Rating Prediction for Mobile Applications via Collective Matrix Factorization Considering App Categories

Nalinsak Gnotthivongsak1,* and Dong Jun Huang 2
1,2School of Computer Science and Technology, Central South University Changsha, China.

Noy_gtvs@hotmail.com1*; Djhuang@csu.edu.cn2

Abstract. The rapid development of Internet and mobile communication, smart phone, tablet computer and other mobile devices become widespread, they have become an indispensable part of life, changing the human lifestyle, way of working and learning style, it brings great convenience and fun to us. This paper mainly discusses the status of personalized recommendation system, mobile application and analyse the main problems of recommendation system. This research proposes a rising mobile application domain, which uses web crawling technique to get explicit feedback from a real-world mobile application market. The explicit feedback is the users' ratings data on Apps and the corresponding category information. To establish a mobile application recommendation model based on user interest and categorization information and propose a novel collective matrix factorization algorithm. Through the crawled data, the improved algorithm, and the traditional collaborative. The traditional mobile application recommendation systems are mostly based on keyword search, popularity, download usage and categories, thus they do not make personalized recommendation to users, and invalid to find applications that users are interested in. Through the user's historical behaviour, a personalized recommendation system can be established to recommend the applications that fit user's interests effectively. This paper is based on the collective matrix factorization using user interest such as user’s ratings and category information, designs a personalized recommendation system for mobile applications by using the relationship between them.

1. Introduction
In recent years, with the rapid development of internet and mobile communication, smart phone, tablet computer and other mobile devices has played an important role in people life, these devices have become an indispensable part of life, changing the human lifestyle, way of working and learning style, it brings great convenience and fun to us. It also creates economic opportunities for app developers, companies, and marketers. At present, users are able to access a substantial number of apps via App Stores like Apple Store and Google Plays. Furthermore, the selection available in app stores is growing rapidly as new apps are approved and released daily. While this growth has provided users with a myriad of unique and useful apps.
Since the emergence of Apple and Android apps markets, at the end of March 2017, Android applications had reached more than 2.8 million, while Apple had more than 2.2 million apps [1].
Therefore, the explosive growth of mobile application increases the difficulty of users to discover the apps that fit their interests.

Mobile personalized application recommendation system [2] can better solve the following problems, it is for not only developers to provide more opportunities for exploring the applications, so more applications can be seen by users and it is a good way to reduce the cost of the promotion of the developers. Furthermore, it takes the initiative to help users filter some unrelated information, recommend the applications which users might be interested in, it better reduces the user screening application time costs, allowing users to find the desired application more easily.

This research aims to build mobile application recommender system for customers using collective matrix factorization and update rule use the stochastic gradient descent algorithms to make predictions and recommendations and experimentations using a real-world dataset from Apple Store and compare with state-of-art recommender system algorithms. Matrix factorization [3] is one of model-based method that becomes extensive and widely used because it is able to perform high scalability and accuracy, it also gets success in Netflix challenge, which is a competition for implementing movie recommender system [4], matrix factorization uses talent factor of users and items in share talent space of user-item matrix and calculate by inner product [5].

This research paper is divided into the following: section 2 presents the related works on recommendation system; section 3 discusses the system implementation and experimental setup for proposed recommendation system and mobile app predictions; section 4 analyses the results of the experiment involving the system; section 5 provides a conclusion.

2. Related works

2.1. Recommender system

Search engine requires the explicit needs of users, and it finds what they want when they cannot describe a clear message or when it is difficult for users to describe it with simple keywords. Google and Baidu are the examples of popular search engine, these websites provide billion of information and numerous users around the world. In addition, there are some websites that use category to make it easier for users to find what they need by category. With the continuous expansion of the Internet, this method can only cover the popular information and does not effectively meet the needs of users. There are some examples of category site, such as Yahoo, DMOZ, hao123.

Recommender system can solve the deficiencies of above techniques [6] [7]. It is a tool for contacting users and information to help users find useful information. The users do not need to provide explicit requirements. It builds a model of interest to users through past historical behaviour, and then proactively recommend information that will satisfy their interests.

In order to make a recommendation, the system should collect the information of user interaction, where user interaction divided into two types, there are explicit and implicit feedback [8] [9]. Explicit feedback obtains user preferences data from displayed and ready to use information, such as users’ ratings. Implicit feedback obtains user preferences data learned and inferred from user actions. For example, the number of clicks on web page, usage times, user searching pattern and so on [10].

Recommendation system (in Figure 1) such as YouTube that is a web application giving suggestion to the user about some relevant videos based on user browsing’s histories. Another example is Facebook, the system will suggest some friends based on concept “People you may know” by analysing the common friends of friends. Taobao is one of Chinese e-commerce website. There are millions of products sold in there. The system recommends some products based on users’ purchase histories, the clicks, and ratings for products.
2.2. Personalized Recommender system

Personalized recommender system was first proposed by Malone and other researchers in 1987, personalized recommender system is mainly divided into three categories: cognitive filtering, social filtering, and economy filtering [11]. The informal concept of the currently widely recommender system is given by Resnick and Varian in 1997 [12]. It is the use of e-commerce site to provide customers with product information and recommendations to help users decide what products should be purchased, imitate salesmen to help customers complete their purchase process.

A general personalized recommendation system flow consists of three parts:
- Input module for user information collection, mainly used to collect the user's personal input. The target user's personal input refers to the information related to the target user, including the user's personal information, item attributes, user historical behaviour, such as reviews and rating information, page tour history, shopping basket information, purchase history, etc.
- Recommendation module is the core module of personalized recommender system [13], it is directly related to the performance of the recommender system. The main function of the recommendation module is to predict the user’s evaluation of items based on user information, item information, and other information that is used for recommendation.
- Output module is mainly responsible for the results according to the recommendation module to determine what to show to the user and how to display content.

**Figure 1.** Recommendation on YouTube, Facebook and Taobao.
Figure 2. Architecture of Recommender System.

The recommender algorithm [14] is the most critical and core part of the recommender system, which largely determines the performance of recommender system [15]. At present, the recommender algorithms are mainly divided into 3 types: content-based approach, collaborative filtering and hybrid approach. Currently, recommendation system that mentioned in Figure 2 is classified into two types, namely traditional and context-aware recommendation system. Traditional recommendation system [16] does not use contextual information while processing. Among of different approaches in a recommendation system, collaborative filtering is one of the most popular and commonly used traditional recommendation system techniques in many industries [17] [18].

2.3. Context-aware recommender system

Context-aware recommender system is classified as three types [19] [20]: Contextual Prefiltering, Contextual Postfiltering and Contextual Modelling. The three paradigm of context recommender system show as Figure 3.

2.3.1. Contextual Prefiltering: in this paradigm, context information [21] is firstly utilized for filtering irrelevant set of data and selecting relevant information before showing the results of recommendations. After that, predict the rating by using any 2D traditional recommender systems according to selected data, traditional recommender systems include collaborative filtering, content-based, hybrid approaches, etc.

2.3.2. Contextual Postfiltering: in this paradigm, context information is firstly ignored before generating a predicted result, entire data is applied on 2D traditional recommender systems [19]. After getting the list of recommendations, the model will filter out the irrelevant items according to a given context that the users request.

2.3.3. Contextual Modelling: in this paradigm, context information is used in the whole process of generating recommendations, design appropriate algorithms and models to deal with multidimensional context user preferences. This approach requires the processing of high dimensional data, which is the most complex, but it can also effectively excavate the relationships between users, items, and contexts.
2.4. Previous research works.

Kaoungku et.al [22] presented a technique for association rule mining on multiple dataset to help in the association rule mining, replace association rule mining from large dataset which require high performance computer to process, a technique combined with Fact++ Reasoner to check conflicts of rule. dataset that similar efficient to the association rules from one dataset can be checked by ontology using protégé editor, the association rule dataset used Breast-cancer dataset from the UCI Machine learning repository that consists of 10 attributes and 286 data instances, the experiment was divided into three datasets using minimum support for association rule mining at 0.3 and 0.5. the results of association rule identified that the association rule from multiple datasets is missing a lot with minimum support 0.3, but one dataset is clearly different. There are some association rules from multiple datasets are still missing and effectively close association rules from one dataset with minimum support 0.5. Siddiqui et.al [23] introduced the problem of mining association rules in large relational tables containing both quantitative and categorical attributes using k-mean clustering, the success of the algorithm is mainly relied on the supervised multivariate procedure user for discretizing the continuous attributes in order to generate the rules. The proposed method avoided three main drawbacks by rule mining algorithm, production of a high number of rules, discovery of uninteresting patterns and low performance that linearly scaled the algorithm with number of records.

Shine and Kulkarni [24] proposed a fast k-medoids clustering algorithm which is used for hybrid personalized recommender system that provide clustering of user-item rating matrix and the recommendations for the active user with goof quality rating using similar measures. The result using Iris data showed that the proposed k-medoids clustering algorithm performed better than k-medoid, k-mean and FCM that enhance to improve the rating quality, furthermore, hybrid personalized recommender system performed better than web personalized recommender system.
3. Methodology and experiment setup
Matrix factorization as collaborative filtering technique is proposed to be able to solve prediction problem. The quality of the prediction will be improved when incorporating other contextual information such as time context, location context. Collective matrix factorization (CMF) is a technique that expand from matrix factorization models which not only uses the user-item pairs, but also considers about others contextual information. Moreover, Term frequency-inverse document frequency (TF-IDF) is also proposed to evaluate the important of words to information search and text mining in a document set or whole corpus. This research makes a collective matrix factorization from mobile application rating data and category data that consisting of two matrices such as user-app matrix and item-category matrix. Suppose $U$ is a set of all users and $A$ is a set of applications, then donate $R_{u,a}$ as a rating matrix for user and app. Data is crawled from www.itunes.apple.com which is apple app store, it means real world dataset is used to conduct in this research, each application contains application name, description, price, version, size, language, customer rating and reviews, after that the provided data will be converted into user-app matrix and app-category matrix by given the id data number for each user, app and category. The estimation of this matrix will be donated $\hat{R}_{u,a}$. $R_{u,a}$ is observed matrix for every $(u,a) \in U \times A$. The goal of mobile application prediction is to estimate missing values in the user-app matrix which incorporate in matrix $\hat{R} \in \mathbb{R}^{U \times A}$. To do this, we initially decompose the user-app matrix into $P$ and $Q$. $P$ represents the relation between user and talent factor. $Q$ denotes the relation between item and talent factor. The inner product of these two vectors can define as below:

$$\hat{R}_{u,a} = p_u q_a^T$$

(1)

For the second matrix denotes $S_{a,c}$ as app-category matrix, $A$ is set of app and $C$ is set of categories. $Q \in \mathbb{R}^{U \times M}$ is observed matrix for every $(a,c) \in A \times C$. The estimation of this matrix will denotes as $\hat{S}_{a,c}$. The app-category matrix is decomposed into $Q$ and $V$. $Q$ represents the relation between app and talent factors, $V$ denotes the relation between category and talent factors. The inner product of these two vectors can define as below:

$$\hat{S}_{a,c} = q_a c_c^T$$

(2)

Furthermore, Due to the imbalanced distribution of the number of each category in the data set (Figure 4), some users usually like to download one category of applications more than others, thus the system will normally recommend that kind of application to the users, ignoring the recommendation of other categories. It is found that the number of downloads does not fully represent the user’s preference for
the application, the rating for the application is also critical. For example, a user like to download games, but on among of those games, there are some games which have low rating, the system still recommends it to user because of the high distribution of games, but this user also likes other categories of application, but system cannot capture, when we utilize category information into the system. Therefore, this research uses TF-IDF technique to reflect the importance of each application in each category.

The concept of TF-IDF used weighting technique based on statistical methods to identify the number of time and word increases in proportion which a word appears in the document, in a given document, TF referred to the frequency at which a given word appears in the document, as follows:

\[
TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,k}}
\]  

(3)

Where \(n_{i,j}\) is the number of times the word \(t_i\) appears in the document \(d_j\), and the denominator is the sum of the occurrences of all the word in the documents \(d_j\).

IDF is a measure of the universal importance of words, which reflects in the corpus, IDF of particular word can be calculated as the number of documents which contain the word divides by the number of the total number of documents, then the result is taken from the logarithm of obtained quotient as follow:

\[
IDF_j = \log \frac{|D|}{|\{j : t_i \in d_j\}|}
\]  

(4)

Therefore TF-IDF of a word can be calculated as follow:

\[
TFIDF_{i,j} = TF_{i,j} \times IDF_j
\]  

(5)

If occurrences of a word in a document have high frequency (large TF value) and the word rarely appears in other documents (IDF value is small), it can generate a high weight TF-IDF for the word, it is considered that this word has a good ability of categorization and suitable for classification.

The weight of app over category to reflect the importance of the app in specific category as following:

\[
W_{jl} = r_{jl} \times d_{jl}
\]  

(6)

\[
d_{jl} = \log \frac{|N|}{|N_l| + 1}
\]  

(7)

Where \(r_{jl}\) is the average rating of app \(j\) that belongs to category \(l\), \(d_{jl}\) indicates inverse download rate, \(|N|\) and \(|N_l|\) denote the number of all rating and the number of rating which rate category \(l\) respectively. In addition, the idea of global average is also utilized, it includes item bias, user bias, and user-item interaction to estimate the predicted rating as follow:

\[
\hat{r}_{ui} = \mu + b_i + b_j + P^T Q_j
\]  

(8)

The proposed model incorporates all the information above that is user-app matrix, app-category matrix, and the weight of app in specific category at the same time, the function can describe as bellow:

\[
L = \frac{1}{2} \left( R_y - \mu - b_i - b_j - P^T Q_j^T \right)^T \left( R_y - \mu - b_i - b_j - P^T Q_j^T \right) + \frac{W_{jl}}{2} \left\| S_{jl} - Q_j^T S_{jl} \right\|^2 + \frac{\beta}{2} \left( \|P\|^2 + \|Q\|^2 + \|V\|^2 + b_i^2 + b_j^2 \right)
\]  

(9)

Where \(J^a \in \{0,1\}^{n_a}\) represents the corresponding indicator matrix, \(J(I, J) = 1\) if user I rates app otherwise \(J(I, j) = 0\). \(\beta\) is used for controlling the influence of corresponding terms, the notation \(\|.|\|\) denotes matrix frobenius norm.
4. Experiment and results

4.1 Review Results of Memory-Based Collaborative Filtering Algorithms

As we know, memory-based collaborative filtering is classified into user-based and item-based approaches. User-based uses historical behaviours as ratings, calculating similarities between users, after that select the most similar users and predict ratings of unrated items through similarities of neighbours.

In contract, items-based measures similarities among items, then select the most similar items and evaluate ratings of unrated items through similarities of neighbours. For theses 2 approaches, parameter K which is the number of the most similar users or items significantly affects the performance of algorithms. In Table 1, it is shown the experiments of User-based and Item-based approaches with different parameter K, Pearson correlation similarity is used for calculating similarities between users of apps.

| K  | User-Based RMSE | User-Based MAE | Item-Based RMSE | Item-Based MAE |
|----|-----------------|----------------|-----------------|----------------|
| 1  | 1.678577        | 1.035736       | 1.644330        | 1.033439       |
| 2  | 1.458855        | 1.197359       | 1.427148        | 1.164023       |
| 3  | 1.495762        | 1.330652       | 1.483939        | 1.306277       |
| 4  | 1.532130        | 1.393286       | 1.518197        | 1.362328       |
| 5  | 1.558378        | 1.429831       | 1.538899        | 1.393437       |
| 6  | 1.577839        | 1.454000       | 1.558119        | 1.417002       |
| 7  | 1.595421        | 1.474183       | 1.572309        | 1.432424       |
| 8  | 1.608188        | 1.488446       | 1.588918        | 1.451195       |
| 9  | 1.619424        | 1.501155       | 1.600770        | 1.462488       |

With the growth of K value, value of RMSE and MAE of user-based and item-based collaborative filtering are continuously increased in the range between 2 to 50. When K is greater than 50, the RMSE and MAE value tends to be stable. K = 2, the algorithms achieve the best performance, with RMSE = 1.458855 and MAE = 1.197359, user-based gets the best prediction results, while item-based gets the best performance with RMSE = 1.427148 and MAE = 1.164023. As Seen from the results that illustrated in Figure 5, item-based collaborative approach does better performance than user-based collaborative approach.
4.2 Review results of model-based collaborative filtering approaches

For model-based collaborative filtering, latent factor is used to represent a dimensional matrix. Matrix factorization is one of model-based that utilizes latent factor to analyse the model and split a raw matrix into two low-rank dimensional matrices. In order to find the best performance of proposed algorithm, the first step is to find how each parameter affects the change of computation accuracy. For proposed method, there are 4 parameters, such as: iteration number, learning rate, latent factor $K$, parameter of regularized term which avoids over-fitting in loss function. Iteration number of each algorithm are set as 100, changes 3 other parameters in different numbers, in order to find to best value for each parameter.

4.2.1 Impact of different learning rate: Take proposed method as a sample, set iteration number as 100, regularized term as 0.25, latent factor $K$ as 10. Increase the value of $\lambda$ from $\lambda = 0.001$ to 0.01 and observe the impact of learning rate. From following table (Table 2), $\lambda = 0.01$ performs the best in term of the value of RMSE and MAE that indicated in Figure 6. Thus, choose this value for proposed method.

| $\lambda$  | RMSE         | MAE          |
|------------|--------------|--------------|
| 0.01       | 1.38223701705 | 1.10155025927 |
| 0.005      | 1.29276539706 | 1.0168672809  |
| 0.01       | 1.2922657131  | 1.01463114217 |
| 0.02       | 1.29521460156 | 1.01618700659 |
| 0.04       | 1.30197895376 | 1.02004122867 |
| 0.08       | 1.3099446063  | 1.03034365541 |
| 0.1        | 1.32943881012 | 1.03506696206 |

Figure 5. RMSE performance of Memory-Based Approaches changing over Different Neighbour Number (left) and MAE performance of Memory-Based Approaches Changing over Different Neighbour Number (right).
4.2.2 Impact of different latent factor. Take proposed method as a sample, set iteration number as 100, regularized term $\beta$ as 0.25, latent factor $K$ as 10. Increase the value of $\lambda$ from 0.001 to 0.01 and observe the impact of learning rate. From following (Table 3), $\lambda = 0.01$ performs the best in term of the value of RMSE and MAE in Figure 7. Thus, choose this value for proposed method.

| K   | RMSE          | MAE            |
|-----|---------------|----------------|
| 5   | 1.29272543715 | 1.01479128478 |
| 10  | 1.29226657131 | 1.01463114217 |
| 20  | 1.29323629339 | 1.01566198209 |
| 30  | 1.2948751476  | 1.01586952133 |
| 40  | 1.29538922098 | 1.01686148544 |
| 50  | 1.29860651213 | 1.01762532504 |

Figure 6. RMSE Performance of Proposed Method Changing over Different Learning Rate.

Figure 7. RMSE Performance of Proposed Method Changing Over Different Parameter of Regularized Term (left) and MAE Performance of Proposed Method Changing Over Different Latent Factors (right).
Table 4. Results Comparison of Model-Based Algorithms

| Algorithm | MF            | BMF           | NMF            | BCMF          |
|-----------|---------------|---------------|----------------|---------------|
| λ         | 0.001         | 0.001         | -              | 0.1           |
| K         | 10            | 10            | 120            | 10            |
| β         | 0.1           | 0.25          | 0.25           | 0.25          |
| RMSE      | 1.31965175905 | 1.30519923118 | 1.58863090511  | 1.29226571310 |
| MAE       | 1.04635786350 | 1.03238320488 | 1.25636373608  | 1.01463114217 |

After obtained the best value of parameters of each model-based approach, the next step is considering the results of each algorithm with different latent factors. We run the algorithms with latent factor starting from 5 to 120. There are 4 algorithms such as basic matrix factorization, biased matrix factorization, non-negative matrix factorization and proposed method. As seen in Table 4, with different latent factor K, for MF, learning rate is set as 0.001, regularized term is set as 0.1. For BMF, learning rate is set as 0.01, regularize term is set as 0.25. Meanwhile learning rate of proposed method is set as 0.01 and regularized term is set as 0.25.

In Figure 8, for NMF, when K in the range from 5 to 20, the value of RMSE and MAE is rapidly increased, but when K is greater than 20, these two values start to be decreased. For MF and BMF, while K in the range between 5 and 10, the value of RMSE and MAE are decreased, and these two values are increased when k from 10 to 120. For proposed method (OURS), the value of RMSE and MAE is decreased in the range between 5 and 10, but when K is greater than 10, the values is gradually increased, it indicates that performance of proposed method is more efficient than other model-based collaborative filtering approaches. It demonstrates that incorporating the app category information and the weight of app over category is helpful for developing mobile apps recommender system.

4.2.3 Performance Comparison of All algorithms. After completed all experiments, compare proposed approach (OURS) with the state-of-the art approaches such as user-based, item-based, basic matrix
factorization and biased matrix factorization and non-negative matrix factorization under the condition of best parameters of each approach as following figure:

In Figure 9, it is clear to notice that proposed algorithm achieves the best performance in terms of RMSE and MAE, comparing with user-based, item-based, basic matrix factorization, non-negative matrix factorization and biased matrix factorization. For memory-base approaches, OURS approach improved the performance of user-based and item-based by 12.891% and 10.438% respectively in term of RMSE, in term of MAE, it improved the performance of user-based and item-based by 18.009% and 14.724%.

OURS approach improved the performance of MF, BMF, NMF by 2.119%, 1 % and 22.931% respectively in term of RMSE. Meanwhile, OURS approach improved the performance of MF, BMF and NMF by 3.127%, 1.75% and 23.825% respectively in term of MAE. It reflects that by incorporating the app category information and the weight of app over category can significantly increase the performance of prediction.

5. Conclusion and future works

This paper analyses and summarizes the shortcomings of the existing mobile application recommendation system technologies and algorithms and implements a new recommendation algorithm based on collective matrix factorization. In addition, this study used the real-world data from Apple Store and built a dataset. Specifically, the main contents of this research include the following aspects:

Firstly, crawl mobile application data through the crawler technique, the data is from https://itunes.apple.com, the data includes application category information, ratings, and reviews.

Secondly, this research focuses on mobile application recommendation based on User App ratings and Application Categories, which is part of context-aware recommendation system. This research uses collective matrix factorization model to make prediction and then recommendation. This research conducts several experimentations to compare the state-of-art approaches and proposed method by running six collaborative filtering algorithms. To achieve the goal, this research also presents six steps for implementing mobile app recommendation. There are many challenges in performing these key steps. By learning the pattern of dataset and the logic of algorithm could be a solution of these challenges.

At the end, in order to check the quality of prediction, evaluation metrics such as RMSE and MAE is used, it computes the difference between actual value and predicted value. It also reflects the effectiveness and the robustness of recommender system.

The proposed method achieved the best performance in RMSE and MAE when latent factor $K = \text{This}$ research discovers that collective matrix factorization algorithm makes a prediction of mobile application by incorporating category information. In summary, the proposed method demonstrates the
effectiveness of contextual information for recommending mobile apps comparing with traditional recommendation system. The proposed methods that mentioned above improved the performance of MF, BMF, NMF by 2.119%, 1 % and 22.931% respectively in term of RMSE and improved the performance of MF, BMF and NMF by 3.127%, 1.75% and 23.825% respectively in term of MAE. The future work, it could be finding more factors which are able to improve performance of proposed method by analysing the characteristic of dataset and other potential contextual information, process mobile app recommendation using Tensor Factorization, Context-Aware Recommendation System (CARS) algorithm, Machine learning, deep learning, and other algorithms by incorporating other contextual information.

6. References
[1] “Statista - The Statistics Portal,” Statista. https://www.statista.com/ (accessed Apr. 28, 2021).
[2] X. Wu and Y. Zhu, “A Hybrid Approach Based on Collaborative Filtering to Recommending Mobile Apps,” in 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS), Dec. 2016, pp. 8–15, doi: 10.1109/ICPADS.2016.0011.
[3] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, “Convolutional Matrix Factorization for Document-Aware Recommendation,” in Proceedings of the 10th ACM Conference on Recommender Systems, New York, NY, USA, 2016, pp. 233–240, doi: 10.1145/2959100.2959165.
[4] P. Yin, P. Luo, W.-C. Lee, and M. Wang, “App Recommendation: A Contest between Satisfaction and Temptation,” in Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, New York, NY, USA, 2013, pp. 395–404, doi: 10.1145/2433396.2433446.
[5] R. Zhang, C. Li, H. Sun, Y. Wang, and J. Huai, “Quality of Web Service Prediction by Collective Matrix Factorization,” in 2014 IEEE International Conference on Services Computing, Jun. 2014, pp. 432–439, doi: 10.1109/SCC.2014.64.
[6] X. Su and T. M. Khoshgoftaar, “A Survey of Collaborative Filtering Techniques,” Adv. Artif. Intell., vol. 2009, p. 421425, Oct. 2009, doi: 10.1155/2009/421425.
[7] G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions,” IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, pp. 734–749, Jun. 2005, doi: 10.1109/TKDE.2005.99.
[8] H. Wang, N. Wang, and D.-Y. Yeung, “Collaborative Deep Learning for Recommender Systems,” in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 2015, pp. 1235–1244, doi: 10.1145/2783258.2783273.
[9] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating Collaborative Filtering Recommender Systems,” ACM Trans Inf Syst, vol. 22, no. 1, pp. 5–53, Jan. 2004, doi: 10.1145/963770.963772.
[10] Y. Koren, “Collaborative filtering with temporal dynamics,” in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’09, Paris, France, 2009, p. 447, doi: 10.1145/1557019.1557072.
[11] Y. Koren and R. Bell, “Advances in Collaborative Filtering,” 2015, pp. 77–118.
[12] Y. Koren, “Factorization Meets the Neighborhood: A Multifaceted Collaborative Filtering Model,” in Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 2008, pp. 426–434, doi: 10.1145/1401890.1401944.
[13] A. Karatzoglou, L. Baltrunas, K. Church, and M. Böhmer, “Climbing the app wall: Enabling mobile app discovery through context-aware recommendations,” in ACM International Conference Proceeding Series, 2012, pp. 2527–2530, doi: 10.1145/2396761.2398683.
[14] L. Yezheng, D. Fei, J. Yuanchun, L. Xiao, and W. Quidan, “A Novel APPs Recommendation Algorithm Based on APPs Popularity and User Behaviors,” in 2016 IEEE First International Conference on Data Science in Cyberspace (DSC), Jun. 2016, pp. 584–589, doi: 10.1109/DSC.2016.86.
[15] S. Wenxuan and Y. Airu, “Interoperability-Enriched App Recommendation,” in *2014 IEEE International Conference on Data Mining Workshop*, Dec. 2014, pp. 1242–1245, doi: 10.1109/ICDMW.2014.23.

[16] E. Costa-Montenegro, A. B. Barragáns-Martínez, and M. Rey-López, “Which App? A recommender system of applications in markets: Implementation of the service for monitoring users’ interaction,” *Expert Syst. Appl.*, vol. 39, no. 10, pp. 9367–9375, 2012, doi: https://doi.org/10.1016/j.eswa.2012.02.131.

[17] J. S. Breese, D. Heckerman, and C. Kadie, “Empirical Analysis of Predictive Algorithms for Collaborative Filtering,” in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, San Francisco, CA, USA, 1998, pp. 43–52.

[18] T. W. Malone, K. R. Grant, F. A. Turbak, S. A. Brobst, and M. D. Cohen, “Intelligent Information-Sharing Systems,” *Commun ACM*, vol. 30, no. 5, pp. 390–402, May 1987, doi: 10.1145/22899.22903.

[19] G. Adomavicius, B. Mobasher, F. Ricci, and A. Tuzhilin, “Context-Aware Recommender Systems,” *AI Mag.*, vol. 32, pp. 67–80, 2011.

[20] C. Basu, H. Hirsh, and W. Cohen, “Recommendation as Classification: Using Social and Content-Based Information in Recommendation,” p. 5.

[21] R. J. Mooney and L. Roy, “Content-Based Book Recommending Using Learning for Text Categorization,” in *Proceedings of the Fifth ACM Conference on Digital Libraries*, New York, NY, USA, 2000, pp. 195–204, doi: 10.1145/336597.336662.

[22] N. K. Kaoungku, K. Kerdprasop, and N. Kerdprasop, “A Technique to Association Rule Mining on Multiple Datasets,” *J. Adv. Inf. Technol.*, vol. 5, no. 2, pp. 53–57, May 2014, doi: 10.4304/jait.5.2.53-57.

[23] T. Siddiqui, M. A. Aalam, and S. Jain, “Discovery of Scalable Association Rules from Large Set of Multidimensional Quantitative Datasets,” *J. Adv. Inf. Technol.*, vol. 3, no. 1, pp. 69–76, Feb. 2012, doi: 10.4304/jait.3.1.69-76.

[24] S. K. Shinde and U. V. Kulkarni, “Hybrid Personalized Recommender System Using Fast K-medoids Clustering Algorithm,” *J. Adv. Inf. Technol.*, vol. 2, no. 3, pp. 152–158, Aug. 2011, doi: 10.4304/jait.2.3.152-158.

**Acknowledgments**

I would like to express my deepest gratitude to my advisors and colleagues for their technical and/or material support, assistance and inspired me during the completion of this research. Financial support from the School of Computer Science and Engineering, Central South University, Changsha, China is greatly appreciated.