Pertanika J. Sci. & Technol. 27 (4): 1783 - 1802 (2019)

PERTANIKA JOURNALS

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Time Adaptive Collaborative Filtering for Movie Recommendation

Suganeshwari Gopalswamy and Syed Ibrahim Peer Mohamed*

School of Computing Science and Engineering, Vellore Institute of Technology, Chennai, Tamil Nadu 600 127, India

ABSTRACT

Collaborative filtering is the most widespread recommendation system technique deployed in e-commerce services nowadays. It recommends products based on the historical preference of the user. The biggest challenges in these techniques are data sparsity and growing volume of data, specifically in e-commerce sites like movie recommendation. Clustering algorithms are used for scaling up the performance of collaborative filtering in dynamically growing datasets. Most of the existing clustering based recommendation algorithms improve scalability but produce low quality recommendations. This is mainly due to data sparsity, as the user tends to rate very few items from a large number of options available. Moreover, users with a similar taste for a group of items may show different likings for another group of items over a period, i.e.., user's interest dynamically changes over time. Finding the sub-groups that are more relevant to each other than the entire user-item matrix is more affordable. Since the user's recent ratings can better represent their interest and preference, a Time Adaptive Collaborative Filtering Method —TACF is proposed, that adopts time to generate a recommendation. Experimental results on the MovieLens dataset show that the proposed system outperforms other state-of-art collaborative filtering algorithms in terms of accuracy and efficiency.

Keywords: Clustering, collaborative filtering, matrix factorization, recommendation system, temporal information, user drifts

ARTICLE INFO

Article history:
Received: 24 January 2019
Accepted: 04 Jun 2019
Published: 21 October 2019

E-mail addresses:
g.suganeshwari2015@vit.ac.in (Suganeshwari Gopalswamy)
syedibrahim.sp@vit.ac.in (Syed Ibrahim Peer Mohamed)
* Corresponding author

INTRODUCTION

In recent years, the recommender system has emerged as a tool to cope up with the information overload problem. With the growing volume of information on the web, it has become inevitable for companies to provide the most relevant information to the user. Currently, companies are deploying

ISSN: 0128-7680
e-ISSN: 2231-8526

© Universiti Putra Malaysia Press

intelligent recommender systems to personalize recommendations according to the customer's taste. The Recommendation system has been a great success in dealing with an information overload problem generating more relevant recommendation in a variety of applications like recommending movies, CDs, books and webpages.

Recommendation System (RS) is broadly classified as content-based, collaborative filtering (CF) and hybrid approaches. Content-based approaches generate a user profile based on the purchase history of the user (Desrosiers & Karypis, 2011). This profile is compared with item features and then suitable recommendations are generated. Collaborative filtering (Sarwar et al., 2001) provides a recommendation based on the idea of people who share a similar taste in certain items sharing the same interest in the rest of the items. It works well for complex items like movies (Harper & Konstan, 2016) music (Tan et al., 2011) and tourism (Jiang et al., 2016). Hybrid methods work by combining the content and collaborative approaches.

CF method is the most popular method used in the recommender system. It is further classified as memory and model-based approaches (Adomavicius & Tuzhilin, 2005). It finds extensive implementation in the e-commerce industry due to its justifiability and ease of implementation (Desrosiers & Karypis, 2011). However, this method suffers from data sparsity, which occurs mainly as a user's rate only a few products from a large number of available options. Furthermore, when the number of users and items increase, the computational complexity also increases leading to poor scalability becoming infeasible to produce recommendations within the elapsed time interval.

Model-based techniques have been introduced for meeting the challenges from memory-based CF. These techniques discover the rating pattern from historical data and provide highly accurate and efficient recommendations from some sample data. Bayesian network (Luo et al., 2012), Clustering model (Li & Kim, 2003) (West et al., 2016) and matrix factorization (Takács et al., 2008) are familiar techniques used in model-based approaches. SVD, SVD++ (Koren, 2009) and ALS (Chen et al., 2017) are popular MF techniques used in the recommender system that gained importance following the Netflix price challenge (Bell & Koren, 2007).

In the clustering-based recommendation approach, the users and items are segmented in such a way that objects in the same cluster have a large similarity. The CF method gathers ratings from all the users and then computes the predictions. It is a time-consuming process and in the previous work (Suganeshwari & Ibrahim, 2016) has focused on the incorporation of the temporal information for generating a time-aware recommendation and improving efficiency. However, despite the improved performance in recommendation quality, scalability still remains an issue in a real-time environment. Motivated by these observations and understanding of the likelihood of popular items causing bias in the recommendations generated, we propose a Time Adaptive Collaborative Filtering (TACF).

This paper provides a scalable algorithm which shows a significant improvement in precision on recommending top-k items when compared with other existing algorithms ALS, UBCF, IBCF, and time-sensitive CF (TSCF) (Sun et al., 2016). The experiments have been performed using benchmark MovieLens dataset.

The rest of the paper is organized as follows. Section II provides an introduction to the preliminary concepts of RS and discusses its related work and its limitations. Section III presents the proposed method Time Adaptive Scalable Neighborhood (TACF). Section IV describes the experimental setup and enlists the evaluation results followed by the conclusion.

BACKGROUND AND RELATED WORK

Section 2 introduces the preliminaries on memory-based and model-based collaborative filtering methods. Table 1 represents the symbol and their representation used in this paper.

Table 1
Table of symbols

Symbol	Definition
U	Set of users
I	Set of items
R^{mxn}	Utility matrix
u	User
i	Item
u_{ij}	User i rating on item j
W	Item clusters
K	Dimension of the user and item latent factors
P	User latent factor matrix
Q	Item latent factor matrix
λ	Regularization parameter
α	# of neighbors
δ	Recent transactions of user
S_{ij}	Similarity between item i and j
b_u	User bias
b_i	Item bias

Memory-Based Collaborative Filtering

Memory-Based CF methods are also referred to as neighborhood methods. They recommend items on the basis of the views of other like-minded people. Similarity computation plays a vital role in CF methods (Herlocker et al., 2002). The most popular statistical techniques used for computing similarity are Pearson correlation (Adomavicius & Tuzhilin, 2005) and cosine based (Meng et al., 2014) measures. Here the entire utility

matrix of size *User X Item* is taken as input. The similarity is evaluated by employing user correlation in user-based (Meng et al., 2014) and item correlation in item based (Sarwat et al., 2014) methods. The bottleneck in CF methods computation of the similarity between users/items. Though these methods pioneer the e-commerce industry, it is affected by data Sparsity. The neighborhood-based CF methods rely on exact matches for computing correlations resulting in an extreme sparse dataset. This insufficient *user X item* matrix leads to inaccurate predictions where similar items do not show any correlation and are termed as reduced coverage (Billsus & Pazzani, 1998). Traditional CF algorithms suffer in scalability when dealing with the tremendously growing volume of data. Several methods have been proposed for dealing with problems like imputation techniques (Ren et al., 2013), and incorporation of the contextual parameters that include location, the company of other people and time. LARS, a location-aware recommendation system has been proposed that provides recommendations on the basis of spatial ratings (Sarwat et al., 2014). Time is used as a special type of context in Ding & Li, (2005) and Panniello et al. (2014) for improving the recommendation quality. Memory-based methods rely on the similarity computation between users/items for the determination of the predictions, but changes in user preference drifts can mislead recommendations that may not be of interest to the user in the current situation. Time is used to compute the similarity between the users in a Time-aware RS (Hu et al., 2015). Ratings within the similar timestamps are assigned a high weight than the other older ratings by employing an exponential decay function. The function value drops rapidly when the time difference increases. It is a time-weighted approach where the older values are penalized. This method cannot be deployed in movie recommendation application as user rates the movie only once and the rating remains static. Sun et al., (2016) proposed a time-sensitive CF (TSCF) approach to discover the latest preference of the user by ordering the items based on the time behavior sequence. The system considers only one transaction which is insufficient to generate better recommendations. The proposed method claims that recent *n* transaction of each user is adequate to compute similarity and to reflect the user's current preference.

Model-Based CF

In real-time applications deployment of the memory-based method is infeasible especially with the explosive increase in the number of items and users. The model-based CF utilizes the rating matrix to construct a model and later exploits it for future predictions, thus improving the efficiency of the system. Different machine learning techniques like classification, clustering, and matrix factorization are employed to construct the model. The classification algorithms used in CF are Simple Bayesian Algorithm (Miyahara & Pazzani, 2000 and Su & Khoshgoftaar, 2006) and Baseline Bayesian model (Heckerman et al., 2000). Matrix Factorization is a widely used dimensionality reduction technique

in RS. An extensive survey on recommendation (Takács et al., 2008 and Ba et al., 2013) shows the help rendered by the application of matrix factorization techniques in revealing the latent factor. The results can be easily interpreted for predicting the rating for an item.

User's interest is always drifting with time and a dynamic model based RS is needed to address this. The user rating behavior is influenced by the user rating style and the item's popularity. SVD++ incorporated temporal information to improve the quality of recommendation (Bell & Koren, 2007). The lessons learnt from Netflix price challenge (Bell & Koren, 2007), help the neighborhood method and model-based methods in the exploration of different levels of data patterns and hybrid methods alone have the ability to produce more optimal results. In the proposed work, a time-adaptive hybrid model is designed to address the scalability and sparsity problem.

Large-Scale Datasets and Clustering

Data scalability is one of the key challenges in providing recommendation in a real-time environment. This occurs mainly as a result of the tremendous growth of users and items in modern e-services. The aim of clustering algorithms is to partition the data into meaningful subgroups. High-quality clusters are produced when elements within the cluster are more similar and elements from different clusters are dissimilar. Clustering algorithm segments items based on the user rating data (O'Connor & Herlocker, 1999). Similarly, partition based on user-user similarity was proposed by Sarwar & Karypis (2002a). Xue et al. (2005) used the clustering method to smooth the missing value in the utility matrix. West et al. (2016) had proposed a scientific article based on the citation-based network. It used a hierarchical representation of scientific structure as domains, fields, subfields, and sub-subfields for different levels of influence. Scalability and sparsity problems were addressed in Ma et al. (2016) where different clustering techniques were adopted on the basis of the user, item and trust relationships. Guo et al. (2015) had utilized the rating information along with social trust information for iterative segmentation of users. There are myriad applications in a recommendation system that exploits the clustering methods. All these methods get adversely affected due to sparsity, resulting in a low-quality recommendation. Hence, we argue that better recommendations can be produced if the data to be clustered is dense. The proposed TACF method applies a clustering algorithm to produce clusters based on latent factors and exploits time to generate recommendations. The proposed method can improve the efficiency and mean average precision of the CF method, and this is demonstrated in the experimental results.

TIME ADAPTIVE COLLABORATIVE FILTERING FRAMEWORK

The primary objective of TACF is to find the item sub-groups drowned in the *user x item* utility matrix for improving the quality of the CF recommendation algorithm. It is difficult

to identify the retention of the user's interest in the item as the user rates an item only once. To address this problem, the matrix factorization method is utilized where the user and item features are found and used as input to the clustering method. The features are more compact and denser. Items are clustered based on item features. Recommendations are generated based time adaptive function. The proposed method fully considers the user's preference drifts, addresses sparsity issues and guarantees the achievement of better performance.

Problem Formulation

With the implication of the presence of n user's $U = \{u_1, u_{2, \ldots,} u_n\}$ and m items $I = \{i_1, i_{2\ldots,} i_m\}$ represented as matrix R of size m x n. Each user's rating is represented as u_{ij} where u_i represents the i^{th} user and u_j represents the j^{th} item. The aim is to divide the items into w subgroups $\{w_1, w_2\ldots, w_c\}$. This is the clustering problem, but a matrix factorization model comprising user and item factors has been created for framing more meaningful segments. The rating matrix is decomposed into lower rank matrices P and Q^T using the ALS method as given in equation 1.

$$R = PQ^T \quad (1)$$

$P \varepsilon R^{mxk}$ represent the user latent factors, $Q \varepsilon R^{kxn}$ represent the item latent factors, k is the latent factor formed and T is the transpose operator. The aim of this function is to frame P and Q^T such that the Frobenius norm is minimized. Once the factor matrices are built, the x^{th} row vector $p_x \varepsilon P$ represents the user latent factor and y^{th} column vector $q_y \varepsilon Q$ represents the item latent factor In the proposed method, the item factors are used as inputs to the clustering model as they generate subgroups based on the user-rating behavior.

The TACF encodes the item factors Q^T into w clusters and represents them as $w \varepsilon [0,1]$. Here each data point w_{ij} is a value that indicates the presence of the element in the subgroup. Where $w_{ij} = 1$ the element is in the subgroup else, it is not present in the subgroup. The item factor Q^T is formally partitioned into w_c $\{w_1 \cup w_2 \ldots w_c\} = Q^T$, $w_x \cap w_y = \phi$, for $x \leq 1$ and $y \leq c$ as shown in equation 2.

$$Q^T = \{w_1, w_2, \ldots w_c\} \quad (2)$$

The squared Euclidean distance is used in classical clustering.

$$K_p = \sum_{i=1}^{n} \sum_{j=1}^{c} u_{i,j}(p_{i,t}) \quad (3)$$

Design of TACF

TACF proposes a scalable neighborhood formulation for improving the efficiency of collaborative filtering. The steps involved in the algorithm framework are shown in Figure

1. In the first stage, an ALS model is created from the user-item matrix R. The decomposed matrix consists of user features and item features. This item factor acts as input to the TACF model.

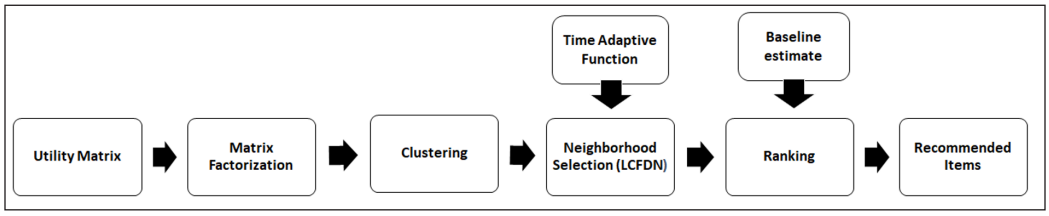

Figure 1. Steps in TACF method

Matrix Factorization

Matrix Factorization is the most popular CF algorithm used for solving the co-clustering problem. CF methods produce accurate recommendations on dimensionality reduced dataset rather than sparse dataset (Sarwar et al., 2001). The utility matrix of dimension U x I get reduced to U x K and I x K. MF techniques can take the input both explicit and implicit. User's rating for an item can be predicted through simple multiplication of the matrix as given in equation 4.

$$r_{u,i} = p_u^T q_i \quad (4)$$

The following quadratic function should be minimized for learning the factors obtained from the factorization techniques.

$$argmin_{p,q} \sum_{u,i} \left(r_{u,i} - p_u^T q_i\right)^2 - \lambda(|p_u|^2 + |q_i|^2) \quad (5)$$

$\sum_{u,i}\left(r_{u,i} - p_u^T q_i\right)^2$ is the Mean square error of the original matrix and the approximation matrix. The $\lambda(|p_u|^2 + |q_i|^2)$ is the regularization term added for avoiding the overfitting of the data. The idea is to produce a minimum value for the cost function using the parameters k and λ. The popular methods used in recommendations are SVD, SGD and ALS.

Singular Value Decomposition (SVD). SVD is the well-established factorization technique used in RS. In SVD, the matrix R gets reduced to three matrices p, σ, and q.

$$r_{u,i} = p\sigma q^T \quad (6)$$

Here p represents the left singular matrix in which each user is represented in some latent factor, q represents the right singular matrix in some latent factor and ∑ represents the strength of the latent factor. Maximum Margin Matrix Factorization is a low-rank approximation (DeCoste, 2006) that addresses the noise problem in CF. SVD has been

extensively researched in CF methods, but it suffers from computational complexity and poor recommendation when data is sparse.

Stochastic Gradient Descent (SGD). In SGD (Yu et al., 2014) the cost functions are computed, and the factors in the opposite side of the opposite side of the gradient are updated. SGD is not practical when the explicit preferences are converted to implicit ones. User time items would be in the order of huge numbers. SGD requires many numbers of iterations for obtaining the good model. Its performance is also based on the learning rate (Park et al., 2017). This has helped the proposed method in the adoption of the alternative least square method which is more efficient

Alternative Least Square. Alternate Least Square (ALS) is another optimizing technique that can handle data sparsity and still achieve good performance (Chen et al., 2017). The main advantage of this method is its parallel implementation and suitability for large datasets. TACF utilizes explicit user input as ratings. P_u and Q_i ε R^f are the user and item vectors that represent the value that measures the latent factor, the user and item possessed. By calculating the partial derivative of p_u, q_i and substituting it to 0, equation 7 and equation 8 are obtained.

$$p_u = \left(Q^T Q + \lambda I\right)^{-1} Q^T r_u \quad (7)$$

$$q_i = \left(P^T P + \lambda I\right)^{-1} P^T r_i \quad (8)$$

Here I is the unit matrix, r_u, r_i is the u^{th} row and i^{th} column of the matrix R. The solution given by ALS is unique and the model is constructed until convergence. ALS is adopted in TACF since it has the advantage of parallel implementation with the ability to attain accurate recommendations in the sparse dataset. After decomposing the original matrix *R* into pq^T, the item factor vector q_i^T is used as input to the clustering method.

Clustering Items

Clustering methods groups objects in segments where members within the cluster have similar features and members of different clusters are dissimilar. This dimensionality reduction technique produces low-quality recommendation when compared to the nearest neighborhood method (Sarwar & Karypis, 2002b). K-means clustering partitions the data points into *k* distinct clusters. The proposed work utilizes the Euclidean distance measure. The number of clusters (K) is selected based on the elbow method and shown in Figure 2. The objective function of the *K-means* is to minimize the squared error function as in equation 9.

$$J = \Sigma_{i=1}^{n} \Sigma_{j=1}^{x} |x_i - c_j|^2 \quad (9)$$

The utility matrix is decomposed into item and user factors using the ALS method for dealing with these issues. The item factors are then given as input to the clustering method. The clusters framed from the item factors provide a better relationship between the items when compared to the classical K-means clustering. In each round, each point is examined with centroid to find the closest cluster. So, the computational complexity is O(KN). Here k is a number of clusters and n is the number of points. But in a sparse dataset, the number of iterations to converge can be very large. The K-means converges in a fewer number of iterations when the points are represented in a low dimensional matrix using MF as shown in Figure 3.

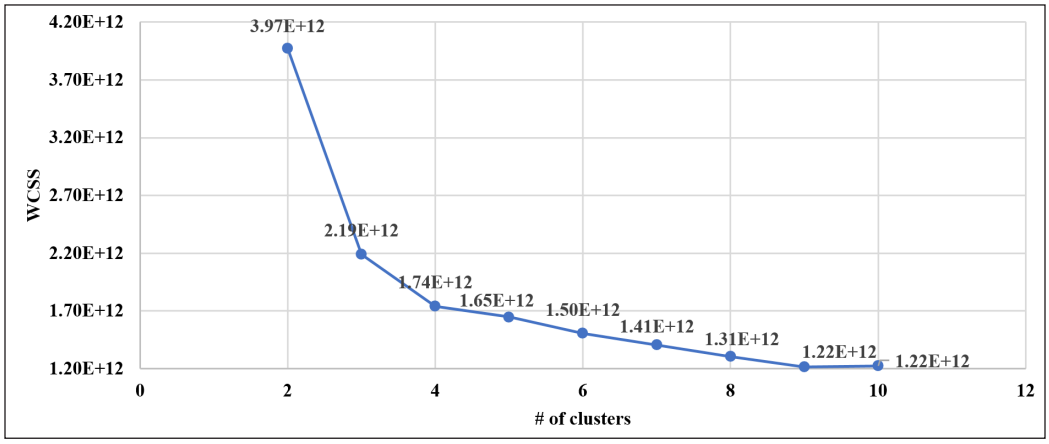

Figure 2. Number of clusters along With in Cluster Sum of Squared Errors (WCSS) in K-means for 1M dataset

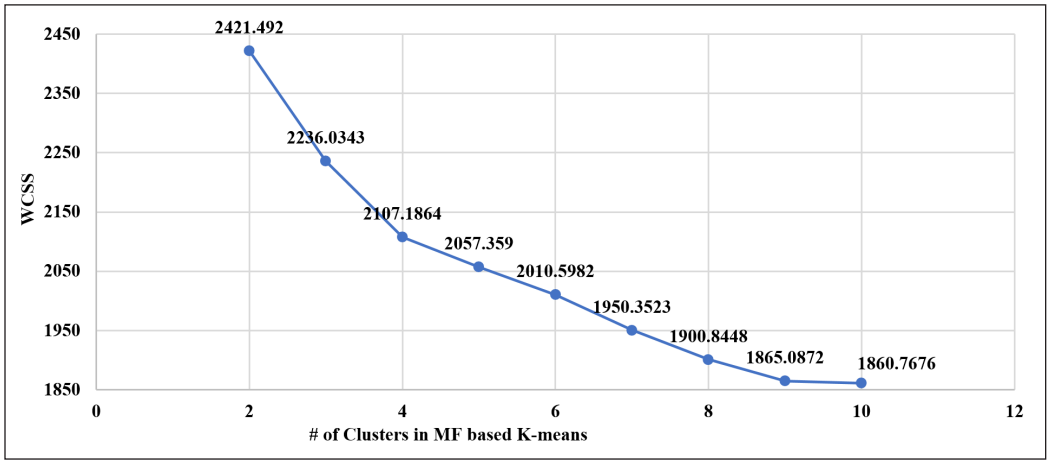

Figure 3. Number of clusters along With in Cluster Sum of Squared Errors (WCSS) in MF based K-means for 1M dataset

Lazy Collaborative Filtering with Dynamic Neighborhood

TACF utilizes the LCFDN (Suganeshwari et al., 2018) approach as its primal recommendation method. The clusters framed are meaningful but do not address the user preference drifts of the user. The LCFDN concentrates on improving the accuracy of RS by adopting time with two defined features α and δ. Recommendations generated based on time yielded better quality and accurate results.

Phase 1: Prediction Computation. A numerical value is formulated by computing the weighted average given in equation 10. Computation of similarity is done between items i and j is done by the isolation of ratings given by the users to both the items and statistical measures is applied. In this work the correlation ratio found is shown in equation 11. Using these similarity ratios for each item i ε I, a similar items list s is stored with k items with highest similarity ratios sim (i, j).

$$P_{u,i} = \frac{\sum_{\delta,\ n}\left(S_{i,n} * R_{u,n}\right.}{\sum_{\delta,\ n}\left|S_{i,j}\right|} \tag{10}$$

Phase 2: Recommendation Generation. For every active user u_a, predicted rating P(u, i) is calculated for each item i ε I not rated by the active user u using the equation (11). Recommendations are generated by computing the sum of the predicted rating related to the items that are similarly weighted by the similarity score and normalized by using the sum of the similarity values.

$$Sim(i,j) = \frac{\sum_{u\epsilon U_{i,j}}^{\alpha}\left(R_{u,i} - \bar{R}_i\right)\left(R_{u,j} - \bar{R}_J\right)}{\sqrt{\sum_{u\epsilon U_{i,j}}^{\alpha}\left(R_{u,i} - \bar{R}_i\right)^2}\sqrt{\sum_{u\epsilon U}^{\alpha}\left(R_{u,j} - \bar{R}_J\right)^2}} \tag{11}$$

The computational complexity of the TACF approach after clustering is O(KN) + $O(\alpha^2\delta)$. Here α is neighborhood size and δ is the recent transactions of the user. As the size of α is 20 the complexity is reduced to O(KN) + O(δ). This method is computationally efficient when compared to traditional K-means and IBCF method which is O(KN) + $O(n^2m)$. Here n is the number of items and m is the number of users.

Baseline Estimate

Ranking. In CF patterns or hidden feature can be extracted depending on the user rating behavior. The user may be biased in his rating depending upon their personal preferences and certain items may receive better ratings than the others. All these impacts were clearly captured by Koren (2009) as shown in equation 12 using baseline estimate.

$$u = \mu + b_i + b_u \tag{12}$$

Here u is the unknown rating, μ is the average rating for the dataset, b_i and b_u are the items and user bias calculated as in equation 13 and equation 14.

$$b_i = \frac{\sum_{u \epsilon N}(r_{u,i} - \mu)}{|\{u|(u,i)\ \varepsilon\ N\}|} \quad (13)$$

$$b_i = \frac{\sum_{u \epsilon N}(r_{u,i} - \mu)}{|\{u|(u,i)\ \varepsilon\ N\}|} \quad (14)$$

These parameters are added up to the top-k items and ranked again. These improvements are shown in the experiment section in MAP as TACF + Ranking.

Algorithm 1: Time Adaptive Cluster Method

Input: An active user u_a ε U,
R: The user-item rating matrix, Rank, numIterations,
nClusters, nIterations, nRuns, top-K items.

Output: The items with top-k highest rating $\{i_{r1}, i_{r2}, i_{r3}, \ldots, i_{rk}\}$

1: P ← 0, Q ← random initialize
2: repeat
3: **for** row u ← 1, m **do**
4: $p_u = (Q^T Q + \lambda I)^{-1} Q^T r_u$
5: **end for**
6: **for** column i ← 1, n **do**
7: $q_i = (P^T P + \lambda I)^{-1} P^T r_i$
8: **end for**
9: **until** max iterations
10: $K_p = \sum_{i=1}^{n} \sum_{j=1}^{c} u_{i,j}(p_{i,t})$
11: APPLY LCFDN algorithm on each User Cluster
12: $r' = \mu + b_i + b_u$
13: $r''_{u,i} = u + r'$
14: SORT IN DECREASING ORDER

Algorithm 1 that explains the pseudocode of time-aware scalable neighborhood consists of 3 phases: In the first phase, the rating matrix R, rank and number of iteration that the system that converges are taken as inputs and a matrix factorization model is generated. The matrix R is decomposed into user Factors and item Factors. Each item is represented in some latent factor. The item factor from ALS model is given as input to the time adaptive K-means clustering method. Items are segmented on the basis of the rating behavior of the user by incorporating time.

TACF has better scalability and high performance when compared to traditional methods as item similarity is computed only within the item clusters. The computational complexity is expensive for TACF method as it is a combination of two models. The model is created offline. Since items represent the rating behavior pattern, they produce highly relevant recommendations.

EXPERIMENT ANALYSIS

This section enlists the experiments conducted on the benchmark MovieLens (Harper, & Konstan, 2016) dataset which includes one million ratings from 6040 users on 3900 movies and one lakh ratings from 943 users and 1682 movies. The ratings are represented in numerical form (1~5) along with the rating timestamp. All experiments reported in this section were performed on the machines with Intel(R) Xeon(R)2 CPU 3.36GHz and 32GB RAM and were implemented using Spark. Precision @ k and Mean Square Error (MSE) has been used to evaluate the TACF.

Mean Square Error

The prediction quality of the CF methods is often measured using the statistical accuracy metric MSE (Pentreath, 2015). It is defined as the sum of the squared errors divided by the number of observations as given in equation 15. The lower MSE means high-quality predictions.

$$\mathrm{MSE} = \frac{\sum_{i,j \epsilon X} \left| r_{ij} - r'_{ij} \right|^2}{|X|} \qquad (15)$$

where r_{ij} is the actual rating given by user i for an item j, r'_{ij} is the predicted rating

Figure 4 compares the MSE value of ALS, UBCF, IBCF, TSCF, and TACF. MSE values of TACF is found to be much lower compared to other methods. Table 2 represents the ratings predicted by different algorithms for the random user for ml-100k. The predicted values of TACF is more desirable when compared to the other methods. Sparsity helps the K-means produces large deviations from the original ratings. ALS predicts ratings on the basis of the latent factor model. Hence, there are less deviation and lower MSE.

Figure 5 represents the MSE value for the ml-1m dataset. The inference from the figure is that the proposed method TACF excels ALS, UBCF, IBCF and TSCF method. Thus, the proposed method TACF has shown the ability to provide better prediction compared to other traditional models. Similarly, Table 3 represents the predicted value for each movie by random users for ml-1m datasets. There is a greater deviation in K-means and the proposed method TACF shows lower MSE when compared to other methods.

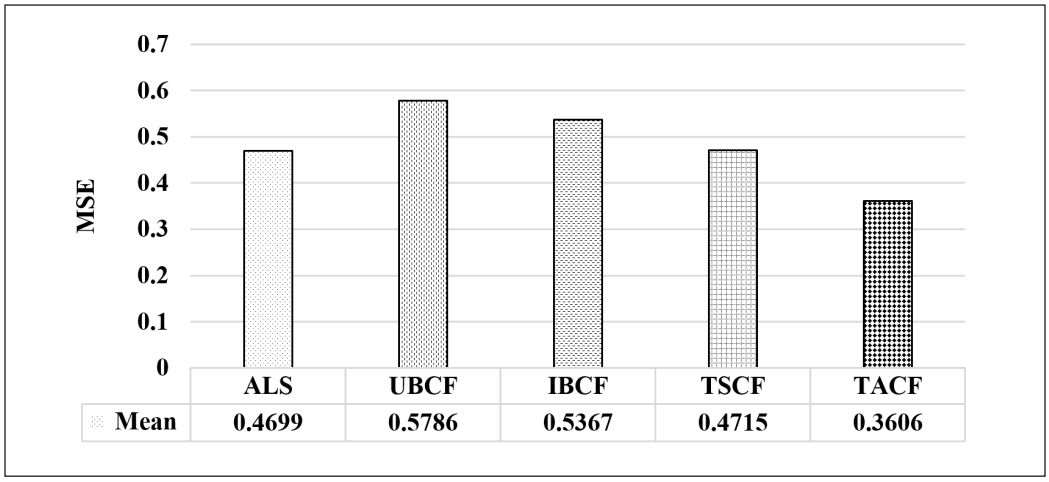

Figure 4. MSE for ml-100k

Table 2
The Predicted value for movies for ml-100k

Random Users	Rating	ALS	UBCF	IBCF	TSCF	TACF
u1	3	4.2455	2.9868	3.2312	3.5517	3.9646
u2	3	3.0646	3.9836	3.3425	3	3
u3	4	4.3988	3.6547	2.6543	3.5813	3.9905
u4	4	4.3993	3.1234	3.4587	3.9651	4.1532
u5	5	4.5889	3.6978	4.3	3.4777	4.7833

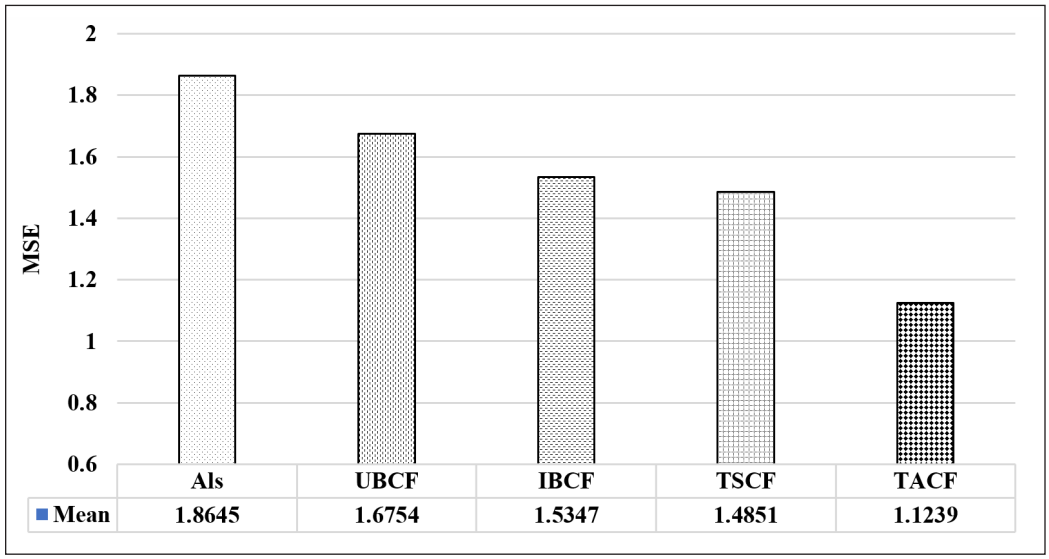

Figure 5. MSE for ml-1m

Table 3
The Predicted value for movies for ml-1M

Random Users	Rating	ALS	UBCF	IBCF	TSCF	TACF
u1	5	3.3576	3.789	3.546	3.6754	3.9687
u2	5	3.2735	4.099	4.2342	4.2034	4.2231
u3	4	3.7959	4.234	3.987	3.7687	3.8687
u4	5	3.746	4.167	4.342	4.407	4.608
u5	5	3.5607	3.774	4.234	4.2314	4.1299

The Mean Average Precision

It is the popular performance measure used for measuring the significance of the top-k items. It is the mean of the average precisions at K (Pentreath, 2015) as indicated in equation 16. Higher MAP values predict a higher quality of the recommendation system.

$$\mathrm{MAP} = \frac{\sum_{q \epsilon Q} \mathrm{AP}(q)}{|Q|} \qquad (16)$$

AP is average precision for each query q, |Q| is mean of the queries.

Figure 6 shows the comparison of the MAP values of the ALS, UBCF, IBCF, TSCF, TACF and TACF+Baseline method for ml-100k. TACF method exceeds the UBCF by 65.95%, 69.24%, 89.95%, IBCF by 32.72%, 31.91% and 34.86% and TSCF by 23.73%, 25.63% and 30.24% at top-3, top-5 and top-7 respectively. The percentages calculated and given in Table 4 are meant for a clear illustration.

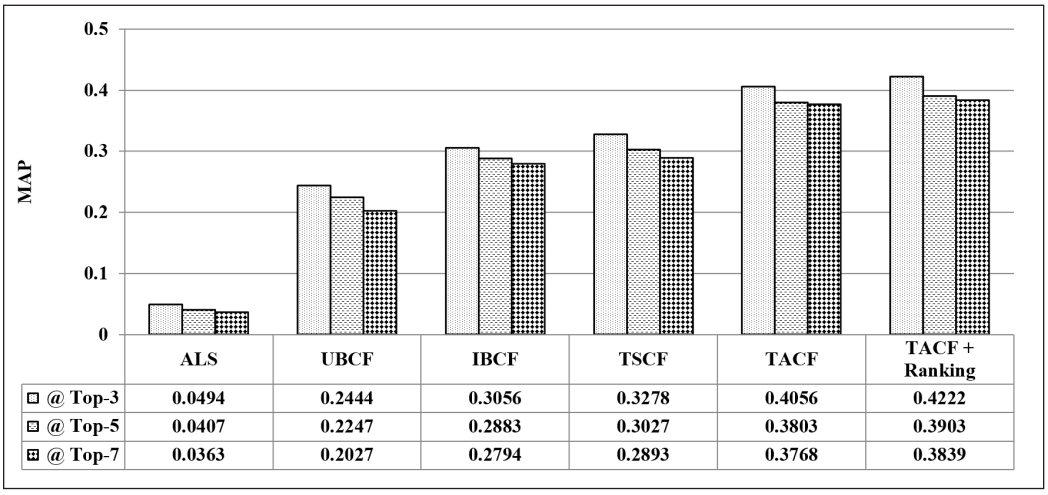

Figure 6. MAP for ml-100k

Table 4
The Percentage Values that excels other methods in MAP for ml-100k

MAP	Top-3	Top-5	Top-7
TACF/ UBCF	65.95 %	69.24 %	85.89 %
TACF/ IBCF	32.72 %	31.91 %	34.86 %
TACF/ TSCF	23.73 %	25.63 %	30.24 %

Figure 7 displays the MAP values for the ml-1M dataset. TACF exceeds UBCF by 84.62%, 50.57% and 37.02%, IBCF by 63.62%, 46.19% and 41.78%, TSCF by 38.44%, 38.53% and 24.36% respectively at top-3, top-5 and top-7. The MAP values of TACF are better compared to other methods. The MAP value decreases with an increase in the number of K.

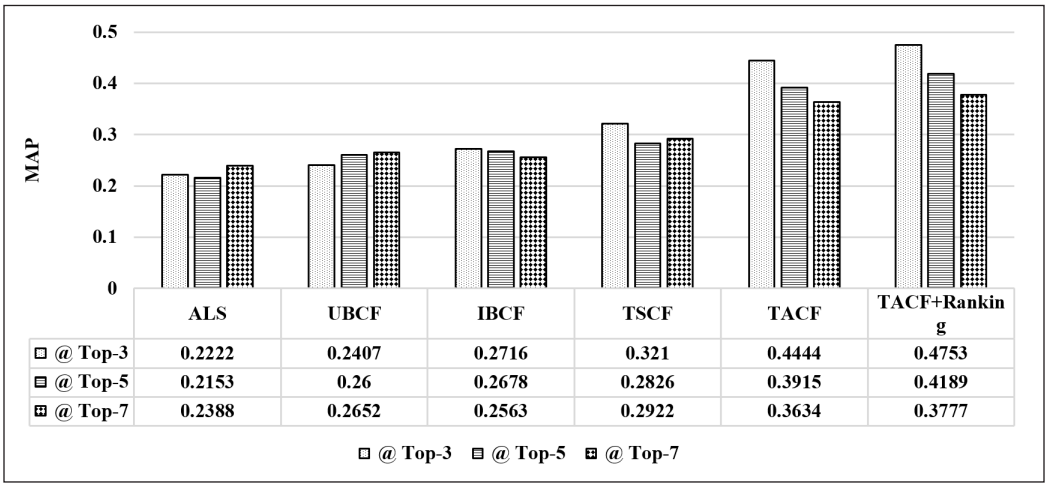

Figure 7. MAP for ml-1m

Table 5 is a clear illustration of MAP values for the ml-1m dataset. The TACF perform better than the other state-of-art methods ALS, UBCF, IBCF and TSCF in MSE and MAP. Thus, the top-k items recommended by the proposed method would satisfy the user's expectation with the recommendation for the items based on the current context of the user.

Table 5
The Percentage Values that excels other methods in MAP for ml-1m

MAP	Top-3	Top-5	Top-7
TACF/ UBCF	84.62 %	50.57 %	37.02 %
TACF/ IBCF	63.62 %	46.19 %	38.53 %
TACF/ TSCF	38.44 %	38.53 %	24.36 %

Execution Time

Figure 8 and Figure 9 show the execution time of different algorithms for ml-100k and ml-1M dataset. The proposed TACF method is computationally faster than traditional methods. The performance of the proposed system will not degrade even if there is a large number of ratings in the rating matrix, as this system considers only 2% of the recent transactions for each user.

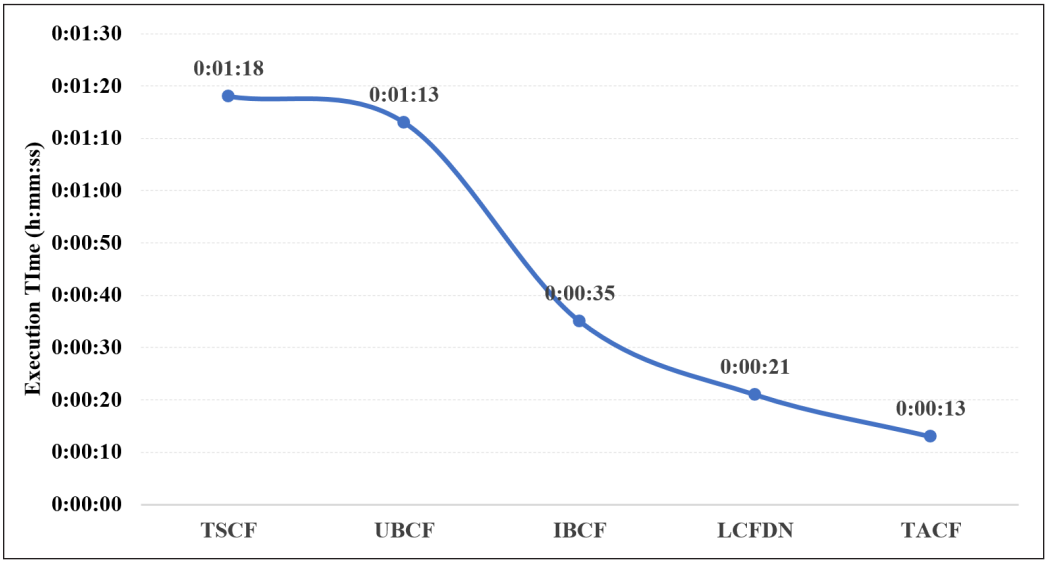

Figure 8. Execution time for generating Recommendation for ml-100k

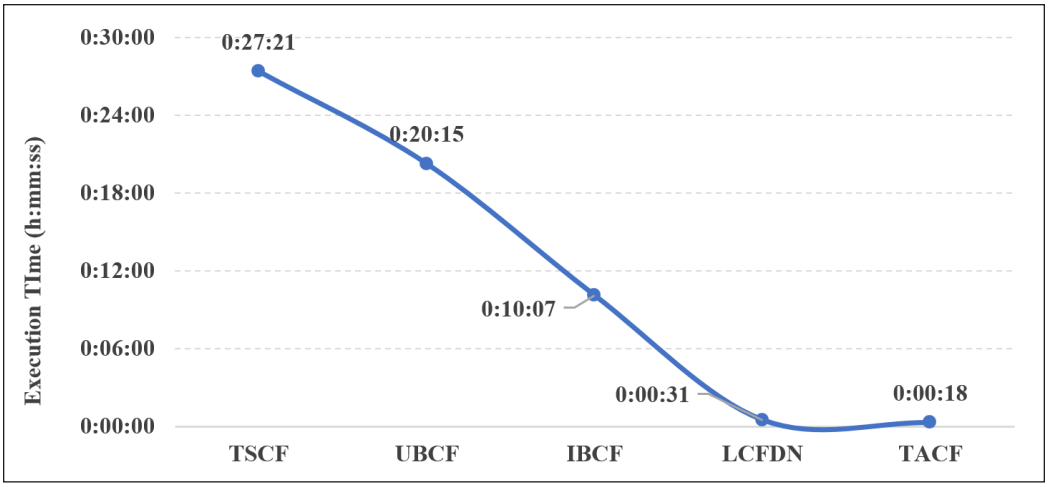

Figure 9. Execution time for generating Recommendation for ml-1M

CONCLUSION

This paper presents a time-adaptive recommendation approach on item subgroups that exploits the recency of the user's transaction to address the data sparsity problem. The items are clustered based on the item latent factors and time is used as additional information in generating recommendation on each subgroup. Experimental results show that using time in item subgroups is a promising way to improve the top-k recommendation performance when compared with the other state-of-the-art collaborative filtering methods. Future work can be extended by forming better user or item subgroups by adding additional information based on context.

ACKNOWLEDGEMENT

We are immensely grateful to the anonymous reviewers for their valuable comments and suggestions to improve the quality of the article.

REFERENCES

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 6*(17), 734-749.

Ba, Q., Li, X., & Bai, Z. (2013, May 23-25). Clustering collaborative filtering recommendation system based on SVD algorithm. In *2013 4th IEEE International Conference on Software Engineering and Service Science (ICSESS)* (pp. 963-967). Beijing, China.

Bell, R. M., & Koren, Y. (2007). Lessons from the Netflix prize challenge. *Acm Sigkdd Explorations Newsletter*, *9*(2), 75-79.

Billsus, D., & Pazzani, M. J. (1998, July 24-27). Learning Collaborative Information Filters. In *Proceedings of the 15th International Conference on Machine Learning, Madison, WI,* (Vol. 98, pp. 46-54). Morgan Kaufmann, San Francisco.

Chen, J., Fang, J., Liu, W., Tang, T., Chen, X., & Yang, C. (2017, May 29-June 2). Efficient and portable ALS matrix factorization for recommender systems. In *IEEE International conference on Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (pp. 409-418). ake Buena Vista, FL, USA.

DeCoste, D. (2006, June 25-29). Collaborative prediction using ensembles of maximum margin matrix factorizations. In *Proceedings of the 23rd international conference on Machine learning* (pp. 249-256). Pittsburgh, Pennsylvania, USA.

Desrosiers, C., & Karypis, G. (2011). A comprehensive survey of neighborhood-based recommendation methods. In F. Ricci, L. Rokach, B. Shapira & P. Kantor (Eds.), *Recommender systems handbook* (pp. 107-144). Boston, Massachusetts: Springer.

Ding, Y., & Li, X. (2005, October 31-November 5). Time weight collaborative filtering. In *Proceedings of the 14th ACM international conference on information and knowledge management* (pp. 485-492). Bremen, Germany.

Guo, G., Zhang, J., & Yorke-Smith, N. (2015). Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems. *Knowledge-Based Systems*, *74*, 14-27.

Harper, F. M., & Konstan, J. A. (2016). The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, *5*(4), 1-19.

Heckerman, D., Chickering, D. M., Meek, C., Rounthwaite, R., & Kadie, C. (2000). Dependency networks for inference, collaborative filtering, and data visualization. *Journal of Machine Learning Research*, *1*(Oct), 49-75.

Herlocker, J., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, *5*(4), 287-310.

Hu, Y., Peng, Q., Hu, X., & Yang, R. (2015). Time-aware and data sparsity tolerant web service recommendation based on improved collaborative filtering. *IEEE Transactions on Services Computing, 8*(5), 782-794.

Jiang, S., Qian, X., Mei, T., & Fu, Y. (2016). Personalized travel sequence recommendation on multi-source big social media. *IEEE Transactions on Big Data*, *2*(1), 43-56.

Koren, Y. (2009, June 28-July 01). Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 447-456). Paris, France.

Li, Q., & Kim, B. M. (2003, October 13-17). Clustering approach for hybrid recommender system. In *Proceedings IEEE/WIC International Conference on Web Intelligence (WI 2003)* (pp. 33-38). Halifax, NS, Canada.

Luo, X., Liu, H., Gou, G., Xia, Y., & Zhu, Q. (2012). A parallel matrix factorization based recommender by alternating stochastic gradient decent. *Engineering Applications of Artificial Intelligence*, *25*(7), 1403-1412.

Ma, X., Lu, H., Gan, Z., & Zhao, Q. (2016). An exploration of improving prediction accuracy by constructing a multi-type clustering based recommendation framework. *Neurocomputing*, *191*, 388-397.

Meng, S., Dou, W., Zhang, X., & Chen, J. (2014). KASR: a keyword-aware service recommendation method on MapReduce for big data applications. *IEEE Transactions on Parallel and Distributed Systems*, *25*(12), 3221-3231.

Miyahara, K., & Pazzani, M. J. (2000, August). Collaborative filtering with the simple Bayesian classifier. In R. Mizoguchi & J. Slaney (Eds.), *PRICAI 2000 Topics in Artificial Intelligence-PRICAI 2000. Lecture Notes in Computer Science* (pp. 679-689). Heidelberg, Germany: Springer.

O'Connor, M., & Herlocker, J. (1999, August). Clustering items for collaborative filtering. In *Proceedings of the ACM SIGIR workshop on recommender systems* (Vol. 128). Berkeley, California.

Panniello, U., Tuzhilin, A., & Gorgoglione, M. (2014). Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction*, *24*(1-2), 35-65.

Park, H., Jung, J., & Kang, U. (2017, December 11-14). A comparative study of matrix factorization and random walk with restart in recommender systems. In *Big Data (Big Data), 2017 IEEE International Conference on* (pp. 756-765). Boston, Massachusetts.

Pentreath, N. (2015). Building a Recommendation Engine with Spark. In N. Pentreath (Ed.), *Machine Learning with Spark* (pp. 83-116). Birmingham UK: Packt Publishing Ltd.

Ren, Y., Li, G., Zhang, J., & Zhou, W. (2013). Lazy collaborative filtering for data sets with missing values. *IEEE transactions on cybernetics*, *43*(6), 1822-1834.

Sarwar, B. M., Karypis, G., Konstan, J., & Riedl, J. (2002a, December 27-28). Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering. In *Proceedings of the Fifth International Conference on Computer and Information Technology* (Vol. 1, pp. 291-324). Dhaka, Bangladesh.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2002b, December 27-28). Incremental singular value decomposition algorithms for highly scalable recommender systems. In *Fifth International Conference on Computer and Information Science* (pp. 27-28). Dhaka, Bangladesh.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295). New York, NY, USA: Association for Computing Machinery.

Sarwat, M., Levandoski, J. J., Eldawy, A., & Mokbel, M. F. (2014). LARS*: An efficient and scalable location-aware recommender system. *IEEE Transactions on Knowledge and Data Engineering*, *26*(6), 1384-1399.

Su, X., & Khoshgoftaar, T. M. (2006, November 13-15). Collaborative filtering for multi-class data using belief nets algorithms. In *18th IEEE International Conference on Tools with Artificial Intelligence, 2006 (ICTAI'06)* (pp. 497-504). Arlington, VA, USA.

Suganeshwari, G., & Ibrahim, S. P. S. (2016). A survey on collaborative filtering based recommendation system. In V. Vijayakumar & V. Neelanarayanan (Eds.), *Proceedings of the 3rd International Symposium on Big Data and Cloud Computing Challenges (ISBCC–16')* (pp. 503-518). Cham, Switzerland: Springer.

Suganeshwari, G., Ibrahim, S. P. S., & Li, G. (2018). Lazy collaborative filtering with dynamic neighborhoods. *Information discovery and delivery*, *46*(2), 95-109.

Sun, L., Michael, E. I., Wang, S., & Li, Y. (2016, December 15-18). A Time-Sensitive Collaborative Filtering Model in Recommendation Systems. In *IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)* (pp. 340-344). Chengdu, China.

Takács, G., Pilászy, I., Németh, B., & Tikk, D. (2008, December 15-19). Investigation of various matrix factorization methods for large recommender systems. In *IEEE International Conference on Data Mining Workshops, 2008 (ICDMW'08)* (pp. 553-562). Pisa, Italy.

Tan, S., Bu, J., Chen, C., Xu, B., Wang, C., & He, X. (2011). Using rich social media information for music recommendation via hypergraph model. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, *7*(1), 1-22.

West, J. D., Wesley-Smith, I., & Bergstrom, C. T. (2016). A recommendation system based on hierarchical clustering of an article-level citation network. *IEEE Transactions on Big Data*, *2*(2), 113-123.

Xue, G. R., Lin, C., Yang, Q., Xi, W., Zeng, H. J., Yu, Y., & Chen, Z. (2005, August 15-19). Scalable collaborative filtering using cluster-based smoothing. *In Proceedings of the 28th Annual International ACM SIGIR conference on Research and Development in Information Retrieval* (pp. 114-121). New York, NY, USA.

Yu, H. F., Hsieh, C. J., Si, S., & Dhillon, I. S. (2014). Parallel matrix factorization for recommender systems. *Knowledge and Information Systems*, *41*(3), 793-819.