LARRS*: Location-Aware Recommendation And Rating System

Swati S. Avhad¹, Santosh R. Durugkar²

Department of Computer Engineering, SND COE & RC, Yeola, India

Abstract—LARS* is a location aware recommendation system that detects the location of user from IP address of system, gives details of that location and uses location based rating to produced recommendations. Recommender systems which are traditional do not consider spatial properties of users nor items; LARS*, on the other hand, supports a technique of three classes of location-based ratings, i.e. the spatial ratings for non-spatial items, the non-spatial ratings for the spatial items, and the spatial ratings for the spatial items. Details about user’s location are provided by using FivaTech. It propose an unsupervised, page-level data extraction to deduce the schema and templates for each individual huge Websites, which contains either singleton or multiple data records in one Webpage. FiVaTech applies tree matching, tree alignment, and mining techniques to achieve the challenging task. FivaTech is going to take place for web mining purpose.

Keywords—Recommendation system, spatial location performance, FivaTech efficiency, web mining, spatial rating system.

I. INTRODUCTION

The Location Aware Recommendation System is to develop such a system which will automatically identify location of system from from an IP Address of that system.

Next objective is to provide all location details of detected location. Ex: If Nashik city is detected by your system by running this project, then the objective is to show all the location details such restaurants, hospitals, picnic spots present at that location so that it will very much usefull for new user for any new city to find emergency places in that city.

Then next objective is provide rating system for movies according to location. Ex: Suppose timepass movie has rating 3 stars (**) in Nashik, 4 stars (****) in Pune, then this system should automatically change these movie ratings when system’s location will change.

Also this system is going to fulfill objective that it will display all Hindi, English and Marathi news of detected location & also will display helpline numbers of city for emergency. All these objectives are going to be satisfied by this system. of ratings when produce location aware recommendations. Also FivaTech is going to be used for giving details of detected location.

II. RELATED WORK

First of all location of system is detected from IP address of system using infosnipper.com. Details of detected location are provided by web mining.

Whatever city will be detected, dynamic web mining will be done only for that city name.

Spatial rating system is implemented. User can give rating as well as view rating. 5 star rating is implemented.2 parts are there for rating system i.e. user and admin. User can only view and give rating. Admin adds released movie in database as well as admin can delete movies from database.

Location Based Services
- Automatic location identification from an IP address
- Location based ratings such as spatial and non spatial ratings
- Giving all detailed information of detected location
- Extracting only news for detected location from news papers
- Giving all helpline information of location in case of emergency
- Giving whether information of detected location
III. PROBLEM STATEMENT

First of all, this system will identify location of user from an IP address of user’s system. After location detection, system is going to provide us 2 facilities i.e. providing location details and providing Rating system.

LARS* produces recommendations using special ratings for non special items, i.e., the rows by employing a user partitioning technique that exploits preference locality. This technique uses an adaptive pyramid structure to partition ratings by their user location parameters into spatial areas of different sizes at different levels. For a querying user located in a areas R, we apply an existing collaborative filtering technique that utilizes only the ratings located in R. The challenge, however, is to determine whether all regions in the pyramid must be maintained in order to balance two contradicting factors: scalability and locality. Maintaining a maximum number of areas increases locality (i.e., recommendations unique to smaller spatial regions), till now adversely affects system scalability because each area requires storage and maintenance of a collaborative filtering data structure necessary to produce recommendations (i.e., the recommender model). The LARS* pyramid dynamically adapts to find the right pyramid shape that balances scalability and recommendation locality.

There are a novel classification of 3 types of location-based ratings not supported by existing recommender systems: spatial ratings for non-spatial items, non-spatial ratings for spatial items, and spatial ratings for spatial items.

IV. ALGORITHMS USED FOR IMPLEMENTATION

1. Pyramid maintenance algorithm

2: Function Pyramid Maintenance(Cell C, Level k)
4: Maintain cell C statistics
6: if (Cell C is an α-Cell) then
7: Rebuild item-based collaborative filtering model for cell C
8: end if
10: if (C children quadrant q cells are α-Cells) then
11: CheckDownGradeToSCells(q, C)
12: else if (C children quadrant q cells are γ -Cells) then
13: CheckUpGradeToSCells(q, C)
14: else
15: iSProvidedToMcells ← CheckUpGradeToMCells(q, C)
16: if (iSProvidedToMcells is False) then
17: CheckDownGradeToECells(q, C)
18: end if
19: end if
20: return

α-Cell is considered the highly ranked cell type in LARS*. a β-Cell is the secondly ranked cell type as it only maintains statistics about the user/item ratings.

The storage and maintenance overhead incurred by a β-Cell is less expensive than an α-Cell. The statistics maintained at a β-Cell determines whether the children of that cell need to be maintained as α-Cells to serve more localized recommendation. Finally, a γ -Cell (lowest ranked cell type) has the least maintenance cost, as neither a CF model nor statistics are maintained for that cell. Moreover, a γ -Cell is a leaf cell in the pyramid.

2. Travel penalty algorithm

1: Function LARS* SpatialItems(User U, Location L, Limit K)
4: for (K iterations) do
5: i ← Retrieve the item with the next lowest travel penalty
6: Insert i into R ordered by RecScore(U, i) computed by Equation 7
7: end for
8: LowestRecScore← RecScore of the kth object in R
10: while there are more items to process do
11: i ← Retrieve the next item in order of penalty score
12: MaxPossibleScore← MAX RATING - i.penalty
13: if MaxPossibleScore≤ LowestRecScore then
14: return R
15: end if
16: RecScore(U, i) ← P(U, i) - i.penalty
17: if RecScore(U, i) > LowestRecScore then
18: Insert i into R ordered by RecScore(U, i)
19: LowestRecScore← RecScore of the kth object in R
20: end if
21: end while
22: return R

V. IMPLEMENTATION MODULES

FivaTech

FivaTech technology is used for Web mining for providing location details to users.

By using various algorithms stated above, FivaTech is implemented.

Spatial Ratings for Non Spatial Ratings

Spatial ratings for non-spatial items stated by the tuple (user, ulocation, rating, item). The idea is to states preference locality, i.e., the observation that user opinions are spatially unique. We identify three requirements for producing recommendations using spatial ratings for non-spatial items:

(1) Locality: recommendations should be influenced by those ratings with user locations patially close to the querying user location (i.e., in a spatial neighborhood);

(2) Scalability: the recommendation procedure and data structure should scale up to large number of users.

(3) Influence: system users should have the ability to control the size of the spatial neighborhood (e.g., city block, zip code, or county) that affects their recommendations.
Non Spatial Ratings for Non Spatial items

The traditional item-based collaborative filtering (CF) method is a special case of LARS*. CF takes as input the classical rating triplet \((user, rating, item)\) such that neither the user location nor the item location are specified. In such case, LARS* directly employs the traditional model building phase to calculate the similarity scores between all items. Moreover, recommendations are produced to the users using the recommendation generation phase. During the rest of the paper, we explain how LARS* incorporates either the user spatial location or the item spatial location to serve location-aware recommendations to the system users.

Non Spatial Ratings for Spatial Items

This section tells that how LARS* produces recommendations using non-spatial ratings for spatial items represented by the tuple \((user, rating, item, location)\). The idea is to exploit travel locality, i.e., the observation that users limit their choice of spatial venues based on travel distance based on analysis in Traditional (non-spatial) recommendation techniques may produce recommendations with burdensome travel distances (e.g., hundreds of miles away). LARS* produces recommendations within reasonable travel distances by using travel penalty, a technique that penalizes the recommendation rank of items the further in travel distance they are from a querying user. Travel penalty may incur expensive computational overhead by calculating travel distance to each item. Thus, LARS* employs an efficient query processing technique capable of early termination to produce the recommendations without calculating the travel distance to all items.

Pyramid Maintenance Algorithm

In this algorithm 1, A \(\alpha\)-Cell requires the highest storage and maintenance overhead because it maintains a CF model as well as the user or item ratings statistics. On the other hand, an \(\alpha\)-Cell (as opposed to \(\beta\)-Cell and \(\gamma\) - Cell) is the only cell that can be leveraged to answer recommendation queries. A pyramid structure that only contains \(\alpha\)-Cells achieves the highest recommendation locality, and this is why an \(\alpha\)-Cell is considered the highly ranked cell type in LARS*. A \(\beta\)-Cell is the secondly ranked cell type as it only maintains statistics about the user/item ratings. The storage and maintenance overhead incurred by a \(\beta\)-Cell is less expensive than an \(\alpha\)-Cell. The statistics maintained at a \(\beta\)-Cell determines whether the children of that cell need to be maintained as \(\alpha\)-Cells to serve more localized recommendation.

Finally, a \(\gamma\)-Cell (lowest ranked cell type) has the least maintenance cost, as neither a CF model nor statistics are maintained for that cell. Moreover, a \(\gamma\)-Cell is a leaf cell in the pyramid. LARS* upgrades (downgrades) a cell to a higher (lower) cell rank, based on trade-offs between recommendation locality and system scalability. If recommendation locality is preferred over scalability, more \(\alpha\)-Cells are maintained in the pyramid. On the other hand, if scalability is favored over locality, more \(\gamma\)-Cells exist in the pyramid. \(\beta\)-Cells comes as an intermediary stage between \(\alpha\)-Cells and \(\gamma\)-Cells to further increase the recommendation locality whereas the system scalability is not quite affected.

Travel Penalty Algorithm

The algorithm starts by running a \(k\)-nearest-neighbor algorithm to populate the list \(R\) with \(k\) items with lowest travel penalty; \(R\) is sorted by the recommendation score. This initial part is concluded by setting the lowest recommendation score value (LowestRecScore) as the RecScore of the \(kth\) item in \(R\). Then, the algorithm starts to retrieve items one by one in the order of their penalty score. This can be done using an incremental \(k\)-nearest-neighbor algorithm. Travel penalty requires very little maintenance. The only maintenance necessary is to occasionally rebuild the single system-wide item-based collaborative filtering model in order to account for new location-based ratings that enter the system.

VI. Results

The proposed system will give us following results:
- First of all location of user is going to be detected from an IP address of system.
- Then one whether module is also provided which will give temperature of detected location.
- Details of detected location will be provided such as restaurants, hospitals, picnic spots, temples etc available on the location.
- In short, system is going to provide location detection, detailed information of detected location and rating system for movies in detected location.

Ex:- Suppose Nasik city is detected, then it will provide all information regarding picnic spots, hospitals, theaters etc available in nasik. Then system will provide rating system for movies in detected city Nasik.
Table 1:
Example of rating system of movie

<table>
<thead>
<tr>
<th>Maharashtra State</th>
<th>Top movies</th>
<th>Avg. Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mumbai</td>
<td>Duniyadari</td>
<td>70%</td>
</tr>
<tr>
<td></td>
<td>Roy</td>
<td>55%</td>
</tr>
<tr>
<td></td>
<td>Baby</td>
<td>68%</td>
</tr>
<tr>
<td></td>
<td>Lokmanya</td>
<td>63%</td>
</tr>
<tr>
<td>Pune</td>
<td>Lokmanya</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td>Baby</td>
<td>77%</td>
</tr>
<tr>
<td></td>
<td>Mitawa</td>
<td>45%</td>
</tr>
<tr>
<td></td>
<td>Duniyadari</td>
<td>70%</td>
</tr>
</tbody>
</table>

VII. RESULT ANALYSIS
VIII. CONCLUSION

This system is going to provide 2 functionalities i.e., after location detection, it will provide details of detected location and also going to provide rating system.

LARS* faces a problems which are unsolved by traditional recommender systems by dealing with three types of location-based ratings: spatial ratings for non-spatial items, non-spatial ratings for spatial items, and spatial ratings for spatial items. LARS* employs user partitioning and travel penalty techniques to support spatial ratings and spatial items, respectively.

REFERENCES


