Fine-Grained Preference-Aware Location Search Leveraging Crowdsourced Digital Footprints from LBSNs

Dingqi Yang†, Daqing Zhang†, Zhiyong Yu†, Zhiwen Yu‡
† Institut Mines-Télécom/Télécom SudParis, CNRS UMR 5157 SAMOVAR, Evry, France
‡ Northwestern Polytechnical University, Xi'an, Shaanxi, China
{dingqi.yang, daqing.zhang, zhiyong.yu}@it-sudparis.eu, zhiwenyu@nwpu.edu.cn

ABSTRACT

The crowdsourced digital footprints from Location Based Social Networks (LBSNs) contain not only rich information about locations, but also individual's feeling about locations and associated entities. This new data source provides us with an unprecedented opportunity to massively and cheaply collect location related information, and to subtly characterize individual's fine-grained preference about those places and associated entities. In this paper, we propose SEALs - a fine-grained preference-aware location search framework leveraging the crowdsourced traces in LBSNs. We first collect user check-ins and tips from Foursquare and use them as direct user feedback on locations. Second, we extract users' sentiment about locations and associated entities from tips to characterize their fine-grained location preference. Third, we incorporate such fine-grained user preference into personalized location ranking using tensor factorization techniques. Experimental results show that SEALs can achieve better location ranking comparing to the state-of-the-art solutions.

Author Keywords

Personalized Location Search; Crowdsourcing; Location
Based Social Networks; Fine-Grained User Preference;
Sentiment Analysis; Tensor Factorization

ACM Classification Keywords

J.4 Computer Applications: Social and Behavioral Sciences;
H.3.3 Information Search and Retrieval: Search Process

General Terms

Algorithms, Design, Experimentation

INTRODUCTION

With the ubiquity of GPS-equipped smartphones, location based services have gained great popularity in recent years. Typical location based services include location search [6], location recommendation [27], mobile navigation [23], etc. As a typical location based service, location search allows users to search places using keywords, e.g., finding a "burger" restaurant or an "art museum" in a particular city. Building such a location search service usually needs to solve two problems: 1) how to collect location related information to ensure a good coverage of interesting places and characterization of those places; 2) how to rank locations according to users' search objectives.

Location related information is usually collected from various data sources by commercial location search service providers. For example, Google Maps builds its location database by combining information sources[1] such as web search results, data submitted directly by local business owners, user submitted photographic content, street view imagery and third party sources (e.g. the Yellow Pages[2]). However, collecting and compiling such a usable location database might incur substantial cost. In contrast, the booming of Location Based Social Networks brings us a new opportunity to collect rich location related data at a low cost [15]. In LBSNs such as Foursquare[3], users explore and tag places, post reviews, upload photos, and share locations and experiences with others. *Check-ins* are performed at physical locations (i.e., venues[4]), such as museums, restaurants, or bars; user comments are termed *tips*, containing rich information about individual's opinions about the check-in places or the associated entities. For example, a tip left at a restaurant may recommend a special dish or give positive/negative comments about the restaurant environment. The crowdsourced digital footprints [26] from Foursquare users provide us unprecedented opportunities to massively collect information about locations as well as users' sentiment about the locations and entities, which are the key ingredients for enabling personalized location search.

Location ranking in location search services is usually performed based on the relevance between the query keywords and venues under certain geographical constraints. With continuous expansion of location database, users may be returned a long list of venues for each query, causing difficulty for users to select their interested places. In addition, due to the limited screen size of mobile devices and user's unwillingness to scroll beyond the first page, the search results at the top of the list become crucial. Thus, a well-designed location search service should consider users' needs and put one's desired venues at the top of the returned venue list.

UbiComp'13, September 8–12, 2013, Zurich, Switzerland.
Copyright © 2013 ACM 978-1-4503-1770-2/13/09...$15.00.
http://dx.doi.org/10.1145/2493432.2493464

[1] http://support.google.com/maps/bin/answer.py?hl=en&answer=7103

[2] http://www.yellowpages.com/

[3] https://foursquare.com/

[4] here "venue" refers to a logical "location" in Foursquare, we do not differentiate these two terms throughout this paper

Personalized location search is usually fulfilled from two perspectives in current literature, i.e., via context-awareness and preference-awareness. The *context-aware location search* leverages user's context, e.g., current time, location, weather condition, user's activity, to augment the search queries and deliver the appropriate search results to users. For example, a query of looking for a burger restaurant can be interpreted as finding "where is the *nearest* burger restaurant" (i.e., location context is taken into account). The *preference-aware location search* provides results according to the individual's preference about venues. The same query above may be interpreted as asking "where is the burger restaurant serving *my favorite taste of burgers*". Most of the research efforts on location search personalization focus on search according to context. Even though there are less efforts focused on preference-aware location search, as a special type of information in the web, locations can be retrieved using personalized web search approaches. Web search personalization is an extensively studied topic where user preference has been widely used to enhance user's search experience. In Web search, preference-aware approaches usually provide users with a personalized list of results using certain ranking algorithm by incorporating user preference which is mainly extracted from users' historical search records.

In this paper, we present SEALs (Sentiment-EnhAnced Location search), a fine-grained preference-aware location search framework leveraging the crowdsourced data in LBSNs. In particular, we exploit or introduce unique features in three key phases of the preference-aware location search scheme, i.e., in *user feedback capture*, *user preference modeling* and *search result ranking*. The main contributions of this work can be roughly summarized as follows:

1) *Collecting users' direct feedback on venues from LBSNs*. Users' interaction in LBSNs can be regarded as user feedback on locations. Different from the classical user feedback on web based location search (e.g., click-through data, browsing history, past queries), the user feedback from LBSNs is direct and more precise. For example, a user intends to search a French restaurant in New York, clicking on one restaurant's website does not indicate that she would go to that place. Even if she goes to the restaurant later, this might not necessarily mean that she would like the restaurant. However, in LBSNs, users physically visit and leave comments directly on the venues. Thus collecting the user feedback would better characterize users' actual feeling about the venues and what entities users like/dislike at those venues.

2) *Modeling fine-grained user preference extracted from heterogeneous user feedback in LBSNs.* User generated traces at venues in LBSNs are usually heterogeneous, including check-ins, tags in terms of keywords, and tips in the form of short text. These crowdsourced contents imply user preference at different granularity levels. For example, a user named Jane who often visits a restaurant (check-in there) usually implies that she has positive preference about the restaurant. This type of preference describes a user's general and broad feeling, which can be treated as *coarse-grained user preference*. In LBSNs such as Foursquare, users may give very detailed comments about a venue or even entities at a venue. For example, Jane left a tip saying "I like cheese burgers at restaurant X, but not the beer there". By applying sentiment analysis on this tip, we can extract her positive preference for "cheese burgers" and negative preference for "beer". This type of preference indicates a user's different feeling about different items at a venue (in the form of user-keyword-venue), which can be defined as *fine-grained user preference*. Apparently, *fine-grained user preference* contains more precise and detailed information about a location comparing to *coarse-grained user preference* (in the form of user-venue). It provides us with new possibilities to rank locations more accurately according to one's preference.

3) *Incorporating fine-grained user preference into personalized location ranking using tensor factorization techniques*. Preference-aware search personalization approaches can be roughly classified into three categories based on how user preference is used. (i) User preference is used to augment the submitted query. In this case, user preference and the original query keywords are put together to feed into the traditional search engine, thus user preference does not directly affect the ranking process in the way to bring user preferred venues to the top of the list. (ii) User preference is used to re-rank the search results. Approaches in this category first employ keyword-venue relation to find relevant venues and then leverage user-venue relation to re-rank venues. They fail to consider user-keyword-venue relation simultaneously in the search process. (iii) The *fine-grained user preference* in the form of user-keyword-venue is incorporated into the location search and ranking process. The most popular approach is based on tensor factorization. Concretely, a three-way tensor is adopted to model the *fine-grained user preference*. In order to handle both positive and negative preference, we propose a novel Multi-Tuple based Ranking Tensor Factorization (MT-RTF) algorithm for personalized location ranking.

RELATED WORK

Personalized location search mainly employs user specific information such as user context and user preference, to provide customized search results. Most of the existing personalized location search approaches exploit user context. For example, with consideration of user's current location, Choi [4] utilized fuzzy query techniques to re-rank the search results. Leveraging user's current location and time, Waga et al. [22] built a location search system using context-aware recommendation techniques. Iwata et al. [5] extracted user's situation, e.g., being in the office at lunch time on weekdays, or going downtown on holiday, to perform personalized search. Lane et al. [6] proposed a framework that considered rich context such as weather and activity. Even though few studies have been conducted for preference-aware location search, as a special type of information in the web, locations can be retrieved by leveraging the common personalized web search approaches using user preference.

Based on how user preference is obtained, preference-aware search approaches can be roughly classified into two categories. The first approach leverages user explicit feedback, i.e., let user explicitly state their preference. However, users usually do not want to spend extra efforts on providing such feedback [1]. The second approach uses implicit feedback. Since it can be collected without extra user efforts, they are widely used in search personalization to extract user preference. The classical implicit feedback sources include browsing history [19], click-through data [10] and user personal information (e.g., email, desktop data) [3]. Recently, the booming of social network brings an opportunity for collecting user direct feedback. In social network sites, users add tags to items (e.g., photos, video, blogs, locations) and/or make comments. Users' interaction in such crowdsourcing platform usually implies their preference that can be used in search personalization [2, 14, 24, 28]. In this work, we explore preference-aware search using crowdsourcing data as user direct feedback.

Preference-aware search personalization approaches can be divided into three categories according to how user preference is used. First, query expansion techniques using user preference usually aim at augmenting user submitted query with keywords associated with user preference. Using crowdsourcing data from the social bookmarking web service Delicious[5], Zhou et al. [28] extended original query using user profile extracted from one's social annotation history. The approaches in this category usually have limited personalization effect due to the lack of user preference in ranking process. Second, user preference can be used to re-rank the search results generated from a non-personalized search engine. For example, Xu et al. [24] extracted users' interests from social annotation data and ranked documents according to both query-document relevance and the similarity between users' interests and documents' topics. By constructing user profiles and resource profiles, Cai et al. [2] first modeled query-document relevance and then leveraged user-document preference to adjust the result ranking. These works all separated query-document relevance ranking and user-document preference ranking, and then merged them together. Although the approaches in the first two categories leverage *coarse-grained user preference*, i.e., user-keyword preference and user-document preference, they do not handle *fine-grained user preference*. The third scheme leverages user preference in the document indexing and searching process. The most popular approach supporting this scheme is tensor factorization. In web search, Sun et al. [20] conducted an early work by modeling click-through data as a three-way tensor and then using High Order Singular Value Decomposition (HOSVD) techniques to factorize the built tensor for personalized ranking. Since HOSVD cannot practically handle sparse tensors, Rendle et al. [11] proposed a ranking with tensor factorization approach to specifically address element ranking problem in tensor, which can alleviate the problem of sparsity. Sang et al. [14] proposed a multi-correlation ranking approach in tensor along with a user-specific topic modeling to personalize image search using social annotation data on Flickr[6]. Since ranking tensor factorization process is usually time-consuming, Rendle et al. [13] designed a pairwise interaction tensor factorization model which dramatically reduced the learning time while maintaining the performance. Shi et al. [17] addressed the top-N context-aware recommendation problem by leveraging tensor factorization to maximize mean average precision. However, existing tensor factorization approaches can merely handle positive preference. As fine-grained preference might include both positive and negative preference, we propose Multi-Tuple based Ranking Tensor Factorization to consider both positive and negative preference simultaneously in the factorization process.

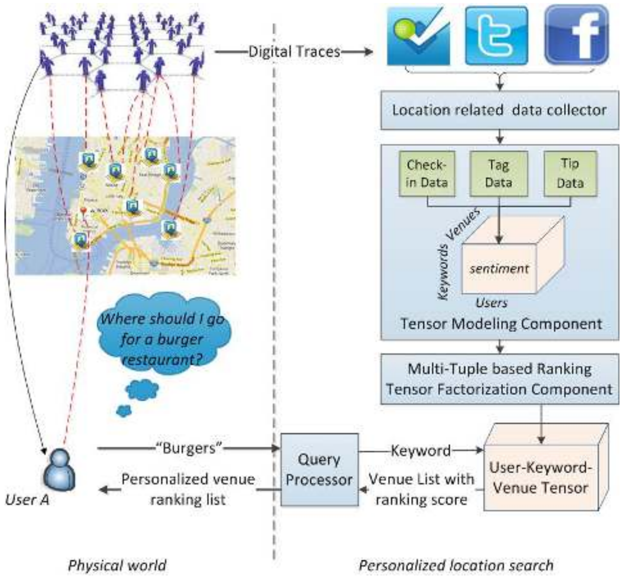

Digital Traces
Location related data collector
Check-in Data
Tag Data
Tip Data
Venues
Keywords
sentiment
Users
Tensor Modeling Component
Multi-Tuple based Ranking Tensor Factorization Component
Where should I go for a burger restaurant?
"Burgers"
Query Processor
Keyword
User-Keyword-Venue Tensor
Personalized venue ranking list
Venue List with ranking score
User A
Physical world
Personalized location search

Figure 1. Overview of SEALs Framework

PROBLEM DESCRIPTION AND SEALS FRAMEWORK

In general, for a given query, personalized search tries to provide customized search results for different users based on their preference. User preference is usually extracted from their historical behaviors. The problem of targeted location search based on users' interaction in LBSNs can be described as: *Given users' interaction history with locations, i.e., the places users visited and the comments users left, for each query, the objective of personalized location search is to generate a customized ranking list of venues for different users, i.e., top ranked results include user liked venues while exclude disliked ones.* In this paper, we extract user preference from digital footprints in LBSNs. Along with venue information (venue tags), we intend to generate personalized ranking of venues for each user. Figure 1 illustrates the overview of SEALs framework. The left panel shows users' activities in physical world while the right part presents the SEALs components. The location related data collector is in charge of gathering raw data from LBSNs. The tensor modeling component then constructs a user-keyword-venue tensor by extracting fine-grained user preference from raw data. Afterwards, the Multi-Tuple based Ranking Tensor Factorization component decomposes the tensor in order to reveal the latent correlation among the three factors, i.e., user, keyword, and venue, resulting in a fulfilled tensor with the predicted ranking score for all user-keyword-venue triplets. When a

[5] http://www.delicious.com/

[6] https://www.flickr.com/

user looks for places using a keyword, the query processor returns the personalized ranking results.

DATA COLLECTION

We first collect raw check-ins and filter the noise ones from Foursquare (check-ins from 24 October 2011 to 20 February 2012). Then we crawl tips and tags from the refined venues and users. Because of privacy policy in Foursquare, personal check-in information cannot be accessed by public. However, in Foursquare, some users choose to post their check-ins via Twitter. Therefore we can capture check-ins by crawling foursquare-tagged tweets from Twitter Public Stream[7]. For the purpose of filtering out noisy check-ins, we first select users who have performed at least one check-in per week and then eliminate users who have ever performed "sudden-move" check-ins (consecutive check-ins which correspond to a speed faster than 1200 km/h: the common airplane speed). Details about the dataset collection process and statistics can be found in our previous work [25].

In this study, due to implementation limitation of sentiment analysis, we only process tips in English but the proposed framework is not bound to one language. We select one large city in an English-speaking country, i.e., New York, and then extract tips and tags of the venues in this city. Venues in Foursquare are classified into nine categories (i.e., Arts & Entertainment, College & University, Food, Great Outdoors, Nightlife Spot, Professional & Other Places, Residence, Shop & Service, Travel & Transport) and 400 subcategories[8]. Similar to [15], we focus on restaurant search to verify the key idea of SEALs. Thus, we select food related venues ("Food" category, containing 87 sub-categories such as French restaurant, Italian restaurant, etc.) for evaluation. In this work, the following three types of Foursquare data is collected for user preference extraction:

- Check-in Data: they are in the form of user-venue pairs. A user-venue pair indicates that the user visits the venue once.
- Tag Data: they are represented by venue-tags pairs where tags are a set of keywords used to label and index the venue.
- Tip Data: they are defined as user-venue-tip triplets indicating that the user leaves the tip at the venue.

MODELING USER PREFERENCE BY TENSOR

Figure 2 demonstrates the tensor modeling process. First, we extract user preference for entities (in the form of keywords) on a venue (denoted as *u-k-v* preference) from two data sources, i.e., check-ins and tips, separately. Then, we propose a fusion method to merge the two sets of preference. In Foursquare, tags[9] can be used to index venues and added by users without any constraint. Thus, they might be diverse and sometimes odd, or not even a correctly spelled word. To avoid uncommon tags, we only choose the tags that have been used at least twice in all venues as keywords. The tag cloud is shown in Figure 3. Most of the popular tags that are usually used to describe restaurants (e.g., coffee, brunch, bar, burgers, pizza, etc.) are shown as expected.

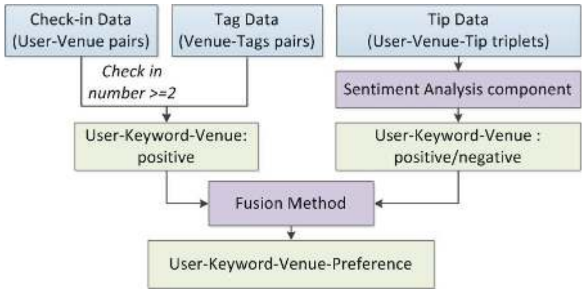


Figure 2. User-Keyword-Venue Tensor Modeling

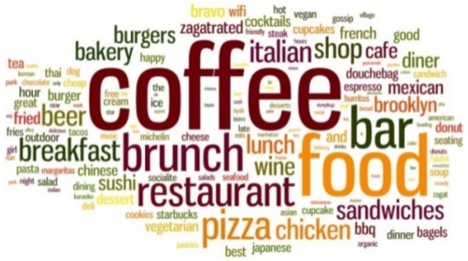


Figure 3. Tag cloud of Keywords in the Dataset (Larger font size implies higher frequency, and vice versa.)

User Preference from Check-ins

A user may not necessarily like a venue if she has visited there only once, while repeated visits usually indicate she likes the place, i.e., expressing positive feedback to this venue. Based on this common sense, for each user, we select the venues being checked-in at least twice as her liked places. Since check-ins cannot provide further information on what items the users like or dislike on a venue, we assume that users have only positive feedback to all keywords at this venue. Hence, we get a set of *u-k-v* triplets with positive preference.

User Preference from Tips

Tips left by users on venues usually describe what users like or what users complain. By applying aspect-based sentiment analysis techniques, we can extract users' different opinions on different items/aspects. In this work, dictionary based unsupervised sentiment analysis method is used to process tips for the sake of its simplicity in implementation. More sophisticated sentiment analysis techniques can also be applied to improve the performance, but they are not the focus of this paper. Figure 4 demonstrates the sentiment analysis process of tips. The left part shows the processing workflow. The right part illustrates the sentiment analysis results of an example tip about an Italian restaurant.

In our study, we merely deal with tips in English. The language detection component firstly filters out non-English tips. We use a language detection library developed by Cybozu Labs[10]. Then tips are split into sentences and identified the part-of-speech for each word, e.g., "good" is an adjective,

[7] https://dev.twitter.com/docs/streaming-apis/streams/public

[8] https://developer.foursquare.com/docs/venues/categories

[9] Since tags are used in Foursquare to index venues, we do not differentiate it with the term "keywords" throughout this paper.

[10] http://developer.cybozu.co.jp/archives/oss/2010/10/language-detect.html

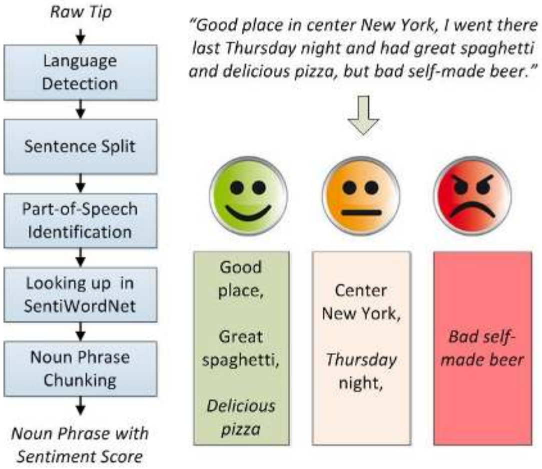

Raw Tip
Language Detection
Sentence Split
Part-of-Speech Identification
Looking up in SentiWordNet
Noun Phrase Chunking
Noun Phrase with Sentiment Score
"Good place in center New York, I went there last Thursday night and had great spaghetti and delicious pizza, but bad self-made beer."
Good place,
Great spaghetti,
Delicious pizza
Center New York,
Thursday night,
Bad self-made beer

Figure 4. Sentiment Analysis of Tips

"place" is a noun. Afterwards, we can obtain a sentiment score for each word referring to SentiWordNet [18] with the corresponding part-of-speech type. The positive, zero and negative values of the sentiment score indicate the positive, neutral and negative sentiment, respectively. Noun-Phrase Chunking is then performed to get the phrases e.g., "good place", "delicious pizza", which describe what users like or dislike at a venue. By summing up the sentiment score of each word in a phrase, we obtain the sentiment of the phrase. We adopt NLTK toolkit [9] for implementation. The output of sentiment analysis on tips is a set of user-phrase-venue triplets with the corresponding sentiment score. In order to map a phrase to keywords, we simply find out the keywords (i.e., tags) contained in a phrase. For example, the phrase "delicious pizza" is mapped to keywords "delicious" and "pizza" if they exist in the keyword set. Then, we can get a set of *u-k-v* triplets with positive or negative preference.

To test the accuracy of our sentiment analysis method, we randomly choose 1000 tips and manually label their fine-grain preference. The experiments give a precision of 63.91% and a recall of 88.13%. More sophisticated methods can be used to achieve better performance. For example, using a domain-dependent sentiment lexicon may generate more appropriate sentiment scores [8] and using sequence classifier such as Conditional Random Field can better handle adversative conjunctions [16].

Fusion Method

From the check-in data, only positive preference of keywords can be extracted, while from tip data both positive and negative preference can be extracted. The user preference extracted from tips is fine-grained and contains more precise information. Hence, the fusion policy is: when the same *u-k-v* triplet is observed from both data sources, the preference from tips analysis is used. For example, a user checked in twice at a restaurant (tagged by burgers, pizza and beer) and left a tip complaining about the burgers there. The preference extracted from her check-in for burgers, pizza, beer in that restaurant is positive while the tip reports negative preference for burgers there. The preference extracted from tips is considered to be more accurate. Hence, the user preference for burgers in that place is negative, and user preference for pizza and beer remains positive. Finally, to assign tensor with a score, we assign 1 and -1 to positive and negative preference, respectively. The unknown ones are assigned 0.

MT-RTF ALGORITHM

The purpose of MT-RTF algorithm is to rank venues in the order of user preferred, with unknown preference, and with negative preference. To achieve this goal, MT-RTF algorithm predicts the ranking of user preference for venues in tensor. First, we select an appropriate tensor factorization model. Then, based on this model we define an objective function which measures the multi-tuple ranking quality. Finally, we extend the learning framework in [12] to maximize the objective function in the learning process.

Tensor Factorization Model

Tensor factorization techniques intend to decompose a tensor into multiple factors. For the *u-k-v* tensor, let $\hat{U}$, $\hat{K}$ and $\hat{V}$ denote the user, keyword and venue feature matrices, with dimension of $|U| * l$, $|K| * l$ and $|V| * l$, respectively. Note that l is called latent space dimension (or factorization dimension) which is the most important parameter in tensor factorization. It controls the number of features used in the factorization process. The U, K, V are finite sets of users, keywords and venues, respectively. The decomposition can be formulated as:

$$\hat{Y} = \hat{C} \times_U \hat{U} \times_K \hat{K} \times_V \hat{V} \tag{1}$$

where $\times_n$ is the mode-n tensor product with matrix. The core tensor $\hat{C}$ with dimension $l * l * l$ handles the correlation among different factors. The value of each element in $\hat{Y}$ is calculated as:

$$\hat{y}_{u,k,v} = \sum_{\tilde{u}} \sum_{\tilde{k}} \sum_{\tilde{v}} \hat{c}_{\tilde{u},\tilde{k},\tilde{v}} \cdot \hat{u}_{u,\tilde{u}} \cdot \hat{k}_{k,\tilde{k}} \cdot \hat{v}_{v,\tilde{v}} \tag{2}$$

where $\tilde{u}, \tilde{k}, \tilde{v} \in \{1, ..., l\}$ are indices of latent space. This model is called Tucker decomposition model [21]. If we set the core tensor as a diagonal tensor:

$$\hat{c}_{\tilde{u},\tilde{k},\tilde{v}} = \begin{cases} 1, \ if \ \ \tilde{u} = \tilde{k} = \tilde{v} \\ 0, \ else \end{cases} \tag{3}$$

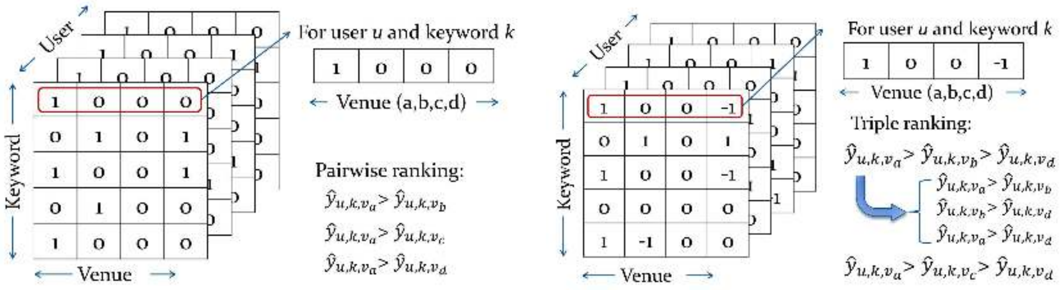
We obtain a Canonical decomposition model with each element calculated as:

$$\hat{y}_{u,k,v} = \sum_{\tilde{f}} \hat{u}_{u,\tilde{f}} \cdot \hat{k}_{k,\tilde{f}} \cdot \hat{v}_{v,\tilde{f}} \tag{4}$$

where $\tilde{f} \in \{1, ..., l\}$ is the indices of latent space. As a special case of Canonical decomposition model, the pairwise interaction model [13] explicitly captures the pairwise interaction among the three factors:

$$\hat{y}_{u,k,v} = \sum_{\tilde{f}} \hat{u}^K_{u,\tilde{f}} \cdot \hat{k}^U_{k,\tilde{f}} + \hat{u}^V_{u,\tilde{f}} \cdot \hat{v}^U_{v,\tilde{f}} + \hat{k}^V_{k,\tilde{f}} \cdot \hat{v}^K_{v,\tilde{f}} \tag{5}$$

where $\hat{u}^K$ represent the interaction between user and keyword from user's perspective, and so on. When predicting venue ranking, the interaction between user and keyword vanishes.


User
Keyword
Venue
1 0 0 0
0 1 0 1
1 0 0 1
0 1 0 0
1 0 0 0
For user u and keyword k
1 0 0 0
← Venue (a,b,c,d) →
Pairwise ranking:
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_b}$
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_c}$
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_d}$
User
Keyword
Venue
1 0 0 -1
0 1 0 1
1 0 0 -1
0 0 0 0
1 -1 0 0
For user u and keyword k
1 0 0 -1
← Venue (a,b,c,d) →
Triple ranking:
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_b} > \hat{y}_{u,k,v_d}$
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_b}$
$\hat{y}_{u,k,v_b} > \hat{y}_{u,k,v_d}$
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_d}$
$\hat{y}_{u,k,v_a} > \hat{y}_{u,k,v_c} > \hat{y}_{u,k,v_d}$

(a) Pairwise ranking relation

(b) Triple ranking relation

Figure 5. Multi-Tuple Ranking Scheme

Using vector representation we get:

$$\hat{y}_{u,k,v} = \hat{u}_u \cdot (\hat{v}_v^U)^T + \hat{k}_k \cdot (\hat{v}_v^K)^T \tag{6}$$

where $\hat{u}_u$ and $\hat{k}_k$ are the feature vectors in $\hat{U}$ and $\hat{K}$. Additionally, $\hat{v}_v^U$ and $\hat{v}_v^K$ are the feature vectors in $\hat{V}^U$ and $\hat{V}^K$. Note that in this model, a tensor is decomposed into four factors, i.e., $\hat{U}$, $\hat{K}$, $\hat{V}^U$ and $\hat{V}^K$. This model is used in our work to factorize the *u-k-v* tensor.

Optimization criterion

Optimization criterion is represented by an objective function when performing tensor factorization. Existing approaches can only handle positive preference [11, 13]. Given a user u and a keyword k, those works aim at ranking user preferred venues in front of others, which can be formulated as:

$$Obj_{po} = \sum_{v^+ \in V_{u,k}^+} \sum_{v^0 \in V_{u,k}^0} (\hat{y}_{u,k,v^+} - \hat{y}_{u,k,v^0}) \tag{7}$$

where $V_{u,k}^+$ and $V_{u,k}^0$ represent venues with positive preference and with unknown preference, respectively. Maximizing the above function is able to rank the venues with positive preference in front of the others, as shown in Figure 5(a), where rank of venue a is higher than that of venue b, c and d. However, since our *u-k-v* tensor further includes negative preference, this objective function can not handle such case. As shown in Figure 5(b), for a user u and a keyword k, the rank of venue a (with positive preference) is higher than that of venue b and c (unknown preference), and the rank of venue b and c are higher than that of venue d (with negative preference). Then, each triple ranking relation can be seen as three pairwise ranking relations, as shown in Figure 5(b). Let $V_{u,k}^-$ denote venues with negative preference. In addition to Equation 7, the two other pairwise ranking relations are between $V_{u,k}^0$ and $V_{u,k}^-$, $V_{u,k}^+$ and $V_{u,k}^-$.

$$Obj_{on} = \sum_{v^0 \in V_{u,k}^0} \sum_{v^- \in V_{u,k}^-} (\hat{y}_{u,k,v^0} - \hat{y}_{u,k,v^-}) \tag{8}$$

$$Obj_{pn} = \sum_{v^+ \in V_{u,k}^+} \sum_{v^- \in V_{u,k}^-} (\hat{y}_{u,k,v^+} - \hat{y}_{u,k,v^-}) \tag{9}$$

The optimization is then performed successively for the three pairwise ranking relations. Let $\vec{v}_{u,k}$ denote the venue vector given the user u and keyword k. The vector $\vec{v}_{u,k}$ belongs to *pairwise ranking* if $\vec{v}_{u,k}$ only contains two different values, while $\vec{v}_{u,k}$ belongs to *triple ranking* if $\vec{v}_{u,k}$ contains three different values, i.e., 1, -1 and 0. Considering both pairwise and triple ranking relations, the optimization criterion of MT-RTF is defined as:

$$Obj = \begin{cases} Obj_{xx}, \ if \ \ \vec{v}_{u,k} \in pairwise \ ranking \\ Obj_{po} + Obj_{on} + Obj_{pn}, \ if \ \ \vec{v}_{u,k} \in triple \ ranking \end{cases} \tag{10}$$

where Obj_{xx} represents Obj_{po}, Obj_{on} or Obj_{pn} for $\vec{v}_{u,k}$ containing 1/0, 0/-1 or 1/-1, respectively. By maximizing Obj for all observed (u,k) pairs, we get finally the objective function as:

$$\max_{\hat{U},\hat{K},\hat{V}^U,\hat{V}^K} \sum_{\{(u,k)|\exists v, y_{u,k,v} \neq 0\}} Obj \tag{11}$$

Learning Process

We adopt Bayesian personalized ranking learning algorithm [12] as the learning framework. A bootstrap sampling method is used to reduce learning time. Since our objective function considers all the data in tensor, to target the data obtained from sampling approach, we extract an atomic objective function for each ranking venue pair, denoted as $\hat{y}_{u,k,v^a,v^b}$ for a given ranking pair $\hat{y}_{u,k,v^a}$ and $\hat{y}_{u,k,v^b}$, where $(a,b) \in \{(+,0),(0,-),(+,-)\}$.

$$\hat{y}_{u,k,v^a,v^b} = (\hat{y}_{u,k,v^a} - \hat{y}_{u,k,v^b}) \tag{12}$$

Gradient descent approach is used to update parameters $\hat{U}$, $\hat{K}$, $\hat{V}^U$ and $\hat{V}^K$ in each iteration. Combining with Equation 6, the gradients of $\hat{y}_{u,k,v^a,v^b}$ are:

$$\frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \hat{u}_u} = (\hat{v}_{v^a}^U - \hat{v}_{v^b}^U), \frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \hat{k}_k} = (\hat{v}_{v^a}^K - \hat{v}_{v^b}^K) \tag{13}$$

$$\frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \hat{v}_{v^a}^U} = \hat{u}_u, \frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \hat{v}_{v^b}^U} = -\hat{u}_u \tag{14}$$

$$\frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \hat{v}_{v^a}^K} = \hat{k}_k, \frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \hat{v}_{v^b}^K} = -\hat{k}_k \tag{15}$$

Given a tensor T and a set of parameters Θ i.e., $\hat{U}$, $\hat{K}$, $\hat{V}^U$ and $\hat{V}^K$, MT-RTF learning algorithm is illustrated in Algorithm 1. Note that $g(x) = \frac{1}{1+e^{-x}}$ is the logistic function. The α controls the learning step and λ is the regularization parameter. In each iteration, we first select one (u,k) pair randomly (Line 3), and then randomly select pairwise ranking relation (Line 4-5) or triple ranking relation (Line 9-10) according to $\vec{v}_{u,k}$. For pairwise ranking, the optimization is conducted

Algorithm 1 MT-RTF Learning Algorithm

```
Require: $T, \Theta$
 1: initialize $\Theta$
 2: repeat
 3:     draw a $(u, k)$ pair uniformly from $T$
 4:     if $\vec{v}_{u,k} \in$ pairwise ranking then
 5:         draw $(v^a, v^b)$ uniformly from $\vec{v}_{u,k}$
 6:         $\rho = (1 - g'(\hat{y}_{u,k,v^a,v^b}))$
 7:         $\Theta = \Theta + \alpha \cdot [\rho \cdot \frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \Theta} - \lambda \cdot \Theta]$
 8:     end if
 9:     if $\vec{v}_{u,k} \in$ triple ranking then
10:         draw $(v^+, v^0, v^-)$ uniformly from $\vec{v}_{u,k}$
11:         for $(v^a, v^b) \in \{(v^+, v^0), (v^0, v^-), (v^+, v^-)\}$ do
12:             $\rho = (1 - g'(\hat{y}_{u,k,v^a,v^b}))$
13:             $\Theta = \Theta + \alpha \cdot [\rho \cdot \frac{\partial \hat{y}_{u,k,v^a,v^b}}{\partial \Theta} - \lambda \cdot \Theta]$
14:         end for
15:     end if
16: until convergence of $Obj$
17: return $\hat{\Theta}$
```

only for v^a, v^b (Line 6-7). For triple ranking, the optimization is conducted successively for $\{(v^+, v^0), (v^0, v^-), (v^+, v^-)\}$ (Line 11-14). The algorithm converges until no further improvement for the objective function Obj. The output of MT-RTF is the optimized Θ. Using Equation 6, the predicted ranking score can be obtained. Based on such ranking score, for a given user and a keyword, venues can be ranked.

EXPERIMENTAL EVALUATION

For personalized search, evaluation is not an easy task because the returned results can be judged only by the searchers themselves. Obviously, such an approach is costly in our case because it is difficult to interview Foursquare users by questionnaire. We choose to evaluate SEALs framework from two aspects. First, by random splitting dataset into training set and test set, a series of experiments are conducted to evaluate the performance of MT-RTF algorithm in personalized ranking, i.e., whether the top ranked results contain more user liked venues and less disliked venues. Second, a case study of SEALs is presented to show how fine-grained user preference can improve the performance of personalized location search.

Performance Evaluation of MT-RTF

The performance evaluation of MT-RTF algorithm intends to answer the following questions:

- How does the latent space dimension influence venue ranking performance of MT-RTF?
- Can MT-RTF achieve better performance compared with the state-of-the-art approaches? What advantages can be brought out by considering fine-grained and negative preference?
- Does MT-RTF algorithm perform consistently for different types of users?

To answer these questions, we first test MT-RTF performance using different latent space dimensions. By fixing to one latent space dimension, we then compare its performance with other state-of-the-art approaches. Finally, we show the performance of MT-RTF algorithm for different types of users. To get a relatively dense tensor in experiments, we select 20-core data[11], resulting in a *u-k-v* tensor with dimensionality of (994*728*1008). The characteristics of the experimental tensor are shown in Table 1.

User number	994
Keyword number	728
Venue number	1008
Number of the observed u-k-v triplets	51091
Data density	0.007%
Positive feedback number	43924
Negative feedback number	7167

Table 1. Characteristics of the Experimental Tensor

A test set S is constructed by randomly selecting *u-k* pairs and all related venues, i.e., $\vec{v}_{u,k}$. The remaining is used as the training set. ($\vec{v}_{u,k}$ is set to 0 in training set for those *u-k* pairs selected by the test set). Using MT-RTF on the training set to perform the venue ranking, the predicted ranking for *u-k* pairs in the test set S is then evaluated. The classical evaluation metrics in IR (Information Retrieval) often evaluate whether a result is relevant or not. However, in our case, for a given *u-k* pair, the venues may fall into three categories, i.e., venues with positive, negative, or unknown preference. While the ones with positive or unknown preference can be treated as “relevant” or “non-relevant”, the negative ones cannot be simply considered as “non-relevant”. Because putting a user disliked venue on the top will decrease user experience more than a non-relevant venue. Hence, by adjusting Mean Average Precision (MAP) which is a widely used metric in IR community, we introduce a metric named *Mean Average Satisfaction* (MAS). To introduce MAS, we first explain the definition of MAP. For a test set S, MAP is defined as follows:

$$MAP = \sum_{(u,k) \in S} \frac{\sum_{i=1}^{n} \sum_{j=1}^{i} \frac{r(j)}{i} \cdot r(i)}{N^+} \qquad (16)$$

where N^+ and n are the number of relevant venues (i.e., venues with positive preference) and number of retrieved venues, respectively. The relevance function $r(i)$ is set to 1 if the i^{th} venue in the results is relevant and 0 otherwise. Since this definition does not consider the venues with negative preference, the extension of MAS comparing with MAP is to introduce a punishment against ranking user disliked venues on the top. Its definition is as follows:

$$MAS = \sum_{(u,k) \in S} \frac{\sum_{i=1}^{n} \sum_{j=1}^{i} \frac{sat(j)}{i} \cdot r(i)}{N^+} \qquad (17)$$

where the satisfaction function $sat(i)$ is set to 1, 0 or -1 if the i^{th} venue is the one with positive, unknown or negative preference, respectively. A higher value implies the top results contain more venues with positive preference and fewer venues with negative preference. Hence, MAS can be regarded as an indicator of user experience for the retrieved venue ranking.

[11] The p-core of a tensor is the largest subset of the tensor with the property that every user, every keyword and every venue has to occur in at least p records.

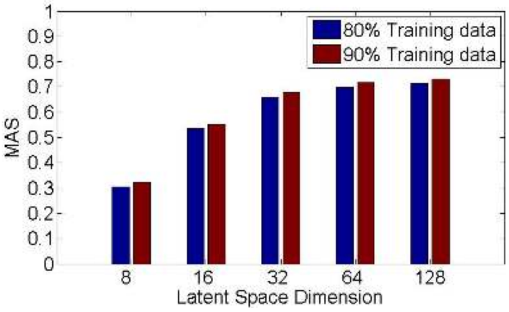


Figure 6. Performance with Different Latent Space Dimensions

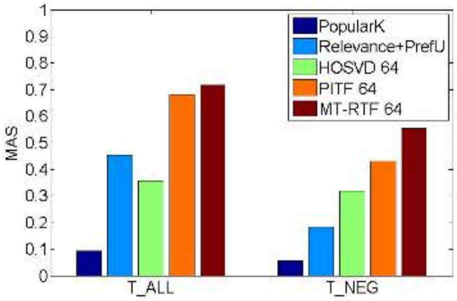


Figure 7. Performance Comparison with Other Approaches

Performance test with different latent space dimensions

In this test, we choose 80% and 90% of the dataset as training set, and then vary the latent space dimension in the order of 8, 16, 32, 64 and 128. The learning step α is set as 0.1 and regularization parameter λ is set as 0.00001, respectively. Due to the space limitation, we do not detail the parameter tuning process for α and λ. In all the performance tests, each result is the mean value of ten repeated trials. Figure 6 reports the results. With the increase of the latent space dimension, the ranking performance of MT-RTF increases. A slight improvement is observed for using 90% of the data as training set comparing the case of using 80%. We also find that no significant improvement of MAS for dimension higher than 64, which indicates the convergence of the algorithm in terms of latent space dimension. Hence, in the following experiments, the latent space dimension is fixed as 64 and training data percentage is set to 90%.

Comparison with other approaches

In order to further validate the effectiveness of MT-RTF, we compare it with the existing personalized search approaches shown below:

- PopularK: for a given keyword, venues are ranked by its popularity in a descending order, regardless of users. This is deemed as a non-personalized search approach because it returns the identical search results to all users.
- Relevance+PrefU: for a given keyword and a user, venues are firstly filtered by venue-keyword relevance and then reranked by the user preference on venues and keywords. This can be regarded as a typical personalized search approach using coarse-grained user preference.
- HOSVD: high order singular value decomposition [7] which performs the low-rank approximation. It corresponds to a Tucker decomposition optimized for square-loss.
- PITF: pairwise interaction tensor factorization [13] which only incorporates positive preference into factorization. Using this approach, we consider negative preference (i.e., -1) as unknown preference (i.e., 0) in the training set in order to ignore negative preference.

We set latent space dimension as 64 for all the tensor based approaches, i.e., MT-RTF, HOSVD and PITF, and keep other parameters the same as in the previous section. Firstly, we report the overall performance on the whole test set (denoted as T_ALL). In order to deeply investigate the improvement of considering negative preference, we choose the partial test set that only contains *u-k* pairs with negative preference (T_NEG) to show what performance can be achieved.

The left part of Figure 7 illustrates the overall performance. Obviously, all personalized approaches outperform the non-personalized search PopularK, which indicates that personalization is able to enhance user experience, i.e., leading higher MAS. Among the personalized approaches, HOSVD performs the worst. This might be caused by two reasons, viz. the tensor sparsity problem and weakness of HOSVD for ranking problem. Relevance+PrefU approach that considers coarse-grained user preference of venues performs better than HOSVD but still gets unsatisfactory results. The high performance of MT-RTF and PITF proves that the ranking tensor factorization approach is efficient in solving such ranking problem. Furthermore, MT-RTF that considers both positive and negative preference achieves higher performance comparing with PITF.

The right part of Figure 7 illustrates the performance for T_NEG. The proposed MT-RTF outperforms other approaches. Considering negative preference can significantly improve the user experience for those users with negative preference. In our dataset, the total number of observed negative preference (7167) is only 1/6 of the positive ones (43924). Such statistic explains that the improvement of MT-RTF for all users is not as much as that for users with negative preference. However, we believe that with more data collected in the future, the number of users with negative preference will increase. Thus, the advantages of MT-RTF will become more significant. An interesting observation is that the performance of Relevance+PrefU dramatically decreases when tackling negative preference. Because *coarse-grained user preference* on venue fails in the case that user has both positive and negative preference in one venue. On the contrary, *fine-grained user preference* can fully capture such detailed preference.

Performance test for different types of users

In LBSNs, users often behave differently in terms of active level. For example, some active users may check in or leave tips very frequently while other users may be inactive and report less digital traces. In our dataset, the average number of observed fine-grained preference per user is 51.40. Therefore, we split each test set into two subsets: low active users (observed preference number < 50) and high active users (ob-

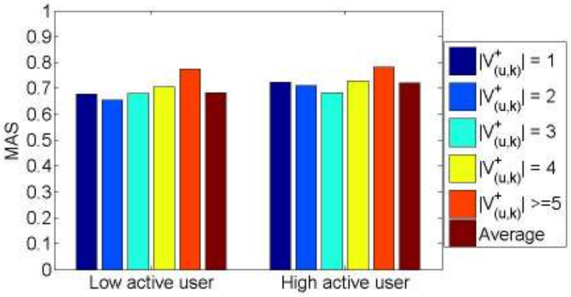


Figure 8. Performance for Different Types of Users

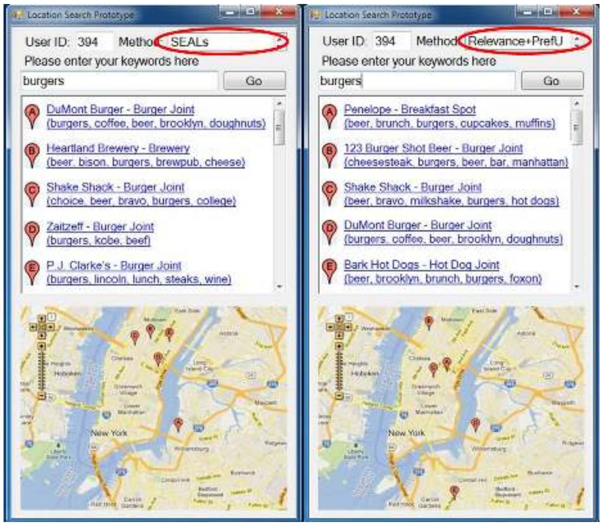
served preference number $>= 50$). Moreover, for a given *u-k* pair, the number of venues with positive preference in the test set (i.e., the ground truth length $|V^+_{u,k}|$) might be different. In order to prove that MT-RTF algorithm achieves consistently good performance, we report separately the results for different $|V^+_{u,k}|$, and the average performance as well. Figure 8 illustrates the results for both low active users and high active users.

First, MT-RTF algorithm gets consistent results (MAS around 0.7) for venue ranking with different length $|V^+_{u,k}|$. The result confirms that MT-RTF preforms well regardless the ground truth length. Moreover, with regard to user active level, a slight improvement could be observed for the high active users (average MAS is 0.7308) comparing with the low active users (average MAS is 0.6840). This observation implies that the more activities users have in LBSNs, the better location search experience they can get from SEALs.

Case study of SEALs

In this section, we present an example to illustrate how fine-grained user preference helps improve personalized search. When a user has different feelings about the items in one place, the coarse-grained user preference based location search sometimes fails to return the results matching such preference, but SEALs can manage to return satisfactory results. To illustrate this example, we select a keyword and a user to perform queries using two different search schemes, viz. SEALs and a coarse-grain preference based personalized search, i.e., Relevance+PrefU. We select "burgers" as keyword because it is popular for most New Yorkers and has a high frequency of occurrence. We then choose a user from the dataset (ID 394) and send the query. Figure 9 presents the search results displayed on our prototype. The information of each venue is described in the format "Venue Name - venue category (its top 5 frequent keywords)". For those venues with less than 5 keywords, we list all of them.

In the test set, this user has positive preference for burgers in the venue "Dumont Burgers". In addition, she has negative preference for burgers in the venue "123 Burger Shot Beer" and positive preference for pizza and beer in "123 Burger Shot Beer". From the user's all preference records, we find that her top 3 preferred keywords are "beer", "coffee" and "burgers".



User ID: 394 Method: SEALs

Please enter your keywords here

burgers

A DuMont Burger - Burger Joint (burgers, coffee, beer, brooklyn, doughnuts)

B Heartland Brewery - Brewery (beer, bison, burgers, brewpub, cheese)

C Shake Shack - Burger Joint (choice, beer, bravo, burgers, college)

D Zaitzeff - Burger Joint (burgers, kobe, beef)

E P.J. Clarke's - Burger Joint (burgers, lincoln, lunch, steaks, wine)

User ID: 394 Method: Relevance+PrefU

Please enter your keywords here

burgers

A Penelope - Breakfast Spot (beer, brunch, burgers, cupcakes, muffins)

B 123 Burger Shot Beer - Burger Joint (cheesesteak, burgers, beer, bar, manhattan)

C Shake Shack - Burger Joint (beer, bravo, milkshake, burgers, hot dogs)

D DuMont Burger - Burger Joint (burgers, coffee, beer, brooklyn, doughnuts)

E Bark Hot Dogs - Hot Dog Joint (beer, brooklyn, brunch, burgers, foxon)

Figure 9. Search results of "burgers" using different search schemes

In the left panel of Figure 9, we can observe that using SEALs, "Dumont Burgers" is the first venue appeared in the returned search results and "123 Burger Shot Beer" is ranked far behind other venues (20th, we only display five venues at the same time on the screen), which matches the user's preference for burgers in these venues. In the right panel of Figure 9, using coarse-grained preference, we observe that "123 Burger Shot Beer" is ranked at 2nd and "Dumont Burgers" becomes the 4th in the list, which is against her preference. Such observation can be explained by two reasons: 1) even if the user has negative preference for burgers in "123 Burger Shot Beer", her overall preference of this restaurant is positive because she expressed positive sentiment about two entities and negative sentiment about one entity; 2) this venue has both burgers and beer that match the user's coarse-grained preference. This observation shows that SEALs can delicately handle fine-grained user preference and efficiently provide a desired list of places according to each user's preference, given a search keyword.

DISCUSSION

Integrating location-awareness with SEALs. The focus of SEALs framework is to build a preference-aware location search service. As most of location-aware search approaches target at retrieving locations within a certain geo-span, this location-aware feature can be easily incorporated into SEALs to filter out the locations by the geo-span.

Scalability of tensor factorization. Practically, tensor factorization has scalability issue. If the tensor scale is large, the factorization process is quite time-consuming. Although the bootstrap sampling can alleviate such a problem to certain extent, the problem persists for extra large scale tensor processing. Fortunately, the computation of factorization can be done off-line, thus we might conduct tensor factorization beforehand to generate the ranked venues corresponding to each combination of users and keywords. When a user issues a query with a keyword, the corresponding list of ranked venues are retrieved. With more data accumulated from LB-

SNs, computation of tensor factorization needs to be performed regularly.

Incentives in crowdsourcing. LBSNs provide users with a socializing platform which can be regarded as a crowdsourcing platform. While most crowdsourcing platforms purposely design incentive mechanisms to attract users, the SEALs framework has an inherent incentive mechanism. The more check-ins and comments the user contributes to Foursquare, the better location search experience the SEALs framework can provide using the proposed method. Moreover, better user experience in LBSNs will in turn incentivize users to perform more activities, and thus contribute more data to LBSNs.

CONCLUSION AND FUTURE WORK

In this paper, we proposed SEALs, a fine-grained preference-aware location search framework leveraging the crowdsourced LBSNs traces. With the rich information contained in the massively and cheaply contributed contents from LBSNs, we collect and extract user's sentiment about locations and associated entities to characterize user's fine-grained location preference. We further model such fine-grained user preference using tensor and develop a Multi-Tuple based Ranking Tensor Factorization algorithm to ensure that only the liked places with the liked items of individual are shown at the top of the returned location search list, when a user conducts a location search with a simple keyword. Extensive evaluations were conducted using our collected Foursquare tips and check-ins. The results show that SEALs can efficiently handle fine-grained user preference and provide users with great location search experience.

In the future, we plan to broaden this work in several directions. First, we plan to build a more comprehensive personalization location search framework by considering both user context and preference. Second, we intend to exploit the social relationship among users hidden in LBSNs to further improve the personalized location ranking algorithm. Third, we plan to explore new ways of accommodating the accumulated crowdsourced digital footprints from LBSNs and enabling effective, scalable and personalized location search.

ACKNOWLEDGMENTS

This work is supported by the EU FP7 Project SOCIETIES (No. 257493), the National Basic Research Program of China (No. 2012CB316400) and the National Natural Science Foundation of China (No. 61222209, 61103063).

REFERENCES

1. P. Anick. Using terminological feedback for web search refinement: a log-based study. In *SIGIR '03*, pages 88–95, 2003.
2. Y. Cai and Q. Li. Personalized search by tag-based user profile and resource profile in collaborative tagging systems. In *CIKM '10*, pages 969–978, 2010.
3. P. A. Chirita, C. S. Firan, and W. Nejdl. Personalized query expansion for the web. In *SIGIR '07*, pages 7–14, 2007.
4. D.-Y. Choi. Personalized local internet in the location-based mobile web search. *Decis. Support Syst.*, 43(1):31–45, 2007.
5. M. Iwata, T. Hara, K. Shimatani, T. Mashita, K. Kiyokawa, S. Nishio, and H. Takemura. A location-based content search system considering situations of mobile users. *Procedia Computer Science*, 5(0):426 – 433, 2011.
6. N. D. Lane, D. Lymberopoulos, F. Zhao, and A. T. Campbell. Hapori: context-based local search for mobile phones using community behavioral modeling and similarity. In *Ubicomp '10*, pages 109–118, 2010.
7. L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, Mar. 2000.
8. F. Li, S. J. Pan, O. Jin, Q. Yang, and X. Zhu. Cross-domain co-extraction of sentiment and topic lexicons. In *ACL '12*, pages 410–419, 2012.
9. E. Loper and S. Bird. Nltk: The natural language toolkit. *CoRR*, cs.CL/0205028, 2002.
10. F. Qiu and J. Cho. Automatic identification of user interest for personalized search. In *WWW '06*, pages 727–736, 2006.
11. S. Rendle, L. Balby Marinho, A. Nanopoulos, and L. Schmidt-Thieme. Learning optimal ranking with tensor factorization for tag recommendation. In *KDD '09*, pages 727–736, 2009.
12. S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI '09*, pages 452–461, 2009.
13. S. Rendle and L. Schmidt-Thieme. Pairwise interaction tensor factorization for personalized tag recommendation. In *WSDM '10*, pages 81–90, 2010.
14. J. Sang, C. Xu, and D. Lu. Learn to personalized image search from the photo sharing websites. *IEEE Transactions on Multimedia*, 14(4):963–974, 2012.
15. P. Shankar, Y.-W. Huang, P. Castro, B. Nath, and L. Iftode. Crowds replace experts: Building better location-based services using mobile social network interactions. In *PerCom'12*, pages 20–29, 2012.
16. S. Shariaty and S. Moghaddam. Fine-grained opinion mining using conditional random fields. In *ICDMW'11*, pages 109–114, 2011.
17. Y. Shi, A. Karatzoglou, L. Baltrunas, M. Larson, A. Hanjalic, and N. Oliver. Tfmap: optimizing map for top-n context-aware recommendation. In *SIGIR '12*, pages 155–164, 2012.
18. B. Stefano, E. Andrea, and S. Fabrizio. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *LREC'10*, 2010.
19. K. Sugiyama, K. Hatano, and M. Yoshikawa. Adaptive web search based on user profile constructed without any effort from users. In *WWW '04*, pages 675–684, 2004.
20. J.-T. Sun, H.-J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *WWW '05*, pages 382–390, 2005.
21. L. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, Sept. 1966.
22. K. Waga, A. Tabarcea, and P. Franti. Context aware recommendation of location-based data. In *ICSTCC '11*, pages 1–6, oct. 2011.
23. A. Wu and X. Zhang. Location-based information fusion for mobile navigation. In *UbiComp '11*, pages 593–594, 2011.
24. S. Xu, S. Bao, B. Fei, Z. Su, and Y. Yu. Exploring folksonomy for personalized search. In *SIGIR '08*, pages 155–162, 2008.
25. D. Yang, D. Zhang, Z. Yu, and Z. Wang. A sentiment-enhanced personalized location recommendation system. In *HT '13*, pages 119–128, 2013.
26. D. Zhang, B. Guo, and Z. Yu. The emergence of social and community intelligence. *IEEE Computer*, 44(7):21–28, 2011.
27. Y. Zheng, L. Zhang, Z. Ma, X. Xie, and W.-Y. Ma. Recommending friends and locations based on individual location history. *ACM Trans. Web*, 5(1):5:1–5:44, Feb. 2011.
28. D. Zhou, S. Lawless, and V. Wade. Web search personalization using social data. In *Theory and Practice of Digital Libraries*, volume 7489, pages 298–310. 2012.