# Effective Hybrid Recommendation Combining Users-Searches Correlations Using Tensors

Rakesh Rawat, Richi Nayak, Yuefeng Li
Faculty of Science and Technology, Queensland University of University
Brisbane Australia
r.rawat,r.nayak,y2.li{@qut.edu.au}

**Abstract.** Most recommendation methods employ item-item similarity measures or use ratings data to generate recommendations. These methods use traditional two dimensional models to find inter relationships between alike users and products. This paper proposes a novel recommendation method using the multi-dimensional model, tensor, to group similar users based on common search behaviour, and then finding associations within such groups for making effective inter group recommendations. Web log data is multi-dimensional data. Unlike vector based methods, tensors have the ability to highly correlate and find latent relationships between such similar instances, consisting of users and searches. Non redundant rules from such associations of user-searches are then used for making recommendations to the users.

**Keywords:** Tensor, clustering, association rule mining, web log data, recommendation.

## 1 Introduction

With the popularity of World Wide Web, use of recommenders to suggest relevant products and services to online users is gaining momentum. Collaborative filtering (CF) techniques of recommendation have been used by many websites like Amazon, ebay, CdNow, Netflix, Yahoo Answers and many more. CF techniques can be grouped into two general classes, namely the neighbourhood and model-based methods [1]. Neighbourhood (or memory-based or heuristic-based) methods use Item-to-Item or User-to-Item correlations to find the nearest neighbours and then subsequently use this information to make recommendations. The Item-to-Item correlation methods adopt a content-based approach where knowledge about the products (contents) is used for recommendation and only similar matching content/products are recommended [2], [4]. The User-to-Item correlation methods combine interests of a group of people to find the highest rated interests and then interests consisting of items/ products/people are recommended to the individuals in a group [4],[14].

A user's search generally consists of multiple attributes e.g. in the case of a car website, a user may search for a particular make, model, body type, cost, new or old car type. A user may have made *n* number of searches within a website. Modeling or comparing such users-items data having multi-dimensional properties is a complex process. Traditional CF methods employed for finding similarity between users-items ignore this multi-dimensional nature of search log data and are unable to recommend unique items to different users [2]. These methods consider an item as an object, whereas, the item may be a combination of many features, represented as a vector itself. Finding the latent relationships between user's searches and item's features is often ignored by two dimensional data models such as vectors and matrix. Recommendation systems need to handle very high dimensional profiles of users-items, in order to find the correlations between users and items. A noteworthy consideration as also discussed by [3], is that distance measures used for clustering or comparisons may reflect strange properties in high dimensional space and might not be as useful as they seem.

In this paper, we propose a novel recommendation method which utilizes the implicit information about users by using the search log data. This methodology utilizes the search log data to infer the user rating about items. This is a collaborative approach which group users based on their common searches and then finds users–items correlations within a group. To find the correlations between users according to their usage of items, we employ tensor, the high dimensional data model. Once users are clustered using the proposed tensor based clustering method, the associations shared by a group of users represented as top '*n*' items, are used for making recommendations within the group. Unlike most of previously adopted tensor models consisting of three dimensions, we have modeled users search log into more than three dimensions, and used the tensor factorization information for making recommendations. Empirical results on real car sales datasets show that the recommendation for all users suggested by the proposed tensor based recommendation method outperforms the recommendations given by the traditional collaborative based methods, which mostly employ vector/matrix methods to find similarity between users-items. Taking the average of recommendations done by CF methods and tensor based methods, on an average there was an improvement of about 40% in the precision, 52.78% in recall and 36.84% in F-score values.

## 2 Related Work

There are a myriad of work published, we present some of the recent related work employing CF and hybrid recommendations techniques. A collaborative filtering method to provide an enhanced quality of recommendations, derived from user-created tags is discussed by [4]. In this work collaborative methods of tagging item are employed to find users preferences for items. Data cubes consisting of three 2 dimensional matrices (User–item, User–tag and Tag–item frequency) which are transformed from three dimensional space for collaborative tagging are used. For recommendation Naïve Bayes classifier are used. The performance of such an approach was found out to be far superior than the plain collaborative recommendation approaches. A genetic algorithm that formulates purposeful association rules out of the transactions database of a transportation management system has been proposed by [5]. The constructed rules are recommended to the associated users. The recommendation process takes into account the constructed rules and techniques that are derived from collaborative filtering.

In another work [6] a novel hybrid recommendation method that combines the segmentation-based sequential rule method with the segmentation-based KNN-CF method is proposed. Here a sequential rule-based recommendation method analyses customers purchase behaviour over time to extract sequential rules. Sequential rules are extracted for each group from the purchase sequences to make recommendations. Consequently, the segmentation-based KNN-CF method provides recommendations based on the target customer's purchase data for the current period. The results of the two methods are combined to make final recommendations. A hybrid recommendation for an online retail store is proposed by [7]. The method adopts six steps for recommendation which are product taxonomy formation, grain specification, extracting product, category attributes, user (customer) profile creation, and user–user and user–product similarity calculation and recommendation generation. Experimental results show that proposed technique improves recommendation when compared to other similar CF based methods. Another recent work that proposes a hybrid approach that uses neural nets for recommendation is [8]. The proposed approach trains the artificial neural networks to group users into different clusters, and applies the well-established Kano's method for extracting the implicit needs of users in different clusters. The approach is applied on a tour and travel website to demonstrate the improvement for the problem of information overload.

Tensors have previously been used extensively in chemometrics and psychometric and some Web mining tasks like Web link analysis [9] and chat room analysis [10]. Recently some recommendation models, which have used three dimensional tensors for recommending music, tags and objects, have been proposed. A recommender model, using HOSVD for dimension reduction, have been proposed for recommending personalized music [11] and Tags [12]. Researchers [13] have used TSM based tag recommendation model which uses tensor factors by multiplying the three features matrices with core matrix each consisting of user, items and tags. Another collaborative filtering approach based on tensor factorization for making recommendations, where, the users, items and related contextual information are modeled as a three dimensional tensor is proposed by [20].

## 3. Model Construction, Decomposition and Clustering

Tensor notations and conventions used are similar to the notations used by previous authors [14]. Scalars are denoted by small letters *a, b*, vectors are denoted by boldface small letters like $\boldsymbol{a}$, $\boldsymbol{b}$. All subscript are shown by small letters starting from *i..n*. Matrices are shown using capital bold letters like **A, B** and the element *(i, j)* of a matrix is shown by $a_{ij}$. All tensors are represented using calligraphic fonts $\mathcal{T}$, e.g. $\mathcal{T} \in \mathbb{R}^{M_1 \times M_2 \times M_3 \times .. M_n}$ and the entries are shown using $a_{ijk}$ and the subscript *(i, j, k)* range from 1..to *I, J, K* in each mode. A tensor is a multi-dimensional data array which has $n$ dimensions (or modes). The order of a tensor is the number of dimensions. For example, the tensor $\mathcal{T} \in \mathbb{R}^{M_1 \times M_2 \times M_3 \times .. M_n}$ is of an order $n$ with $n$ dimensions. Each element of a tensor needs $n$ indices to represent or reference its precise position in a tensor, for example, the element $\mathrm{a}_{ijklmn}$ is an entry value at the *i, j, k, l, m* and *n* modes. In various tensor decomposition techniques the dimensions are flattened to represent matrices of various sizes before the subsequent decomposition technique is applied. Matricizing, unfolding or flattening of a tensor is a useful operation for transforming a given multi-dimensional array into a matrix. A third order tensor $\mathcal{A} \in \mathbb{R}^{I \times J \times K}$ is able to form three matrices of $I \times JK, J \times IK, K \times IJ$. More details on tensors and their properties can be found out in [14],[15].

### 3.1 Model Construction

Prior to creating the tensor model, the data is preprocessed. Pre processing includes arranging searches in different sessions made by a user into records, removing unwanted attributes from such records. This data is then grouped for each user based on the various searched parameters, and frequency of such records as grouped is counted. The prominent attributes from the users' data are identified and such attributes are then represented as modes of the tensor model. A tensor is created with all such features and the users as one dimension. For example, the structure of a tensor created, consisting of 5 searched dimensions and the users are as follows:

$$\mathcal{T} \in \mathbb{R}^{Make \times Model \times Bodytype \times Search\ Type \times Cost\ Type \times Users} \tag{1}$$

For each user the term frequency of each similar search is counted. A similar search is a search whose all searched parameters are same. As an example for a given user, the different searched parameters like make, model, body type, search type and cost of a car may be same, and user may have searched them many times in different sessions. The term frequency value for all the searches of a user are found out. Next theses values are populated in the tensor. As an example, the term frequency $t_{ijklmn}$ is an entry value at the *i, j, k, l, m* and *n* modes, where *i* represents the *Make*, *j* the *Model*, *k* the *Bodytype*, *l* the search type, *m* the cost ranges and *n* the user id.

### 3.2 Decomposition

In multi-dimensional data modeling, the decomposition process enables to find the most prominent components (i.e. tensor entries and modes) as well as the hidden relationships that may exist between different components. We have used the popular and widely used PARAFAC [16] tensor decomposition technique to decompose the constructed model. PARAFAC is a generalization of PCA (Principal Component Analysis) to higher order arrays. Given a tensor of rank 3 as $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$, a R-component PARAFAC model can be represented as

$$x_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} + E \tag{2}$$

where $a_i, b_i, c_i$ are the $i^{th}$ column of component matrices $\mathbf{A} \in \mathbb{R}^{I \times R}, \mathbf{B} \in \mathbb{R}^{J \times R}$ and $\mathbf{C} \in \mathbb{R}^{K \times R}$ respectively and $E \in \mathbb{R}^{I \times J \times K}$ is the three way array containing residuals. $x_{ijk}$ represents an entry of a three way array of $\mathcal{X}$ and in the $i^{th}$ row, $j^{th}$ column and $k^{th}$ tube. Thus in our case when the users tensor (equation 1) is decomposed using [17], the various matrices formed are as shown in the figure 1 below. In figure 1, $\mathbf{M_1, M_2..M_n}$ are the various component matrices formed after the decomposition of the tensor, and *R* is the desired best rank tensor approximation, which is set as 1, 2, and 3 in all the experiments.

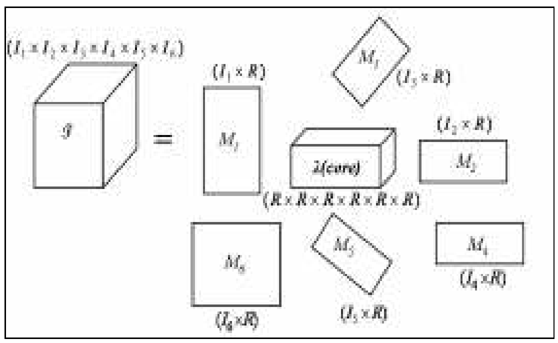


**Fig. 1.** PARAFAC Decomposed tensor of users-searches, gives component matrices as shown.

### 3.3 Clustering

Clustering is done on the component matrices $\mathbf{M}_1, \mathbf{M_2..M_n}$ obtained after PARAFAC decomposition and representing decomposed values of each mode, from $M_1, M_2, ...M_n$ respectively. Each component matrix $\mathbf{M}_1, \mathbf{M_2..M_n}$ is of dimension $\mathbf{M}_{i \times r}$ where $i$ is the number of ways in a mode $M_n$, and $r$ is the value of best rank approximation of the tensor. The n$^{th}$ matrix obtained after PARAFAC decomposition (Shown as $\mathbf{M}_6$ in fig. 1) represents the users' dimension. Clustering on row values of this matrix is achieved by using two clustering methods the EM (Expectation Maximization) and k means [18]. We have taken the last component matrix $\mathbf{M}_n$ as clustering input since it represents the users dimension in the proposed data model. Entries (or values) in the matrix $\mathbf{M}_n$ depicts the correlations between users based on the multiple factors of the tensor. Therefore, clustering on the matrix $\mathbf{M}_n$ results in grouping users according to similar search behaviour which is based on multiple search components as modeled in the tensor.

## 4 Discovering Association within clusters

All searches made by users in a cluster are grouped and frequent associations based on desired query components (like make-model in our case) are mined. Thus each cluster contains the searched parameters, as searched by users of the respective cluster. Association rules are mined from each individual cluster. We have considered associations of length two, as the occurrence of associations of length greater than 2 was very rare, especially when the number of users in a cluster is small. The whole recommender process is explained in the algorithm in figure 2.

**Input:** Processed search data users and searched parameter wise, *n* the number of recommendations desired.
**Output:** Top *n* recommendations.
Let *k* be the number of clusters (here *k*=10, 20, 30). Let $\text{Rule}_{rk} = X_r \Rightarrow Y_r$ be the rule mined for cluster *k* on item set $X_m$ and $Y_m$ such that $X_r, Y_r \in X_m, Y_m$ and $X_r \cap Y_r = \varnothing$, where *m* is the number of item sets in a cluster, on which association rules are mined and *r* be number of rules generated for each cluster k. Let *j* be a counter variable, initialized at zero.

```
Begin
   Step 1. Create users-searches tensor (𝒯).
   Step 2. Decompose the tensor (𝒯), and cluster the last component matrix M_n.
   Step 3.  // Find Association rules for each cluster from  1 to k.
   Step 4.  //Order Association rules in decreasing order of confidence score.
     for i=1 to k //Evaluating distinct rules for each cluster from 1 to k
              j=0; r=1;
              Do While (j!=n)
                  Extract (r); //Function retrieves top r^th rule with highest confidence from
                  cluster k.
                  If (X_r ⇒ Y_r) = (Y_r ⇒ X_r) then
                        Save(X_r ⇒ Y_r) ;  //Save(r) function saves the distinct item sets from
                                               association rules for the cluster k.
                        j=j+1;  r=r+1;
                  Else
                        //Retrieve rules with next highest confidence score.
                        Save(X_r ⇒ Y_r) ;
                        j=j+1;
                        Save(Y_r ⇒ X_r) ;
                        j=j+1;  r=r+1;
     End for
    // Cluster wise Top n recommendations
     Top_k n  =X_n ⇒ Y_n;
  End
```

**Fig 2.** Complete Recommendation algorithm.

## 5.1 Experimental Design

All experiments have been done on server log data collected from a live Australian car sales website[1]. Search log data of duration 1 month was used. Out of a total number of users, 949 users who had made searches in that period were used in experiments. These users were identified based on IP address (IP), web browser and Geo segmentation details like location and PIN number. These users had made leads or enquiries about a car of their interest, showing that these users were interested in buying the particular car, for which they had made the lead. All leads about a car were made by emailing the dealer through the e-mail feature (contact us) provided by the website. In all the experiments, the data used for making recommendations were taken prior to a user had made a lead. The major objective was to match leads with the correctly made recommendations by various traditional collaborative methods and collaborative methods based on tensor model. Some statistics for the user searches and leads are shown below in Table 1:

**Table 1.** Statistics of filtered data used for tensor modeling.

| No. of Sessions | No. of Users | Average searches per user | No. of Leads | Avg. Lead/User |
|---|---|---|---|---|
| 2692 | 949 | 14 | 1649 | 1.74 |

Five parameters used for searching like make, model, body type, cost, search type (e.g. like new or used) plus a sixth dimension (users) were taken as dimensions of the tensor model (Table 2).

**Table 2.** Number and Sample of Dimensions Used in the Tensor Model.

| Dimension Name | No of unique ways or modes | Sample dimension values |
|---|---|---|
| Make | 68 | Toyota, Holden, Ford |
| Model | 644 | RAV4, Liberty. |
| Body Type | 12 | Sedan, Hatch |
| Search Type | 5 | New, Used, Ex Demo. |
| Cost Type | 13 | $1-2500 |
| Users Id | 949 | 1, 2,..949 |

Once the data was pre processed the users-items tensor model was created as $\mathcal{T} \in \mathbb{R}^{68 \times 644 \times 12 \times 5 \times 13 \times 949}$. Subsequently, decomposition was achieved using the PARAFAC model. In the absence of an ideal clustering solution, three cluster solutions consisting of 10, 20 and 30 clusters were used for evaluation. For each cluster, association rules with highest confidence score and make/model as associated items were taken. For each cluster the number of association rules generated was 3, 5, 10, and 15. Once frequent patterns in the form of association rules are mined cluster wise, only rules having highest support and distinct make-model were taken for recommendations from these rules. These top *n* make-models

[1] Due to privacy issues we are unable to specify the details about the website.

were given as recommendation to each of the users belonging to a same cluster. A sample of some association rules are shown as below.

| **Best rules found:** |
|---|
| 1. X-TRAIL=X-TRAIL 10 ==> NISSAN=NISSAN 10 conf:(1) |
| 2. NISSAN=NISSAN 10 ==> X-TRAIL=X-TRAIL 10 conf:(1) |
| 3. X-TRAIL=GRAND VITARA 2 ==> NISSAN=SUZUKI 2 conf:(0.89) |
| 4. NISSAN=SUZUKI 2 ==> X-TRAIL=GRAND VITARA 2 conf:(0.75) |

**Fig.3** Sample of associations found out for a cluster1.

Example for the above case as shown in Figure 3 the top 3 cars recommended with make-model were: 1. NISSAN- X-TRAIL, 2 SUZUKI- GRAND VITARA and 3 SUBARU- OUTBACK. All users with similar interests belonging to a cluster (Evaluated based on users-searches using tensor) will be recommended these top *n* cars.

### 5.2 Evaluation Criteria

To evaluate the quality of top-*m* recommendations given by each method we used the following metrics. Let $L_n$ be the number of leads made by a user $U_n$, and let $R_n^m$, be the top-*m* recommendations given by various methods to $U_n$, where $m \geq 3 \text{ and } m \leq 15, m \in \{\{3\},\{5\},\{10\},\{15\}\}$. Precision $(Pr_n)$ and recall $(Re_n)$ for each user $U_n$ is evaluated as

$$Pr_n = \frac{R_n^m \cap L_n}{R_n^m \cap L_n + (R_n^m - L_n)} \qquad \text{And recall as } Re_n = \frac{R_n^m \cap L_n}{R_n^m \cap L_n + (L_n - R_n^m)} \tag{3}$$

### 5.3 Results

The average details of clustering results for each method are shown below in Table 3, where the acronyms used in the table 3 are U-ESM (Users-users Euclidian Similarity Measure) and U-CSM (Users-users Cosine Similarity Measure). U-ESM and U-CSM are evaluated based on users-items relationships, where such relationships are represented as vectors in two dimensional spaces. Clustering is done on these vectors to find users with similar interests. P1-EM, P2-EM and P3-EM are the PARAFAC best rank approximation of rank 1, 2, 3 respectively, with clustering achieved using Expectation Maximization (EM) [18]. Similarly P1-kM, P2-kM and P3-kM are the PARAFAC best rank approximation of rank 1, 2, 3 respectively, with clustering achieved using k means. The other values are Pr=precision, Re=recall and Fs=F Score.

**Table 3.** Average Recommendations for various methods.

| | Top 3 Recommendation | | | Top 5 Recommendation | | | Top 10 Recommendation | | | Top 15 Recommendation | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Pr** | **Re** | **Fs** | **Pr** | **Re** | **Fs** | **Pr** | **Re** | **Fs** | **Pr** | **Re** | **Fs** |
| **U-ESM** | 0.08 | 0.10 | 0.06 | 0.06 | 0.12 | 0.06 | 0.07 | 0.16 | 0.06 | 0.07 | 0.16 | 0.06 |
| **U-CSM** | 0.24 | 0.46 | 0.31 | 0.23 | 0.56 | 0.31 | 0.22 | 0.59 | 0.30 | 0.22 | 0.68 | 0.29 |
| **P1-EM** | **0.35** | **0.60** | **0.43** | **0.32** | **0.59** | **0.40** | 0.22 | 0.67 | 0.30 | 0.21 | 0.69 | 0.28 |
| **P2-EM** | 0.25 | 0.38 | 0.25 | 0.24 | 0.4 | 0.25 | 0.17 | 0.55 | 0.18 | 0.16 | 0.61 | 0.18 |
| **P3-EM** | 0.34 | 0.56 | 0.40 | 0.29 | 0.57 | 0.36 | **0.24** | **0.69** | **0.29** | **0.23** | **0.72** | **0.28** |
| **P1-kM** | 0.15 | 0.25 | 0.17 | 0.11 | 0.26 | 0.15 | 0.13 | 0.53 | 0.21 | 0.11 | 0.57 | 0.17 |
| **P2-kM** | 0.22 | 0.47 | 0.29 | 0.21 | 0.51 | 0.27 | 0.14 | 0.65 | 0.22 | 0.12 | 0.73 | 0.19 |
| **P3-kM** | 0.22 | 0.44 | 0.28 | 0.19 | 0.48 | 0.25 | 0.16 | 0.63 | 0.24 | 0.14 | 0.67 | 0.21 |
| **Average** | **0.23** | **0.41** | **0.27** | **0.21** | **0.44** | **0.26** | **0.17** | **0.56** | **0.23** | **0.16** | **0.60** | **0.21** |

## 6 Discussion

Due to very high number of dimensions of interest vector of users (742 dimensions excluding users and as shown in table2) clustering based on Euclidean (ESM) using k means and EM was unable to produce good clustering of users which ultimately resulted in not so high quality of recommendations. This can happen because as the number of dimensions grows significantly, ESM (Euclidian Similarity Measure) and CSM (Cosine Similarity Measure) eventually become less similar. In very high dimensional spaces as dimension gets higher ($\geq$ 128) [19], the two similarity measures start having small variations between them. However, the rate of decrease of similarity is very slow. Similarity vectors of instances in such high dimensional spaces starts loosing inter component relationships, when traditional two dimensional distance based methods are employed. On the other hand cosine (CSM) measure produced average quality of results. This happens because cosine similarity is able to map the different dimensions, but due to the two dimensional model, latent relationships between users-items are lost.

In contrast the tensor based methods are able to extract hidden relationships between the datasets and give much improved similarity results for the users-items data. For making CF based recommendations, traditional k means algorithm with Euclidian and Manhattan similarity measures performed worse, whereas CSM methods performed average. On the other hand, EM based methods performed exceptionally well. These contrasting results confirm that distance based approaches using Euclidian, Manhattan or cosine base similarity measures used in high dimensional data mappings reflect strange properties [18] which include loss of inter component relationships and unable to map inter component latent relationships. EM is density based clustering algorithm, and rather than using a distance based clustering measure, it assigns a probability distribution to each instance, indicating the probability of it belonging to each of the other clusters.

From the results (table 3, figure 4) it is clear that, the number of best recommendations made to a user is around 3-5. The aggregate recommendation scores of precision, recall and F Score for each datasets using simple CF based methods and PARAFAC –EM and PARAFAC-kM based methods are shown below in table 4.

**Table 4.** Average Aggregate Recommendations for various methods.

| Methods | Precision | Recall | F-Score |
|---|---|---|---|
| CF-Distance Based | **0.15** | **0.36** | **0.19** |
| PARAFAC-kM | 0.16 | 0.52 | 0.22 |
| PARAFAC-EM | 0.25 | 0.59 | 0.30 |
| Avg. PARAFAC-EM+kM | **0.21** | **0.56** | **0.26** |

The other noteworthy observation was that in most associations with large lengths ( > 2) , there was a reduction in number of frequent item sets discovered in the process, which had high support and confidence values. Association with length=1 had too many rules, and such rules were biased towards rules with highest frequency. Hence such rules were ignored for analysis in the experiments. An interesting observation which is shown in figure 5 shows relevant F-score when tensor best rank decomposition are considered. In the figure 5, 1EM, 2EM, 3EM refer to PARAFAC decomposition with best rank approximation of 1, 2 and 3 respectively, where clustering was achieved using EM clustering algorithm. Similarly 1KM, 2KM, 3KM refer to PARAFAC decomposition with clustering achieved using k-means clustering method. In both cases, using EM and KM performance starts decreasing with the increase in rank and number of recommendations. In case of KM clustering, rank 1 performs worst. KM clustering methods use distance measure for clustering. Due to the availability of singular values per instance for clustering, some useful relationships between instances may not be clearly distinguishable. When decomposing with higher ranks the extra factors available for clustering, which have the ability to preserve some information, KM's performance starts to increase. However decomposition at higher ranks may start loosing valuable inter component relationships, due to complete flattening of the tensor.

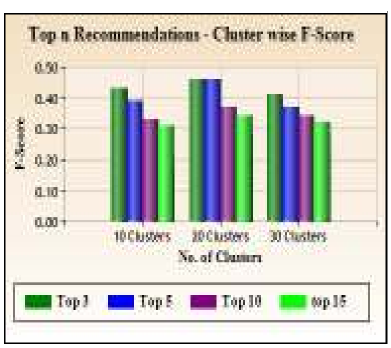


**Fig.4** F-Score for Top *n* recommendations cluster sizes.

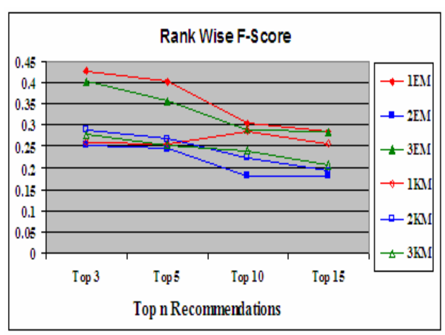


**Fig.5** F-Score Rank Wise, top n with various recommendations.

The figure 5 clearly separates top3, top 5 recommendations from top10 and 15. Rank 1 approximation using EM clustering performed the best among all cases. In case of KM, rank 1 gave best results when number of clusters was 30 and for rank 3, KM gave best when it was 30. These contrasting results in case of KM are clear indicator that distance measures may not work well when intra distance between instances is small. The other observation is that distance measures need larger cluster sizes to maximize the distance between instances and in hence improve overall performance.

## 7 Conclusion

This paper presented a novel method of recommendation based on tensor clustering and associations. Users-items similarity measures were evaluated using high dimensional data model tensors. Once the model was decomposed using PARAFAC, clustering was achieved on the users matrix. Association rules for users, clustered in a group were found out. Once such rules were found out, only unique rules with highest confidence were taken as top *n* recommendation for the users in a cluster. Experimental results show that tensor based recommendation method and unique association rule generation for making recommendations outperforms the traditional CF based methods, which perform user-items similarity measure using vector or matrix based methods. Since most of the processes for generating rules, creating and clustering users can be done offline, the system can effectively be used for generating high quality of online recommendations, thus limiting recommendation to top 3-5 recommendations per user. However as of now, since the process of identifying top *n* unique recommendations from association rules is not automated, a lot of time is needed to generate such top *n* recommendations for each group.

## Acknowledgement

This research has been funded by CRC (Co-operative Research Centre), Australia and Queensland University of Technology, Brisbane Australia under the CRC Smart Services Web Personalization Project 2009-10.

## References

1 C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," in *Handbook on Recommender Systems, Springer*, 2009.

2 Y. J. Park and K. N. Chang, "Individual and group behavior-based customer profile model for personalized product recommendation." vol. 36: Elsevier, 2009, pp. 1932-1939.

3 D. Skillicorn, *Understanding complex datasets: data mining with matrix decompositions*: Chapman & Hall/CRC, 2007.

4 H. N. Kim, A. T. Ji, I. Ha, and G. S. Jo, "Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation," *Electronic Commerce Research and Applications,* vol. 9, pp. 73-83, 2009.

5 A. Lazanas and N. Karacapilidis, "Enhancing Recommendations through a Data Mining Algorithm," in *Proceedings of the 12th international conference on Knowledge-Based Intelligent Information and Engineering Systems*, Zagreb, Croatia, 2008, pp. 525-532.

6 D. R. Liu, C. H. Lai, and W. J. Lee, "A hybrid of sequential rules and collaborative filtering for product recommendation," *Information Sciences,* vol. 179, pp. 3505-3519, 2009.

7 A. Albadvi and M. Shahbazi, "A hybrid recommendation technique based on product category attributes," *Expert Systems with Applications,* vol. 36 ,Issue 9, pp. 11480-11488, 2009.

8 C. C. Chang, P. L. Chen, F. R. Chiu, and Y. K. Chen, "Application of neural networks and Kano's method to content recommendation in web personalization," *Expert Systems with Applications,* vol. 36, Issue 3, pp. 5310-5316, 2009.

9 T. G. Kolda, B. W. Bader, J. P. Kenny, C. A. Livermore, and N. M. Albuquerque, "Higher-Order Web Link Analysis Using Multilinear Algebra," in *Fifth IEEE International Conference on Data Mining (ICDM'05)* 2005, pp. 242-249.

10 S. A. C. a. Evrim Acar, Mukkai S. Krishnamoorthy, and B¨ulent Yener, " Modeling and Multiway Analysis of Chatroom Tensors," in *Intelligence and Security Informatics*. vol. 3495/2005: Springer Berlin / Heidelberg, 2005, pp. 256-268.

11 P. S. M. Ruxanda and A. N. Y. Manolopoulos, "Ternary Semantic Analysis of Social Tags for Personalized Mucis Recommendation," in *Proceedings of the 9th International Conference on Music Information Retrieval*, 2008, p. 219.

12 P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "Tag recommendations based on tensor dimensionality reduction," in *RecSys '08'*, Lausanne, Switzerland, 2008, pp. 43-50.

13 S. Rendle and B. Marinho, "Learning optimal ranking with tensor factorization for tag recommendation," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, Paris, France 2009, pp. 727-736.

14 T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," Technical Report SAND2007-6702, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, , November 2007.

15 T. G. Kolda and J. Sun, "Scalable Tensor Decompositions for Multi-aspect Data Mining," *time,* vol. 18, p. 19.

16 R. A. Harshman, "Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multi-modal factor analysis," *UCLA working papers in phonetics,* vol. 16, p. 1, 1970.

17 B. W. Bader and T. G. Kolda, "Efficient MATLAB computations with sparse and factored tensors," *SIAM Journal on Scientific Computing,* vol. 30, pp. 205-231, 2007.

18 E. F. Ian H. Witten, *Data Mining: Practical machine learning tools and techniques, 2nd Edition* San Francisco: Morgan Kaufmann, 2005.

19 G. Qian, S. Sural, Y. Gu, and S. Pramanik, "Similarity between euclidean and cosine angle distance for nearest neighbor queries," in *ACM Special Interest Group on Applied Computing* ACM, 2004, pp. 1232-1237.

20 A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering," in *ACM Recommender Systems*, Barcelona, Spain, 2010, pp. 79-86.