A Naïve Recommendation Model for Large Databases

Hosein Jafarkarimi1, Alex Tze Hiang Sim, and Robab Saadatdoost

***Abstract*—It is difficult for users to find items as the number of choices increase and they become overwhelmed with high volume of data. In order to avoid them from bewilderment, a recommender could be applied to find more related items in shorter time. In this paper, we proposed a naive recommender model which uses Association Rules Mining technique to generate two item sets enabling to find all existing rules for a certain item and has the capability to search on demand which decrease the response time dramatically This model mines transactions' database to discover the existing rules among items and stores them in a sparse matrix. It also searches the matrix by means of a naive algorithm to generate a search list. We have applied and evaluated our model in Universiti Teknologi Malaysia and the results reflect a high level of accuracy.**

***Index Terms*—Data mining, recommendation model, association rule mining.**

I. Introduction

As the number of items in markets increases, the quantity of available choices also increases. Consequently, users might face loads of irrelevant data when looking for an item. This problem could be felt when users face a large number of goods which can confuse them to choose what they are looking for [8]. Moreover, this strenuous toil would not always provide the users with the best available choice. In this case a recommendation model can be applied in order to help users to find what they are looking for faster and easier.

A recommendation model provides information on services, movies, music, news, tools, books and web pages for users. The model analyzes the existing data to generate the recommendation list. This study can be done on previous user requests or previous related searches in an existing database. In fact the recommendation model searches the database, finds data relationships and presents it to users. As a result, users could find what they are looking for, in a shorter time and with a noticeable higher precision.

Different classifications have been proposed for recommendation models. One idea is to classify them into two main categories which are content-based and preference-based [7] [3]. Some researcher also mentioned other categories like cluster models and search-based method [9], web mining-based [10], social networking [14], demographic [12]. Most of recommender systems use collaborative filtering (CF) [11] [9] and association rule mining (ARM) [4] [14].

In this paper we proposed our model in details. We applied Association Rules Mining (ARM) to discover existing relations in the dataset. We used library circulation data of Universiti Teknologi Malaysia (UTM) between years 2007 and 2009 to evaluate our model for large databases.

We have presented our studies as follows: discussing about basic concepts of ARM, followed by our proposed model discussed in detail, and finally, the implementation and evaluation of our model.

H. Jafarkarimi is with Department of Computer, Damavand Branch, Islamic Azad University, Damavand, Iran. (email:hoseinkarimi@gmail.com)

A. T. H. Sim and R. Saadatdoost are with Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia, Malaysia.

II. Association Rule Mining

ARM is a technique in data mining which was introduced by Agrawal et al. [1] in 1993. They proposed the famous Apriori algorithm in 1994 [2]. The main idea in ARM is to find relation among different items in databases; for example, items that are bought together in a supermarket or books which are borrowed together in a library's circulation database. ARM helps to extract these kinds of patterns from database.

An association rule is representing using (A => B). In this rule, we name 'A' as head and 'B' as body. It shows that item A has a relation with item B and depends on the substance. For example, assuming A is cheese and B is bread in a supermarket's transactions database. It can be claimed that, cheese is bought together with bread in that database. ARM depends on two user-specified parameters which are minimum support and minimum confidence. These two parameters show the strength of a rule, in the dataset. Support (Supp) is the probability of an item in a dataset which can be defined as bellow:

$$\text{Supp (A)} = \frac{Quantity\ \ of\ item\ set\ (A)}{Total\ number\ \ of\ transactions\ \ in\ a\ database} \quad (1)$$

For example, in a supermarket, we define item set (A) as "the number of transactions including cheese". If this event of "cheese" occurs 1000 times in a database that consists of 100000 transactions, the support is calculated to be:

$$\text{Supp (Cheese)} = \frac{Quantity\ \ of\ (Cheese\,)}{Total\ amount\ \ of\ Transactions} = \frac{1000}{100000} = 0.01 \quad (2)$$

Confidence (Conf) of association rule (A => B) is the probability of occurring B in a transaction which also includes A. bellow is the way that Conf can be calculated:

$$\text{Conf (A=>B)} = \frac{Probability\ \ (A \cap B\,)}{Probability\ \ (A)} \quad (3)$$

As an example, if we assume P (cheese) = 0.03
and $P\ (Cheese\ \cap Bread\ ) = 0.01$, then the confidence of (cheese => bread) will be calculated below:

$$\text{Conf (Cheese=>Bread)} = \frac{P\ (Cheese\ \cap Bread\ )}{P(Cheese\,)} = \frac{0.01}{0.03} = 0.33 \quad (4)$$

This means in 33% of transactions in the supermarket,

cheese has occurred together with bread.

An item set like (cheese, bread) that passes the minimum support, is called a frequent item set, and if an association rule like (cheese => bread) passes the minimum confidence, it is called a strong association rule. ARM is easy to implement and has a great ability to hide sensitive data [14]. A good algorithm for mining association rule is FPGrowth (frequent pattern growth) which is relatively a new technique [5]. This algorithm implements an extended prefix-tree (FP-tree) to store dataset in a smaller data structure with which there is no need to scan the database repeatedly. It uses a divide and conquers approach to do the mining task and dramatically decreases the search space [6]. We used FP-Growth for mining in this project.

III. Proposed Recommendation Model

In our proposed recommender model, there are four main steps to recommend items to patrons. The first step is to pre-process the available data in database. In the second step, we used FP-Growth to perform association rules mining in order to discover patterns among different books. Thirdly, we proposed to store those extracted rules and their confidence values into a sparse matrix. And in our final step, we search the matrix and find the relation in order to make the recommendation list.

A sparse matrix of transactions' dataset is made in step 3. Each row denotes a transaction and each column represents a unique item. Whenever an item exists, the value for column and row will be greater than zero.

The number of rules that will be generated with FPGrowth technique is depending on the number of items in the database and a set minimum support value. The number increases exponentially in proportion to the number of items in a database. For a constraint of *k* items in a rule with *n* number of items, the total number of generated rules can be calculated using equation below:

$$\text{Total (permutation)} = 1 * \binom{n}{2} + 2 * \binom{n}{3} + 3 * \binom{n}{4} + \cdots + k - 1 * \binom{n}{k} = \sum_{i=2}^{k} (i-1) * \binom{n}{i} \quad (5)$$

In case of 5 items, the total number of permutations is 980 as shown below:

$$T(\text{maximum 5 items in a rule}) = 1 * \binom{5}{2} + 2 * \binom{5}{3} + 3 * \binom{5}{4} + 4 * \binom{5}{5} = 980 \quad (6)$$

This means, there are 980 different rules / combinations based on items A, B, C, D, E that could be borrowed together. Among these rules, a large number of them are lengthy in size, for examples A, B → C, D, E and B, C → A, D, E. Since the search space is exponentially large, it is often impractical to generate and analyse all longer rules for making a recommendation. We argue that shorter rules are good enough for making a recommendation under the constraint of the storage size and search time.

In this research, we recommend an approach to focus on pair-wise items that could be associated together. Based on this model, we are able to build a list of related and recommended items. Despite that our model only searches for two item sets, it is useful in many scenario. For example, in a library, a user searches for a book, instead of a bulk of books. They typically start by searching a title whence other related items. In contrast, typical ARM searches associations on groups of items – it starts from two item sets and continues for more items to form (a group of) head and body of a rule. Not only that it takes much time to find all these rules, the results are confusing. Our model is fast and easy to implement and does not need complicated infrastructure and heavy computational resources. Based on our model, mining are performed on pairs of two items, and all the discovered rules are stored into a simple sparse matrix to depict items that are related to one another. In fact, we could utilise mining algorithm such as FP-growth to generate the confidence values. For N number of items, we build a (N x N) matrix. An example of a (7 x 7) matrix which contains seven items is shown below:

	A	B	C	D	E	F	G
A		1			0.97		
B	1		0.92			0.80	
C				1	0.93		
D			1				
E	0.85		0.82				
F							0.94
G						0.89	

Fig. 1. A sample sparse matrix to store discovered rules.

For item A (on the left), there is a relation with item B and item E. Each cell in this matrix stores the confidence value of a rule. For example, the rule "A→E" comes with 97% in confidence. Although the matrix could be huge, it is also sparse. Our matrix is similar to the matrix used in collaborative filtering; however, it stores values of confidence instead of vectors of user interests.

It has to be said that this matrix is appropriate for pair wise rules and cannot cover more than that. In order to find all recommendation items in related to an item, we performed a depth search on the matrix to build a hierarchical tree. For example, to make the recommendation list for a user who is searching for item A, the following hierarchical tree is generated based on our previous sparse matrix:

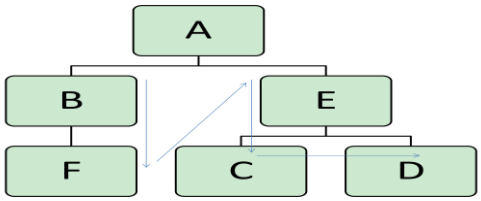

Fig. 2. A sample search generated for rules matrix presented in Fig. 1.

As shown in Fig. 2, to make a list for item A, our search algorithm takes item A as head and finds possible body among item A to item G. In this case, item A is associated with two items – B and E with confidence of 100% and 97% respectively. Item B is first detected and reported. And, based on the latter, it finds the association with item F recursively. A similar process is repeated on item E. The entire process repeats itself until no item is then associated to items that are heads in a rule. Below shows our algorithm to generate a

hierarchical tree:

```
For each head
    While candidate body exists
        For every candidate child
            If conf(body, head) > 0
Link(head, body);
    head= body;
        else
                break;
```

Our search algorithm is based on a confidence value which does not inherit anti-monotone properties. Typically, if "A→B" and "B→C" are reported, it is understood that "A→C" may not be true. It is otherwise true based on *support* value. In this case, we successfully report the association rules, A→B, A→E, B→F, B→C, E→F, E→C and C→D.

IV. Evaluation

In order to evaluate our model, we have tested our model on a library circulation database of Universiti Teknologi Malaysia (UTM) during year 2007- 2009.

A. Dataset

Due to privacy matters, UTM library released a data which consists of only the book IDs, date of transaction and department name of a student to us. Without a patron ID, we could not identify the unique transactions from an individual. In this experiment, we then formulated a combination key that comprises of a date and a department ID for unique transaction. We assumed that items borrowed on the day from the same department are borrowed together (by an individual or a group of patrons).

B. Experiment Results

There are more than ten thousands of items in our dataset. We utilised FP Growth to help pruning at a low minimum support at 0.08. And we managed to build recommendation lists for various items. Showing below is a sample recommendation list for a book named "Mechanical behavior of rapidly solidified materials".

TABLE I: An Example of Recommendation List

Title	Call number
Mechanical behavior of rapidly solidified materials solidified materials	TS247.M42 1986
Jigs,fixtures & gages	TJ1187.J56 1986
Pressure diecasting	TS239.U68 1982j1n1
Fundamentals of foundry technology	TS230.F86 1980
Tool design	TJ1186 P58 1988
Modern vacuum physics	QC166C424 2005

As shown in the above table, the first book is the root and the rest of books are generated by our depth search in a sequence. The call number shows the place of the book in the library shelves – the two characters (e.g. TS, TJ) represent a shelf in the library. And the rest of digits denotes the exact location of the book. Due to the fact that call numbers are based on relevant subjects, we can judge if these books are related. In this case, the first letters of all books (except one) in our recommendation list are the same (i.e. 'T'). This shows that the recommendation is relevant. In fact, the titles also suggest that they are related. We investigate the last item which has a different call number (i.e. 'Q'), and concluded that the book is relevant but was moved to a new shelf.

In order to evaluate the quality of generated recommendation lists, we used a questionnaire [13] made from 40 different lists of recommendations and presented them to 20 lecturers from different faculties and 5 librarians. The recommendation lists was provided to lecturers based on the field of their major. On average, their recommendation scored 89.4% which is reasonably good in view of our assumption on the dataset (see section 4.1).

V. Conclusion

In this research, we proposed a simple approach utilising data mining to make a recommendation list. Our approach was based on pairs of items which is faster than typical ARM. Experimental results show that we derived reasonably good results on a large dataset.

Acknowledgements

We would like to thank officers from Universiti Teknologi Malaysia Library (PSZ) for their help and kind cooperation. This research is funded by Research University Grant (vot.7128) from Universiti Teknologi Malaysia and Damavand Branch of Islamic Azad University of Iran.

References

[1] R. Agrawal, T. Imielinski, and A. Swami, "Mining association rules between sets of items in large databases Proceedings of the 1993 ACM SIGMOD international conference on Management of data, ACM, "Washington, D.C., United States, pp. 207-216, 1993.

[2] R. Agrawal, and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases Proceedings of the 20th International Conference on Very Large Data Bases," Morgan Kaufmann Publishers Inc, pp. 487-499, 1994.

[3] M. Balabanović, \#263 and Shoham, Y. Fab: content-based, collaborative recommendation. Commun. ACM, vol. 40, no. 3.pp. 66-72.

[4] K. Choonho, and K. Juntae, "A recommendation algorithm using multi-level association rules," in Web Intelligence, in *Proc. IEEE/WIC International Conference on*, 524-527, 2003.

[5] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," SIGMOD Rec, vol. 29, no. 2, pp. 1- 12.

[6] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent- Pattern Tree Approach. Data Mining and Knowledge Discovery," vol. 8, no. 1, pp. 53-87.

[7] C H. Ming, W. L. Chuan, and C. C. P, "A Study on the Comparison between Content-Based and Preference-Based Recommendation Systems," *In Semantics, Knowledge and Grid*, 2008. SKG '08. Fourth International Conference on, 477-480,2008.

[8] W. J. Hui, L. Qiang, and S. L.Wen, "Clustering Technology Application in e-Commerce Recommendation System. in Management of e- Commerce and e-Government," 2008. ICMECG '08. *International Conference on*, 200-203, 2008.

[9] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering. Internet Computing," *IEEE*, vol. 7, no.1, pp.76-80.

[10] H. Longjun, D. Liping, W. Yuanwang, and H. Minghe, "A Personalized Recommendation System Based on Multi-agent," *in Genetic and Evolutionary Computing*, WGEC '08. Second International Conference on, pp. 223-226, 2008.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, GroupLens: an open architecture for collaborative filtering of netnews

Proceedings of the 1994 ACM conference on Computer supported cooperative work, ACM, Chapel Hill, North Carolina, United States, pp.175-186, 1994.

[12] J. Sobecki, and K. Piwowar, "Cofmparison of Different Recommendation Methods for an e-Commerce Application. in Intelligent Information and Database Systems," ACIIDS 2009. First Asian Conference on, pp. 127-131,2009.

[13] K. Sugiyama and M. Y. Kan, "Scholarly paper recommendation via user's recent research interests Proceedings of the 10th annual joint conference on Digital libraries," ACM, Gold Coast, Queensland, Australia, pp. 29-38,2001.

[14] L.Yongcheng, L. Jiajin, and C. Huilan, "A Privacy- Preserving Book Recommendation Model Based on Multi-agent. in Computer Science and Engineering," 2009. WCSE '09. Second International Workshop on, pp. 323-327, 2009.