Advances in Economics, Business and Management Research, volume 50
3rd International Conference on Management Science and Engineering (MSE 2017)

Research on Recommendation Algorithm for Mobile Application Crowdsourcing Testers

Liu YING
College of Software and Microelectronics
Northwestern Polytechnical University
Xi'an, China
894749065@qq.com

Zhang TAO
College of Software and Microelectronics
Northwestern Polytechnical University
Xi'an, China

Li KUN
College of Software and Microelectronics
Northwestern Polytechnical University
Xi'an, China

***Abstract*—The anonymous crowdsourcing testers determine the quality of tests, and the low matching degree between testers and tasks reduce testers' enthusiasm. To match the recommended testers with tasks, a two phase recommendation method based on Top-K algorithm was proposed. Category was introduced to reduce time complexity of Top-K algorithm. By classifying the tasks and calculating the category matching scores, the testers were most suitable for the category of the task were obtained. After calculating the similarity between tester portrayal and tasks, the top K testers were recommended from selected categories. Experiment shows that the proposed Top-K-Worker algorithm can greatly improve the matching degree between testers and the recommended task.**

Keywords-Mobile application crowdsourcing testing; Top-K algorithm; tester recommendation; matching degree

I. INTRODUCTION

Mobile Application crowdsourcing testing is a distributed problem solution that outsources mobile application test tasks to anonymous network users in a voluntary way[1]. Compared with the traditional test methods, crowdsourcing test has freedom, high innovation, low cost and other advantages.[2] However, due to the lack of effective crowdsourcing testers recommendation method, the task publishers often select testers directly from the massive applicants. The matching degree between tasks and testers is low. Not only does the problem affects the test efficiency and quality, but also reduce testers' enthusiasm.

The research on the recommendation of mobile application crowdsourcing testers is also scarce, and the research related to personalized recommendation is mainly about collaborative filtering and content filtering.

Collaborative filtering is the most successful and popular recommendation method currently. [3]By analyzing the correlation between the users and the tasks, the method can determine the hidden relation between the users and the tasks. The significant advantage of collaborative filtering is that it is of high accuracy. However, the biggest problem with this approach is the cold start problem because it relies on the historical information, which is not available to new users or tasks.[4] Another significant problem with this approach is the high time complexity generated by the combination of a large number of users and tasks. Therefore, this method can not be implemented in principle.

Content filtering is one of the most basic methods in information filtering, which is usually implemented by means of probability statistics and machine learning.[5]But the method is mainly applicable to the recommendation of learning resources, and content filtering can't select high-quality resources.

To solve these problems, the paper studies personalized recommendation algorithm of crowdsourcing testers. First, an improvement on Top-K algorithm was proposed by introducing category. By calculating the category matching score, we can find the testers who are most suitable for the category. By computing the similarity between tester portrayal and tasks, top K testers who is most suitable for the task is recommended to the crowdsourcing task publisher.

II. TESTER RECOMMENDATION ALGORITHM FOR MOBILE APPLICATION CROWDSOURCING TESTING

Top-K algorithm is a real-time recommendation algorithm proposed by Mejdl Safran[6]. The algorithm predicts the user's preferences by setting the weights of the user's different attributes[7], returning the previous K results based on the user's preferences[8]. Because the algorithm can extract useful information from a large amount of data, it has been widely used in many fields such as e-commerce, search engines and so on. The paper will study and improve Top-K algorithm, and propose a method for task recommendation.

A. *Top-K Recommendation Algorithm Based on Category*

In the application scenario of Top-K algorithm, there are a lot of Crowdsourcing tasks and testers data. An effective intermediate mechanism is introducing category to avoid a large-scale operation. Category acts as an intermediary mechanism between crowdsourcing and task. Depending on the skills required, tasks can be put into different categories, and testers can be associated with a variety of categories based on their own features, historical task information. The crowdsourcing task can be recommended as a limited 1-K match between testers and tasks. The data structure is shown in figure 1.

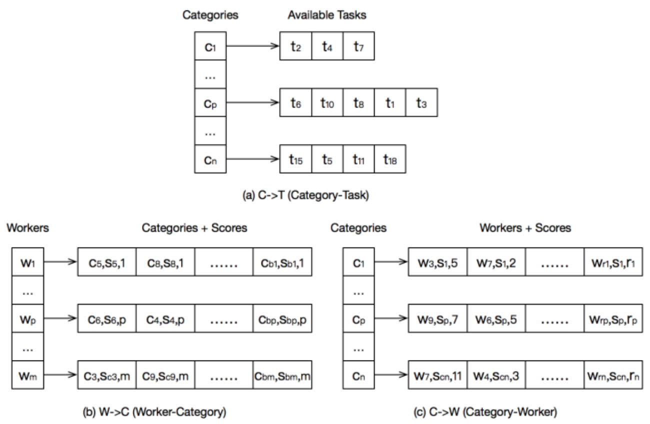
Categories
Available Tasks
c_1
c_p
c_n
t_2 t_4 t_7
t_6 t_{10} t_8 t_1 t_3
t_{15} t_5 t_{11} t_{18}
(a) C->T (Category-Task)
Workers
Categories + Scores
W_1 → $C_5,S_5,1$ | $C_8,S_8,1$ | …… | $C_{b1},S_{b1},1$
W_p → C_6,S_6,p | C_4,S_4,p | …… | C_{bp},S_{bp},p
W_m → C_3,S_{c3},m | C_9,S_{c9},m | …… | C_{bm},S_{bm},m
(b) W->C (Worker-Category)
Categories
Workers + Scores
C_1 → $W_3,S_1,5$ | $W_7,S_1,2$ | …… | W_{r1},S_1,r_1
C_p → $W_9,S_p,7$ | $W_6,S_p,5$ | …… | W_{rp},S_p,r_p
C_n → $W_7,S_{cn},11$ | $W_4,S_{cn},3$ | …… | W_{rn},S_{cn},r_n
(c) C->W (Category-Worker)

Figure1. Data structure of Top-K algorithm based on category

C→T(category-task) is introduced as an array of many categories, each category c_i contains a large number of tasks, the data structure is shown in Figure 1 (a). Figure 1 (B) shows a worker Wj in W→C (worker-category) points to many categories.However, it is not enough to get the W→C data structure because W→C only represents the relationship from the testers to categories. Additional data structures are needed to match the appropriate categories with the appointed testers, as is shown in Figure 1 (C).

For the given tester Wj and categories c_i, the matching score S_{ij} between them is defined as the product of three factors: the acceptance rate, the proficiency level, and the user satisfaction:

$$S_{ij} = AR_{ij} \times TSP_{ij} \times ATRS_{ij} \times AWBN_{ij} \tag{1}$$

Acceptance rate AR_{ij} is defined as the ratio of tasks number accepted by tester Wj in the category c_i to the total number of accepted tasks of Wj:

$$AR_{ij} = \frac{AT_{ij}}{ACCT_{ij}} \tag{2}$$

Task completion rate SP_{ij} is defined as the ratio of completed tasks by tester Wj in the category c_i to the total number of completed tasks of Wj:

$$TSP_{ij} = \frac{CT_{ij}}{FT_{ij}} \tag{3}$$

The satisfaction degree is divided into four grades: very satisfied, satisfied, not very satisfied, not satisfied .The definition of customer satisfaction (Customer Satisfaction, Rating, CSR) is $CSR = csr_m$,where $csr_1 = 5$, $csr_2 = 4$, $csr_3 = 3$, $csr_4 = 0$. The test average satisfaction of tester Wj in category c_i is defined as:

$$ATRS_{ij} = \frac{\sum_{n=1,m=1}^{n,3} result_n \times csr_m}{\sum_{n=1}^{n} result_n} \tag{4}$$

The average weighted Bug number is used to describe the severity of the Bug found by tester Wj in category c_i. The influence of Bug can be classified into different levels: especially important, important, general important, unimportant, not Bug. The Bug severity class (Bug, Severity, Level, BSL) is defined as $BSL = bsl_m, j_m = 1,\dots,5$, where $bsl_1 = 5$, $bsl_2 = 2$, $bsl_3 = 1$, $bsl_4 = 0.5$, $bsl_5 = 0$. Then the average weighted Bug numbers can be defined as:

$$AWBN_{ij} = \frac{\sum_{n=1,m=1}^{n,4} bug_n \times bsl_m}{\sum_{n=1}^{n} bug_n} \tag{5}$$

B. *Similarity between Tester Portrayal and Tasks*

The above can only screen out the testers which is most suitable for the specified task type. So it is important to consider how to recommend the most suitable testers, and the introduction of Cosine similarity can solve the problem.

Cosine similarity is a commonly used similarity measurement, especially for high-dimensional spaces[9]. The Cosine similarity is more applicable than Euclidean distance when the dimensions of different attribute values are inconsistent[10]. The range of Cosine similarity is [-1,1], and the bigger the result is, the higher the similarity is[11].The Cosine similarity between $a = [a_{1,\dots,} a_n]^T$ and $b = [b_{1,\dots,} b_n]^T$ are:

$$similarity(a, b) = \frac{\sum_{k=1}^{h} a \times b}{\sqrt{\sum_{k=1}^{h} a^2} \times \sqrt{\sum_{k=1}^{h} b^2}} \tag{6}$$

Similarity between tester portrayal and tasks can be defined as:

$$similarity(p_j, r_i) = \frac{\sum_{k=1}^{h} p_{jk} \times r_{ik}}{\sqrt{\sum_{k=1}^{h} p_{jk}^2} \times \sqrt{\sum_{k=1}^{h} r_{ik}^2}} \quad (7)$$

Where tester portrayal p_j is the collection of test experience p_{j1}, number of certificates p_{j2}, response speed p_{j3}; r_i is the collection of requirements of tasks.

C. Top-K-Worker Recommendation Algorithm

For published tasks, the Top-K-Worker algorithm recommend the most appropriate and testers to the customer. The Top-K-Worker algorithm is shown in figure 2. The algorithm first obtains the categories to which the publishing task belongs. Then the matching score was calculated and get a list of the 2K testers that match the task's type. Then, by calculating similarity between tester portrayal and tasks , the 2K name is further sorted by calculating Similarity between tester portrayal and tasks. At last, we can get a list of top K testers.

```
input:
    C→Task: category-task data structure
    Worker→C: worker-category data structure
    index: the serial number of testers searching for the task
    k: the number of recommended tasks
output:
    L: the first K testers recommended for the task

1   Initializing output list L
2   Get the category which the publication task belongs
3       for i=1 to AllTestersnumber(n)
4       calculate matching score $S_{ij}$
5       All testers are arranged in descending order according to the matching score $S_{ij}$
6           List←Put the top 2K testers in list L
7           for  j=1 to testersnumber(List)
8           Arrange all tasks in descending order according to portrait task similarity
9               L ← similarity between tester portrayal and tasks
10      end
11      end
```

Figure 2. the Top-K-Worker algorithm

It is possible that the recommended K testers are unable to accept tasks for any special reasons. The Top-K-Worker algorithm solves this problem by using a list of testers stored in the Cindex→Tester. The algorithm periodically adds K values until one of the recommended testers receives the task. At the same time, the list of recommendation retains the previously recommended testers, which provides opportunities to score higher score testers, because their status may change at any time, rather than simply recommend alternative testers.

The Top-K-Worker algorithm requires O(α+k) to generate recommendations for appointed tester, where α is the number of task categories, and K is the number of testers to be recommend. Obviously, the time complexity of the Top-K-Worker algorithm is very low. No matter how big the data size is, the algorithm still works.

D. Data Update and Cold Start

When testers complete the task, match score S_{ij} defined by formula (1) and similarity between tester portrayal and tasks defined by the formula (7) should be updated, which will affect the sort of testers. However, it would be time-consuming if you update these scores and arrays every time the testers complete the tasks. Therefore, we assume that testers complete a task, the impact on the testers' matching score is negligible, and these scores and arrays can be updated periodically.

Cold start is a difficult for recommendation system. New testers face cold start problems, which means that these testers don't have matching scores or similarity between tester portrayal and tasks, and these new testers will never be recommended. To solve the problem, we can assume that the mobile application Crowdsourcing test system requires the new testers to select some preference at the time of registration. Once the new testers have completed a certain number of tasks, the formula(1) and (7) is used to calculate the matching degree between the tester and the tasks.

III. EXPERIMENTAL VERIFICATION

A. Experimental Data

In this paper, we access experimental data from the mobile application crowdsourcing test website Testin by crawling method .The paper selects data from March to April, the final experimental data is shown in table 1.

TABLE I. EXPERIMENTAL DATA

Data set	Testers number	Category	Tasks available	Tasks finished
Testin	1798	20	3612	9865

B. Evaluation Index

The recommendation system commonly used evaluation criteria NDCG to divide correlation into several levels. CG (cumulative gain) is calculated as follows:

$$CG_p = \sum_{i=1}^{p} rel_i \quad (8)$$

Where i presents position, p presents the position of the task in the recommended result list, rel_i indicates the graded relevance of the task .

It would be more reasonable to rank more relevant results in the front, whereas CG would not consider their relevance. Therefore, DCG (Discounted Cumulative Gain) is introduced.

$$DCG_p = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)} \quad (9)$$

Although DCG considers the location factor, it does not consider that the sorting position is more advanced, the correlation degree has more influence on the quality of sorting. In addition, another shortcoming of DCG is that the number of recommended entries returned by different recommendations is different. Therefore, NDCG (Normalized DCG) is introduced to solve these problems.

$$NDCG_p = \frac{DCG_p}{IDCG_p} \quad (10)$$

$$IDCG_p = \sum_{i=1}^{|REL|} \frac{2^{rel_i} - 1}{\log_2(i+1)} \quad (11)$$

Where IDCGP is maximum value in the DCG value after sorting the recommended tasks.

C. *Experimental Results and Analysis*

In order to accurately and objectively evaluate Crowdsourcing task recommendation algorithm, the experiment compares the original data from existing crowdsourcing task distribution and the proposed algorithm matching degree were compared. They use the same data set and parameter settings in the experiments,

Through the statistics of each data set of tasks, applications and the types of crowdsourcing testers, we select 15 categories as experimental parameters, as shown in table 2.

TABLE II. EXPERIMENTAL PARAMETER

	Class parameter
Task type	Bug exploration, use case execution, and use case design
Application type	Social communication, health care, shopping preferences, financial management, office, business, travel, audio-visual, news, education,
Crowdsourcing testers' attributes	Occupation, geography, and testing experience

In addition, another important parameter testers' number n is obtained from the task information. The range of n is from 1 to 60, and can be divided into 1-10,10-30,30-60 three intervals, which are represented by @1, @10 and @30 respectively.

The data acquisition and preprocessing are implemented in Python language. nDCG@1, nDCG@10 and nDCG@30 were used to evaluate the experimental results. The higher the score is, the better the accuracy is. The experiment results is shown in the table 3.

TABLE III. EXPERIMENT RESULTS

	nDCG@1	nDCG@10	nDCG@30
Original data	0.0905	0.1356	0.1570
Top-K-Worker	0.0938	0.1386	0.1593
Growth proportion	3.65%	2.21%	1.46%

From the above experiment results, the matching degree between tasks and testers in original data is low. The matching degree of Top-K-Worker algorithm is higher than the original data, which proves the accuracy of the Top-K-Worker algorithm.

IV. CONCLUSION

The paper explores and improves the Top-K algorithm. The proposed Top-K-Worker algorithm has the following characteristics:(1)Introducing task categories as intermediate mechanism, which can effectively avoid matching sparse matrix and large scale calculating. (2) Cosine similarity is used to calculate task similarity, which is more applicable than Euclidean distance when the dimensions of different attribute values are inconsistent. (3)Considering the problem of data updating and cold start, the algorithm is practical.

We will further improve the Top-K-Worker algorithm by considering the solution that occurs in parallel when sorting testers.So that the match between the testers and recommended tasks is closer, and the algorithm is more practical.

REFERENCES

[1] HOWE J.Crowdsourcing: why the power of the crowd is driving the future of busines[J]. Journal of Consumer Marketing, 2009, 26(4) : 305 - 306

[2] Zhiqiang Zhang,Jusheng Pang,Xiaoqin Xie,.Kang Zhou.Research on Crowdsourcing Control Strategies and evaluation Algorithm[J].Chinese Journal of Computers,2013,36(08):1636-1649.

[3] Wang LC, Meng XW, Zhang YJ. Context-Aware recommender systems: A survey of the state-of-the-art and possible extensions.Ruan Jian Xue Bao/Journal of Software, 2012,23(1):1-20 (in Chinese with English abstract).http://www.jos.org.cn/1000-9825/4100. htm [doi: 10.3724/SP.J.1001.2012.04100]

[4] Yu Hong, Li JunHua. Algorithm to Solve the Cold-Start Problem in New Item Recommendations[J]. Journal of software,2015,26(06):1395-1408.

[5] Wang Yanran, Chen Mei, Wang Hanhu, Zhang Xin. A content Based Filtering Algorithm for scientific Literature Recommendation[J].Computer Technology And Development.21(02):66-69.

[6] Safran M, Che D. Real-time recommendation algorithms for crowdsourcing systems[J]. Applied Computing & Informatics, 2016.

[7] Ci Xiang,Ma Youzhong,Meng Xiaofeng.Method for Top-K Query on Big Data in Cloud[J].Journal of Software,2014,25(04):813-825.

[8] Zhou Li-Yong. Algorithm on Top-k Keyword Search of Uncertain XML[A]. International Informatization and Engineering Associations, Atlantis Press.Proceedings of 2015 4th International Conference on Mechatronics,Materials,Chemistry and Computer Engineering (ICMMCCE 2015)[C].International Informatization and Engineering Associations, Atlantis Press:,2015:6.

[9] Qiu Wei. Cosine similarity measures for dual hesitant fuzzy sets[A]. Proceedings of 2016 4th International Conference on Machinery,Materials and Computing Technology(ICMMCT 2016)[C],2016:4.

[10] Li ZHANG,Peipei XIA,Fanzhang LI. A Method Based on Cosine Similarity in Supplier Selection.

[11] Tingting LIANG,Chunqing LI,Haisheng LI .Top-K Learning Resource Matching Recommendation Based on Content Filtering Pagerank[J].Computer Engineering,2017,43(02):220-226.