Postprint: Collaborative Filtering Recommendation Quality Evaluation Based on Empirical Distribution and KL Divergence

Authors: Zhang Wen, Jiang Yipan, Zhang Siguang, Cui Yangbo, Du Yuhang

Date: 2018-05-24T00:00:00+00:00

Abstract

How to evaluate the quality of collaborative filtering recommendations—specifically, conducting quality assessment of recommendation results before pushing recommended items to users—is a problem worthy of research. This paper proposes a collaborative filtering recommendation quality evaluation method based on empirical distribution and KL divergence, termed RQE-EDKL (recommendation quality evaluation based on empirical distribution and KL divergence). RQE-EDKL first utilizes historical user-item data to generate historical usage probability distributions of items under varying item quantities; then, it compares this distribution with the user item usage probabilities obtained from various collaborative filtering recommendation methods, calculating their KL divergence; finally, it regards the recommendation result with the minimum KL divergence as the optimal recommendation result and pushes it to users. Experimental results on the TalkingData dataset demonstrate that the RQE-EDKL evaluation method can effectively select recommendation results that better align with users' real needs among different recommendation results, thereby improving the quality of collaborative filtering recommendations.

Full Text

Preamble

Study on Recommendation Quality Evaluation Based on Empirical Distribution and KL Divergence

Zhang Wen[1], Jiang Yipan[1], Zhang Siguang[2], Cui Yangbo[1], Du Yuhang[1]

[1]School of Economics & Management, Beijing University of Chemical Technology, Beijing 100029, China

[2]Institutes of Science & Development, Chinese Academy of Sciences, Beijing 100190, China

Abstract: Evaluating the quality of collaborative filtering recommendations before pushing recommended items to users is a problem worth studying. This paper proposes a method called RQE-EDKL (Recommendation Quality Evaluation based on Empirical Distribution and KL Divergence) to evaluate collaborative filtering recommendation quality. RQE-EDKL first generates historical usage probability distributions of items under different item quantities using historical user-item data. It then calculates the KL divergence between these distributions and the item usage probability distributions obtained from various collaborative filtering recommendation methods. Finally, the recommendation result with the minimum KL divergence is considered the best and pushed to users. Experimental results on the TalkingData dataset demonstrate that the RQE-EDKL evaluation method can effectively select recommendation results that better match users' real needs among different recommendation results, thereby improving the quality of collaborative filtering recommendations.

Keywords: Empirical distribution; Recommendation algorithm; KL divergence; Collaborative filtering

0 Introduction

Recommender systems match users with the most suitable items or services based on sufficient user data. For ordinary users, recommender systems recommend interesting items, enabling "one-to-one" services. By analyzing user behavior data, they can even uncover new interest points that users themselves are unaware of and dynamically adjust as user needs change, reducing information collection costs. For product or service providers, more accurate recommendations lead to higher user engagement and greater profits from loyal users. Consequently, e-commerce websites, social software, and video/audio streaming platforms have all introduced recommender systems to provide personalized choices. According to VentureBeat statistics, Amazon' s recommender system contributes to 35% of its product sales.

Based on this analysis, this paper proposes RQE-EDKL, a collaborative filtering recommendation quality evaluation method based on empirical distribution and KL divergence. The fundamental idea is to introduce empirical distribution into recommendation algorithms and use KL divergence (Kullback-Leibler divergence) to measure the similarity between the recommendation result distribution of traditional recommendation algorithms and the empirical probability distribution, thereby filtering the most credible recommendation results for end users. Unlike general recommendation accuracy evaluation metrics, RQE-EDKL provides a more scientific measurement approach by introducing probability distributions from statistics, increasing the utilization of user information and item information.

Current research on recommendation algorithms is extensive, including rule-based, content-based, collaborative filtering, and matrix factorization methods. Collaborative filtering has gained widespread favor in academia and industry due to its ease of implementation and high recommendation accuracy. Collaborative filtering recommendations fall into two categories: user-based collaborative filtering and item-based collaborative filtering. The former essentially recommends items used by similar users, while the latter recommends items similar to those the user has already used. Many scholars have improved upon collaborative filtering algorithms in two main directions. First, measuring similarity in different ways. In this direction, Nikolaos et al. proposed a multi-level collaborative filtering algorithm that divides similarity rankings obtained from the commonly used Pearson Correlation Coefficient (PCC) into different levels, adding corresponding constraints to each level to improve similarity measurement accuracy and recommendation effectiveness. Wang Fuqiang et al. proposed a location-based asymmetric similarity collaborative filtering algorithm (LBASCF) that fuses cosine similarity with location-based similarity to obtain a new asymmetric user similarity that reflects both location and interest preferences. Choi et al. considered the similarity degree between all items and target items when measuring user similarity—the more similar an item is to the target item, the greater its weight in measuring user similarity. Second, optimizing and improving the recommendation model. Deng Xiaoyi et al. established a collaborative filtering recommendation model based on context clustering and user ratings, clustering users according to context information, introducing social network theory to analyze relationships between users, and building a user rating model to evaluate users' recommendation capabilities for score prediction. George et al. proposed a joint clustering algorithm based on singular value decomposition, designing a parallel version of the co-clustering algorithm to build an efficient real-time collaborative filtering framework. Liu Fuyong et al. proposed a recommendation algorithm based on an improved Bayesian probability model, setting an association vector for each user and item, and designing a decision algorithm to reduce computational complexity and storage overhead by utilizing the sparsity of user vectors.

However, there is no complete quality evaluation system for the final recommendation results of many recommendation algorithms. The question of how to integrate results from different recommendation algorithms and filter the most suitable items for users through certain methods needs attention after multiple recommendation algorithms generate recommendation lists. Additionally, analyzing the historical download records of Apps in the TalkingData dataset reveals two fundamental facts. First, for a single user, the "user popularity" of downloaded Apps follows a typical Low-Rank-Plus-Shift distribution. That is, among the set of Apps downloaded by a single user, a few Apps have very high popularity among all users, while most Apps are downloaded by relatively few users. This confirms the commonality and uniqueness of user preferences for Apps. Second, for a single App, the "App popularity" among its users also follows a typical Low-Rank-Plus-Shift characteristic. Regardless of how many

users have downloaded an App, only a few of these users have historically downloaded a large number of Apps, while most users have not downloaded many Apps. This confirms the commonality and uniqueness of Apps for users. Specific analysis is provided in Section 4.1.

---

1 Related Work

1.1 Recommendation Problem Statement

Assume there are m items to be recommended $\{v_1, v_2, ..., v_i, ..., v_m\}$. For each item v_i, its historical user set is $U(v_i) = \{u_{i1}, u_{i2}, ..., u_{ij}, ..., u_{in}\}$ where $1 \leq i \leq m$ and $1 \leq j \leq n$. For each user u_j, the set of items they have currently used is $V(u_j) = \{v_{j1}, v_{j2}, ..., v_{jk}, ..., v_{js}\}$ where $1 \leq j \leq n$ and $1 \leq s \leq n$. The mainstream approach of current recommendation algorithms is to recommend a set of unused items $\{v_{j1}, v_{j2}, ..., v_{jq}\}$ that user u_j is most likely to be interested in, based on the item' s historical usage record $U(v_i)$ and the user' s already-used item set $V(u_j)$. Without loss of generality, assume the set size is N, i.e., $|V(u_j)| = N$. The purpose of the recommendation algorithm is to recommend N unused items that user u_j is most likely to be interested in.

1.2 User-Based Collaborative Filtering Algorithm (User-CF)

The user-based collaborative filtering recommendation algorithm consists of two steps. First, find the set of users similar to the target user. Second, find items liked by this set of users but not yet selected by the target user. When measuring similarity between users, the cosine similarity method can be used. For users u_1 and u_2, let $V(u_1)$ represent the set of items used by user u_1, and $V(u_2)$ represent the set of items used by user u_2. The similarity between user u_1 and user u_2 is:

$$w_{12} = \frac{|V(u_1) \cap V(u_2)|}{\sqrt{|V(u_1)| \cdot |V(u_2)|}}$$

After obtaining the user similarity matrix, the following formula is used to measure user u' s interest degree $p(u, v)$ in item v:

$$p(u, v) = \sum_{u_k \in S(u,K) \cap U(v)} w_{uu_k}$$

where $S(u, K)$ represents the K users most similar to user u, $U(v)$ is the set of users who have interacted with item v, and w_{uu_k} represents the similarity between user u and user u_k. Since single-behavior implicit feedback data is used, all r_{uv} values are 1. Through the above formula, we can obtain the interest degree $p(u, v)$ values that connect the target user with all items through its K most similar users. By summing these $p(u, v)$ values, we can obtain each user'

s interest degree in each item and rank Apps accordingly to form the final recommendation sequence.

1.3 Item-Based Collaborative Filtering Algorithm (Item-CF)

Item-based collaborative filtering assumes that people will like items similar to those they have previously used. This recommendation algorithm also has two steps. First, calculate similarity between items. Second, recommend potentially interesting items to users based on item similarity and the target user's historical behavior. The similarity w_{ij} between items v_i and v_j can be defined as:

$$w_{ij} = \frac{|U(v_i) \cap U(v_j)|}{\sqrt{|U(v_i)| \cdot |U(v_j)|}}$$

where $|U(v_i)|$ is the number of users who have used item v_i, and $|U(v_j)|$ is the number of users who have used item v_j. The numerator is the set of users who have used both item v_i and item v_j. After obtaining the similarity between all item pairs, analyze each item used by the user, take the top K items most similar to each item that the user has not used, and sum the similarities to obtain the user' s interest degree in each item. Finally, rank items according to the user' s interest degree to form the recommendation sequence.

1.4 Joint User-Item Collaborative Filtering Algorithm (K-UNN)

Verstrepen et al. proposed combining user-based and item-based collaborative filtering algorithms, introducing a fusion algorithm K-UNN based on nearest neighbor theory. This algorithm targets Boolean-type data, a pattern Pan et al. call OCCF (one-class collaborative filtering). The K-UNN algorithm performs weighted summation of user-based and item-based collaborative filtering recommendation algorithms, with the calculation formula:

$$LNGS(u_j, v_i) = L \cdot N \cdot G \cdot S \cdot p(u_j, v_i) \cdot w_{u_j u_k} \cdot w_{v_i v_p}$$

where $LNGS$ represents four different dimensions for measuring interest degree. L represents the user' s direct interest degree in an item, N represents the user' s interest degree in an item through neighbor relationships, G represents the overall interest degree of all users in the item, reflecting the user' s global interest situation, and S represents the choice of scaling function, which can be adjusted according to actual conditions.

2 RQE-EDKL: Collaborative Filtering Recommendation Quality Evaluation Based on Empirical Distribution and KL Divergence

2.1 Empirical Distribution of User-Used Items

Most recommendation algorithms focus on the selection relationship between users and items, with less analysis of the user distribution of the item set itself. As the object of recommendation, items themselves contain much information of important reference value, and how many users have selected each item is one of the important items. When analyzing the distribution of items used by a user, first calculate the usage frequency of each item in the user' s used item set among all users. Second, place the occurrence counts of all items used by the user into different equal intervals. We specify the number of intervals as 10, as experiments show that setting the number of intervals to 10 yields the best results. That is, Apps installed by user u_j are placed into 10 equal intervals in descending order according to their total installation counts. Finally, count the number of items in each interval and calculate the proportion of items in each interval relative to the total number of items used by the user, using this as the probability distribution of items used by the user.

The empirical distribution of items used by a single user refers to the probability distribution of the number of items in different intervals used by that user. As shown in the third column of , this is the empirical probability distribution of Apps used by a user. The standard empirical distribution refers to the probability distribution of the number of items in different intervals for users who have used the same number of items, as shown in the fifth column of , which is the standard empirical distribution for users who have installed 12 Apps. It is worth noting that empirical distributions are discrete probability distributions.

Through the above method, different empirical distributions can be generated for users who have installed different numbers of Apps. For example, shows the installation status of Apps by a user, where the value c represents the number of times the App has been installed among all users. The table shows that the user has installed a total of 12 Apps, with App number 12 having the highest occurrence count of 91, and App number 4 having the lowest occurrence count of 12. The c values are divided into 10 equal intervals: $[12, 20)$, $[20, 28)$, $[28, 36)$, $[36, 44)$, $[44, 52)$, $[52, 60)$, $[60, 68)$, $[68, 76)$, $[76, 84)$, $[84, 91]$. The proportion of Apps in each interval relative to the total number of Apps installed by the user and the standard distribution for users who have installed this number of Apps are shown in . When visualized, this distribution is shown in [Figure 1: see original paper], where the empirical distribution of Apps installed by the user and the standard user distribution for this quantity level are represented by two line graphs.

2.2 Recommendation Quality Evaluation

In studying collaborative filtering recommendation quality evaluation based on empirical distribution and KL divergence, this paper introduces probability distributions from statistics. After obtaining recommendation lists from different recommendation algorithms, it statistics the user distribution of recommended items. The user distribution of recommendation list L_j refers to the user distribution of the j-th recommended item in the recommendation list for user u_j. By comparing this distribution with the predetermined standard distribution $S(u_j)$ of user u_j and calculating their similarity, the recommendation result most similar to the historical user empirical distribution can be selected to improve recommendation accuracy.

The process involves four steps:

a) **Calculate the standard distribution.** When calculating the standard distribution, the method proposed in Section 3.1 for the standard distribution of user-used items is adopted. First, users who have used the same number of items need to be merged to obtain U_N, which refers to the set of users with a total count of n_N who have all used N items. For each user in a single quantity level, extract the items used by the user, merge all items used by all users, and obtain the set of all appeared items V_N, which refers to the standard item set for users who have used N items at this quantity level. Next, the distribution calculation method for user-used items mentioned in Section 3.1 can be used to calculate the standard distribution. Divide 10 equal intervals according to the overall installation count of Apps, and place each App in its corresponding interval. These 10 equal intervals serve as the reference intervals for users who have installed this number of Apps. That is, different standard distributions can be found corresponding to different numbers of Apps installed by users. In [Figure 2: see original paper], $S(u_1)$ is the standard distribution corresponding to the number of items used by user 1.

b) **Use different recommendation algorithms to form different recommendation lists and calculate the empirical distribution of user-used items for the recommendation lists.** This paper adopts User-CF, Item-CF, and K-UNN algorithms from Section 2 as baseline recommendation algorithms. Using these three recommendation algorithms, three recommendation lists are formed. The empirical distribution calculation method for user-used items proposed in Section 3.1 is used to calculate the user distribution of recommendation results formed by different recommendation algorithms for users.

c) **Calculate the KL distance between the user distribution obtained from the three algorithms' recommendation lists and the standard user distribution of items.** KL distance, also called relative entropy, essentially measures the difference between two probability distributions P and S for the same space event. The smaller the KL distance,

the more similar the distributions of P and S.

For discrete distributions, the KL distance from P to S is calculated as follows:

$$D_{KL}(P||S) = \sum_{j=1}^{N} P_j \log\left(\frac{P_j}{S_j}\right)$$

where P refers to the empirical distribution of user-used items, S refers to the standard empirical distribution matched according to the number of items used by user u_j, and N refers to the number of items recommended to the user. Here, formula (5) is used to calculate the KL distance between the standard distribution and the empirical distribution formed by the baseline recommendation algorithm. Using the data in as an example, we can obtain:

$$D_{KL}(P||S) = 0.25\times\log\left(\frac{0.25}{0.18}\right)+0.19\times\log\left(\frac{0.19}{0.17}\right)+0.17\times\log\left(\frac{0.17}{0.08}\right)+...+0.08\times\log\left(\frac{0.08}{0.03}\right) = 0.27$$

That is, the KL divergence between the empirical distribution of the 12 Apps installed by user u_j and the standard distribution for the 12-App quantity level is 0.27.

d) **Select the recommendation list corresponding to the user distribution with the minimum KL distance as the best recommendation algorithm to form the final recommendation for the user.** Therefore, filter the recommendation results corresponding to distributions with larger KL distances for the user, and select the recommendation result corresponding to the distribution with the minimum KL distance as the best recommendation result provided by the collaborative filtering recommendation quality evaluation method based on empirical distribution and KL divergence proposed in this paper. The flowchart of the four steps is shown in [Figure 2: see original paper].

---

3 Experiments

3.1 Experimental Data

Experimental data comes from the Kaggle website (http://www.kaggle.com) provided by TalkingData, a big data company, containing real information about Android phone users' App usage. This includes user profile information such as gender, age group, and mobile phone brand and model for nearly 80,000 Android phone users, as well as dynamic information on user location, mobile App downloads, usage, and categories from May 1, 2016 to May 7, 2016, totaling more than 30 million records. For privacy and security considerations, each user is represented by a unique ID in the data.

When conducting experiments, due to the large data volume and for practical recommendation usability, this paper filtered the above data. In the original data, each user behavior automatically generates an event when using an App, including accessing the Internet via the App, using a new App, deleting an old App, etc. These events contain user behavior time information (specific to the second) and the App currently being used by the user (including background opening behavior). After counting and sorting user behavior frequencies, to minimize the impact of data sparsity, this paper selected 2,020 users with behavior frequencies between 500-1,000 as the experimental dataset. The reason is that these users' behavior frequencies are in the middle range of all users, with relatively regular and stable behavior. They neither stick to already-used Apps nor blindly follow trends, making the data relatively representative. Among these 2,020 users, 250 users used fewer than 10 Apps. For App recommendation, the information from these users is insufficient to generate reasonable recommendations, so these 250 users were removed, leaving 1,770 users as the experimental subjects in this paper. After summarizing the personal information and dynamic behavior of these 1,770 users, they serve as the experimental dataset for this study.

As shown in [Figure 3: see original paper], the set of App IDs downloaded by a user is {1,2,3,4,5,6···}, totaling 63 Apps. The historical download counts of these Apps among all users follow a typical Low-Rank-Plus-Shift distribution. Although the user downloaded 63 Apps, [Figure 3: see original paper] shows that only 10 Apps (16%) have historical download counts greater than or equal to 700, while the remaining 53 Apps (84%) have download counts less than 700.

Second, for a single App, the "App popularity" among its users also follows a typical Low-Rank-Plus-Shift characteristic. That is, regardless of how many users have downloaded an App, only a few of these users have historically downloaded a large number of Apps, while most users have not downloaded many Apps. This confirms the commonality and uniqueness of Apps for users. As shown in [Figure 4: see original paper], the set of users who have historically downloaded a certain App is {1,2,3,4,5,6···}, totaling 114 users. In other words, the App was downloaded by 114 users. The number of Apps downloaded by these users in history follows a typical Low-Rank-Plus-Shift distribution. [Figure 4: see original paper] shows that only 25 users (22%) have downloaded 70 or more Apps in history, while the remaining 89 users (78%) have downloaded no more than 70 Apps.

3.3 Accuracy Metrics

When measuring the recommendation accuracy of each recommendation algorithm, this paper adopts two widely used evaluation metrics from the information retrieval field: MAP (Mean Average Precision) and MRR (Mean Reciprocal Rank). Specifically:

$$MAP = \frac{1}{Q}\sum_{q=1}^{Q} AveP(q)$$

where Q is the number of users in the test set, and

$$AveP(q) = \frac{1}{N}\sum_{k=1}^{N} \frac{k}{rank_{ik}}$$

where N is the number of Apps installed by the user, $rank_{ik}$ is the ranking position of App v_i expected to be recommended for user u_k, and $rank_{iv}$ is the actual recommendation ranking position of App v_i.

$$MRR = \frac{1}{Q}\sum_{i=1}^{Q} \frac{1}{rank_i}$$

These two metrics are relatively simple to calculate and provide good measurement effects. MAP is a single-value metric reflecting system performance across all relevant documents. The more relevant results the system recommends at the front, the higher the MAP. If the system returns no relevant results, the MAP value defaults to 0. MRR takes the reciprocal of the ranking of the standard result in the evaluated system' s output as its accuracy, then averages all results, serving as an important metric for measuring recommendation results. When different algorithms recommend Apps to users, each algorithm generates a ranked list of recommended Apps. MAP and MRR values are then calculated. When Apps that users are truly interested in are ranked higher, MAP and MRR values are higher, and the recommendation effect is better.

3.4 Novelty and Diversity Metrics

Recommendation novelty refers to the ability to recommend items or services that users have never heard of. Recommendation diversity includes individual diversity and overall diversity. Individual diversity refers to the diversity of items in the recommendation list generated by the recommender system for a single user. Improving individual diversity can solve the problem of high similarity among items within recommendation lists. Overall diversity means that recommendations for different users should be as different as possible.

This paper adopts the novelty measurement method proposed by Hurley et al.:

$$NOV = \frac{1}{m}\sum_{v_i \in L}\left(1 - \frac{d(v_i, v_c)}{m}\right)$$

where L is the recommended App list, m is the number of recommended Apps, $d(v_i, v_c)$ is the distance measurement function used to measure the similarity between App v_i and App v_c, and $c(v_i)$ is the number of times App v_i was downloaded by users.

For diversity measurement, this paper only considers individual diversity, using the Internal List Distance (ILD) metric:

$$ILD = \frac{1}{|V(u_i)|(|V(u_i)|-1)} \sum_{v_i \in V(u_i)} \sum_{v_j \in V(u_i)} (1 - sim(v_i, v_j))$$

where $|V(u_i)|$ is the number of Apps installed by user u_i, and $sim(v_i, v_j)$ is the similarity between App v_i and App v_j.

3.5 Experimental Setup and Results

This paper uses User-CF, Item-CF, and K-UNN as baseline algorithms. The proposed RQE-EDKL method calculates KL distances on top of the recommendation lists from these three algorithms to obtain the best recommendation list. Specifically, the experiments consist of two parts. The first part fixes the test set proportion R and continuously adjusts the number of Apps masked for each user K, starting from 2 and increasing by intervals of 2 up to 10. The second part fixes the number of masked Apps K at 8 and varies the test set proportion R from 5% to 25% in increments of 5%.

In the first part of the experiment, for simplicity, the test set proportion is set to 5%, meaning 90 users are randomly selected from 1,770 users to form the test set, with the remaining 1,680 users forming the training set. During the experiment, the number of masked Apps K for each user in the test set is continuously changed from 2 to 10 in increments of 2, with the selection of masked Apps being random. According to the proposed RQE-EDKL method, the standard distribution is first calculated using the App usage information of users in the training set. Then, for the 90 users in the test set, three different recommendation lists are obtained using User-CF, Item-CF, and K-UNN algorithms. For each test user, these three recommendation lists essentially rank all Apps not used by the user according to different criteria, with the ranking standard being the user' s interest degree.

To match real-world situations, this paper does not select all recommended Apps for distribution calculation but instead takes the top 20 Apps from each recommendation list as the new final recommendation list. For the randomly selected 90 test users, each user first receives three different recommendation lists from the three algorithms. Then, the KL distance between each algorithm' s App distribution and the standard distribution is calculated, and the recommendation algorithm result with the minimum KL distance is selected as the recommendation result of the proposed RQE-EDKL method, ultimately forming a new recommendation list. This list is the final result of the method proposed in this

paper. Since some test users' numbers of used Apps cannot find corresponding standard distributions in the training set, this paper selects the standard distributions for the user' s actual number of used Apps and the two adjacent quantities (i.e., 5 standard distributions) and fuses them to form a new standard user distribution. Finally, this list is compared with the baseline methods —User-CF, Item-CF, and K-UNN algorithms—and MAP and MRR values are calculated.

[Figure 5: see original paper] shows the performance of the proposed RQE-EDKL method and baseline methods on MAP and MRR metrics when using 5% of experimental data as the test set and recommending different numbers of Apps. From [Figure 5: see original paper], we can see that first, when the test set proportion is fixed at 5%, whether for the proposed RQE-EDKL method or User-CF, Item-CF, and K-UNN algorithms, their MAP values are above 0.5, indicating good recommendation performance. At the same time, as the number of masked Apps increases from 2 to 10, their MAP and MRR values show a clear downward trend. That is, the more historical App usage information is hidden from users, the worse the recommendation effect; conversely, the more complete the user' s historical App usage information, the better the recommendation effect.

Second, combining MAP and MRR values, the figure clearly shows that the proposed RQE-EDKL method can significantly improve recommendation accuracy. Essentially, the RQE-EDKL method utilizes the Low-Rank-Plus-Shift characteristic of historical usage frequency of Apps used by users, combined with KL divergence to measure recommendation result quality. As shown in [Figure 1: see original paper] and [Figure 3: see original paper], Apps used by users can generally be divided into two types: popular Apps reflecting users' common preferences, and unpopular Apps reflecting users' personalized preferences. The RQE-EDKL method essentially considers the rationality of users' usage of these two types of Apps, i.e., whether the recommendation result contains both popular and unpopular Apps, and whether the distribution of popular and unpopular Apps is consistent with the empirical distribution. Therefore, RQE-EDKL can extract useful information to the greatest extent possible to improve recommendation result quality and achieve better recommendation performance.

In terms of novelty and diversity, as shown in [Figure 6: see original paper], the proposed RQE-EDKL method performs significantly better than other recommendation algorithms. Because this algorithm considers the distribution of all Apps used by users, it can maximize the possibility of recommending unpopular Apps to users, thereby increasing novel Apps in recommendations. At the same time, through the above experiments, it can be found that when the number of masked Apps K is 8, the MAP values of the four recommendation algorithms considered in this paper gradually show a relatively stable trend.

[Figure 7: see original paper] shows the performance of the proposed RQE-EDKL method and baseline methods on MAP and MRR metrics when fixing K at 8 and continuously changing the proportion of the test set. From [Fig-

ure 7: see original paper], we can see that first, with K fixed and the test set proportion R changing, as the test set proportion continues to increase and the training data continuously decreases, the MAP values of User-CF, Item-CF, and K-UNN algorithms generally show a decreasing trend. The proposed RQE-EDKL method remains relatively stable when test set proportions are 5%, 10%, and 15%, at approximately 0.65. When the test set proportion is 20%, the MAP value shows a significant increase, reaching about 0.75, but when the test set proportion increases to 25%, the MAP returns to the previous level of 0.65. This paper considers that a test set proportion of 20% can be treated as an outlier, where some accidental factors caused fluctuations in the MAP value. For MRR values, there is no obvious pattern in the change trend. When test set proportions are 10% and 20%, the MRR values of the proposed RQE-EDKL method show two small peaks. User-CF only shows a significant increase when R is 10%, then shows a downward trend. When R takes other values, except for a very obvious decrease in Item-CF' s MRR at a test set proportion of 25%, other recommendation algorithms maintain a relatively stable state. Second, regardless of the amount of training data, the recommendation performance of the RQE-EDKL method is basically higher than the other three baseline methods, demonstrating the effectiveness of this recommendation algorithm. No matter what the training set proportion R is or what the value of K is, better recommendation results can be obtained. When using MAP to measure recommendation effectiveness, a test set proportion of 20% yields the best results. When using MRR to measure recommendation effectiveness, test set proportions of 10% or 20% yield the best results.

[Figure 8: see original paper] shows the performance of the proposed RQE-EDKL method and baseline methods on NOV and ILD metrics when fixing K at 8 and continuously changing the test set proportion. From the figure, we can see that the proposed recommendation algorithm is very stable in novelty performance, stable at about 0.85 with small fluctuations and higher than the other three baseline recommendation algorithms. In diversity performance, the gap with the other three baseline algorithms is relatively small, but overall it is still higher than the other three baseline recommendation algorithms.

---

4 Conclusion

With the exponential explosion of online information, the cost of obtaining information that matches user needs continues to increase. Improvements to recommendation algorithms can save time costs for both product/service providers and users. This paper proposes a collaborative filtering recommendation quality evaluation method based on empirical distribution and KL divergence. On real Android market data, it compares the method with user-based collaborative filtering, item-based collaborative filtering, and joint user-item collaborative filtering algorithms. Experimental results show that the RQE-EDKL method performs better in both recommendation accuracy and recommendation result

diversity. It can maintain stable recommendation results when the size of the test set or the number of masked Apps changes. The reason is that the RQE-EDKL method integrates the concept of distribution from statistics into machine learning, integrates currently popular recommendation algorithms with relatively good recommendation effects, and improves upon them by filtering their recommendation results. The method proposed in this paper narrows the recommendation scope, making it more focused, uses items with high user interest probability as candidate recommendation targets, and further screens them using a user distribution method, improving recommendation result quality. It is worth mentioning that this paper not only focuses on recommendation accuracy but also innovates in how to improve recommendation novelty, providing users with unique items that match their personal interests.

Future research will incorporate user profile information into the recommendation process, including user gender, age group, and other information, to find more effective methods for improving App recommendation algorithm performance.

References

[1] Schafer J B, Konstan J A, Riedl J. E-commerce Recommendation Applications [C]// Proc of Applications of Data Mining to Electronic Commerce. Boston: Springer, 2001: 115-153.

[2] 刘建国, 周涛, 汪秉宏. 个性化推荐系统的研究进展 [J]. 自然科学进展, 2009, 19 (1): 1-15. (Liu Jianguo, Zhou Tao, Wang Binghong. Personalized recommender systems: a survey of the state-of-the-art. Chinese Journal of Progress in Natural Science, 2009, 19 (1): 1-15.)

[3] Resnick P, Iacovou N, Suchak M, et al. GroupLens: an open architecture for collaborative filtering of netnews [C]// Proc of ACM Conference on Computer Supported Cooperative Work. New York: ACM Press, 1994: 175-186.

[4] Sarwar B, Karypis G, Konstan J, et al. Item-based collaborative filtering recommendation algorithms [C]// Proc of International Conference on World Wide Web. New York: ACM Press, 2001: 285-295.

[5] Polatidis N, Georgiadis C K. A multi-level collaborative filtering method that improves recommendations [J]. Expert Systems with Applications, 2016, 48: 100-110.

[6] 王付强, 彭甫镕, 丁小焕. 基于位置的非对称相似性度量的协同过滤推荐算法 [J]. 计算机应用, 2016, 36 (1): 171-174. (Wang Fuqiang, Peng Furong, Ding Xiaohuan. Location-based asymmetric similarity for collaborative filtering recommendation algorithm [J]. Journal of Computer Application, 2016, 36 (1): 171-174.)

[7] Choi K, Suh Y. A new similarity function for selecting neighbors for each target item in collaborative filtering [J]. Knowledge-Based Systems, 2013, 37

(1): 146-153.

[8] 邓晓懿, 金淳, 韩庆平. 基于情境聚类和用户评级的协同过滤推荐模型 [J]. 系统工程理论与实践, 2013, 33 (11): 2945-2953. (Deng Xiaoyi, Jin Chun, Han Qingping. Improved collaborative filtering model based on context clustering and user ranking [J]. Systems Engineering-Theory & Practice, 2013, 33 (1): 2945-2953.)

[9] George T, Merugu S. A scalable collaborative filtering framework based on co-clustering [C]// Proc of IEEE International Conference on Data Mining. 2005: 4.

[10] 刘付勇, 高贤强, 张著. 基于改进贝叶斯概率模型的推荐算法 [J]. 计算机科学, 2017, 44 (05): 285-289. (Liu Fuyong, Gao Xianqiang, Zhang zhu. Improved Bayesian probabilistic model based recommender system [J]. Computer Science, 2017, 44 (05): 285-289.)

[11] Zha Hongyuan, Zhang Zhenyuan. On matrices with low-rank-plus-shift structure: Partial SVD and latent semantic indexing [J]. SIAM Journal on Matrix Analysis & Applications, 1998, 21 (2): 522-536.

[12] Zeng Yifei, Doshi P, Pan Yinghui, et al. Utilizing partial policies for identifying equivalence of behavioral models [C]// Proc of AAAI Conference on Artificial Intelligence. Palo Alto: AAAI Press, 2011.

[13] Verstrepen K, Goethals B. Unifying nearest neighbors collaborative filtering [C]// Proc of the 8th ACM Conference on Recommender Systems. 2014: 177-184.

[14] Pan Rong, Zhou Yunhong, Cao Bin, et al. One-Class Collaborative Filtering [C]// Proc of the 8th IEEE International Conference on Data Mining. Washington DC: IEEE Computer Society, 2008: 502-511.

[15] Zheng Zibin, Ma Hao, Lyu Michael R, et al. QoS-aware Web service recommendation by collaborative filtering [J]. IEEE Trans on Services Computing, 2011, 4 (2): 140-152.

[16] 王斌, 曹菡. 基于新颖性和多样性的旅游推荐模型研究 [J]. 计算机工程与应用, 2016, 52 (6): 219-222. (Wang Bin, Cao Han. Research on tourism recommendation model based on novelty and diversity. Computer Engineering and Applications, 2016, 52 (6): 219-222.)

[17] Ziegler C N, Mcnee S M, Konstan J A, et al. Improving recommendation lists through topic diversification [C]// Proc of International Conference on World Wide Web. New York: ACM Press, 2005: 22-32.

[18] Hurley N, Zhang Mi. Novelty and diversity in top-N recommendation: analysis and evaluation [J]. ACM Trans on Internet Technology, 2011, 10 (4): 1-30.

[19] Ziegler C N, Lausen G. Making product recommendations more diverse [J]. Bulletin of the Technical Committee on Data Engineering, 2010, 32 (4): 23-30.

[20] Godoy D, Amandi A. Modeling interests of web users for recommendation: a user profiling approach and trends [M]// Evolution of the Web in Artificial Intelligence Environments. Berlin: Springer, 2008: 41-68.

Note: Figure translations are in progress. See original paper for figures.