International Journal of Modern Electronics and Communication Engineering (IJMECE) ISSN: 2321-2152
Volume No.-7, Issue No.-4, July, 2019

User Perspective Based APP Recommendation System

Sarun K T 1st
Department of Information Technology
Government Engineering College
Idukki, Kerala, India
sarunkt777@gmail.com

K R Remesh Babu 2nd
Department of Information Technology
Government Engineering College
Sreekrishnapuram, Palakkad, Kerala, India
remeshbabu@yahoo.com

Praveen A N 3rd
Department of Information Technology
Government Engineering College
Idukki, Kerala, India
praveenan79@gmail.com

Elizabeth Shelry 4th
Indian Institute of Information Technology and Management-Kerala
Technopark, Trivandrum, India
rsherly@iiitmk.ac.in

Abstract: Now huge numbers of smart phone applications are part of our daily life. Since there are several applications with different features, users often experience difficulty in selecting the best mobile application for their exact purpose. It is also observed that and these applications may not be useful for particular user needs. Identifying an exact application for the user needs from a large catalogue is pretty hard. Hence a good recommendation system plays a key factor in selecting these applications. It is also a fact that for a user it is time consuming and burden to read all the reviews about each search suggestion. These reviews may not be always genuine, and fraudulent reviews are very common. In the proposed user perspective based APP recommendation system (UPBARS) suggests applications based on individual interests. Here usage statistics of each application is considered as the measure of user likeness towards an application. In this proposed method collaborative filtering is used to identify similar users based on their likeness. Once similar users are clustered, application liked by the clustered users form a candidate for the recommendation. Each candidate is ranked using a deep neural network model based on user perspectives as input features. The feature includes privacy preferences, size, functional category, in-app interests, ad-content, ratings, and user demography. The score for each application represents the probability that a user likes the application. The experimental result shows the effectiveness of the proposed method.

Keywords: Collaborative filtering, Neural network, User perspective, Recommendation, Personalization.

I. INTRODUCTION

Since Smartphone becomes very common, the mobile APP markets have experienced explosive growth. Now the number of mobile application in popular platforms like in Google play store has the exceeded 2.6 million. Users can search and find suitable App from these locations. However, finding an exact application for a user from this large catalogue is a problem. Also selecting the best one for the user from these redundant featured application lists is a complex task.

The common application platforms like play-store proposed certain mechanisms such as rating, reviewing and ranking for each individual application in their platform. These parameters help the user to make a decision on whether to consider a particular application or not. The current scenario is that users have to input the required content and the recommendation system generates a list of applications that matches the requirement according to the rank calculated. The major problem faced by the recommendation system is to parse the user input to the applications. Another factor is that App developers try to gain profit by performing some fraudulent mechanism and manipulate these reviews, ranks, and ratings of applications. Several surveys show 3 out of 10 popular Apps are fraudulent hence these reviews, ratings, and ranks are not justifiable parameters for application recommendation. Considering all the problems mentioned above some efficient mechanism for App recommendation is needed. Here comes the importance of a context aware app recommendation mechanism. Te existing context aware recommendations consider general user requirements. This is impractical since specific user requirements are different. Also, context aware recommendation should be independent of reviews, ratings, and rankings. The fraud manipulation has nothing to do in context-aware recommendation. Another fact is that most of the third party applications have privacy privileges to user's sensitive data. So users choose app not only based on functionality but also based on the privacy preference.

There exist some context-aware recommendation systems which mainly consider general contextual features such as location, demography etc [15][19]. But these methods not focused to the user perspectives. Thus user a perspective based recommendation system (UPBARS) which consider user preferences and recommend applications that matches these requirements is necessary. However previous download history is mere indicator of personal interests of a user. Sometimes the downloaded applications are unsuitable for their required purpose. The proposed UPBARS reads the personal interest from application usage statistics. Based on each individual application usage statistics in a particular period of time, the system determines a

person's interest in an application. Then broad collaborative filtering is used to generate a candidate application. The candidate applications are then ranked based on user provided preference list, which includes privacy preferences, functional categories, etc.

This paper aims to design and develop an efficient and effective personalized APP recommendation system. The rest of the paper is organized as follows. Section II describes related works, and section III deals with a detailed overview of the proposed system. Section IV describes the experimental setup and result analysis. Finally, this paper concludes in section V.

II. RELATED WORKS

A detailed review is carried out to identify the problems and challenges existing in the current application recommendation for Smart-phones. Also, the advantages of context-aware recommendation over traditional recommendation system and major challenges in the existing context-aware recommendation systems are considered. An APP with a higher rank on leader-board makes large revenue [1]. Therefore the developers try to boost their application and also try to manipulate rank through fraudulent mechanisms. The work in the paper [1] proposes a mechanism to identify fraudulent applications. Based on statistical observation of historic rankings, it is observed that the fraudulent APP has a different ranking behavior. Observation done on play store data reveals that 3 out of 10 popular APPs are subjected to rank, review or rating based manipulation. Thus it can be confirmed that Review, Rating, and Ranking alone can be justifiable parameters to download an application.

There are some studies [2][3][18][19] been done on context based recommendations. The paper [2] proposes a mobile application category recommendation framework using Bayesian networks. The location, stance, time and date are the contextual features considered. Location and stance information is collected from the Smartphone itself using built-in sensors.

Papers [3] consider user interaction with downloaded applications apart from the location context. Thus similar applications can be recommended. A logger is maintained to monitor user location and usage time of each application. The logger periodically uploads collected data to the server. The server responds with the recommendation which matches the context. There may exist millions of applications within a category thus category recommendation is not sufficient. Location, stance and date and time alone can't be the distinguishable factor to determine user interests.

Another development is personalization in recommendation many works contribute to generates personalized recommendations [4][6]. The paper [4] proposed a personalized application recommending system. The user downloading history is a weak indicator of user likes, therefore usage statistics of individual applications considered to determine whether the user liked the application. Collaborative filtering (CF) is used to make automatic predictions about the interests of a user by collecting many users' interests. One slope prediction algorithm predicts user score for each application and top scored are recommended.

Multiple relevant features can make efficient recommendation rather than single feature. The research in [8] the authors integrate multiple features together using tensor structure. Thus complex relationships can be learned and it can be used to predict the user rating on applications.

User not only considers application features but also privileges required by the applications. Most case third party application may request unwanted access privileges which put user privacy at risk. Many works focus on user privacy based recommendations. The paper [9] proposes a method to identify potential security risk of application from the mobile apps permission requests. After filtering popular applications are recommended.

Paper [10] suggests an efficient recommendation for users by integrating user interest features and privacy preferences together as a latent vector. Then the probabilistic matrix factorization method is used to make a personalized recommendation. Paper [11] considers various constraints such as battery power, network limit along user preference and popularity of application for recommendation.

III. SYSTEM DESIGN

The user perspective based APP recommendation system (UPBARS) considers user preferences and recommends applications those match. Usage statistics of each application provide what all applications the user has liked. The interaction time with each app measures likeness towards the application. Thus similar applications to the liked ones can be recommended to the user. To make personalize effective recommendation other contexts that user preferred can be considered. Here user needs to explicitly provide his preferences based on certain attributes. The user liked application and preference alone form the input for the recommendation algorithm. The algorithm learns by deep collaborative filtering from multiple users then scores each application based on the preference of the user. The score determines the predicted range of likeness towards the application by the user. Top scored application can be recommended to the user. The attributes to define the user preferences are, interests in paid APPs, privacy preferences, user willing to support in-app purchase, users interest towards pop up ads, memory preferences i.e.; interested in lite versions of

applications, user current functional category of interest, rating preference if any, device capability based on android version and user age.

Once the user preferences are obtained the context can be used to identify the apps that match exactly. For that, a deep ranking neural network model is proposed. The overall architecture of the model is shown in Figure1. Here, the user required to explicitly provide their preference context to the system. Collected context from different users is stored in the system which became one of the training input for the ranking model. User liked application lists are obtained from client-apk installed on user's mobile devices. The client system measures usage statistics of downloaded applications and export collected data to the system. Liked application list and preference contexts of different users are stored in the system. Based on the stored data a deep ranking classifier can be trained. The classifier can predict the probability that user going to like an application when the context and app id is given.

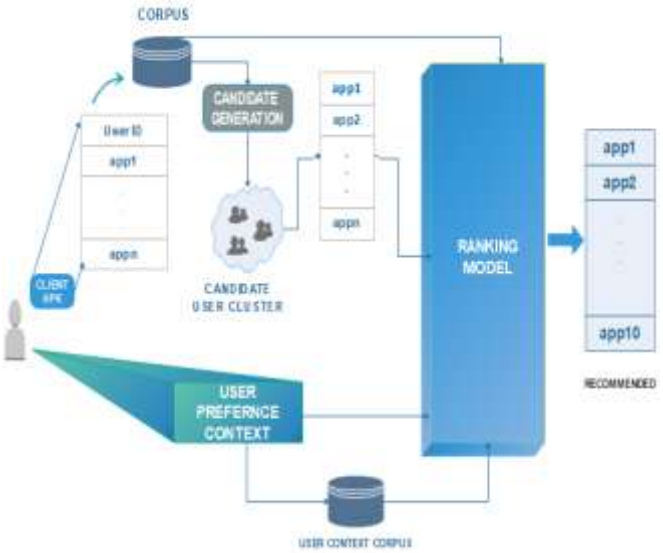

Figure1. UPBARS- Architecture

However, the number of total applications in the corpus is too large thus rank each one is a burden and time-consuming. Thus a smaller set of applications need to be considered. For that broad collaborative filtering can be used to obtain a candidate set. That is similar users can be clustered and application liked by similar users became the candidate. The candidate list consists of a limited number of applications. Each application in the candidate can be ranked to the probability that the user going to like it. Top ten ranked applications in the candidate can be recommended to the user. Thus proposed system consists of three functional modules a client-apk, candidate generation, and ranking module.

A. User Preference Contexts

User preference contexts determine user interests. To identify user preferences different attributes are considered. The first attribute is usage statistics which determines user liked applications. To obtain usage statistics of user downloaded application an additional client system is required. Remaining contexts need to be explicitly expressed by the user, which are:

- Paid or Free - Attribute determines user interested in paid applications.
- Device capability - It is the android version of the user's smart phone. Thus applications that support the user's device can only be recommended.
- In-app purchase - Full features of application would not be available until purchased such purchases are called as in-app purchase. Attributes determine the user interest towards in-app provided applications.
- Ad content - Developer to gain profit enable popup ads in their application and in some cases it is annoying to see ads during functioning of applications. This attribute specifies the user's willing to compromise on popup ads.
- Size preference - Some users may interest in graphics-rich applications while others may prefer lite applications. Size preference is bounded to below 20MB, 50MB, and 100MB.
- Functional category - Even businessman tries entertainment application to kill time that is, demography alone not specify users current interest. Therefore the user needs to specify the current interested category. Functional categories considered are Business, Art and design, Game, Education, Books and reference, Communication and Entertainment.
- Rating preference - This attribute specifies whether the user interested in the rating of the application, i.e., user interest in platform provided rating.
- User age - This demographic feature is to recommend applications particular to an age group. Also, if sparse data is available to generate candidate age group based candidate applications can be considered.
- Privacy preferences - User not only considers application features but also privileges required by the application. Most case third party application may request unwanted access privileges which put user privacy at risk. The different privileges considered are location access, media read and write permission, storage read and write permission, contact read and write permission and camera access. This attribute determines the access that user may compromise.

User liked applications are the input for collaborative filtering thus to generate the required candidate. Usage statistics is the metric to measure user likeness towards applications. A user interacts more with an application in a period of time de-

termine he likes the application very much. Thus to identify user liked applications usage statistics of each application need to be obtained periodically. So each user needs to install a client apk which periodically reads usage statistics of individual applications within the user's phone. And the client apk export collected data to the system via common communication channels. An example of one such a client system is "Stay-free phone usage tracker" which is openly available in Google play store. It has the functionality to export collected usage statistics as csv file. Once such data is available by simple preprocessing user liked applications can be obtained.

B. Candidate Generation

User preference alone needs to recommend the application to the user. The ranking algorithm can predict the score that the user going to like the application. Based on score recommendations can be done. However, there are millions of application exists in the catalog ranking each is not a straight forward mechanism. Thus needed a smaller set among which recommendation can be made such a set is called as candidate set. Candidate cannot be picked at random since it affects the recommendation accuracy. One method is broad collaborative filtering i.e.; cluster similar users. A user is similar to another if they have something in common. Here a user has the same likeness that is, both users liked the same application can be considered as similar users. Such similar users can be clustered together. Each cluster consists of a number of users and their liked applications. Whenever a user needs a recommendation the user already tried application is subtracted from the applications in the cluster, forming the required candidate.

For example, if the cluster constitutes of m users such as U_1, U_2, …, U_m. To recommend an application to the user u1 the set of application R is generated. R contains applications that are not been used by u1 but liked by other users in the cluster. Let S_1, S_2,…,S_n be the set of application liked by corresponding users U_1,U_2,...U_m in the cluster.

$R = \{i|$ for all i $\in \{S_2||S_2||...||S_n\} - \{S_1\}\}$ where, R is the required candidate.

If the user has no liked application to list suppose the user has bought a new smartphone or he is not willing to share his usage statistics. Such case generating candidate using broad collaborative filtering is not possible. However, each application has specified age group so candidate can be generated based on users age. The user needs to provide his age. Based on the age-group a list of applications can be returned as a candidate.

C. *Ranking*

The ranking algorithm is a feed-forward neural network as shown in Figure2. Which consist 2 inputs one main and an auxiliary input. High dimensional categorical application identifiers can be passed to main input and can be converted to low dimensional vector space using embedding. The contextual preferences need to be encoded and can pass to the auxiliary input. Multi variant features can be one-hot encoded. The binary feature can be represented as '1' if the user allows the feature else represented as '0'. The deep model is used since it has good generalization capability and less feature engineering is required. The prediction problem can be converted to a binary classification problem in which the class '1' indicate user like the application while class '0' not going like the application. Ones the network has trained and optimized user context and candidate for the user can be given as input to the network. The deep model gives a score between 0 and 1 to each application in the candidate. Top scored applications in the candidate are better matching one for the specified contexts. Thus high scored 10 applications from a candidate can be recommended to the user.

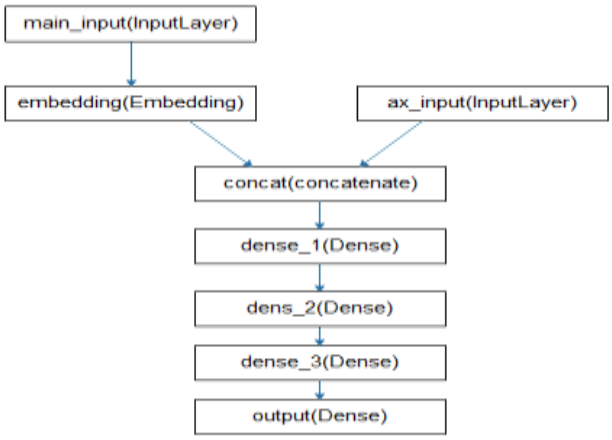

Figure2. Network model

IV. EXPERIMENTAL SETUP AND ANALYISIS

The proposed system is tested with data collected from Google play-store and kaggle-playstore dataset. Which contain application context such as APP name, device support, size, rating, paid or free, ad content, various privacy access required by the application, and category of the application to which it belongs too. 3000 data-points are used to train and test the neural network. The collected data require preprocessing, categorical data are one-hot encoded. Binary value data are converted to 0 or 1. The size feature column represented as below 20, below 50, below 100. Each unique app has given an id which is embedded and represented as low dimensional vectors. The rating column is converted to 3 feature columns such as above 3, above 4 no rating preference. Each data-point has labeled as 1 or 0, based on whether the user with particular context liked the application or not. Label 1 represents the user

with the corresponding context has liked the application. The entire data is split into train and test set further test set is partitioned into validation and test set. Using keras functional model corresponding required neural network architecture is created and the application id and remaining context are input as a separate layer to the network. The network has trained for 100 iterations with a batch size of 32 and on 2400 samples. The metric such as loss, accuracy on the validation set and training set are plotted. The network responds with 93% accuracy and 0.1329 cross-entropy loss on training data and 95% accuracy and 0.9298 loss on 300 validation samples. And the network is optimized to avoid over-fitting.

Usage statistics of downloaded applications are exported using "Stayfree" client apk. The data received as csv file the usage time of application sorted to obtain the liked application names then corresponding ids are retrieved. Users with common likes are clustered. And the list of applications that the similar user has liked is retrieved which is the candidate for the recommendation. The user contexts are explicitly provided and corresponding is passed to predict function. The learned model gives a score for each candidate which is stored in the dictionary structure. The corresponding dictionaries are sorted to obtain top scored applications and among first 10 APP are recommended.

The proposed method is tested to evaluate the effectiveness of the recommendation made. Result analysis is performed based on the parameter such as precision, recall accuracy, intra-list similarity, and personalization. To evaluate the system confusion matrix are created for a different number of user's then precision, recall, and accuracy is calculated. Intra-list similarity score gives the consistency of the recommendation system. To find the intra-list similarity among recommended APPs cosine similarity between the applications is considered. For that each application is represented as a vector of its contextual features. Personalization is measured as 1 -cosine similarity between applications recommended for different contextual users. Personalization score defines the personalization in a recommendation that is; no common applications are recommended to different users.

A. *Precision, Recall and Accruacy*

Confusion matrix is a table that is used to describe the performance of a model on set of data in which true values are known. It is a two dimensional matrix in which one dimension represents true class and other predicted class by the model. It consist of true positive (TP), true negative (TN), false negative (FN) and false positive (FP). Here confusion matrix for 100, 200, 300...upto 700 users from the test set is considered and precision, recall, accuracy is measured using equations.

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FP}$$

$$accuracy = \frac{TP}{TP + FN + FP + TN}$$

Precision is the proportion of the predicted positive cases that were correct. Figure3 shows a graphical representation of precision on the different number of users and it pertains to an average value of 99 percent. The recall is the proportion of positive cases that were correctly identified. Figure4 shows the graphical representation of recall on a different number of users and it pertains to an average value of 93 percent. Accuracy is the proportion of the total number of predictions that were correct. Figure5 shows the graphical representation of accuracy on a different number of users and it pertains to an average value of 95 percent.

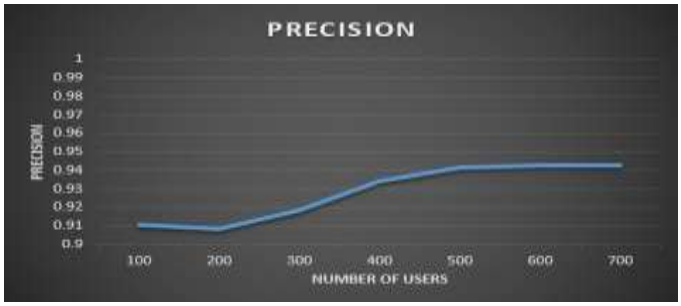

Figure3. Precision on 700 users

Figure4. Recall 700 users

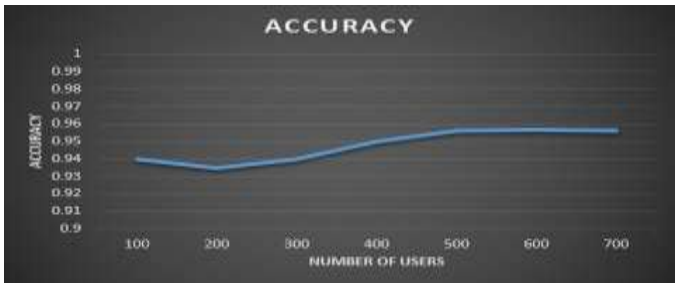

Figure5. Accuracy700 users

B. *Intra-List Similarity*

Applications recommended to a user must have similar contextual characteristics then only we could conclude the recommending engine is making effective predictions. That is, a higher similarity measure should exist between the applications recommended to a particular user. The parameter intra-list similarity measures the consistency of recommendation. Once the preference of the user is obtained and the corresponding candidate generated the system recommends 10 APPs to the user. To find intra-list similarity, each application is represented as a vector of its contexts. Then similarity among these 10 APPPs is measured using cosine similarity and tabulated. Suppose A and B represent corresponding vectors then cosine similarity can be measured as:

$$\text{Cosine similarity} = \frac{A.B}{|A||B|}$$

Here, five different user preferences is considered and their corresponding recommendations are tabulated. The values within the table show the similarity between applications. The table 1 shows recommendation for user 1 and final column represents the average similarity between one application and all other in the list. Similarly, tables for 5 different users are generated. From the table, it can infer that there is more than 90% average similarity exist among each intra-list APPs. Thus can conclude recommended application for a user has a higher similarity so, the recommendation is consistent.

C. *Personalization*

The recommendation must be different for different users this is called personalization. Each individual has his own contextual preferences thus recommendation should be personalized. That is, users having different preferences should have no common recommendations. To measure the personalization for users with different context preferences 1- cosine similarity between the recommended lists are taken. Each recommended application is represented as a vector of contextual features. Then 1-cosine similarity between recommended applications for different users is measured and tabulated. The table 2 shows the personalization score between the application recommended to user1 and user 2. Each column represents applications recommended to user 2 (u_2) and rows in the table represent the application recommended to user1 (u_1). The values within the field show 1-cosine similarity between the applications. Similarly, tables are generated to give personalization value between user 1, user 3, and user2, user 3 respectively. From the tables, it can be concluded that no identical APPs are recommended for different contextual users thus personalization exists. And an average 50% personalization score exists between different applications recommended to different users. The score represents applications recommended to different contextual users are not similar and 50% similarity suggests even different applications should have some common features.

Table 1. Similarity Measure for User 3

user=3	app1-45	app2-47	app3-221	app4-73	app5-72	app6-44	app7-184	app8-71	app9-70	app10-241	Average
app1-45	1	0.985714	0.969528	0.958572	0.969528	0.985659	0.968929	0.986772	0.984824	0.969378	0.97789
app2-47	0.985714	1	0.979169	0.958572	0.968572	0.995418	0.968929	0.998381	0.993859	0.979169	0.982778
app3-221	0.969528	0.979169	1	0.971654	0.968572	0.990375	0.990375	0.986682	0.991408	0.999904	0.984767
app4-73	0.958572	0.958572	0.971654	1	0.986772	0.973064	0.970441	0.968253	0.974539	0.970184	0.973205
app5-72	0.969528	0.968572	0.968572	0.986772	1	0.969378	0.969528	0.958572	0.956757	0.969528	0.971721
app6-44	0.985659	0.995418	0.990375	0.973064	0.969378	1	0.985714	0.999245	0.999886	0.989972	0.988871
app7-184	0.968929	0.968929	0.990375	0.970441	0.969528	0.985714	1	0.979985	0.987541	0.989017	0.981046
app8-71	0.986772	0.998381	0.986682	0.968253	0.958572	0.999245	0.979985	1	0.998543	0.986678	0.986311
app9-70	0.984824	0.993859	0.991408	0.974539	0.956757	0.999886	0.987541	0.998543	1	0.99085	0.987821
app10-241	0.969378	0.979169	0.999904	0.970184	0.969528	0.989972	0.989017	0.986678	0.99085	1	0.984468

Table 2. Personalization between User 2 and User 3

u2/u3	app1-163	app2-155	app3-158	app4-156	app5-278	app6-157	app7-131	app8-147	app9-149	app10-133
app1-178	0.131757	0.240743	0.217376	0.240743	0.112588	0.240743	0.230516	0.333114	0.230516	0.25
app2-205	0.131757	0.240743	0.217376	0.240743	0.112588	0.240743	0.230516	0.333114	0.230516	0.25
app3-221	0.045573	0.165378	0.139691	0.165378	0.046179	0.165378	0.263158	0.315789	0.210526	0.281815
app4-319	0.122942	0.233035	0.209431	0.233035	0.073698	0.233035	0.23825	0.310798	0.23825	0.257538
app5-351	0.122942	0.233035	0.209431	0.233035	0.043817	0.233035	0.274524	0.274524	0.274524	0.292893
app6-184	0.109201	0.221019	0.197045	0.221019	0.046179	0.221019	0.315789	0.315789	0.210526	0.333114
app7-241	0.045573	0.165378	0.139691	0.165378	0.046179	0.165378	0.263158	0.315789	0.210526	0.281815
app8-210	0.131757	0.240743	0.217376	0.240743	0.112588	0.240743	0.230516	0.333114	0.230516	0.25
app9-211	0.132528	0.039129	0.009557	0.039129	0.133079	0.039129	0.138943	0.18678	0.091107	0.160746
app10-311	0.163758	0.122473	0.095466	0.122473	0.088315	0.122473	0.135357	0.204528	0.169943	0.15725

V. CONCLUSION

The rapid growth of the mobile application industry introduced a challenging platform to choose the exact application that the user needs. Existing input based filtering and 3R based recommendations subjected to manipulation. As developers intent to make more profit they began to boost their application and also perform fraudulent mechanisms to make the application top in the chart. Therefore context-aware recommendations that not considering 3R factors are a better option for APP recommendation. UPBARS system tries to include personal preference context along with general context to obtain good results. While other systems consider only single attributes UPBARS includes multiple contexts from the user. The result analysis shows the system is capable of making an effective recommendation with high accuracy, precision, and recall. High value for intra-list similarity among recommended application for a user gives correctness of the recommendation. The system is capable of making a personalized recommendation based on user context, which is evaluated by the personalization metric. Most of the exiting context-aware recommendations subjected to cold start problem UPBARS overcome those by considering user age and generating candidate based on the age group.

REFERENCES

[1] Hengshu Zhu, Hui Xiong, Yong Ge, Enhong Chen, "Discovery of Ranking Fraud for Mobile Apps," IEEE Transaction on Knowledge and Data Engineering, vol. 27, pp. 74 -87, April 2014.

[2] Woo-Hyun Rho, Sung-Bae Cho, "Context-Aware Smartphone Application Category Recommender System with Modularized Bayesian Networks," in Proc. IEEE 10th International Conference on Natural Computation, 2014.

[3] Matthias Bohmer, Moritz Prinz, Gernot Bauer, "Contextualizing Mobile Applications for Context-aware Recommendation," In Adj. Proc. of Pervasive, 2010.

[4] Bo Yan and Guanling Chen, "AppJoy: Personalized Mobile Application Discovery", in Proc. 9th Int. Conf. Mobile Syst., Appl., Serv, 2011.

[5] Adithya Raam, Jeya Balaji, Sreyas Naaraayanan, "CAARD - Context Aware App Recommendation and Delivery using Decision Support Systems," Indian Journal of Science and Technology, 2016.

[6] Cheng Yang, Tao Wang, Gang Yin, Ming Wu, Ming Xiao, Huaimin Wang, "Personalized Mobile Application Discovery," in Proc. 1st International Workshop on Crowd based Software Development Methods and Technologies, 2014.

[7] Hengshu Zhu, Enhong Chen, "Mining Personal Context-Aware Preferences for Mobile Users, " in Proc. IEEE 12th International Conference on Data Mining, 2012.

[8] Tingting Liang, Lifang He, Chun-Ta Lu, Liang Chen, "A Broad Learning Approach for Context-Aware Mobile Application Recommendation", IEEE International Conference on Data Mining, 2017.

[9] Hengshu Zhu, Hui Xiong, Yong Ge, Enhong Chen, "Mobile App Recommendations with Security and Privacy Awareness," in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, 2014.

[10] Bin Liu, Deguang Kong, Lei Cen, "Personalized Mobile App Recommendation: Reconciling App Functionality and User Privacy Preference", in Proc. Eighth ACM International Conference on Web Search and Data Mining, February 2015.

[11] Konglin Zhu, Zexuan Liu, Lin Zhang, Xinyu Gu "A Mobile Application Recommendation Framework by Exploiting Personal Preference with Constraints", Mobile Information Systems, 2017.

[12] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, "Wide & Deep Learning for Recommender Systems," in Proc. 1st Workshop on Deep Learning for Recommender Systems, June 2016.

[13] Paul Covington,Jay Adams, Emre Sargin, "Deep Neural Networks for YouTube Recommendations", in Proc. 10th ACM Conference on Recommender Systems, 2016.

[14] Matthias Bohmer, Lyubomir Ganev, Antonio Kruger "AppFunnel: A Framework for Usage-centric Evaluation of Recommender Systems that Suggest Mobile Applications", Proceedings of the 2013 international conference on Intelligent user interfaces,March 2013.

[15] Bilgehan Erman, Ali Inan, Ramesh Nagarajan, "Mobile Applications Discovery: A Subscriber- Centric Approach", Bell Labs Technical Journal, vol. 15, March 2011.

[16] Francois Chollet, "Deep Learning With Python", Manning Publications Co. Greenwich, 2017.

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, "Neural Collaborative Filtering", Proceedings of the 26th International Conference on World Wide Web, April 2017.

[18] Xinxi Wang, David Rosenblum, Ye Wang "Context-Aware Mobile Music Recommendation for Daily Activities", ACM multimedia, 2012.

[19] Jie Bao, Yu Zheng, Mohamed F. Mokbel, "Location-based and Preference-Aware Recommendation Using Sparse Geo-Social Networking Data", Proceedings of the 20th International Conference on Advances in Geographic Information Systems, November 2012.

[20] Sai Wu, WeichaoRen, Chengchao Yu, "Personal Recommendation Using Deep Recurrent Neural Networks in NetEase", in Proc. IEEE 32nd International Conference on Data Engineering," May 2016.

AUTHOR'S BIOGRAPHIES

Sarun K T obtained his Bachelor's degree in Information Technology from Cochin University of Science and Technology, Kerala in 2017 and currently pursuing Master's degree in Network Engineering from A P J Abdul Kalam Technological University, Kerala. His area of interest include Software defined Networks, Machine Learning and Artificial Neural Network Applications.

K R Remesh Babu is currently working as Associate Professor and head in the department of Information Technology, Government Engineering College Sreekrishnapuram, Palakkad, India. He has published several research papers in International journals and conferences. His research interest includes Distributed and Cloud Computing, Internet of Things, Wireless Sensor Networks, and Big Data Analytics.

Praveen A N is an Assistant Professor in the department of Information Technology at Government Engineering College, Idukki. His area of interest includes embedded systems, machine learning and artificial intelligence.

Elizabeth Sherly is Professor and former Director of the Indian Institute of Information Technology and Management-Kerala (IIITM-K), Technopark, Kerala, India. She took her Ph.D. in Computer Science from University of Kerala in 1995 and has more than twenty five years of teaching experience. Her areas of research involve mainly Pattern Recognition and Machine Learning, Artificial Neural Networks, Medical Image Processing and Computational Linguistics.