3801 to 0.7942, 0.6083, and 0.4781 (26%, 38%, and 26% in percentage gain).

On the other hand, Joachims(Location) method performs the best on location queries, boosting the top 1, 10, and 20 precisions of location queries from 0.5208, 0.4063, and 0.3563 to 0.6989, 0.4269, and 0.3583 (34%, 5%, and 0.5% in percentage gain). The top 1, 10, and 20 precisions for ambiguous queries are also improved from 0.3519, 0.2815, and 0.2528 to 0.5000, 0.3604, and 0.2804 (42%, 28%, and 10% in percentage gain). The performance of Joachims(Location) method is not good for explicit and content queries, because only a limited amount of location information exists in those queries. We observe that Joachims(Content) performs better than Joachims(Location) in general, showing that content information is an important factor in the personalization.

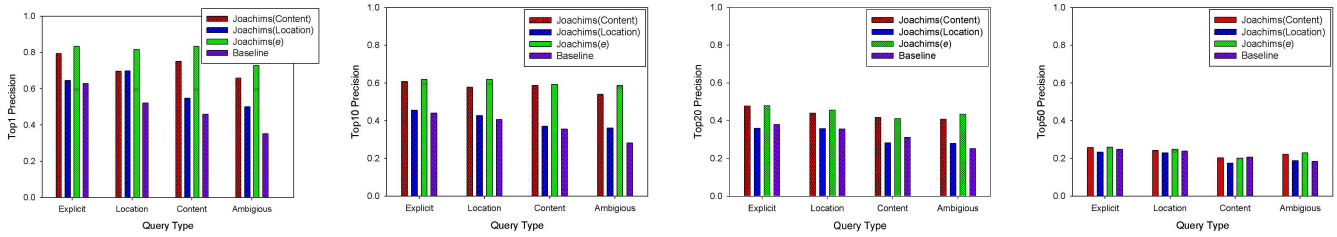
Even though Joachims(Location) by itself does not perform as well as Joachims(Content), it does provide additional improvement for personalization. We observe that when both location concepts and content concepts are employed in our personalization method (denoted by Joachims(e)), the precisions are further improved. This shows that both of the content and location information are useful in the personalization process. Moreover, Joachims(e) method can greatly improve the precisions of all four classes of queries. Joachims(e) method yields a top 1 precision of around 0.8 for explicit, content and location queries. The top 1 precision of Joachims(e) method for ambiguous queries is only slightly lower at 0.7292, which is a 110% improvement over the baseline method.

Figure 8 shows the top N precisions grouped by query classes for our personalization approach using SpyNB for preference extraction. The observation of our personalization method using SpyNB is consistent as above (i.e., using the Joachims methods), providing a validation of our prototype implementations. It also shows that the correct use of content and location user profiles help boosting the performance of our personalization strategy. By comparing Figure 7 and Figure 8, we found that using SpyNB for preference extraction performs better than using Joachims' method in all classes of queries, because SpyNB generates more accurate preferences comparing to Joachims' method. We also observe that SpyNB(e) performs the best among all the methods. It achieves top 1, 10, 20 precisions of 0.9375, 0.6542, and 0.5042, respectively, showing that SpyNB(e) can successfully push the relevant results to the top of the result list according to the user's personal content and location information preferences.



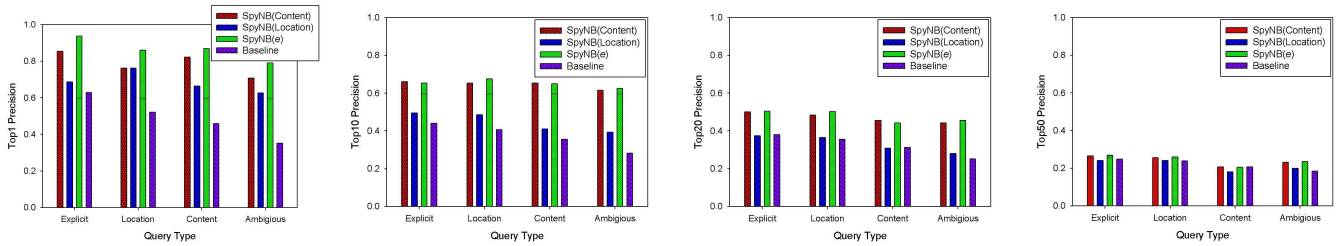
(a) $H_C(q)$ and $H_L(q)$ (b) $H_{\overline{C}}(q,u)$ and $H_{\overline{L}}(q,u)$ (c) $e_C(q,u)$, and $e_L(q,u)$

Fig. 6. Average relevant rank improvement with varying (a) $H_C(q)$ and $H_L(q)$, (b) $H_{\overline{C}}(q,u)$ and $H_{\overline{L}}(q,u)$, (c) $e_C(q,u)$, and $e_L(q,u)$.



(a) Top 1 (b) Top 10 (c) Top 20 (d) Top 50

Fig. 7. Top 1, 10, 20, and 50 precisions for Joachims(Content), Joachims(Location), Joachims(e) and Baseline methods with different query classes.



(a) Top 1 (b) Top 10 (c) Top 20 (d) Top 50

Fig. 8. Top 1, 10, 20, and 50 precisions for SpyNB(Content), SpyNB(Location), SpyNB(e) and Baseline methods with different query classes.

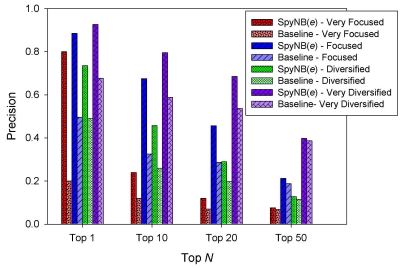


Fig. 9. Top 1, 10, 20, and 50 precisions for SpyNB(e) with different type of users.

Figure 9 shows the top N precisions for the baseline method and SpyNB(e) for different user classes. We observe that the baseline method yields high precisions for Very Diversified users. This is because during relevance judgment they judged many documents as "Good" as a result of their broad interests, thus making it easier for the search results to achieve high precisions. However, the baseline method yields low precisions for Very Focused users because they have very specific needs and would only select some truly relevant results. Finally, the precisions of the baseline method for the Focused and Diversified users fall somewhere between the two extremes.

SpyNB(e) method also yields the best precisions for Very Diversified users, because the baseline method has already performed very well in this user class and a slight boost in precision by SpyNB(e) makes it better than all other user classes. However, the percentage gain obtained is not as impressive as in the Very Focused user class. SpyNB(e) boosts the top 1, 10, and 20 precisions of Very Diversified users from 0.6764, 0.5882, and 0.5360 of the baseline method to 0.9265, 0.7956, and 0.7470 (37%, 35%, and 34% in percentage gain), whereas it boosts the top 1, 10, and 20 precisions of the Very Focused user class from 0.2000, 0.1200, and 0.0800 of the baseline method to 0.800, 0.2400, and 0.1600 (300%, 100%, and 100% in percentage gain). This conforms with our expectation that Very Focused users are expected to have more significant gain of precisions through personalization compared to the other user classes. Finally, the percentage gain of precisions of Focused users is slightly more than that of Diversified users. The percentage gain of the top 1, 10 and 20 precisions are 78%, 107% and 81% for Focused users, and 50%, 80%, and 70% for Diversified users.

E. Evaluation of the Estimated Combination Threshold $e(q,u)$

In Equation (14), we define $e(q,u)$ to combine the content weight vector $\overrightarrow{w_{C,q,u}}$ and the location weight vector $\overrightarrow{w_{L,q,u}}$. To find the optimal combination threshold $oe(q,u)$ (i.e., the optimal value of $e(q,u)$), we repeat the experiment to find the precisions for each query by setting $e(q,u) \in [0,1]$ in 0.05 increments. The $oe(q,u)$ value is then obtained as the the value that results in the highest top N precision. In this subsection,

TABLE IX

AVERAGE $e(q,u)$, $oe(q,u)$, AND ERROR

	Avg. $e(q,u)$	Avg. $oe(q,u)$	Avg. Error
Joachims(e)	0.4789	0.4777	0.1670
SpyNB(e)	0.4789	0.4754	0.1642

TABLE X

TOP N AVERAGE PRECISIONS FOR JOACHIMS(e/oe) AND SPYNB(e/oe)

Joachims(e)	$oe(q,u)$	$e(q,u)$	$\frac{e(q,u)}{oe(q,u)}$
Top 1 Avg. Precision	0.8354	0.8066	0.9655
Top 10 Avg. Precision	0.6267	0.6070	0.9685
Top 20 Avg. Precision	0.4609	0.4467	0.9692
Top 50 Avg. Precision	0.2407	0.2360	0.9805
SpyNB(e)	$oe(q,u)$	$e(q,u)$	$\frac{e(q,u)}{oe(q,u)}$
Top 1 Avg. Precision	0.8765	0.8642	0.9859
Top 10 Avg. Precision	0.6881	0.6560	0.9533
Top 20 Avg. Precision	0.4942	0.4800	0.9713
Top 50 Avg. Precision	0.2461	0.2444	0.9933

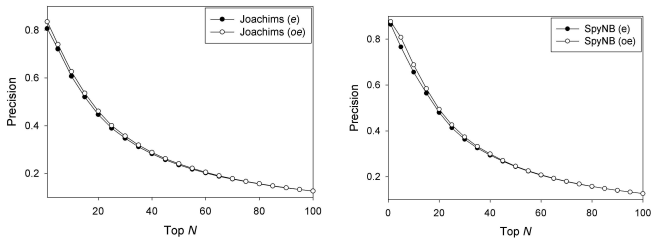
we evaluate the performance of the estimated $e(q,u)$ by comparing it against the optimal combination threshold $oe(q,u)$. Moreover, we evaluate the retrieval effectiveness of the two combination thresholds ($e(q,u)$ and $oe(q,u)$) by analyzing the top N average precisions achieved by them.

Table IX shows the average $e(q,u)$ and $oe(q,u)$ values obtained from all queries. The average $e(q,u)$ and $oe(q,u)$ are very close to each other in both Joachims(e) ($e(q,u) = 0.4789$ and $oe(q,u) = 0.4777$) and SpyNB(e) ($e(q,u) = 0.4789$ and $oe(q,u) = 0.4754$) methods. The average error is only 0.16 for both methods. Moreover, notice that all the combination thresholds ($e(q,u)$ and $oe(q,u)$) are close to 0.5, showing that the content preferences $\overrightarrow{w_{C,q,u}}$ and the location preferences $\overrightarrow{w_{L,q,u}}$ are both very important for determining users preferences in personalization.

Table X shows the top 1, 10, 20, 50 average precisions of Joachims(e) and SpyNB(e) methods using $e(q,u)$ and $oe(q,u)$, where $\frac{e(q,u)}{oe(q,u)}$ is the fraction of precision that $e(q,u)$ can be obtained comparing to the optimal $oe(q,u)$. We observe that $e(q,u)$ is a very good estimation comparing to the optimal $oe(q,u)$, because $\frac{e(q,u)}{oe(q,u)} > 96\%$ shows that $e(q,u)$ is performing almost the same as $oe(q,u)$. Figures 10(a) and 10(b) show the top N precisions obtained using Joachims(e) and SpyNB(e) (using $e(q,u)$ and $oe(q,u)$). Again, we can see that $e(q,u)$ is a very good approximation of $oe(q,u)$, as the two plots ($e(q,u)$ and $oe(q,u)$) are very close to one another.

VIII. CONCLUSIONS

In this paper, we proposed an *Ontology-Based*, *Multi-Facet (OMF)* personalization framework for automatically extracting and learning a user's content and location preferences based on the user's clickthrough. In the OMF framework, we develop different methods for extracting content and location concepts, which are maintained along with their relationships in the content and location ontologies. We also introduced the notion of content and location entropies to measure the diversity of content and location information associated with a query and and click content and location entropies to capture the breadth of the user's interests in these two types of information. Based on the entropies, we derived personalization effectiveness and showed with a case study that personalization effectiveness differs for different classes of users and queries. Experimental results confirmed that OMF can provide more accurate personalized results comparing to the existing methods.



(a) Joachims(e) vs Joachims(oe). (b) SpyNB(e) vs SpyNB(oe).

Fig. 10. Top N precisions for Joachims(e) and SpyNB(e) methods with $e(q,u)$ and $oe(q,u)$.

As for the future work, we plan to study the effectiveness of other kinds of concepts such as people names and time for personalization. We will also investigate methods to exploit a user's content and location preference history to determine regular user patterns or behaviors for enhancing future search.

ACKNOWLEDGMENT

This work was supported by grants 615707 and CA05/06.EG03 from Hong Kong RGC. Wang-Chien Lee was supported in part by the National Science Foundation under Grant no. IIS-0534343 and CNS-0626709.

REFERENCES

[1] Appendix. http://www.cse.ust.hk/~dlee/icde10/appendix.pdf.
[2] National geospatial. http://earth-info.nga.mil/.
[3] svm^{light}. http://svmlight.joachims.org/.
[4] World gazetteer. http://www.world-gazetteer.com/.
[5] E. Agichtein, E. Brill, and S. Dumais, "Improving web search ranking by incorporating user behavior information," in *Proc. of ACM SIGIR Conference*, 2006.
[6] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proc. of ICML Conference*, 2005.
[7] K. W. Church, W. Gale, P. Hanks, and D. Hindle, "Using statistics in lexical analysis," *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, 1991.
[8] Q. Gan, J. Attenberg, A. Markowetz, and T. Suel, "Analysis of geographic queries in a search engine log," in *Proc. of the International Workshop on Location and the Web*, 2008.
[9] T. Joachims, "Optimizing search engines using clickthrough data," in *Proc. of ACM SIGKDD Conference*, 2002.
[10] K. W.-T. Leung, W. Ng, and D. L. Lee, "Personalized concept-based clustering of search engine queries," *IEEE TKDE*, vol. 20, no. 11, 2008.
[11] B. Liu, W. S. Lee, P. S. Yu, and X. Li, "Partially supervised classification of text documents," in *Proc. of ICML Conference*, 2002.
[12] W. Ng, L. Deng, and D. L. Lee, "Mining user preference using spy voting for search engine personalization," *ACM TOIT*, vol. 7, no. 4, 2007.
[13] C. E. Shannon, "Prediction and entropy of printed english," *Bell Systems Technical Journal*, pp. 50–64, 1951.
[14] Q. Tan, X. Chai, W. Ng, and D. Lee, "Applying co-training to clickthrough data for search engine adaptation," in *Proc. of DASFAA Conference*, 2004.
[15] S. Yokoji, "Kokono search: A location based search engine," in *Proc. of WWW Conference*, 2001.
[16] Y. Zhou, X. Xie, C. Wang, Y. Gong, and W.-Y. Ma, "Hybrid index structures for location-based web search," in *Proc. of CIKM Conference*, 2005.