NILR: N-Most Interesting Location-based Recommender System

Sumet Darapisut  
Faculty of Informatics  
Burapha University  
Chonburi, 20130, Thailand  
dearsumet@gmail.com

Komate Amphawan  
Faculty of Informatics  
Burapha University  
Chonburi, 20130, Thailand  
komite@gmail.com

Nutthanon Leelathakul  
Faculty of Informatics  
Burapha University  
Chonburi, 20130, Thailand  
nutthanon@buu.ac.th

Sunisa Rimcharoen  
Faculty of Informatics  
Burapha University  
Chonburi, 20130, Thailand  
rsunisa@buu.ac.th

ABSTRACT
Location-Based Recommender Systems (LBRSs) have gained popularity in recent years as users tend to make decisions based on what are shared in social medias. Such systems depend on each user’s historical behavioral information (or user profile) to determine users’ interests. However, it is impossible for new users to have the profiles, making it difficult and challenging to recommend interesting locations also known as a cold start problem. In order to tackle this issue, we propose an enhanced method, called N-most interesting location-based recommender system (NILR), which effectively recommends the N-most preferred places for each user without leveraging her profile. We also introduce a novel metric (so-called interestingness score) to measure locations’ attractiveness. The metric takes into account both check-in frequencies and number of return visits of previous users already in the system. The method ranks the top-N locations based on the combination of the traditional HITS-based model (Hypertext Induced Topic Search) [3] and the proposed NILR. The results of the experiments on Foursquare dataset reveal that our proposed location recommender system and ranking method perform effectively and efficiently, and outperform the HITS model in terms of accuracies and rankings.

CCS CONCEPTS
- Information systems - Information retrieval - Retrieval tasks and goals - Recommender systems

KEYWORDS
Location-based recommender systems, non-profile users, ranking

1 INTRODUCTION
Traditional Location-Based Recommender Systems (LBRSs) employ users’ profiles (i.e., their check-in information) to infer the users’ preferences in order to suggest attractive places to the users. Nevertheless, in real-world scenarios, most users tend to visit only favorite places in their vicinity, causing relatively few and limited areas contained in their check-in data. Such challenge is considered a cold start problem, where it is difficult to recommend locations in other areas to these users based on their limited profile data. To cope with this issue, several works attempted to suggest interesting locations based on determined local experts. For example, Yu et al. [1] proposed a method, called TBHG, to recommend travel sequences for various users on a variety of geospatial sizes by regarding local experts using HITS-based model. Jie and et al. [2] presented location-based and preference-aware recommendation systems (weighted category hierarchy, WCH). A recommendation list was created based on the similarity between a user and local experts. In [3], CLoRW algorithm that used a random-walk and a page-rank approach were proposed regarding popular locations, friend relations and local experts. A recommendation algorithm for shopping places based on local experts and topical experts was proposed by [4]. The PageRank and Lazy random walk algorithms were used to select local experts for creating recommendation list.

Most existing works [1-3] recommended interesting locations based only on frequencies of check-ins. Recommender systems, depending only on check-in frequencies, might miss certain places attractive to and revisited many times by local experts but not known to foreign users. Recommending locations based only on return visits might also be inefficient as the system might miss locations attractive to boarder audience. To the best of our knowledge, there is no previous research work that leverages both check-in frequencies and numbers of return visits to rank top-N interesting locations for each specific user. Our method also addresses the cold start problem where the system could recommend users with no prior profile.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).

SMA 2020, September 17-19, 2020, Jeju, Republic of Korea  
© 2020 Copyright held by the owner/author(s).
2 THE PROPOSED NILR SYSTEM

To recommend interesting shops to a user that is unfamiliar with her current location, our proposed NILR system takes check-in profiles of all other users as input (See Fig. 1). Each check-in data entry consists of 4 fields: an user name, \( u_x \), the shop’s name, \( s_y \), the shop location, \( loc_{s_y} \), and the number of times (frequency) that the user \( u_x \) checks-in at the shop \( s_y \), \( f_{ux, s_y} \). The first step is to preprocess all check-in histories of all users: the number of users that checks-in at the shop \( s_y \), more than once \((nu_{s_y})\) and the number of shops that the user \( u_x \) checks-in more than once \((ns_{ux})\) are determined. In the second step, calculate, according to each user \( u_j \), the multiplication of i) \( f_{uj, s_y} \), ii) \( nu_{uj} \), and iii) the user score of \( u_j \) \((score_{uj})\), which is firstly initialized as 1. Then, determine the interestingness score of each shop \( s_y \) \((int_{s_y})\), which is the sum of the multiplications (of all users) obtained in the previous step. Subsequently, each user’s score is updated by summing, of all shops, the multiplication of i) \( f_{uj, s_y} \), ii) \( nu_{s_y} \), and iii) \( int_{s_y} \). The interestingness score of each shop \( s_y \) and the user score of each user \( u_x \) are normalized by L2-normalization (as in [5]). The process of calculating interestingness and user scores is repeated until all the values are converged: the interestingness score of all shops are ready to be used for recommendation.

Figure 1. Framework of the NILR system.

Whenever a new user, \( u_x \), asks for nearby shop recommendation, she needs to send a request enclosing her current location, \( loc_{u_x} \). Upon receiving the request, the NILR system starts ranking interesting shops nearby the enclosed location. Note that a distance threshold \( d \) could be specified to filter out shops too far away from the location \( loc_{u_x} \). We sort all shops’ interestingness scores in descending order. In addition, we also calculate the HITS-based scores for all shops [3] and then order them in descending order. Lastly, to provide the list of recommended shops, all shops are ranked by regarding both interestingness and HITS-based scores. Only top-10 shops close to the location \( loc_{u_x} \) (within the distance \( d \)) are recommended to the user \( u_x \).

3 EXPERIMENTAL RESULTS

A comparative study between our proposed system and HITS is conducted on a well-known Foursquare check-in data of food shops in Tokyo [6]. The dataset is divided into 5 segments of the famous areas in Tokyo: Chiyoda, Shibuya, Minato, Shinjuku, and Chuo. Each segment is split into 5 equal sub-segments for 5-fold cross validation. The performance is measured in terms of three metrics: precision, recall, and average ranking. The number of shops to be recommended and ranked is varied from 1 to 10.

As shown in Table 1, our system yields higher recalls and precisions than those of HITS in the most cases (except only the case of Shinjuku area). It is because users in Shinjuku usually go to the shops of high check-in frequencies and large numbers of return visits. Meanwhile, users (with no profile in Shinjuku) mostly go to the shops that frequently visited. Moreover, we also measure ranking accuracies (in terms of average ranking) of recommended-location lists. As shown in the table, our system can yields the average ranks in the range of 2 and 5, better than those of HITS in the range of 3 and 8. (The top 10 the ranks of the shops in the list, which users actually visit, the higher the ranking performance.)

<table>
<thead>
<tr>
<th>Metric</th>
<th>Area</th>
<th>HITS</th>
<th>NILR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>Chiyoda</td>
<td>0.062</td>
<td>0.068</td>
</tr>
<tr>
<td></td>
<td>Shibuya</td>
<td>0.016</td>
<td>0.019</td>
</tr>
<tr>
<td></td>
<td>Minato</td>
<td>0.023</td>
<td>0.024</td>
</tr>
<tr>
<td></td>
<td>Shinjuku</td>
<td>0.038</td>
<td>0.032</td>
</tr>
<tr>
<td></td>
<td>Chuo</td>
<td>0.029</td>
<td>0.094</td>
</tr>
<tr>
<td>Recall</td>
<td>Chiyoda</td>
<td>0.167</td>
<td>0.172</td>
</tr>
<tr>
<td></td>
<td>Shibuya</td>
<td>0.062</td>
<td>0.076</td>
</tr>
<tr>
<td></td>
<td>Minato</td>
<td>0.063</td>
<td>0.073</td>
</tr>
<tr>
<td></td>
<td>Shinjuku</td>
<td>0.118</td>
<td>0.099</td>
</tr>
<tr>
<td></td>
<td>Chuo</td>
<td>0.094</td>
<td>0.099</td>
</tr>
<tr>
<td>Average ranking</td>
<td>Chiyoda</td>
<td>3.245</td>
<td>2.817</td>
</tr>
<tr>
<td></td>
<td>Shibuya</td>
<td>7.484</td>
<td>4.098</td>
</tr>
<tr>
<td></td>
<td>Minato</td>
<td>6.407</td>
<td>4.139</td>
</tr>
<tr>
<td></td>
<td>Shinjuku</td>
<td>5.499</td>
<td>4.476</td>
</tr>
<tr>
<td></td>
<td>Chuo</td>
<td>6.579</td>
<td>4.020</td>
</tr>
</tbody>
</table>

4 CONCLUSIONS

In this paper, we propose a novel system for recommending most interesting places based on user’s current location without leveraging their profiles. The metrics based on 1) the check-in frequencies (obtained from HITS algorithm) and 2) the compound...
scores of both frequencies and numbers of return visits are calculated and applied to identify attractiveness of each location. Given a user’s location and the maximum distance allowed, a set of the N-most interesting places ranked by the two metrics are generated. The experiments on a well-known real-world dataset were conducted and the results show that, in most cases, our proposed system yields higher recall, precision and more accurate ranking than the ones of the HITS algorithm.

REFERENCES