Atlantis Highlights in Intelligent Systems (AHIS), volume 1
3rd International Conference on Control, Automation and Artificial Intelligence (CAAI 2018)

Improving Collaborative Filtering Recommendation

Pang-Ming Chu[1], Hong-Ru Tsai[1], Shie-Jue Lee[2,*] and Shing-Tai Pan[3]
[1] Electrical Engineering National Sun Yat-Sen University, Kaohsiung 804, Taiwan
[2]Electrical Engineering, Intelligent Electronic Commerce Research Center National Sun Yat-Sen University, Kaohsiung 804, Taiwan
[3]Computer Science and Information Engineering National Sun Yat-Sen University, Kaohsiung 804, Taiwan
[*]Corresponding author

***Abstract*—Collaborative filtering recommender systems traditionally recommend products to users solely based on the user-item rating matrix and are simple, convenient to use. In this paper, we focus on two main issues, data sparsity and scalability. Data sparsity can lead to inaccurate recommendations, while scalability may cause an unacceptably long delay before valuable recommendations are acquired. We propose a novel approach to deal with these two issues. Word2Vec is employed to build item vectors from the user comments. Through the user-item rating matrix, user vectors of all the users are then obtained. A clustering technique is applied to reduce the time complexity related to the large numbers of items and users. Experimental results of real data sets are shown to demonstrate the effectiveness of our proposed approach.**

Keywords—data sparsity, scalability, Word2Vec, self-constructing clustering, word vectors

I. INTRODUCTION

Recommender systems [1], [2] are able to analyze the past behavior of customers and predict the products they might be interested in. Recommender systems can roughly categorized into two types, collaborative filtering and content-based filtering. Content-based filtering [3] assumes that customers will buy things that are similar to what they have bought in the past. Detailed information about products and users are therefore required for recommendations. However, the information needed is either enormous or hard to get, and the information about products are mostly provided by the manufacturers or related vendors, which could be biased. Collaborative filtering [4], [5], on the other hand, assumes that similar users will have similar interest to items and similar items will have similar ratings by users. It makes recommendations by analyzing the previous interaction information from users.

Allahbakhsh and Ignjatovic [1] propose an iterative method which regards all the users as one user. The ratings by individual users are integrated together according to each user's credibility which is iteratively updated. This can reduce the sparsity difficulty. However, it cannot give personalized recommendations to users; all the users have the same recommendations. Park [4] proposes a method which identifies tail items, having only a few ratings, from head items, having an enough amount of ratings, according to their popularities. The recommendations for tail items are based on the ratings of clustered groups, while the recommendations for head items are based on the ratings of individual items or groups clustered to a lesser extent. However, this method tends to recommend tail items as a whole to users. Zhang et al. [5] introduce the concepts of popular items and frequent raters. They assume that the ratings match some probability model. To overcome the data sparsity and rating diversity, smoothing and fusion techniques are employed. However, the method may encounter difficulties with a user-item rating matrix having a high-degree of sparsity.

In this paper, we propose a novel approach to deal with data sparsity and scalability. First of all, we apply Word2Vec [6], which is a word semantic tool developed by Google, to analyze the user comments on their previously bought goods. Each word is given a unique vector which represents its semantics. A set of item vectors, one item vector for each product, are then developed. Through the user-item rating matrix, user vectors of all the users are obtained. A self-constructing clustering algorithm is applied to reduce the time complexity related to the large numbers of items and users. Recommendation work is then done with the resulting clusters. Finally, reverse transformation is performed and a ranked list of recommended items is offered to each user.

The rest part of this paper is organized as follows. Our proposed approach is described in detail in Section II. Experimental results are presented in Section III. Finally, a conclusion is given in Section IV.

II. PROPOSED APPROACH

Suppose there are a set of N users, u_i,$1 \le i \le N$, and a set of M products, p_i ,$1 \le j \le M$. A user u_i may express his/her evaluation to a product p_i by providing a rating r_{ij}, a positive integer, for p_i. Usually, a higher rating is assumed to indicate a more favorable feedback from the user. If user u_i has not provided a rating for product p_i, $r_{ij} = 0$. Such information can be represented by the following user-product rating matrix **R**:

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1M} \\ r_{21} & r_{22} & \cdots & r_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ r_{N1} & r_{N2} & \cdots & r_{NM} \end{bmatrix} \quad (1)$$

which is a N × M matrix. The goal of a collaborative filtering recommender system is, based on a given rating matrix, to recommend a predicted preference list of the products to each user.

We propose a novel approach to overcome the inaccuracy and inefficiency caused by data sparsity and scalability. We aim

to overcome the sparsity problem by exploiting extra information from online user comments. Word2Vec is applied on user comments to extract semantic relationships between products. Each product is converted to an item vector. Then, through the given user-item rating matrix, user vectors, one vector for each user, are derived. Next, we aim to deal with the scalability problem. Principal component analysis (PCA) [7] is applied to reduce the dimensionality of item and user vectors. Then a self-constructing clustering algorithm [8] is adopted to cluster items and users into item groups and user groups, respectively. A reduced user-item rating matrix which involves item groups and user groups is obtained.

A. Developing Item and User Groups

Word2Vec was proposed by Mikolov et al. [6] and released by Google in 2013. The main purpose of Word2Vec is to project the words contained in a set of training documents to a semantic space of specified dimensionality. Through a neural network learning process, each word is assigned a unique word vector in the semantic space. The resulting word vectors reflect semantic relationships among the words.

Training patterns, each consisting of an input-output word pair (*x, y*), are extracted from the training documents which are divided into a sequence of non-overlapping windows. For each window, the central word is taken as input *x* and each of its context words is taken as output *y* to form training patterns. After extracting the training patterns from the user comments, the training patterns are fed into the Word2Vec network for training. At the end of learning, the input weights are extracted from the network to form item vectors. Then user vectors, $\boldsymbol{u}_k$, $1 \le k \le N$, are derived.

The main idea of PCA [7] is to reduce the dimensionality of a data set consisting of many variables correlated with each other while retaining the variation present in the dataset up to the maximum extent. This is done by transforming the original variables to a new set of variables, which are known as the principal components and are orthogonal, ordered such that the retention of variation present in the original variables decreases. So, the first principal component retains maximum variation that was present in the original variables, and the last principal component retains minimum variation that was present in the original variables.

Suppose $\mathbf{X} = \{\boldsymbol{x}_i \mid 1 \le i \le N\}$ is a finite set of N unlabeled training instances, where $\boldsymbol{x}_i = [x_{i,1}\ x_{i,2}\ \ldots\ x_{i,n}]\ \mathrm{T} \in \mathbb{R}^n$ is the *i*th instance. Each instance is a vector with *n* features. SCC [8] does clustering in a progressive way. Only one training cycle on the instances is performed. Each cluster is described by a Gaussian-like membership function characterized by the center and deviation induced from the data assigned to the cluster. The instances are considered one by one sequentially, and clusters are created incrementally.

Let *J* denote the number of currently existing clusters. Initially, no clusters exist and thus $J = 0$. When instance 1 comes in, the first cluster, C_1, is created and instance is assigned to it and $J = 1$. Then, for instance *i*, which is $\boldsymbol{x}_i$, $i \ge 2$, the *z*-distance between instance *i* and every existing cluster, C_j, is calculated by

$$Z(i,j) = \sum_{k=1}^{n} \left(\frac{x_{i,k} - c_{j,k}}{\sigma_{j,k}} \right)^2 \tag{2}$$

for $j = 1, 2, \ldots, J$. Note that $\boldsymbol{c}_j = [c_{j,1}\ c_{j,2}\ \ldots\ c_{j,n}]^T$ and $\boldsymbol{\sigma}_j = [\sigma_{j,1}\ \sigma_{j,2}\ \ldots\ \sigma_{j,n}]^T$ denote the mean and deviation, respectively, induced from the instances assigned to cluster C_j. The membership degree (MD) of instance *i* belonging to cluster C_j is defined as

$$\mu_j(\mathbf{X}_i) = exp\{-Z(i,j)\} \tag{3}$$

with the value lying in the range of [0, 1]. Let S_j denote the number of instances assigned to cluster C_j. There are two cases:

- If all the MD's are less than a threshold, ρ^n, where ρ is a pre-specified threshold in each dimension, i.e.,

$$exp\{-Z(i,j)\} < \rho^n \tag{4}$$

  for all existing clusters C_j, $1 \le j \le J$, a new cluster, C_{J+1}, is created and instance *i* is assigned to it.

- Otherwise, instance *i* is assigned to the cluster with the largest MD, say cluster C_a, and the characteristics of C_a are updated. Note that J is not changed in the second case.

When all the instances have been considered, SCC stops with *J* clusters.

We apply SCC to the item vectors to form item groups. We feed the item vectors, $\mathbf{p}_i$, $1 \le i \le M$, as the training patterns, to SCC. Then a $M \times M^g$ matrix $\mathbf{T}^p$ is formed, representing the membership degrees of the products p_i, $1 \le i \le M$, to the item groups p_i^g, $1 \le i \le M^g$. Next, we apply SCC to the user vectors to form user groups. We feed the the user vectors, $\mathbf{u}_i$, $1 \le i \le N$, as the training patterns, to SCC. Then a $N \times N^g$ matrix $\mathbf{T}^u$ is formed, representing the membership degrees of the users u_i, $1 \le i \le N$, to the user groups u_i^g, $1 \le i \le N^g$.

B. Getting Recommendations

We transform the original user-item rating matrix $\mathbf{R}$ to a reduced form $\mathbf{R}^g$. $\mathbf{R}$ is a $N \times M$ matrix, while $\mathbf{R}^g$ is a $\mathrm{N}^g \times \mathrm{M}^g$ matrix.

To do this, we first convert $\mathbf{R}$ to the reduced user-item rating matrix $\mathbf{R}^g$. Next, a preference list of item groups is derived for each user group. A correlation graph is built from the reduced user-item rating matrix $\mathbf{R}^g$. Then a series of random walks based on ItemRank are performed. For any user group u_i^g, $1 \le i \le \mathrm{N}^g$ we obtain $\mathbf{h}_i$ which is the predicted preference list of product groups for the user group u_i^g. Then we do reverse transformation to get predicted preference lists of products for each user. Firstly, we find a preference list of products, y_i[j], for user group u_i^g. Finally, we compute vector s_i for $1 \le i \le N$. Therefore, we end up with predicted preference lists of products, $s_1, s_2, \ldots, s_N$, for users $u_1, u_2, \ldots, u_N$, respectively.

III. Experimental Results

To evaluate the performance of our proposed approach, we conduct a set of experiments on several benchmark data sets. We also compare our approach with some other collaborative filtering recommender systems. Two metrics are adopted for comparison on recommendation accuracy, mean absolute error (MAE) and root mean squared error (RMSE).

Here we experiment with a real-world data set to show the effectiveness of our proposed approach. This dataset was downloaded from Amazon [9]. The number of users is 5541, the number of products is 3568, the number of ratings is 64706, and the sparsity of this set is 99.7%. Note that the data set is heavily sparse. Its sparsity is $1 - \frac{64706}{5541 \times 3568} = 99.7\%$.Comparisons on MAE, RMSE, and Time(s) among our ap- proach, SCC [10], and ICRRS [1] are:

- MAE. Our approach gets 0.715 in MAE, while SCC gets 0.837, and ICRRS gets 0.987. Clearly, our approach is the best, having the smallest value in MAE in this case.
- RMSE. Our approach gets 1.000 in RMSE, while SCC gets 1.171, and ICRRS gets 1.393. Clearly, our approach is the best, having the smallest value in RMSE in this case.
- Time(s). Our approach takes 51.9 seconds to get the results, while SCC takes 945.3 seconds, and ICRRS takes 358.5 seconds. Clearly, our approach is runs most efficiently, consuming the least amount of time in this case.

Note that for MAE, RMSE, and Time, the smaller the value a method obtains, the better the method performs.

IV. Conclusion

We have presented a novel approach to deal with these two issues. Word2Vec is employed to build item vectors from the comments made by users on their previously bought goods. Through the user-item rating matrix, user vectors of all the users are then obtained. A clustering technique is applied to reduce the time complexity related to the large numbers of items and users. With the proposed approach, the inaccuracy caused by the sparse ratings is overcome and the processing time for making recommendations is much reduced.

Acknowledgment

This work was supported in part by the NSYSU-NUK Joint Research Project #NSYSUNUK-107-P006, by the "Intelligent Electronic Commerce Research Center" from The Featured Areas Research Center Program within the framework of the Higher Education Sprout Project by the Ministry of Education (MOE) in Taiwan, and by grant B106002, STSP AI Robot Project, MOST, Taiwan.

References

[1] M. Allahbakhsh and A. Ignjatovic. An iterative method for calculating robust rating scores. *IEEE Transections on Parallel and Distributed Systems*, 26(2):340-350, 2015.

[2] S. Zahra, M. Ghazanfar, A. Khalid, M. Azam, U. Naeem and A. bennett. Novel centroid selection approaches for KMeans-clustering based recommender systems. *Information sciences*, 320:156-189, 2015.

[3] S.Debnath, N.Ganguly and P.Mitra.Feature weighting in content based recommendation system using social network analysis. *Proceedings of the 17th international conference on World Wide Web*, ACM, pp. 1041- 1042, 2018.

[4] Y-J. Park, The adaptive clustering method for the long tail problem of recommender systems, *IEEE Transections on knowledge and data engineering*, 25(8):1904-1915, 2013.

[5] D. Zhang, C-H. Hsu, M. Chen, Q. Chen,N. Xiong and J. Lloret, Cold- start recommendation using bi-clustering and fusion for large-scale social recommender systems, *IEEE Transections on Emerging Topics in Computing*, 2(2):239-250, 2014.

[6] T.Mikolov, I.Sutskever, K.Chen, G.S.Corrado and J.Dean. Distributed representations of words and phrases and their compositionality. *Proceedings of the 26th International Conference on Neural Information Processing Systems*, Vol. 2, pp. 3111-3119, 2013.

[7] S. Wold, K. Esbensen and P. Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37-52, 1987.

[8] S.-J. Lee and C.-S. Ouyang. A neuro-fuzzy system modeling with self-constructing rule generation and hybrid SVD-based learning. *IEEE Transactions on Fuzzy Systems*, 11(3):341–353, 2003.

[9] M. Wan, M. and J. McAuley. Modeling ambiguity, subjectivity, and diverging viewpoints in opinion question answering systems. *Proceedings of the 2016 IEEE 16th international conference on Data Mining (ICDM)*, IEEE, pp. 489-498, 2016.

[10] C.-L. and S.-L. Lee. A clustering based approach to improving the efficiency of collaborative filtering recommendation. em Electronic Commerce Research and Applications, 18:1-9, 2016.