Collaborative Filtering Recommendation Algorithm Based on Xml Fuzzy Data

Yan Jiang and Meihua Dong
Department of Software, Shenyang University of Technology, Shenyang, China
Email: jiangyansygd@sut.edu.cn; 247023087@qq.com

Abstract. To analyse the limitations of traditional recommendation algorithms, we propose a collaborative filtering recommendation algorithm based on XML fuzzy data. Firstly, the method grasped and modelled the fuzzy attribute features of items and established the membership matrix of fuzzy attribute features, then used XML similarity calculation to extract the similarity between project fuzzy attribute features and combined it with the traditional collaborative similarity to get the comprehensive similarity. Finally, recommendations were made through this comprehensive similar. Experimental results show that, compared with the traditional collaborative filtering recommendation algorithm, the establishment of membership matrix of fuzzy attribute features solves the problem that the description of goods or items is usually fuzzy, which makes the calculation of neighboring items more accurate and improves the accuracy of the recommendation system. At the same time, when the new item has no user scoring information, it can be recommended through the similar of the fuzzy attribute features among items, thus effectively solve the cold-start problem.

1. Introduction

With the blowout development of the Internet industry, more and more information acquisition methods have been applied. Traditional search methods can no longer meet the needs of users to find information, resulting in a significant increase in the cost of users to obtain valuable information. At this time, the recommendation system came into being.

In recommendation techniques and algorithms, collaborative filtering recommendation algorithm is widely recognized and used. Collaborative filtering is a typical method of using collective intelligence, which makes recommendations based on users with similar interests [1], [2]. However, collaborative filtering recommendation system also has problems such as cold-start and data sparsity. The cold-start problem is a hot topic. To some extent, it can be regarded as an extreme case of data sparsity [3]. With the continuous research of scholars at home and abroad, many methods have been proposed to improve this problem. Zhang and Zeng [4] used popular recommendation methods to prove that new users prefer to focus on popular products, so popular recommendations can also achieve good results. However, every user sees the same recommendation content, which is not personalized. Yu and Li [5] used the concept of user time weight information to solve the cold-start problem of new items.

"Fuzzy Logic" is a concept proposed by American engineer L A Zadeh in 1965 when he improved the "fuzzy set theory" of computer programs [6]. Another problem faced by the recommendation system is that there is nothing to do with these fuzzy concepts, which leads to a decline in recommendation accuracy. Currently, XML has become the standard for the representation and exchange of important data on the Web, and XML databases with fuzzy data have been extensively studied. It can be seen that the processing and discovery of fuzzy information in the recommendation system requires the processing ability of XML. Therefore, the needs of all walks of life to find
valuable knowledge and rules from a large amount of XML data to improve the quality of recommendations are very strong.

Focus on the above problems; we propose a collaborative filtering recommendation algorithm based on XML fuzzy data. Firstly, we dealt with the fuzzy attribute features of items and calculated their similarity, then combined this similarity with the collaborative similarity to obtain a comprehensive similarity, finally, got recommendations through this comprehensive similarity. The experimental results show that this method can effectively improve the cold-start problem and improve the accuracy of the recommendation system.

2. Traditional Collaborative Filtering Recommendation Algorithm and Its Analysis

In the collaborative filtering recommendation system, the similarity is first performed through the user-item scoring matrix. It can be the similarity of users or the similarity of items. These two methods are similar and the conversion between them only needs to transpose the dataset. The user-item scoring matrix is represented by $R_{m,n}$, where $m$ is the number of users, $U = \{U_1, U_2, ..., U_m\}$, and $n$ is the number of items, $I = \{I_1, I_2, ..., I_n\}$. At present, cosine similarity is widely used. Therefore, we calculates the similarity between items based on the cosine similarity, which is defined as follows:

$$
sim(I_a, I_b) = \frac{\sum_{k=1}^{m} R_{k,a} \times R_{k,b}}{\sqrt{\sum_{k=1}^{m} (R_{k,a})^2} \times \sqrt{\sum_{k=1}^{m} (R_{k,b})^2}}
$$

(1)

where $sim(I_a, I_b)$ is the similarity between items $I_a$ and $I_b$. $R_{k,a}$ represents the score of user $k$ for item $a$. $R_{k,b}$ is similar to it.

Finally, the following weighted equation (2) is used to obtain the target user's predictive score for the recommended items:

$$
P_{u,j} = \bar{R}_j + \frac{\sum_{i \in NBS_j, \sim(j,i)} \sim(j,i) \times (R_{u,j} - \bar{R}_j)}{\sum_{i \in NBS_j} (\sim(j,i))}
$$

(2)

where $P_{u,j}$ is the predicted score of user $u$ for item $j$. $\sim(j,i)$ is the similarity between two items. $NBS_j$ is the neighbor collection of item $j$ and $\bar{R}_j$ is the average score of item $j$.

Collaborative filtering has been proved to be very effective. It is based on the previous scoring database of various users and products. However, this database is usually very sparse and unfavorable for the description of fuzzy information, which affects the accuracy of recommendation system. And when a new item appears, the item cannot be recommended because of the lack of rating information.

3. Collaborative Filtering Recommendation Algorithm Based on XML Fuzzy Data

3.1. Modeling of Item Fuzzy Attribute Features

Due to the lack of interpretation of fuzzy information in traditional databases, traditional similarity calculation cannot effectively measure the similarity between items. In order to effectively utilize this fuzzy information, it is necessary to express and process this inaccurate and uncertain information. Therefore, we firstly models the fuzzy attribute features and processes them in XML.

The concept of fuzzy for item attributes is replaced here by a fuzzy number from the interval [0,1]. The fuzzy number represents the membership of an attribute feature to an item. The method of expert evaluation is used to establish the membership function of item attributes.

Suppose item $I$ has $t$ attribute features, then the attribute set of item $I$ is $P = \{P_1, P_2, ..., P_t\}$. Experts with the number of $M$ evaluated the membership of each attribute, which was between [0,1].
$E_j(P_i)$ is the evaluation of the attribute $i$ in item $I$ by the expert $j$. Then the attribute membership function of item $I$ is as follows:

$$
\mu_{P_i}(I) = \frac{1}{M} \sum_{j=1}^{M} E_j(P_i)
$$

(3)

where $\mu_{P_i}(I)$ is the membership degree of attribute $P_i$ to item $I$. The range of $\mu_{P_i}(I)$ is $[0,1]$.

3.2. Similarity of Item Fuzzy Attributes Based on XML

To describe the relationship information in the XML document, the XML document is defined as follows.

Definition 1. An XML document can be represented as a node tag tree $Tree=(N,E,root)$. Where $N$ is the node set, $E$ is the edge set and $root$ is the root node. The set of edges is a binary relationship on $N$, and if $(u,v) \in E$, $v$ is a child element or attribute of $u$.

The similarity calculation of XML generally adopts two methods, which are linear-weighted synthesis method and nonlinear synthesis method [7]. For the item fuzzy attribute similarity, we adopt the linear-weighted synthesis method. The linear-weighted synthesis method is obtained by weighting the similarity of structure and content. Among them, the content similarity of XML documents fully considers the semantic difference of different nodes in XML documents (see section 3.2.1). The structural similarity of XML documents is to transform the structure of the XML document into a vector, and then calculate them by using the correlation coefficient of vector (see section 3.2.2).

$d_1$ and $d_2$ are two XML document, and their comprehensive similarity is represented by $sim^{con}$:

$$
sim^{con} = \gamma \times SemSim(d_1,d_2) + (1-\gamma) \times StruSim(d_1,d_2)
$$

(4)

Separately, $\gamma$ and $1-\gamma$ are the similarity weights of content and structural. In the similarity calculation of XML documents, the distribution of weights is determined according to the influence of content and structure on the discovery of document information. $SemSim(d_1,d_2)$ is the content similarity between $d_1$ and $d_2$. And $StruSim(d_1,d_2)$ is the structural similarity between $d_1$ and $d_2$. They are calculated by equation (5) and equation (7) respectively.

3.2.1. Calculation of Content Similarity. For the content similarity of XML, a WordNet-based method is used to score semantic similarity. Meanwhile, the membership weights of fuzzy attribute features are added to the calculation of content similarity. First, we define semantic similarity scores between different nodes.

Definition 2. $t_1$ and $t_2$ are nodes of the XML document tree, the semantic similarity score of them is $score_{1,2}$, and the range of $score_{1,2}$ is $[0,1]$

- $t_1$ and $t_2$ are considered to match if they are identical, $score_{1,2} = 1$.
- $t_1$ and $t_2$ are partially matched if they are synonymous in WordNet, $score_{1,2} = \sigma$.
- There is no match, $score_{1,2} = 0$.

When $t_1$ and $t_2$ are partially matched, their semantic similarity score is $\sigma$. $\sigma$ is calculated by JWS, which is Java WordNet Similarity. It is an open source project based on the semantic similarity calculation between Java and WordNet, and implements many classic semantic similarity calculations. We improve literature [8] to achieve content similarity. Represent two XML documents as vectors $d_1 = (<score^1_1, \mu^1_{t_1}(d_1)>,...,<score^1_m, \mu^1_{t_m}(d_1)>)$ and $d_2 = (<score^2_1, \mu^2_{t_1}(d_2)>,...,<score^2_n, \mu^2_{t_n}(d_2)>)$.

Where $m$ and $n$ are the number of nodes owned by document fragments $d_1$ and $d_2$, $t_j$ represents the $j$-th node of document $i$. $score^j_i$ is the similarity score of node $t_j$, which is the score of the most
similar node between node $i_j t$ and the node from another document. $\mu_{i_j} (d_t)$ is the membership value of the node $i_j t$ to the document $d_t$. which is calculated by equation (3). The content similarity $SemSim(d_t, d_s)$ of $d_t$ and $d_s$ is as follows, and $SemSim(d_t, d_s) \in [0,1]$.

$$SemSim(d_t, d_s) = \left( \sum_{i=1}^{m} score_i^t \times \mu_{i_j} (d_t) + \sum_{j=1}^{n} score_j^s \times \mu_{i_j} (d_s) \right) / (m+n) \tag{5}$$

3.2.2. Calculation of Structural Similarity. Inspired by the linear processing mechanism in the pq-grams similarity algorithm, this paper adopts the structural similarity calculation method based on literature [9]. The method first extract the structure tree of the XML document, and then fill the trunk structure tree to make it become a full structure tree. Secondly, the full structure tree is transformed into a matrix and stored in the lower triangle matrix. The matrix dimension is determined by the number of leaf nodes in the full structure tree. The leaf nodes are stored in the diagonal of the matrix in turn. For intermediate nodes, they should be stored in the lower left corner of all their child nodes. Suppose $a$ is an intermediate node, and its formalization in the matrix is defined as $d(i,j)$. $i$ represents abscissa and $j$ represents ordinate. $i$ is determined by the maximum abscissa of all its child nodes, and $j$ is determined by the minimum ordinate of all its child nodes. For any node stored in the matrix, its value is $O[i,j](i=0,1,2...; j=0,1,2...)$:

$$O[i,j] = \begin{cases} 
1 & \text{If node } O \text{ belongs to trunk structure} \\
0 & \text{If node } O \text{ does not belong to trunk structure} 
\end{cases} \tag{6}$$

In this way, the matrix is transformed into a vector. Suppose the number of leaf nodes of the full structure tree is $m$, the vector dimension obtained by the lower triangle storage is $n = (m \times m) / 2$. Thus, two XML document trees are expressed as two $n$-dimensional vectors. Matrix similarity is eventually transformed into vector similarity, and the vector similarity calculation adopts vector correlation coefficient method. The formula is as follows:

$$StruSim(d_t, d_s) = r(x, y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\left( \sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y_i - \bar{y})^2 \right)^{1/2}} \tag{7}$$

3.3. Item Comprehensive Similarity and Generation Recommendation

After obtaining the similarity between item fuzzy attribute features, it is necessary to calculate the comprehensive similarity of the item. It is calculated by the weighted combination of fuzzy attribute similarity and traditional collaborative similarity.

Suppose there are two items $I_a$ and $I_b$, their comprehensive similarity is $Com\_sim(I_a, I_b)$:

$$Com\_sim(I_a, I_b) = \omega^\text{con} \cdot sim^\text{con}(I_a, I_b) + \omega^\text{rating} \cdot sim^\text{rating}(I_a, I_b) \tag{8}$$

where $sim^\text{con}$ is the similarity of the item on the fuzzy attribute feature, and $\omega^\text{con}$ is the weight of this similarity. $sim^\text{rating}$ is the collaborative similarity obtained by the user-item scoring matrix, which is calculated by equation (1), and $\omega^\text{rating}$ is the weight of collaborative similarity. The weight can be estimated and selected by experiments or the sparsity of scoring data, and $\omega^\text{con} + \omega^\text{rating} = 1$. 


4. Experimental Results and Analysis

4.1. Experimental Preparation
The experiment runs under Mahout [10], which is an open source project of Apache. It provides some implementations of classic algorithms for the scalable machine learning. MovieLens is a movie-based dataset created by the GroupLens project team at Minnesota University in the United States. It includes movie attribute matrix, user scoring matrix, and user-movie label matrix. In order to analyse the influence of the increased dataset on the algorithm, we randomly selected 10, 20, and 50 users. The scoring data of these users were composed into three datasets. Three datasets are respectively recorded as DS10, DS20, and DS50. The experimental data were further divided into training set and test set. 90% of the data sets were randomly selected as training sets, and the remaining 10% were used as test sets.

In order to evaluate the recommendation algorithm, MAE (mean absolute error), precision, and recall were used as the measurement criteria. MAE indicates the difference between the predicted score and the actual user score in all test sets and the smaller the better. Precision is the proportion of items that the user likes in the recommendation list. The recall is how many users' favorite items appear in the recommendation list in the test set. Whether precision or recall, the higher the value, the better.

4.2. Analysis of Experimental Results
Assuming that the algorithm proposed in this paper is XFD-CF, in order to verify the effectiveness of this method, we used the Item-CF (item-based collaborative filtering) and ItemKNN (item-based K-nearest neighbour collaborative filtering) as a comparison. Meanwhile, for each movie in the extracted datasets, five movie critics were asked to rate the category of the movie. The score value was between 0 and 1. The membership value of each movie category was obtained by using the membership equation (3) and inputted it into the database. The similarity of three recommended algorithms in the experiment all used the cosine similarity.

The experiment first determined the weight distribution in equation (4) and equation (8). The parameters $\gamma$ in equation (4) are used to adjust the weight of content similarity and structural similarity in item information. In this experiment, we believe that the content of the item is as important as the structure, therefore, the value of $\gamma$ is equal to 0.5. The parameters $\omega_{\text{con}}$ and $\omega_{\text{rating}}$ in equation (8) are used to adjust the weight of item information similarity and item collaborative similarity. We estimated them by the sparsity of the scoring data, and $\omega_{\text{con}} + \omega_{\text{rating}} = 1$. Table 1 shows the number of users, movies, scorings, the sparsity of three datasets, and the selection of parameters in three different datasets.

| Datasets | DS10 | DS20 | DS50 |
|----------|------|------|------|
| Users    | 10   | 20   | 50   |
| Movies   | 20   | 50   | 100  |
| Scorings | 106  | 317  | 954  |
| Sparsity | 0.47 | 0.683| 0.8092|
| $\omega_{\text{con}}$ | 0.5 | 0.7 | 0.8 |
| $\omega_{\text{rating}}$ | 0.5 | 0.3 | 0.2 |

In order to verify the correctness of parameter selection, we performed experiments in different datasets, and only the experimental results in the first dataset DS10 are given here. The value of $\omega_{\text{con}}$ starts from 0, and increases by 0.25 until it increases to 1. Figure 1 shows the precision of XFD-CF for different parameters $\omega_{\text{con}}$ in the DS10 dataset.

Figure 1 shows that that when $\omega_{\text{con}}$ equals 0.5; the precision of XFD-CF is the highest. Therefore, in the subsequent experiments of XFD-CF, the parameters were set to the three groups shown in table.
Recommendation Accuracy. Firstly, in order to verify that the proposed algorithm can effectively improve the accuracy of the recommendation system, according to the weight parameters obtained from above experiments, the recommendation accuracy of three recommended algorithms was compared in the selected different datasets. Figure 2 and figure 3 show the comparison results of the precision and recall of the three algorithms in different datasets. Table 2 shows the results of the above two figures.

Table 2. The result of recommendation accuracy.

|        | Item-CF | ItemKNN | XFD-CF |
|--------|---------|---------|--------|
| Precision | Recall | Precision | Recall | Precision | Recall |
| DS10   | 0.5     | 0.625   | 0.375  | 0.722     | 0.75   |
| DS20   | 0.088   | 0.088   | 0.324  | 0.417     | 0.453  | 0.55   |
| DS50   | 0.011   | 0.011   | 0.178  | 0.256     | 0.453  | 0.453  |

As shown in table 2, compared with Item-CF and ItemKNN, XFD-CF has higher precision and recall in each dataset. At the same time, the increase of dataset samples has the greatest impact on the performance of Item-CF, that is, as the number of samples increases, the precision and recall of Item-CF decrease the most. ItemKNN takes the second place. And the impact on XFD-CF is minimal.
Therefore, compared with the traditional collaborative filtering recommendation algorithm, the collaborative filtering recommendation algorithm based on XML fuzzy data effectively improves the quality and accuracy of the recommendation algorithm.

4.2.2. Cold-Start Problem. Since the cold-start problem of new items is that there is no scoring and label information for new items appearing in the database, it is necessary to consider the influence of the user scoring information on the recommendation algorithm. First, we took the movie in the test set as the new item, and the scoring of the new item corresponding to the training set was set to zero. Then, three recommendation algorithms were run respectively in the dataset before processing and the dataset after processing to compare their recommendation effects. Figure 4 records the MAE of three recommendation algorithms in the DS50 dataset. Table 3 shows all MAE values on three datasets.

| Table 3. MAE comparison of several algorithms before and after three datasets processing. |
|---------------------------------------------|---------------------------------------------|---------------------------------------------|
| Item-CF | ItemKNN | XFD-CF |
| MAE(before) | MAE(after) | MAE(before) | MAE(after) | MAE(before) | MAE(after) |
| DS10 | 0.51 | 0.645 | 0.695 | 0.75 | 0.453 | 0.43 |
| DS20 | 0.666 | 0.712 | 0.708 | 0.731 | 0.523 | 0.53 |
| DS50 | 0.774 | 0.82 | 0.757 | 0.783 | 0.613 | 0.63 |

From table 3, in different datasets, regardless of whether the dataset is processed or not, the MAE value of XFD-CF is significantly lower than that of Item-CF and ItemKNN. Moreover, after data processing, the MAE value of ItemKNN grows fastest, followed by Item-CF, whereas the MAE value of XFD-CF is almost unchanged. Compared with the other two algorithms, XFD-CF can still recommend items in the similarity of fuzzy attribute features when there is no scoring information. Therefore, XFD-CF effectively alleviates the cold-start problem faced by the collaborative filtering recommendation algorithm and improves the recommendation quality.

5. Conclusion and Future Work
Collaborative filtering algorithm is a hot topic in the application of recommendation system, but it is always faced with the problem of cold-start. Meanwhile, the traditional recommendation algorithm is unfavorable to the description of item fuzzy information. Therefore, this paper proposes a collaborative filtering recommendation algorithm based on XML fuzzy data. The method deeply analysed the problem of the fuzziness of item attributes when calculating the nearest neighbor items, then calculated the similarity between fuzzy attribute features, and combined this similarity with the similarity obtained by the collaborative filtering to complete the recommendation. Experimental results show that the processing of item fuzzy attributes can improve the accuracy of the recommendation system. At the same time, considering the similarity between the fuzzy attributes of the item, the cold-start problem and the recommendation quality can be effectively improved.

It can be seen that although the proposed algorithm improves the recommendation quality, the time complexity is not low. Henceforth, we will study how to reduce the complexity and add various social relations among users into the algorithm to further improve the effectiveness of recommendation.

6. Acknowledgments
Authors would like to acknowledge the funding of 2017 Liaoning Natural Science Foundation (Grant no. 20170540680).

7. References
[1] Xu X L and Xu G L 2016 Improved Collaborative Filtering Recommendation Based on Classification and User Trust Journal of Electronic Science and Technology 14 25-31
[2] Liu Q W, Xiong Y and Huang W C 2014 Combining User-Based and Item-Based Models for Collaborative Filtering Using Stacked Regression Chinese Journal of Electronics 23 712-17
[3] Meng X W, Liu S D, Zhang Y J and Hu X 2015 Research on Social Recommender Systems Journal of software 26 1356-27
[4] Zhang C J and Zeng A 2012 Behavior patterns of online users and the effect on information filtering Physica A: Statistical Mechanics & Its Applications 391 1822-30
[5] Yu H and Li J H 2015 Algorithm to Solve the Cold-Start Problem in New Item Recommendations Journal of software 26 1395-408
[6] Zadeh L A 1965 Fuzzy sets Information and Control 8 338-53
[7] Pu Y, Jiang H and Wang X K 2016 Tensor-based approach to XML similarity calculation Control and Decision 31 1711-14
[8] Wu Y Y, Lei Q, Chen D S and Yokota H 2008 A Method of Discovering Relation Information from XML Data Journal of software 19 1422-27
[9] Wu H T, Guo L H and Yang J 2018 XML similarity detection algorithm based on matrix storage Application Research of Computers 35 2025-29
[10] Owen S, Anil R, Dunning T and Friedman E 2011 Mahout in Action (New York: Manning Publications)