Bipartite Graph Network Recommendation Algorithm Based on Differential Path Weights (Postprint)

Authors: Gao Changyuan, Duan Wenbin, Zhang Shuchen

Date: 2018-05-18T00:00:00+00:00

Abstract

Aiming at the phenomenon of unreasonable resource allocation in bipartite graph network structure recommendation algorithms, and simultaneously to enrich the diversity of recommendation results and promote the recommendation of long-tail items, a bipartite graph network structure algorithm utilizing differential path weights to alter resource transmission is proposed. User similarity is employed to construct path weights that modify the resource transmission rules in the first stage, enabling resources to flow more abundantly to user nodes similar to the target user. By constructing path weights based on item attribute similarity, resources in the second stage are directed more towards items with attributes similar to those already purchased by the target user. Experimental results indicate that the proposed algorithm enhances the comprehensive performance of recommendation compared to other prevalent network structure algorithms, and better resolves related issues in recommendation.

Full Text

Preamble

Recommendation Algorithm for Bipartite Graph Network Structure Based on Differential Path Weight

Gao Changyuan a,b, Duan Wenbin a, Zhang Shuchen a,b
(Harbin University of Science and Technology a. College of Economic & Management, b. High-tech Industrial Development Research Center, Harbin 150040, China)

Abstract: To address the unreasonable resource allocation in bipartite graph network structure recommendation algorithms and to enrich recommendation diversity while promoting unpopular items, this paper proposes a novel algorithm

that employs differential path weights to modify resource transfer in bipartite graph networks. The algorithm constructs path weights using user similarity to alter the first-stage resource transfer rules, directing more resources toward user nodes similar to the target user. Path weights are further constructed through item attribute similarity, enabling second-stage resources to flow preferentially toward items with attributes similar to those already purchased by the target user. Experimental results demonstrate that the proposed algorithm improves the comprehensive performance of recommendations compared to other popular network structure algorithms and better addresses related recommendation challenges.

Key Words: bipartite network structure; differential path weight; recommendation algorithm; user similarity; item attributes

0 Introduction

With the rapid development of the Internet and the gradual maturation of social and enterprise informatization, increasingly vast amounts of data are generated through various terminals to serve people' s daily lives. This massive data influx has triggered the "information overload" problem, driving the rapid development of recommendation systems that can satisfy user needs. Recommendation algorithms constitute a crucial module in recommendation systems, primarily responsible for selecting information or items that match user interests and presenting them to users. Current mainstream recommendation algorithms include collaborative filtering recommendation algorithms (CFRA), content-based recommendation algorithms (CBRA), knowledge-based recommendation algorithms (KBRA), and bipartite network structure recommendation algorithms (BNSRA). Among these, BNSRA has gained increasing popularity among recommendation researchers due to its better recommendation diversity and independence from item category limitations.

Currently, CFRA research is relatively systematic and widely adopted by numerous e-commerce platforms. However, extensive experiments by scholars have demonstrated that BNSRA outperforms traditional CFRA in recommendation efficiency and accuracy. BNSRA virtualizes users and items in the system as nodes in a bipartite graph network, with purchase relationships forming connections between nodes. An increasing number of researchers have attempted to optimize BNSRA, particularly with the introduction of diffusion dynamics and heat conduction, leading to two major improvement directions: material diffusion-based BNSRA (MDBN) and heat conduction-based BNSRA (HCBN). Scholars have also optimized BNSRA performance from eight perspectives: improving initial resource allocation, reducing popularity bias, satisfying user personalization, eliminating redundant attributes, considering item content, applying community detection, incorporating temporal effects, and developing hybrid algorithms, thereby enriching recommended item categories and compensating for deficiencies in recommending unpopular items.

Traditional recommendation algorithms typically tend to recommend popular items without considering user demand diversification. Therefore, enriching recommendation diversity and increasing recommendations for unpopular items are practically significant for enabling recommendation systems to serve more users. Based on this, this paper utilizes user similarity to construct path weights for bipartite graph network structure resource transfer, altering the traditional equal resource transfer to improve recommendation accuracy. Simultaneously, to enhance recommendation diversity and promote unpopular items, this paper modifies the resource transfer rules from users to items by calculating Hamming distances between item attributes, thereby increasing recommended item categories. Experimental verification demonstrates that the proposed algorithm improves both recommendation accuracy and comprehensive algorithm performance.

1 Bipartite Graph Network Structure Recommendation Algorithm

The bipartite graph network structure recommendation algorithm posits that items purchased by users have the capacity to recommend other items. This capacity is realized by assigning certain resource quantities to items based on their selection patterns, which are then transferred to other items through the user-item network.

Assume target user u_α has purchased some items in the system, with each purchased item initially assigned a resource quantity of 1. Consider a system with m types of items and n users. Let $O = \{o_1, o_2, o_3, ..., o_m\}$ represent the m items in the system, and $U = \{u_1, u_2, u_3, ..., u_n\}$ represent the n users. The items and users can form an $n \times m$ relationship matrix $a_{i\alpha}$, where $i \in \{1, 2, 3, ..., m\}$ and $\alpha \in \{1, 2, 3, ..., n\}$. The values in relationship matrix $a_{i\alpha}$ are defined as follows:

$$a_{i\alpha} = \begin{cases} 1, & \text{if user } u_\alpha \text{ has selected item } o_i \\ 0, & \text{if user } u_\alpha \text{ has not selected item } o_i \end{cases}$$

The resource transfer process in the bipartite graph network structure is shown in [Figure 1: see original paper]. Initial resources are equally distributed from the item layer to the user layer through item-user selection relationships. After the first-stage resource transfer, the resource quantity obtained by user u_α is:

$$f(u_\alpha) = \sum_{i=1}^{m} a_{i\alpha} k(o_i)$$

where $k(o_i)$ represents the degree of item node o_i, i.e., the number of times item o_i has been selected. In the second stage, resources flow from each user node in

the user layer through selection relationships to the item layer, and the resource quantity of item node o_j is:

$$f(o_j) = \sum_{\alpha=1}^{n} a_{j\alpha} f(u_\alpha)$$

where $k(u_\alpha)$ represents the degree of user node u_α, i.e., the number of items purchased by the user. $f(o_j)$ indicates the contribution of item o_j to recommendations. For a specific target customer, items are sorted by their new resource quantities in descending order, and the top L items are recommended to the target user.

2 Improved Bipartite Graph Network Structure Recommendation Algorithm

2.1 Introducing User Similarity to Construct Path Weights

In the first stage of bipartite graph network structure recommendation algorithms, resources are equally allocated based on item degrees without considering interest differences among users. If users who purchase the same item assign different ratings, their preferences for that item differ. References [16,17] improved algorithm accuracy by clustering to distinguish user interest differences and modifying resource transfer rules. In the first-stage resource transfer, item resources should be directed toward users with preferences more similar to the target user. Therefore, this paper introduces a user similarity coefficient and obtains an inter-user difference function by calculating the mean rating difference among users in the system.

The Jaccard similarity coefficient can be defined as:

$$sim_{Jaccard}(I_\alpha, I_\beta) = \frac{|I_\alpha \cap I_\beta|}{|I_\alpha \cup I_\beta|}$$

where I_α and I_β represent the sets of items with actual ratings by users u_α and u_β, respectively. A larger Jaccard similarity coefficient indicates that users u_α and u_β have rated more items in common, suggesting higher probability of consistent interests and greater similarity. For the common rated items between u_α and u_β, we compute the absolute difference of their ratings: let $d_i = |r_{i\alpha} - r_{i\beta}|$, where $r_{i\alpha}$ and $r_{i\beta}$ are the ratings given by users u_α and u_β to item i, respectively. Let N be the number of items co-rated by u_α and u_β. The mean rating difference between users is:

$$average(\alpha, \beta) = \frac{\sum_{i \in I_\alpha \cap I_\beta} d_i}{N}$$

To modulate the Jaccard similarity coefficient such that it decreases when inter-user rating gaps are large and increases when gaps are small, we normalize the mean rating difference and remove rating scale effects, obtaining the inter-user difference function:

$$D(\alpha,\beta) = 1 - \frac{average(\alpha,\beta)}{r_{max} - r_{min}}$$

where r_{max} and r_{min} represent the maximum and minimum rating values among all items co-rated by the two users. Through this calculation, we can construct the user similarity function:

$$sim(\alpha,\beta) = \frac{|I_\alpha \cap I_\beta|}{|I_\alpha \cup I_\beta| \cdot D(\alpha,\beta)}$$

We construct the first-stage path weight $S_{\alpha\beta}$, enabling user nodes more similar to the target user to obtain greater resource values:

$$S_{\alpha\beta} = \frac{sim(\alpha,\beta)}{\sum_{t=1}^{n} sim(\alpha,t)}$$

$S_{\alpha\beta}$ represents the proportion of similarity between user u_β and target user u_α relative to the sum of similarities between all users who selected item o_i and the target user. The result falls within [0,1]. A larger $S_{\alpha\beta}$ indicates that among all users who have purchased item o_i, target user u_α is more similar to user u_β than to other users, causing more initial resources on item o_i to flow to u_β.

If we take u_4 as the target user in the system, the first-stage resource transfer changes for item o_1 after improving path weights are shown in [Figure 2: see original paper].

2.2 Introducing Item Similarity to Construct Path Weights

When users select items, certain attributes may drive their purchases. Although items may belong to different categories, they often share some similar attributes. Leveraging these attributes can expand recommended item categories and include rare unpopular items. Therefore, studying item attribute similarity has practical significance for recommendation system improvement. Some scholars have improved recommendation performance by using item tags as bipartite graph extensions [18]. For item similarity calculation, different attributes have different weights. Manual weight assignment may cause data smoothing issues. This paper employs Gray code to represent different attribute values and uses Hamming distance to distinguish differences between items, avoiding manual attribute weight setting and enabling the algorithm to automatically analyze the importance of different attributes through self-learning.

For item attributes, we first determine the number of categories c for each attribute and the corresponding code bits b. Different attribute values of an item are Gray-coded and concatenated into a Gray code string Nom^B. The attribute difference between two items is obtained by computing the Hamming distance between their Nom^B strings. Let $HMD(i,j)$ denote the Hamming distance difference between items o_i and o_j. Suppose item o_i has attribute string $Nom_i^B = (0101000000)$ and item o_j has $Nom_j^B = (1100010100)$. The Hamming distance is $HMD(i,j) = 4$.

To achieve an effect similar to $D(\alpha, \beta)$, we normalize the Hamming distance between different item attributes and remove difference scale effects, obtaining the item attribute difference function:

$$D(i,j) = 1 - \frac{HMD(i,j)}{HMD_{max}}$$

where HMD_{max} is the maximum Hamming distance between items o_i and o_j. The item attribute difference function can incorporate attributes describing item features into similarity calculation as much as possible, helping enrich recommended item categories. Integrating the item attribute difference function into the Jaccard similarity coefficient yields the item similarity function:

$$sim(i,j) = \frac{|T_i \cap T_j|}{|T_i \cup T_j| \cdot D(i,j)}$$

where T_i and T_j represent the numbers of characteristic attributes for items o_i and o_j, respectively. Through $sim(i,j)$, we construct the resource transfer path weight $Q_{\beta j}$ from users to items, enabling item nodes more similar to those selected by the target user to obtain more resources:

$$Q_{\beta j} = \frac{\sum_{t=1}^{m} sim(i_t, j)}{\sum_{i=1}^{m} sim(i,j)}$$

$Q_{\beta j}$ represents the proportion of similarity between item o_j and items selected by target user u_α relative to the total similarity. If we take u_4 as the target user and consider item o_3 selected by user u_4, the second-stage resource transfer changes after improving path weights are shown in [Figure 3: see original paper].

After two rounds of resource transfer from the initial resource quantities of items selected by the target user, we can obtain the new resource quantity for each item node. The top L items with the highest resource quantities are selected and recommended to the user.

3 Algorithm Detailed Steps

The specific steps of the improved BNSRA using differential path weights are as follows:

Input: User rating matrix W, item attribute table P, target user u_α
Output: Recommendation list for user u_α

a) Construct the bipartite graph network G between users and items through the user rating matrix, and compute user degrees.

b) In network G, compute different user $sim_{Jaccard}$ coefficients and calculate the mean user rating difference $average(\alpha, \beta)$.

c) Compute the first-stage path weight $S_{\alpha\beta}$ using formula (7) based on $average(\alpha, \beta)$ and $sim_{Jaccard}$ coefficients.

d) Convert data in P to binary, perform XOR operations to obtain item attribute difference quantities $HMD(i, j)$ and maximum difference quantity HMD_{max}.

e) Obtain the intersection and union quantities of attributes between two items from table P, and compute the second-stage path weight $Q_{\beta j}$ using formula (11).

f) Compute the transferred resource quantity of item nodes using formula (13).

g) Sort item resource quantities in descending order, remove already-purchased items, and select the top L items as the recommendation list. Algorithm ends.

Time complexity analysis: Assume the system has n users and m items. In step a), user degrees can be computed offline with time complexity $O(n \times m)$. In step b), since $sim_{Jaccard}$ coefficients and mean user rating differences can be calculated offline, the first-stage path weight complexity is $O(n^2)$, also computable offline. In step d), data processing can be performed directly offline. When computing second-stage path weights in step e), the complexity is $O(m^2)$, but this can also be done offline. Finally, computing new item node resource quantities through two-stage resource transfer has time complexity $O(n \times m)$. Therefore, the overall time complexity is $O(n \times m)$, where n is the number of users and m is the number of items.

4.1 Experimental Data

The experimental data selected for this study includes: 100,000 rating records from 943 users on 1,682 movies in the GroupLens MovieLens dataset; rating information from 72,916 users on 1,628 movies in the EachMovie dataset; and rating data from 480,189 users on 17,770 movies in the Netflix dataset.

4.2 Evaluation Metrics

a) **Hamming distance (H)** serves as an important indicator for detecting recommendation diversity. Let L represent recommendation list length and n represent the number of identical items in recommendation results. The diversity of recommended item categories for the entire system can be expressed as:

$$H = 1 - \frac{n}{L}$$

A larger H value indicates greater diversity in system-recommended items.

b) **Popularity ($\langle k \rangle$)** reflects the system' s ability to recommend relatively unpopular items. System popularity can be represented by the average degree of recommended items:

$$\langle k \rangle = \frac{1}{n} \sum_{j=1}^{n} \frac{\sum_{k=1}^{L} k_{jk}}{L}$$

where n denotes the number of users. A higher $\langle k \rangle$ indicates more popular items in recommendations.

c) **Precision (P)** represents the proportion of items in recommendation results that target users have actually purchased. Higher P values indicate better algorithm accuracy:

$$P = \frac{N_{tp}}{N_{tp} + N_{fp}}$$

where N_{tp} refers to the number of items users have purchased among recommended items, and N_{fp} refers to the number of recommended items.

d) **Recall (R)** is the ratio of recommended items that target users have purchased to the total number of items purchased by target users. Higher R values indicate better algorithm recall:

$$R = \frac{N_{tp}}{N_{tp} + N_{fn}}$$

where N_{tp} is the number of items in the recommendation list that users have selected, and N_{fn} is the total number of items selected by users.

e) **F-measure** evaluates the comprehensive performance of recommendation systems, commonly expressed as the weighted harmonic mean of Precision and Recall:

$$F1 = \frac{2 \times P \times R}{P + R}$$

$F1$ serves as a standard metric for algorithm recommendation capability, where larger $F1$ values indicate better recommendation effectiveness.

5 Experimental Results and Analysis

Baseline algorithms for comparison include: the proposed Differential Path Weight (DPW) bipartite graph network recommendation algorithm, Collaborative Filtering (CF), Network-Based Inference (NBI) [4], Heterogeneous Heat Conduction (HHC) bipartite network recommendation algorithm [14], and Hybrid method of Heat Conduction and Mass Diffusion (HHM) [15].

Recommendation diversity performance: As shown in [Figure 4: see original paper], CF, HHC, HHM, and DPW demonstrate significant improvement in recommendation diversity compared to traditional network structure algorithms. With smaller datasets, DPW' s diversity performance is similar to HHC. As recommendation list length increases, DPW shows better diversity capability. With larger datasets, DPW exhibits superior recommendation diversity, demonstrating that as the number of displayed items and total items in the recommendation system increase, DPW can present more diverse item categories.

Unpopular item recommendation capability: As shown in [Figure 5: see original paper], HHC, HHM, and DPW significantly outperform traditional CF and NBI algorithms in recommending unpopular items. DPW and HHC perform well when recommending very few products. As the number of recommended items and total system items increase, the three superior algorithms gradually enhance their ability to recommend unpopular items. When the number of recommended items exceeds 22, DPW demonstrates better capability across multiple datasets. Therefore, when recommending larger numbers of items, DPW is more likely to suggest unpopular items.

Algorithm accuracy and performance: Since this algorithm modifies resource quantities in items rather than presenting results through user ratings, MAE comparison with other algorithms is not feasible. Therefore, this study primarily uses P, R, and $F1$ values for performance measurement. Experimental results show that DPW' s Precision and Recall gradually improve as recommendation list length increases. In the smaller MovieLens dataset, DPW' s performance is similar to HHC. As dataset size increases, DPW' s recommendation performance gradually strengthens, as shown in [Figure 6: see original paper] and [Figure 7: see original paper].

Comprehensive algorithm performance: DPW exhibits unstable performance and slightly underperforms HHC when the total number of items in the

recommendation system is low. As the total number of system items and recommended items increase, DPW' s performance gradually improves, surpassing both HHC and HHM at multi-item stages, as shown in [Figure 8: see original paper].

When recommending small numbers of items, DPW does not achieve optimal results. Investigation of the MovieLens dataset reveals that this database contains many popular items, resulting in fewer unpopular recommendations when the recommendation list is short. However, across multiple datasets, DPW shows improvement in both recommendation diversity and accuracy compared to current popular network structure algorithms and collaborative filtering algorithms.

6 Conclusion

To address the issues of equal resource allocation in bipartite graph network structure recommendation algorithms and the popularity bias and weak diversity in traditional recommendation algorithms, this paper proposes a bipartite graph network structure recommendation algorithm based on differential path weights. Experimental analysis reveals that introducing user similarity modifies recommendation results, while leveraging item attribute similarity reduces recommendation popularity and further improves recommendation quality. Results indicate that when the system recommends larger numbers of items, the proposed algorithm provides richer item categories and outperforms several popular algorithms. The algorithm also performs effectively in scenarios with sparse experimental data. Overall, the proposed algorithm demonstrates improved comprehensive performance compared to mainstream network recommendation algorithms. By reducing the popularity of recommended items, the algorithm enables personalized recommendation of unpopular items. Future research should adapt the algorithm for different application scenarios and improve its efficiency when processing larger volumes of data.

References

[1] Zhao Hongchen, Zhai Lili, Zhang Shuchen. Research on collaborative filtering recommendation method based on grey correlation clustering and tag overlap factor [J]. Computer Engineering & Science, 2016, 38(1): 171-176.

[2] Pazzani M J, Billsus D. Content-Based Recommendation Systems [C]//Adaptive Web. Springer-Verlag, 2007: 325-341.

[3] Trewin S. Knowledge-based recommender systems [J]. Encyclopedia of library and information science, 2000, 69(32): 180-186.

[4] Zhou T, Ren J, Medo M. Bipartite network projection and personal recommendation [J]. Physical Review E, 2007, 76(4): 046115.

[5] Zanker M, Jessenitschnig M. Case-studies on exploiting explicit customer requirements in recommender systems [J]. User Modeling and User-Adapted Interaction, 2009, 19(1): 133-166.

[6] Zhang Y C, Blattner M, Yu Y K. Heat conduction process on community networks as a recommendation model [J]. Physical review letters, 2007, 99(15): 154301.

[7] Zhou Y, Lu L, Liu W, et al. The power of ground user in recommender systems [J]. PLoS ONE, 2013, 8(8): e70094.

[8] Zhou T, Jiang L L, Su R Q, et al. Effect of initial configuration on network-based recommendation [J]. EPL (Europhysics Letters), 2008, 81(5): 58004.

[9] Sun Yuhua, Zeng Qingduo. Global DEA model for two-stage network systems [J]. Statistics & Decision, 2014, (11): 43-46.

[10] Zhang Xinmeng, Jiang Shengyi, Zhang Qiansheng. Hybrid network recommendation algorithm based on user preference weighting [J]. Journal of Shandong University: Natural Science, 2015, 50(9): 29-35.

[11] Xiong Xiangyun. Research on multi-dimensional recommendation technology based on bipartite network [D]. Jiangsu: Soochow University, 2013: 29-33.

[12] Koren Y. Collaborative filtering with temporal dynamics [J]. Communications of the ACM, 2010, 53(4): 89-97.

[13] Wang Qian, Duan Shuangyan. An improved recommendation algorithm based on bipartite graph network structure [J]. Computer Application Research, 2013, 30(3): 771-774.

[14] Hu Jiming, Lin Xin. Design and implementation of social recommendation algorithm based on user-resource-vocabulary tripartite graph [J]. Information Studies: Theory & Application, 2016, 39(3): 130-134.

[15] Zhou T, Kuscsik Z, Liu J G, et al. Solving the apparent diversity-accuracy dilemma of recommender systems [J]. Proceedings of the National Academy of Sciences, 2010, 107(10): 4511-4515.

[16] Ge Zhipeng, Yan Guangle, Zhang Guoliang. Bipartite graph network recommendation algorithm based on ant colony clustering [J]. Information Technology, 2016, (3): 57-61.

[17] Wang Tongyuan. Bipartite graph network recommendation algorithm based on bisecting K-means clustering [J]. Management Observer, 2015(25): 3.

[18] Xiao Yang, Wang Daoping, Yang Cen. Knowledge recommendation algorithm based on tripartite graph network structure [J]. Computer Application Research, 2015, 32(2): 386-390.

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