Personalized Recommendation Algorithm Considering Time Sensitivity

Fuzhen Sun[1], Haiyan Zhuang[2], Jin Zhang[1], Zhen Wang[1], and Kai Zheng[1(✉)]

[1] School of Computer Science and Technology, Shandong University of Technology, Zibo 255049, China
zhengkai@uestc.edu.cn

[2] Image and Network Investigation Department, Railway Police College, Zhengzhou 450053, China

Abstract. Aiming to solve the problem of goods popularity bias, this paper introduces the prevalence of items into user interest modeling, and proposes an item popularity model based on user interest feature. Usually, traditional model that does not take into account the stability of user's interests, which leads to the difficulty in capturing their interest. To cope with this limitation, we propose a time-sensitive and stabilized interest similarity model that involves a process of calculating the similarity of user interest. Moreover, by combining those two kinds of similarity model based on weight factors, we develop a novel algorithm for calculation, which is named as IPSTS (IPSTS). To evaluate the proposed approach, experiments are performed and results indicate that Mean Absolute Difference (MAE) and root mean square error (RMSE) could be significantly reduced, when compared with those of traditional collaborative filtering Algorithms.

Keywords: Time sensitivity · Stability of interest · Prevalence of item · Personalized recommendation · Popularity bias

1 Introduction

Collaborative Filtering Recommendation Algorithms are mainly divided into two categories: user-based and item-based. The idea of Item Based Collaborative Filtering Algorithm is that, basing on the user historical behavior, making data analysis and calculation, to get a user's behavior preferences and make recommendations for users basing on their preferences. However, the prerequisite of this algorithm should be that, users can not change the interest within a period of time. The idea of Item-Based Collaborative Filtering Recommendation Algorithms is that, basing on the behavior data of the user A, calculating the neighbor users who have similar preferences with user A, and then recommend the items - which are of interest to the neighboring users but user A has not yet found. Therefore, no matter which collaborative filtering recommendation algorithm, the core is the calculation of similarity.

X. Zhang et al. (Eds.): CloudComp 2019/SmartGift 2019, LNICST 322, pp. 154–162, 2020.
https://doi.org/10.1007/978-3-030-48513-9_12

As for the dynamics problem in collaborative filtering technology, a recommendation algorithm is proposed in literature [3], which adapt to user interest to evaluate the time weight and Data weight of resource similarity. Literature [4] proposed a dynamic recommendation technique. As the mobile device flourishesa SUCM model is proposed in [5], which learn fine-grained user preferences to make recommendations to users.

As for the dynamics problem in collaborative filtering technology, a recommendation algorithm, which adapt to user interest, is proposed in literature [3] where the time weight and Data weight of resource similarity are weighted. Literature [4] proposed a dynamic recommendation technique. As the mobile device flourishesa SUCM model is proposed in [5], which learn fine-grained user preferences to make recommendations to users.

Generally, there are two traditional solutions to the problem of scalability in collaborative filtering technology. The first is based on clustering. In literature [6], it proposes to set up a representative user for each cluster, and select the neighbor user set by calculating and representing the similarity of the user. There is a method basing on deep learning to solve the problem of scalability as proposed in [7]. In [8], a hierarchical Bayesian network with cooperative deep learning is proposed to solve the problem of data researchers even propose to use of principal component analysis techniques in statistics to achieve the scale matrix dimensionality reduction [10]. For the problem of cold-starting, the literature [11] proposed the use of proximity, impact and popularity to comprehensively consider the influence of user rating on the similarity between users. In [12], a new heuristic similarity calculation model is proposed, which can alleviate the cold-starting problem and improve the similarity between users. Literature [13] solve the cold-starting problem with self-coding, and improve the quality of top-N recommendation. In [14], the social network is utilized to analyze the strong and weak relationships in the social community, and an EM algorithm is proposed to improve the recommendation quality. In [15], deep learning technique is adopted to generate a learning function between the content and the user interaction, and to transform the user preference into a ranking list that will be recommended to users.

On the issue of popularity bias of item, literature [16] alleviates the problem by changing the popularity distribution of recommended result. This study tries to "box" all items popularity, and then maps the "box" according to the user's actual score. After mapping, a three-dimensional vector will be generated, standing for object popularity basing on the user interest feature - "vectorization". Finally, calculate the similarity between users.

Generally, the higher popularity of the item, the more interest of the user, that is, user interest is related to the popularity of items. Regarding the popularity of items as user interest feature to build a model, this paper proposes a user interest similarity model basing on the popularity of items. This article holds that the user's short-term interest is not necessarily constant, which needs to be determined according to the stability of the user's interest. If the user's interest is stable, thus the user's interest will not change with the past of time. If the

user's interest is unstable, the interest varies. This paper, from the perspective of popularity and stability of interest, proposes a time-aware, user interest stable, hybrid recommendation algorithm.

2 Methodology

2.1 User Interest Feature Similarity Model Basing on Item Popularity

Definition 1 (The popularity of Item i)**:** The ratio of the number of evaluation to the total number of items. The formula is as follows:

$$popularity_i = \frac{count_i}{count_I} \tag{1}$$

Where $count_i$ represents the number of user who had evaluated item I, $count_i$ represents the total number of items.

Algorithm Description

(1) Take 3 intervals of items popularity $[a_1, a_2), [b_1, b_2), [c_1, c_2)$
(2) Use formula $popularity_i$ to calculate all items popularity which had been evaluated by user u.
(3) Box all the item popularity derived in step (2), and then map to step (1). pseudo boxing code are as following:

Algorithm 1. Popularity packing algorithm

Input: Training data: $X = \{x_1, ..., x_n\}$; Binary bits: M
Output: Bitwise weights functions: $U = \{\mu_1, \cdots, \mu_M\}$; Hash functions: $V = \{v_1, \cdots, v_M\}$

```
for p_i → p_n by 1 do
    if p_i ∈ [a_1, a_2) then
        boolean flagfirst=true
    else if p_i ∈ [b_1, b_2) then
        boolean flagse=true
    else if p_i ∈ [c_1, c_2) then
        boolean flaghird=true
    end if
end for
```

Notes: $a_1, a_2, b_1, b_2, c_1, c_2$ are threshold data derived from the experiment. The pseudo mapping code is shown in Algorithm 2 .

(4) Calculate interest feature similarity of user A and user B basing on item popularity with User Feature Vector derived from step (3), using cosine similarity formula. The formula is as following:

$$Item\_pop\_sim = cos{<}A, B{>}\frac{A \cdot B}{\|A\| \cdot \|B\|} \tag{2}$$

The above Model called Item_pop_sim Model (Item Popularity Similarity Model), simply put as IPS Model.

Algorithm 2. Mapping feature vector algorithm

Input: Training data: $X = \{x_1, ..., x_n\}$; Binary bits: M

Output: Bitwise weights functions: $U = \{\mu_1, \cdots, \mu_M\}$; Hash functions: $V = \{v_1, \cdots, v_M\}$

```
if flagfirst&&!flagse&&!flagthird then
    featurevector:=(a,b,b)
else if flagfirst&&!flagse&&flagthird then
    featurevector:=(a,b,a)
else if flagfirst&&flagse&&!flagthird then
    featurevector:=(a,a,b)
else if flagfirst&&flagse&&flagthird then
    featurevector:=(a,a,a)
else if !flagfirst&&!flagse&&!flagthird then
    featurevector:=(b,b,a)
else if !flagfirst&&flagse&&!flagthird then
    featurevector:=(b,a,b)
else if flagfirst&&flagse&&flagthird then
    featurevector:=(b,a,a)
else
    featurevector:=(b,b,b)
end if
```

2.2 Time-Aware Similarity Model with Stability of Interest

In the actual application process, the user's interest is usually volatile, related not only with the item score value given by the user, but also with the popularity of items, and combining the two makes the user interest, which is defined as follows:

Definition 2 (The Interest of User u)**.** The interest vector set made by user u for Items $P_u = (p_{u1}, p_{u2}, p_{u3} \cdots p_{ui})$.

Definition 3 (The Interest Vector of User u to item i)**.** The weighted sum of the ratio of the actual Score of user u for item i to Full Score, and plus the popularity degree of this item. Formula are as following:

$$P_{ui} = \alpha \times \frac{R_{ui}}{R_{max}} + \beta \times popularity_i \tag{3}$$

$$popularity_i = \frac{popularity_i - popularity_{min}}{popularity_{max} - popularity_{min}} \tag{4}$$

where p_{ui} represents The Popularity of user u to item i, R_w is the rating of user u to item, R_{max} represents the full score, indicates the maximum score to the item. $popularity_i$ is the Popularity of item i in formula (1). $popularity_{min}$ is the Minimum Popularity for all items, while $popularity_{max}$ is the Maximum Popularity. α, β are parameters which can be verified through experiments, and $\alpha + \beta = 1$. Calculate the Interest Similarity for users by using Cosine Similarity, formula is as following:

$$sim_p(u, v) = \frac{P_u \cdot P_v}{\|P_u\| \cdot \|P_v\|} \tag{5}$$

The model can effectively alleviate the problem of deviations of user by introducing the weighted popularity of items.

Definition 4 (Stability of Interest to User u)**.** The variance of score for user u to all item.

$$s_{interest(u)} = \frac{\sum_{i=1}^{n}(u_i - E(u))^2}{n} \tag{6}$$

where u_i represents the ratings to item u, n is total number of item evaluated by user u, u is the average value for all items given by the user.

Actually, the variance is to measure the stability of the user's interest, in other words, the smaller the variance, the more stable the user's interest.

Factors influencing the user interest: personal factor, time and environment. In reality, the user's interest is actually volatile, may be affected by their own factors, the surrounding environment, friends and family, interest will silently transform over time, previous interest may fade or disappear, gradually. But, If the user's interest is relatively stable, the paper does deem that user interest will not have much change as time go by. Therefore, from the factors that affect the user's interest, the time-sensitive will be put forward.

Definition 5 (time-sensitive)**.** On the basis of the stability of the two users' interest, the closer of evaluation time of the user's score to the item is, the higher the similarity between the users, that is, the user's interest similarity is time-sensitive.

Overall, this paper presents a time-sensitive similarity calculation model with user interest stability.

$$stability\_Time\_sim_p = \frac{\delta|\sigma_u - \sigma_v|\frac{\sum X^2}{\sum Y^2}\sum_{i \in I_u \bigcap I_v} e^{-\varphi|t_{ui} - t_{vi}|}\overrightarrow{P}_u \cdot \overrightarrow{P}_v}{\|P_u\| \cdot \|P_v\|} \tag{7}$$

where, $\sigma_u = \sigma_u - \sigma_{med}$, $\sigma_v = \sigma_v - \sigma_{med}$, σ_u and σ_v represent the decentralized variance, for user u and user v reviews, σ_{med} is the median of variance, respectively. The model in the above formula (7) called the Stability_Time_Sim model, STS in brief.

2.3 The Fusion of Two Similarity Models

In the above, we introduced two similarity models, and each has its own advantages, among which the user interest similarity model basing on item popularity can effectively alleviate the problem of item bias, while the similarity model introducing time-sensitive stability of user interest in real-time, digs long tail items, improves the novelty of recommendation system. So, in order to make the recommendation system better, we linear weight the two models and put forward the function Item_Pop_Stability_Time_sim model:

$$Item\_Pop\_Stability\_Time\_Sim = \lambda \times IPS + (1 - \lambda) \times STS \tag{8}$$

This is IPSTS Model, which comprehensively puts the stability of users, time-sensitive, and item popularity factor etc. into consideration. We synthesize these factors to build a model, and the experiment shows that, the weighted similarity model has a marked improvement in the quality of recommendation.

3 Experiment Design and Result Analysis

3.1 Data

The data set in this experiment is provided by the MovieLens site and developed by the GroupLens team at the University of Minnesota. MovieLens was founded in 1997, a web-based recommendation system. Currently, the site offers three different levels of data sets: 100,000 records of 943 users to 1682 movies; 6040 users to 3900 films ratings of 1 million data; 71567 users to 10,681 films ratings of 10 million data.

This Experiment adopts the first data sets above, in which each user evaluated 20 films at least. The sparsity for this data sets is $1-1000000/(943\times1682) = 0.937$. In this paper, the data set is divided into training set and test set, among which 80% are the training set, and 20% are test set.

The Evaluation for quality of recommender in this paper is, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE), respectively [16].

3.2 Experimental Results

Experiment in this paper makes a comparison among the IPSTS model, the Pearson model and the Euclidean model.

IPSTS Model Result. Figure 1 (abscissa is the number of selected neighbor-sordinate is RMSE) shows the effect of each similarity model on RMSE when selecting different neighbors. The number of neighbors is 10, 20, 30, 40, 50, 60, 70, 80 and 90 respectively. The results showed that the RMSE of IPSTS Model is less than Euclidean Model and Pearson Model (e.g. When the number of neighbors is 20, the RMSE of Model about 6% lower than Pearson Model,

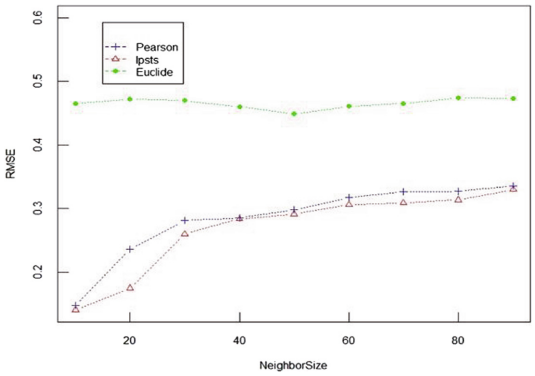

Fig. 1. IPSTS model, pearson model and euclidean model in RMSE comparison

and approximately 30% lower than the Euclide Model). Therefore, the error is reduced and the recommender quality is improved.

Figure 2 (abscissa is the number of selected neighborsordinate is MAE) shows the effect of each similarity model on MAE when selecting different neighbors. The number of neighbors is 55, 60, 65, 70, 75, 80, 85 and 90 respectively. The results showed that the MAE of IPSTS Model is less than Euclidean Model and Pearson Model (e.g. When the number of neighbors is 80, The MAE of IPSTS Model about 1% lower than Pearson Model, and approximately 8% lower than the Euclidean Model). Therefore, the recommender quality is improved, too.

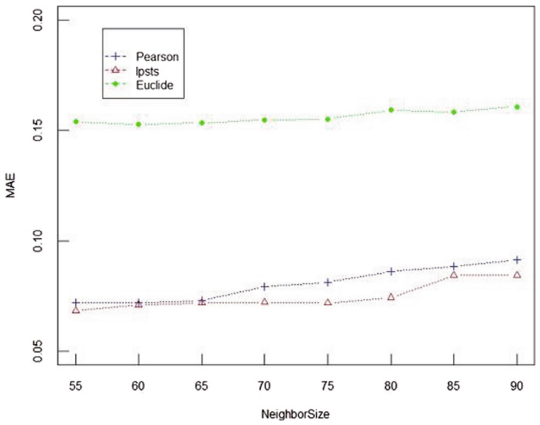

Fig. 2. IPSTS model, pearson model and euclidean model in MAE comparison

The Impact of *IPSTS* Model on Long Tail Items. Figure 3 indicates that these items exhibit long tail phenomenon. Among which the higher the rank of

the item ID, the more prevalent and vice versa, Item ID after 1000 are the long tail items. To test the long tail items mining ability of the model, the experiment is as follows: randomly select 3 users (user ID are 80, 800, 888) and make top-5 recommendation for them respectively. Encode for the user ID that needs to be recommended, the recommended list size is 5, and the number of Neighbor is 90. In the Pearson model, the RMSE is 0.33560, and the RMSE is 0.32965 in IPSTS model.

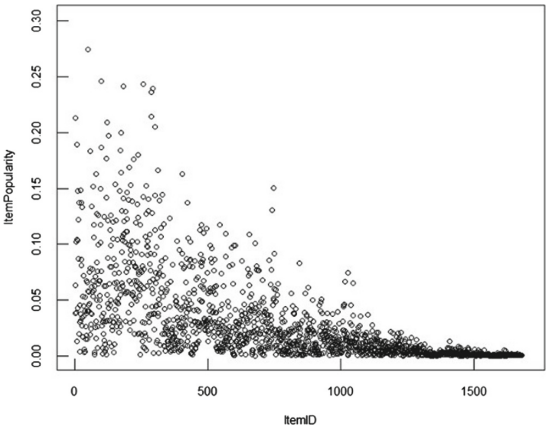

Fig. 3. Diagram for item ID and item popularity

4 Conclusion

Recommendation system is to help user to find useful information that they are interested in big data, then display and recommend to user by appropriate ways, in the situation that they do not have clear requirements. This paper focuses on reducing ratings error and excavating long tail items, alleviating bias of popularity of item and importing stability of interest, at the same time, which also draws the attention of time-sensitive factor thereby, to construct a time-sensitive similarity fusion model of importing interest stability. Experimental data indicates that the model put forward in this paper can dynamically recommend item to users in real-time and improve the recommendation quality simultaneously.

However, user interest can be affected by external factors. With the development of mobile devices and positioning system, which can also be imported into the hybrid model proposed in this paper. Secondly, the fusion model is weighted linearly, and the follow-up study could consider building a non-linear model for recommendation.

Acknowledgements. This research is partially supported by the Chinese National Natural Science Fund (No. 61841602), the Natural Science Foundation of Shandong

Provice (No. ZR2018PF005) and Shandong Province International Cooperation Trianing Project for University Teacher. We express our thanks to Dr. Rongju Li who checked our manuscript.

References

1. Anonymous: statistical report on internet development in China. Internet Commun. (7), 54–59 (2015)
2. Resnick, P., Iacovou, N., Suchak, M., et al.: GroupLens: an open architecture for collaborative filtering of netnews. In: Working Paper Series, pp. 175–186 (2015)
3. Liang, X.: Research on key technologies of dynamic recommendation system. Institute of Automation, Chinese Academy of Science, Beijing (2011)
4. Liu, B., Wu, Y., Gong, N.Z., et al.: Structural Analysis of User Choices for Mobile App Recommendation. ACM Transactions on Knowledge Discovery from Data **11**(2) (2016). Article No. 17
5. Rashid, A.M., Lam, S.K., Karypis, G., et al.: ClustKNN: a highly scalable hybrid model- memory-based CF algorithm. In: The Workshop on in Proceeding of WebKDD (2006)
6. Elkahky, A.M., Song, Y., He, X.: A multi-view deep learning approach for cross domain user modeling in recommendation systems. In: The International Conference, pp. 278–288 (2015)
7. Wang, H., Wang, N., Yeung, D.Y.: Collaborative deep learning for recommender systems. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1235–1244. ACM (2014)
8. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, pp. 426–434 (2008)
9. Kim, D., Yum, B.J.: Collaborative filtering based on iterative principal component analysis. Expert Syst. Appl. Int. J. **28**(4), 823–830 (2005)
10. Ahn, H.J.: A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. Inf. Sci. **178**(1), 37–51 (2008)
11. Liu, H., Hu, Z., Mian, A., et al.: A new user similarity model to improve the accuracy of collaborative filtering. Knowl.-Based Syst. **56**(3), 156–166 (2014)
12. Li, S., Kawale, J., Fu, Y.: Deep collaborative filtering via marginalized denoising auto-encoder. In: The ACM International, pp. 811–820 (2015)
13. Wu, Y., Dubois, C., Zheng, A.X., et al.: Collaborative denoising auto-encoders for Top-N recommender systems. In: ACM International Conference on Web Search and Data Mining, pp. 153–162. ACM (2016)
14. Vuurens, J.B.P., Larson, M., De Vries, A.P.: Exploring deep space: learning personalized ranking in a semantic space. In: Workshop on Deep Learning for Recommender Systems, pp. 23–28 (2016)
15. Adomavicius, G., Kwon, Y.O.: Improving aggregate recommendation diversity using ranking-based techniques. IEEE Trans. Know. Data Eng. **24**(5), 896–911 (2012)
16. Herlocker, J.L.: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22**(1), 5–53 (2004)