Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

Markov Chain Recommendation System (MCRS)

[1]Ahmed Adam Ahmed, [2]Naomie Salim

[1]College of Computer Science and Information Technology, Sudan University of Science and Technology
[2]Faculty of Computing, University Technology Malaysia

Abstract: **Web applications use recommendation techniques that are based on users' preferences for items to recommend interesting items to the active user. Users' preferences can be their activities on these items such as: rate, view, etc. Collaborative filtering (CF) is the most successfully used recommendation system recently; it is based on the similarities between users and the similarities between items. However, users' opinions and items' popularities vary with time. These variations decrease the recommendation accuracy. Many researchers study ways of using Morkov model in recommendation systems; however, they do not consider the time factor in their techniques. In this paper, we propose a new recommendation system that is based on Morkov model, and it considers the time factor. We compare our new technique with the conventional CF recommendation system for the evaluation. We conduct the experiments using MovieLens dataset. The evaluation is done by using precision-recall, and mean absolute error. The result illustrates that our new recommendation system outperforms the conventional CF recommendation system.**

Keywords: **Markov chain, Collaborative filtering, Recommender Systems, Items popularity, Items weights.**

I. INTRODUCTION

The internet providers arrive users at anywhere in the glob these days, and many web applications provide to their users the interaction between each other's[1][2]. Web applications as well as their users face the problem of information overload, because users need to access few interesting items out of millions. Recommendation systems are based on users preferences for items e.g. view, rate; they are used by web application to cope with this problem[3][4][5]. Recommendation systems are categorized into different types such as Collaborative filtering, Content-based [6], Knowledge-based, and Hybrid between two techniques or more[7][8].

Collaborative filtering (CF) is one of the most successfully used techniques recently[9]. It can be memory-based or model-based. Memory-based techniques use user-item databases to predict users' preferences for items by using the similar users preferences, because everyone is a member of a group with similar interests[10][4]. The active user's preferences on unknown items can be produced by memory-based techniques using his nearest neighbors' preferences. Then, the top N most frequent items are listed as recommendations. Model-based techniques use history data, that is taken from users' preferences on items, to formulate the model that can be used later in the recommendation or the prediction processes. Many recommendation techniques use users' ratings for items which can be categorized as follows:

- Scalar: In this case the rating can be numeric stars like (1, 2, 3, 4, or 5) [4]. It can be ordinary rating like very good, good, normal, bad, and very bad.
- Binary: This type can be represented as like or unlike, yes or no, and so on.
- Unary: It can be taken from users' activities; for instance, if a user frequently purchases an item; he most probably likes it.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

This research introduces a new technique using Markov model to recommend items to users. Our new Markov Model use unary users' ratings for items to formulate an initial vector and a transition matrix that used to recommend interesting items to the active user.

The rest of the research introduces the general concepts of CF recommendation system in Section II. In Section III, we discuss ways of using Markov chain in recommendation systems. The limitation of the conventional CFRSs is illustrated in Section 0. The motivation of new recommendation system is given in Section V. In Section VI, we illustrate our new technique, Markov chain recommendation system MCRS. The evaluation is carried out in Section VII. In Section VIII, we analyze the experimental results. Finally, we provide a brief discussion and conclusion in Section IX.

II. COLLABORATIVE FILTERING CF CONCEPTS

Schafer et al.[11] use CF to predict the active user ratings for the unknown items then to recommend interesting items to him. CF is based on users-items matrix, as it represented in Table 1. Users rate on a different subset of items that can be represented as vectors $U_i = \{r_{(i,j)} \mid j=1, 2,\ldots,n\}$, here U_i is the active user, $r_{(i,j)}$ is the rating of the user U_i on item I_j and n is the number of items that accessed by the user U_i. This means user U_i rates on small set of m items.

Table 1, represents the rating of five users on four items. The numbers from 1 to 5 represent users' ratings and the absent cells represent items that can be predicted. For normalization, they can be rated as zero. These ratings can be used to calculate the similarity between users or items.

Table 1 Rating of users on items

	I_1	I_2	I_3	I_4
U_1	2	2	4	5
U_2	1	1	3	0
U_3	2	0	3	4
U_4	4	1	4	3
U_5	5	3	5	4

Cosine similarity [12] is one of the most popular similarity measure. Consider, the set of users is $U=\{U_i: i=1,2,3,...,m\}$ and m is the number of all users; then, the similarity between the user U_x and the user U_y is $S(U_x, U_y)$, and it can be calculated as follow:

$$S(U_x, U_y) = \frac{[\overline{U_x}].[\overline{U_y}]}{\|U_x\|.\|U_y\|} \quad (1)$$

Here, $\overline{U_x}$ and $\overline{U_y}$ are the vectors of ratings of the users U_x and U_y for the n items respectively. E.g. The similarity between U_4 and U_5, as represented inTable1, is given by equation (2):

$$S(U_4,U_5) = \frac{20+3+20+12}{\sqrt[2]{25+9+25+16}.\sqrt[2]{16+1+16+9}} = 0.98 \quad (2)$$

The k nearest neighbor algorithm [13] is one of the most used techniques to predict users' ratings on items and to recommend a list of interesting items to the active user. Schafer et al., for MovieLens Recommendation systems, use average ratings of neighbors of user U_u on item I_i to predict his ratings as in equation (3):

$$\text{Prediction}_{(u,i)} = \overline{r_u} + \frac{\sum_{(\text{n similar users to u})} \text{sim}(u,n) * (r_{ni} - \overline{r_n})}{\sum_{(\text{n similar users to u})} \text{sim}(u,n)} \quad (3)$$

Here $\overline{r_u}$ is the average rating of user U_u on the neighbor of item I_i and r_{ni} is the rating of the nth neighbor of user U_u on item I_i and $\overline{r_n}$ is the average rating of the nth neighbor of user U_u on neighbors of item I_i.

CF techniques are based on the similarities between users and the similarities between items. However, users' opinions and items popularities vary with time, and the similarities calculation do not consider the time factor. Our new technique considers the time factor by using the feature "accessing items by the same user in the same period of time". More details about our new technique will be discussed later in this paper.

III. USING MARKOV CHAIN IN RECOMMENDATION SYSTEMS

Markov proposed that the outcome of a given experiment can affect the outcome of the next experiment [14], [15]. This type of process is called Markov chain.

Markov chain contains three components. The first one is the states, the second one is the process function to move from one state to another, and the last one is the start state(s) [16]–[19]. If we have a set of states S={s1,s2,s3,...,sn} then the process starts from one of these states, and it successively moves an step to another state. If the chain is currently at state si then it moves to state sj in the next step with a probability denoted by p(i,j) and this probability does not depend on which states the chain was in before the current state. The probabilities are called transition probabilities. The process can remain in the state, and this occurs with probability p(i,i). An initial probabilities are given to the starting state. Usually, this is done by specifying a particular state as the starting state.

Markov chain model is used in recommendations system. Shani et al. [20] use Markov Chain model in the recommendation processes. They propose an MDP-Based (Markov Decision Process) Recommender System. The states in their model represent the relevant information about the active user. Their technique considers only the most frequents sequences of 5 items. The transition matrix in their technique can be formulated by the probabilities of accessing a set of items followed by an item. The initial vector is estimated using user's data and users with no data are considered that they access a missing items. MDP-Based Recommender System has these weaknesses:

- It does not consider the time factor in the recommendation process. While, items popularities normally vary with time.
- It face the sparsity problem because web applications provide to their users millions of items; and individual users only access tens from them.
- Shani solution considers sequences of only five items, with no consideration of the time factor. However, users can access more than five items in one session, and the time factor must be considered.
- They use the feature of viewing items in sequential order i.e. viewing the item A leads to viewing the item B. However, they miss that viewing the item B can also leads to viewing the item A.

All the mentioned draw backs are faced when applying MDP-Based Recommender System in web applications. Moreover, social media websites add new features that flood more information in different domains at the same time in the same application. For example, Facebook network contains users, companies, news agents, TV channels, schools, universities, market shops etc. The information that come from all these sources can be used in the recommendation process. Therefore, Shani model cannot be used to handle all this amount of information. Wu et al. [21] proposed Personalized Markov Embedding (PME) to recommend the next song for the active user, and they embed users and songs into Euclidean space. The Euclidean distance between songs and users represent their relationship that is used to generate the next song. They evaluate their new technique on real dataset from ihou.com, and the results clearly demonstrate the effectiveness of PME.

Rendle et al. [22] presented the method that bring together both matrix factorization (MF) and Markov chains (MC). Their method is based on personalized transition graphs over underlying Markov chains. Every user has a transition matrix i.e. all users generate a transition cube. Empirically, the FPMC model outperforms both the matrix factorization and the non-personalized MC model. However, millions of items are available to users, and individual users access only tens of these items. Rendle et. al solution generates a transition cube that used in the recommendation i.e. this solution suffers from sparsity.

Eirinaki et al. [23] present a hybrid probabilistic predictive model based on Markov models using link analysis methods. They propose the use of a PageRank-style algorithm to generate suggestions for websites according to their importance. Empirically, the results show that this approach outperforms the pure usage-based approaches.

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

Huang et. al. [24] propose the use of the learning sequence recommendation system (LSRS) using recommendations based on group-learning paths to recommend web pages to users. They use Markov chain model, which is a probability transition model, and employ an entropy-based approach to assist this model in discovering one or more recommended learning paths through the course material. Their study identified benefits for teachers, providing them with ideas and tools needed to design better online courses.

Fouss et. al. [25] propose Multi Agent System (MAS) that consider each agent as a node and the interaction between any two agents as a link. They use Markov chain model to suggest movies that people should watch based upon what they watched in the past. Experimental results show that their approach outperforms all the other methods.

Many studies use Markov Chain model in recommendation systems, and they consider the sequence events of users' interaction as states of Markov chain. However, these studies faced by sparsity. Recently the amount of information increases exponentially; web applications provide millions of items to their users, and millions of users introduce their opinions that can be used in recommendation systems. Moreover, social media provide a great opportunity to enhance RSs. All these events and activities of users, when they used in the Web, depend on time. The time factor can then be used to predict trends in social media. Social media features and trends analysis can be used to enhance recommendation systems.

We use Markov model to design a new recommendation systems that combine social media information and the time factor and use them as one data source to recommend items to users. All the mentioned techniques based on accessing frequencies sequences of items that followed by an item. Our new proposed solution is based on the new idea that use the feature "accessing items by the same users in the same periods of time".

Table 2 summarizes ways of using Markov chain model in recommendation systems; it gives the used techniques, ways of generating the transition matrix, ways of evaluating the models, and comments.

Table 2 Ways of using Markov chain in recommendation systems

The Reference	The technique	The Transition matrix	Evaluation	Comments
Shani et. al (2005) [20]	An MDP-Based Recommender System.	The probability of accessing an item by the user after accessing a sequence of items.	Suffer from the sparsity problem. The sequence of 5 items only is used.	Million of items are available. Users access of sequence of hundreds items
(Wu et al. (2013) [21]	Personalized Markov Embedding (PME)	The Euclidean distance between songs and users represent their relationship that used to generate the next song	the PME effects positively in RSs.	No consideration of the time factor.
Rendle et al. (2010) [22]	Both matrix factorization (MF) and Markov chains (MC).	Any user has a transition matrix i.e. all users generate a transition cube.	The FPMC effects positively in RSs.	No consideration of the time factor.
Eirinaki et al. (2005) [23]	Page Rank-style algorithm	Markov models using link analysis methods.	outperforms the pure usage-based approaches	No consideration of the time factor.
Huang et al. (2009) [24]	The learning sequence recommendation system (LSRS).	probability transition model	This study identified benefits for teachers.	No consideration of the time factor.
Fouss et al. (2001) [25]	Multi Agent System (MAS).	The suggestion based upon what people watched in the past.	It outperforms all the other methods.	No consideration of the time factor.

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

IV. THE LIMITATION OF THE CONVENTIONAL CFRSs

Collaborative filtering techniques are based on the similarities between users or items [26]. The K nearest neighbor algorithms are used to generate the suggestions for items to users using these similarities. On the other hand, the conventional Markov model techniques are based the sequences of accessing items by users while many users do not work to access items in sequences [20]. The limitation of these techniques can be considered as follows:

A. The first limitation:

The similarity between two users depends on their accessed items [4], [10]. If the two users access the same subset of items; then, they are similar. If they access different items and share the accessing of others; then, they are partially similar. Otherwise, if they access different items and don't share the accessing of any others; then, they are not similar. However, users' opinions vary with the time. Any user can be interested to access items in the earlier periods of time; then, he may change his opinions and accesses different kind of items later. He might be no longer interested in the first ones.

If we look to users' accessed items in the long term; we find that no relationship can be used to link between items; this means, users' opinions cannot be used in this relationship.

Example:

For one year in the first month, if the user u1 has viewed and interested in movies (A,B,C,D), the user u2 has viewed and interested in movies (A,B,C,D) and the user u3 has viewed and interested in movies (F,G,D,J). In the same year, users' opinions might be changed. Then consider in the tenth month, the user u1 has viewed and interested by movies (F,G,D,J), the user u2 has viewed and interested by movies (V,N,M,K), and the user u3 has viewed and interested by movies (Q,R,T,Y).

It clear that users (u1 and u2) are (100%) similar to each other in the first month, and they are not similar in the tenth. In general, if we look to users in long term, u1 and u2 are partially similar the same as u1 and u3.

This means users' similarities are affected positively or negatively by the time factor. The limitation of Collaborative filtering techniques is the lack of using the time factor in users' similarities calculation.

On the other hand, anyone of these two users only accesses tens of items out of millions; this means, CF techniques are from the sparsity problem.

B. The second limitation:

The similarities between items (e.g. movies, photos) depend on users rating for these items [27] [28] [29]. Any item might be accessed (e.g. viewed, marked as like) by users of different opinions. New items might be interesting for all users that can access some of them. Few users access these new items in short term. However, an items similarity is based on users' preferences for items. This means, the approaches that are based on the similarity between items can produce inaccurate lists of recommendations; many of the new items might not be recommended.

C. The third limitation:

Limitations in ways of using the time factor in recommendation processes. Yi Ding and Xue Li [30] propose the time function f(t). The time interval is divided to t periods and the values of the function weights lies in the range (0,1) i.e. while old ratings are weighted by smaller weights, the recent users' preferences are weighted by the higher weights. Wang et al. [31] introduce 'Temporal Summaries' to invest the time factor in the recommendation process. Kostas et al. [32] propose a framework for time-aware recommendations that improve the recommender accuracy. However, their solutions separate between events of accessing items and the time. Values of their function f(t) lay between 0 and 1. The value of f(t) intend to zero in the first period and intend to one in the recent periods; moreover, the value of f(t) is equal for all items at any point of time. However, these values must be distributed randomly according to items popularities, and they must be different from any item to other because item popularities vary with time and depending on users' preferences for it.

D. The last limitation:

Markov chain recommendation techniques are based on sequences of accessing items [20], [16]–[19]. They aim to predict the item that follows a sequence of items. . Shani et al. [20] propose an MDP-Based (Markov Decision Process)

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

Recommender System. The states in their model represent the relevant information about the active user. Their technique considers only the most frequents sequences of k items , and they have k=5. The law order needs user to access less number of items. The law orders violate the accuracy as users can access more items. The high order result in better accuracy, but they increase the complexity of the application.

These limitations can be fixed by using new technique that considers the variation of users' opinions and items popularities with time.

V. THE MOTIVATION OF NEW RECOMMENDATION TECHNIQUES

Recommendation systems have been used by many web applications to ease their usage and to generate suggestion for items to users. The most successfully used techniques are collaborative filtering recommendation systems. However, these techniques have some limitations as mentioned in Section 0. New techniques are needed to address these limitations as follows:

- The new technique can be used to solve the first limitation and consider users' accessed items per session or period of time. These techniques look to users' preferences in short terms. This means the new techniques consider the variation of users' opinions with time.
- This technique can be used to solve the second limitation using the same idea. Here, we consider items that have been accessed by the same user in the same period of time.
- The new techniques can be used to solve the third limitation. They consider the time when users have accessed items and the variation of users' opinions and item popularities.
- The new techniques can be used to solve the last mentioned limitation; since, it don't considers the sequence of accessing items and only look to items that accessed by the same user in the same session or period of time.

VI. MARKOV CHAIN RECOMMENDATION SYSTEM (MCRS)

Users' access (e.g. view) items (e.g. movies) in web applications. The number of the provided items by websites to their users is increasing. Recommendation systems generate suggestion for items that might be interested to users. Initially, any user might choose a random item to access. In this case, the item can be interested by the user or not. He might access the most popular item. In this case the web site needs to use a suitable tool to suggest the most popular item. The user can use search engine to find the interested one. After the active user have accessed his first interested item, our new proposed technique starts working to recommend items to him according to his previous accessed item. The proposed technique aims to recommend items that have been accessed by all users with the first item.

For example:

if the first accessed item by the active user is item A; then the basic MCRS suggests the most accessed items by all users with respect to item A. Hence, we retrieve sessions of all users that access item A to calculate the probability of accessing items with it.

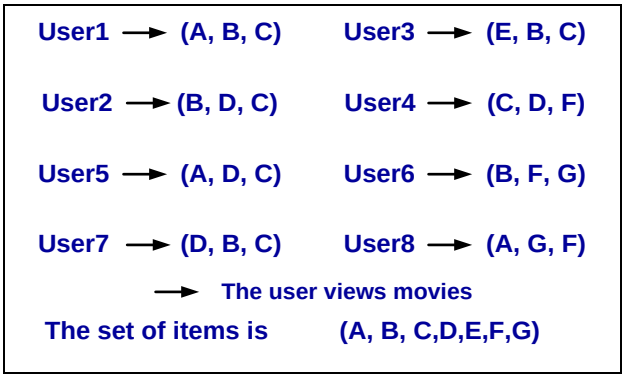

Figure 1 The viewed movies by all users with the movie A

Figure 1 illustrates that three users have accessed item A; and the accessed items with item A are (B,C,D,F,G).

The statistic of accessing items with item A is given as follows:

A = 3, B =1, C=2, D=1, E=0, F=1,G=1 . All items are accessed 9 times.

The probabilities of accessing items with item A can be given as follows:

P = (3/9, 1/9, 2/9, 1/9, 0, 1/9, 1/9).

The most accessed item with item A is item C. If the user access two items then the proposed recommendation system suggests items to him according to items A and C.

A. Probability of accessing items by the same user in the same session (PASS):

In this paper, we introduce a new technique to generate new relations that can link between items. We use the relation between items that accessed by all users' to calculate, the (Probability of accessing items by the same user in the same session (PASS)). The user U normally accesses a list of items in one session. We consider all sessions of all users to calculate the vectors of accessing all items with item j in the same session,

$$S_{(ji)} = \{r_{ji}: i, j = (1,2,3, \dots, n)\}$$

where n is the number of all items r_{ji} is the explicit rating for item i that accessed by all users with item j in the same session. If the user U accesses items (i and j) in the same session, then the rating $r_{ji} = 1$ otherwise it's $r_{ji} = 0$. Our goal is to calculate the vector of accessing all items with item j in the same session, where j=1,2,3,...,n.

$$\boldsymbol{SS}_{(ji)} = \{\sum_{(\boldsymbol{All\ usrs'sessions\ \ that\ contains\ item\ j})}^{\boldsymbol{All\ users'sessions}} (\boldsymbol{r_{ji}}) : \text{i=1,2,3,...,n}\} \qquad (4)$$

The probabilities $P_{(ji)}$ of accessing all items with item j can be given as follows:

$$P_{(ji)} = \{p_{(ji)} = \frac{ss_{(ji)}}{\sum_{i=1}^{n} ss_{(ji)}} : j, i = 1,2,3, \dots, n\} \qquad (5)$$

$$P_{(ji)} = \begin{pmatrix} p_{(11)} & p_{(12)} & p_{(13)} & & p_{(1n)} \\ p_{(21)} & p_{(22)} & p_{(23)} & \cdots & p_{(2n)} \\ p_{(31)} & p_{(32)} & p_{(33)} & & p_{(3n)} \\ & \vdots & & \ddots & \vdots \\ p_{(n1)} & p_{(n2)} & p_{(n3)} & \cdots & p_{(nn)} \end{pmatrix} \qquad (6)$$

$$\sum_{i=1}^{n} p_{(ji)} = 1 \qquad (7)$$

Our new technique solves the problem of the sparsity since it considers all sessions of all users i.e. it guarantees the probabilities of accessing all items with any item. Any user can sit for several sessions in different periods of time i.e. the new technique ties events of user's activities to the time of the session.

B. The basic MCRS:

The main components of Markov Chain Recommendation System are the initial vector and the transition matrix. To generate the initial vector, we need to understand the active user's vector.

a. The active's user:

The active user's vector is the target of the recommendation system. It represents all items that have been accessed by all users, which can be divided into two subsets. The first subset contains items that accessed by the active user which can be used to recommend items from the other subset that are not accessed by him. The first set "set-A" contains s items.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

$$\text{set-A}=\{i_j=1: j=1,2,3 \ldots s \text{ for } 1<=s<=n\} \quad (8)$$

here s is the number of accessed items by the active user and n is the number of all items. The second set set-B contains (n-s) items. set-B=$\{i_z: z=1,2,3, \ldots, (n-s)\}$. The active user vector is user-vector =(set-A)∪(set-B). Normally, items are distributed and items of the active user's vector are not sorted.

Example:

In equation (9), we have 20 items The active user has accessed some of them. In this case n=20 and s=7.

$$\text{user-vector}= \{1,1,0,0,1,0,0,0,0,0,1,1,0,0,1,0,1,0,0,0\} \quad (9)$$

In the above example seven items are accessed by the active user and signed to 1. They can be used to recommend items from the other thirteen items which are signed to 0. The active user's vector can be represented as follows:

$$\text{user-vector} = \{i_e: z=1,2,3,\ldots,n \text{ where n is the number of all items}\} \quad (10)$$

Procedure one can be used to generate the active user vector.

Procedure one:

 # Users-items table is used to generate the user' vector that used to generate (the initial vector).

Consider

"User" : the active user Id.

"Items" : the list of all items in the given dataset.

retrieve "User" from the users-items table.

user-vector = null;

for all records that contains "User"

user-vector = user-vector + (fields of Items in the record);

end for

Figure 2 The creation of the active user vector.

b. The initial vector:

Markov chain initial vector 'I' equals to the active user's vector user-vector divided by the number of times of accessing all items by the active user i.e. the summation of items accessed (ones) in the active user's vector for every item divided by the number times of accessing all items by the active user.

$$I=\frac{user-vector}{(The\ number\ of\ times\ of\ accessing\ all\ items\ by\ the\ active\ user)} \quad (11).$$

'I' is the initial vector that represents the probabilities of accessing items by the active user.

c. The transition matrix of MCRS

Users-items table can be used to formulate Markov Chain transition matrix $T_{(n,n)}$ where n is the number of all items. Any row in $T_{(n,n)}$ represents an item and items that has been accessed with it by the same user in the same period of time. $Row_{(i,j)}$ is the row of $item_i$ where i = 1,2,3,...,n rows and j=1,2,3,..., n columns of items that accessed with $item_i$ i.e. any item has row and column. The value $p_{(i,j)}$ is the probability of accessing $item_j$ with $item_i$ in the same period of time. It can be calculated from the retrieved rows that have the value 1 in the column of the item. The probability vector of that

item is the summation of the retrieved rows divided by the summation of these rows cells, see equation (12). This vector gives the row of $item_i$ in the transition matrix, as represented in Table 3.

$$T_{(i,j)}=p_{(i,j)}=\frac{\sum_{i=1}^{n}(\text{row}_{(i,j)} \text{ where the columnof item}_{(i)}=1)}{\sum_{j=1}^{n}\sum_{i=1}^{n}(\text{row}_{(i,j)} \text{ where the column of item}_{(i)}=1)} \quad (12)$$

Table 3 Markov Chain Transition Matrix

	$item_1$	$item_2$	$item_{13}$	. . .	$item_n$
$item_1$	$p_{(1,1)}$	$p_{(1,2)}$	$p_{(1,3)}$	. . .	$p_{(1,n)}$
$item_2$	$p_{(2,1)}$	$p_{(2,2)}$	$p_{(2,3)}$	. . .	$p_{(2,n)}$
$item_3$	$p_{(3,1)}$	$p_{(3,2)}$	$p_{(3,3)}$	. . .	$p_{(3,n)}$
...	...	...	...	. . .	...
$item_n$	$p_{(n,1)}$	$p_{(n,2)}$	$p_{(n,3)}$	. . .	$p_{(n,n)}$

The basic MCRS is the vector product of (the initial vector) I and (the transition matrix) $T_{(i,j)}$.

$$R=I*T_{(i,j)} \quad (13).$$

The result the equation (13). is the vector R that contains the probabilities of accessing items by the active user. We sort these probabilities descending. Then, items of the highest probabilities are recommended to the active user.

VII. EXPERIMENTAL DESIGN

MovieLens dataset is used to conduct twelve experiments to evaluate the MCRS model. The time interval of users' activities is divided into months. Any month represents a period of time. The number of all periods is 137. In any experiment the periods from 1 to p (here 1< p <137) are used for training; and the next two periods are used for testing. In the first experiment we identify p=70, then we increment p by 3 in the next experiment.

CF user-based algorithm and MCRS technique are used to recommend items to an active user. The active user can access a set "A" of |A| items. Then, there is a set "B" of (n-|A|) items are not accessed by the active user. Items of A are known; but items of B are hidden, and needed to be recommended.

To evaluate MCRS, we can use the set A that accessed by the active user to recommend items to the active user, from the set B which is hidden. The dataset in any experiment split into training data and testing data. The training data can be used to recommend items to the active user as follows:

- In any experiment we identify the active user. His accessed items (elements of A) are used by all models to recommend items; and they used to identify the really accessed items from the testing data.
- The set A can be used by CF user-based algorithms to generate "CF result".
- It can be used by the basic MCRS to generate "The basic MCRS result".

The testing data can be used to retrieve the really accessed items as follows:

- The set A is used to retrieve "The really accessed items" from the testing data i.e. we retrieve all records that contains any item accessed by the active user. Then, we can calculate the accessibility of items by the summation of all retrieved records. Then, we can normalize the vector; such that the summation of the accessibility of all items equal to one.

A. The mean absolute error MAE:

Consider, the set (X) of the most (k) really accessed items, $X=\{x_i: i=1,3,3...,k\}$ where x_i is the ith item and k is the number of elements of X. And the set (R) is corresponding items of X in R, that recommended using MCRS, $R=\{r_i: i=1,3,3...,k\}$ where r_i is the ith item and k is the number of elements of R.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

The probabilities of accessing elements of X are:

$P(X)=\sum_{i=1}^{k} p(x_i) =1.$

The probabilities of accessing elements of R are:

$P(R)=\sum_{i=1}^{k} p(r_i) =1.$

The mean absolute error:

$$MAE\ (P(X), P(R))=\frac{1}{k}\sum_{i=1}^{k}\left|p\left(x_{(i)}\right) - p(r_{(i)})\right| \qquad 14$$

The best result has the less MAE.

The evaluation can be done using the mean absolute error MAE. CF and the basic MCRS results can be compared with the actually accessed items using MAE. From the actually accessed items we can identify the set X of the k highest probability items. The probabilities of accessing items of X can be normalized such that sum(P(X))=1. From the CF and the basic MCRS results we can list the corresponding items of X, "CF-result" and "MCRS-result" respectively. Then we can normalize the probabilities of accessing items of CF-result and items of MCRS-result such that sum(P(CF-result))=1 and sum(P(MCRS- result))=1. We can find the accessibility mean absolute error of CF and X, and of the basic MCRS and X. The best result has the less MAE.

The mean absolute error $mae(e)$ can be calculated for the twelve tests, for e=1,2,...,12. Then, the average means absolute error can be calculated as follows:

B. Precision/recall:

Consider the recommended set of items is R, and it recommended to the active users using MCRS. And the set of the really accessed items X, that taken from the test data. For any r ∈ R there are four situations **Error! Reference source not found.**.

A: r ∈X where r is recommended to the ser and he is actually interested inr.

B: r ∈X where r is recommended to the user but he is not interested inr.

C: r ∉X where r isn't recommended to the user but he is actually interested inr.

D: r ∉X where r isn't recommended to the user and he is not interested inr.

The evaluation can be done as follows:

- The first step is recommending a set of items (R) to the user u.
- The second step is identifying the set X i.e. the really accessed items.
- The final step is comparing X with R using **Error! Reference source not found.**.

Table 4 the result of recommending an item to user

		Recommended	
		(Positive)	(Negative)
Interested	True	TP: True IN& True RD r ∈ R& r ∈X	TN: True IN& False RD r ∉ R & r ∈X
	False	FP :True RD& False IN r ∈ R& r ∉X	FN : False IN& False RD r ∉ R & r ∉X

$$Precision = \frac{TP}{TP+FP}*100\% = \frac{Compared\ (interested\ and\ recommended)}{Compared}*100\% \qquad 15$$

$$Recall = \frac{TP}{TP+FN}*100\% = \frac{Compared\ (interested\ and\ recommended)}{interested(recommended\ or\ not\ recommended)}*100\% \qquad 16$$

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

C. The mean average precision:

The evaluation can be done using the mean average precision. In this case, the K highest probability items can be taken from the really accessed items and the different results. The best result has the highest mean average precision.

Precision and recall are single-value metrics based on number of the recommended and interested items by the system to users. The recommendation is a sequence of items, and it is better to consider the order in which the recommended items are presented. Then, we compute precision and recall at every position in the ranked sequence of the recommendation; we can plot a precision-recall curve, the precision is x-axis and recall is y-axis. Precision p is a function of recall r. Average precision computes the average value of p(r) over the interval of r= (0,1).

$$\text{Average\_precision} = \int \text{p(r)dr} \qquad 17$$

This means we calculate the area under the precision-recall curve. The same value of average precision can be calculated using the following equation:

$$\text{Average\_precision} = \sum_{k=1}^{K} \text{p(k)}\Delta\text{r(k)} \qquad 18$$

where small k is the index in the list of recommended items, capital K is the number of recommended items, p(k) is the precision at the index k, and Δr(k) is the change in recall from items number (k-1) to k.

The average precision, Average_precision(e), can be calculated for the twelve tests for e=1,2,...,12. Then, the mean average precision can be calculated as follows:

$$\text{mean\_Average\_precision} = \frac{1}{12}\sum_{e=1}^{12} \text{Average\_precision(e)} \qquad 19$$

VIII. EXPERIMENTAL RESULTS

A dataset from MovieLens is used in the evaluation processes. It split into months and each month represents a period of time. Twelve experiments are conducted and in each experiment the dataset split into training data and testing data. The actually accessed data is retrieved from the testing data, using the active user's accessed items. The same active user's data is used by CF and MCRS techniques to recommend items to him. The basic MCRS result is compared with standard RS (CF based on vector cosine similarity). Finally, we analyze and discuss the results.

A. The dataset:

To conduct the experiments for MCRS evaluation, a dataset from MovieLens is used [33]. It contains 855598 anonymous ratings of approximately 10,197 movies made by 2,113 users. It has avg. 404.921 ratings per user avg. 84.637 ratings per movie. The dataset is released in the framework of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). The time of users' ratings on movies is divided into months. The dataset is split into periods of time. Any period contains several months. The number of all periods is 137.

The basic MCRS VS CF user-based results:

To evaluate the basic MCRS twelve tests are conducted. In the first test the periods from 1 to p=70 are used for training. And the periods 71 and 72 are used for testing. Then for test from (2 to 12) we increment p by 3 periods. In the last test, periods from 1 to 103 are used for training and the periods 104 and 105 are used for test. Twelve users are used in the twelve tests. In any test the same user data is used by the basic MCRS and CF user-based algorithm in the training data to formulate the models. And the same user accessed items are used to identify the really accessed items from the testing data.

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (11-26), Month: January-April 2016, Available at: www.noveltyjournals.com

There are two cases with respect to the result of the recommendation results. The first case includes the active user's accessed items in the recommendation results. The second case excludes these items from the results.

Including the active user's accessed items in the results In this case, the active user's accessed items are included in the list of the actually accessed items that are retrieved from the testing data i.e. the recommendation lists, which are generated by the basic MCRS and CF user-based algorithms. It can include items that are accessed by the user in the training data. In the second case we exclude these items from the recommendations lists and the actually accessed items i.e. we recommend novel items.

In this case the basic MCRS and CF user-based algorithm have similar result with small variation as shown in **Error! Reference source not found.**. The result ,the average precision, of CF is 0.887218177 and the result the basic MCRS is 0.874561479. CF outperforms the basic MCRS by 0.012656698.

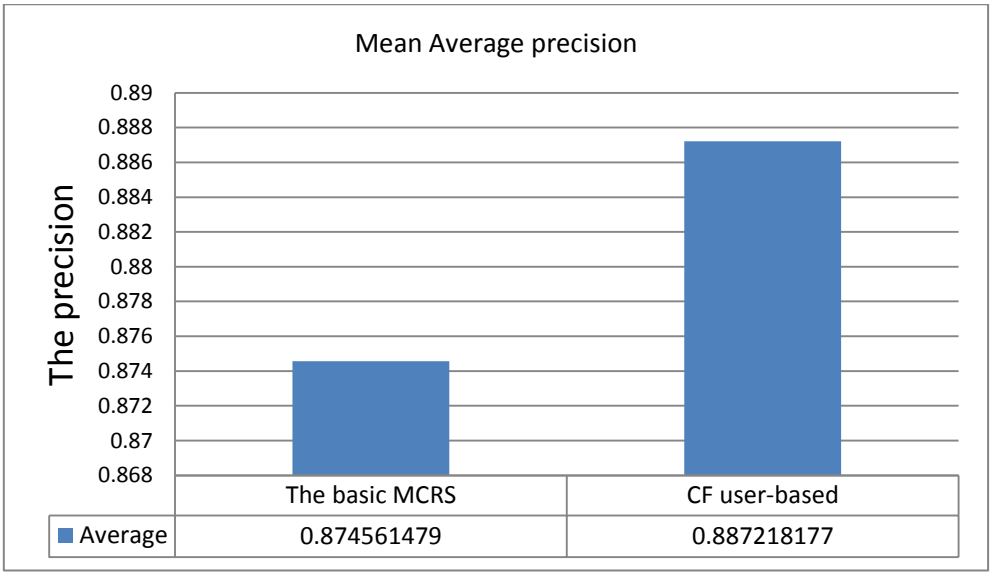

Figure 3 Mean Average precision of the basic MCRS VS CF user-based algorithm

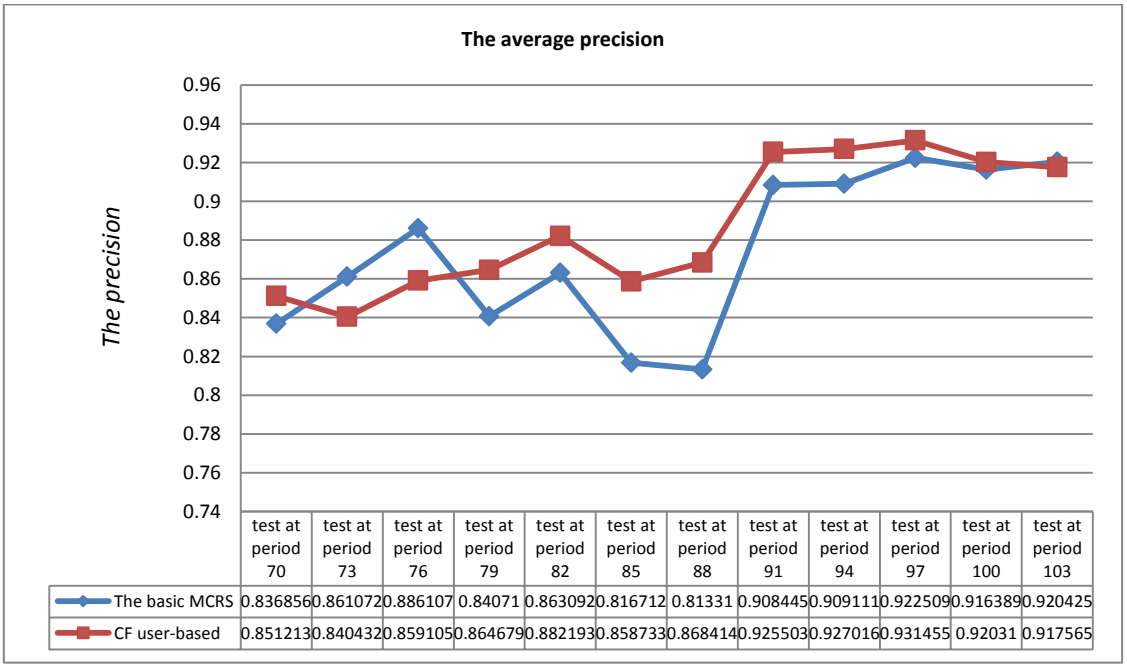

Figure 4 The average precision of the basic MCRS VS the CF user-based algorithm in the case of including the active user's accessed items in the recommendation results

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (27-29), Month: January-April 2016, Available at: www.noveltyjournals.com

Significant test:

Table 5 The average precision of the basic MCRS VS the CF user-based

The basic MCRS	The user-based CF
0.8368	0.8512
0.861	0.8404
0.8861	0.8591
0.8407	0.8646
0.863	0.8821
0.8167	0.8587
0.8133	0.8684
0.9084	0.9255
0.9091	0.927
0.9225	0.9314
0.9163	0.9203
0.9263	0.9175

Unpaired t test:

Mean of the basic MCRS = 0.875017 (n = 12)

Mean of the CF user-based = 0.887183 (n = 12)

Assuming equal variances:

Combined standard error = 0.015612

The degree of freedom df = 22

t = 0.779311

One sided P = 0.222

Two sided P = 0.4441

95% confidence interval for difference between means = -0.044544 to 0.020211

Power (for 5% significance) = 18.18%

Assuming unequal variances:

Combined standard error = 0.015612

df = 21.217126

t(d) = 0.779311

One sided P = 0.2222

Two sided P = 0.4444

95% confidence interval for difference between means = -0.044544 to 0.020211

Power (for 5% significance) = 10.39%

Comparison of variances:

Two sided F test is not significant

No need to assume unequal variances

However, our target is the suggestion of novel items to the active user; therefore, Items that have been accessed by the active user must be excluded from the recommendation results before the evaluation.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (27-29), Month: January-April 2016, Available at: www.noveltyjournals.com

B. Excluding the active user's accessed items from the results:

In the second case items that have been accessed by the active user are excluded from the results. The evaluation is done using the mean average precision, and the mean absolute error (MAE).

The mean average precision:

In the second case, the active user's accessed items are excluded from the recommendation lists and the really accessed item. In this case the basic MCRS has the mean average precision (0.826424068); it is better than the CF user-based algorithm by (0.539729201) as represented in Figure 5.

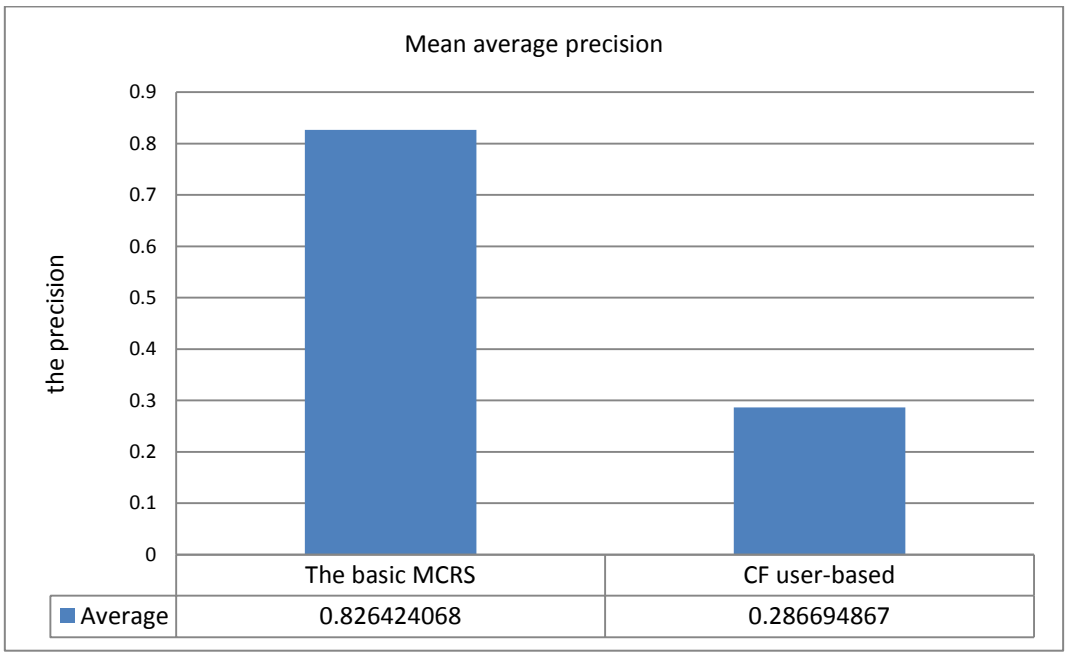

Figure 5 The mean average precision of the basic MCRS VS the CF user-based algorithm in the case of excluding the active user's accessed items in the recommendation results

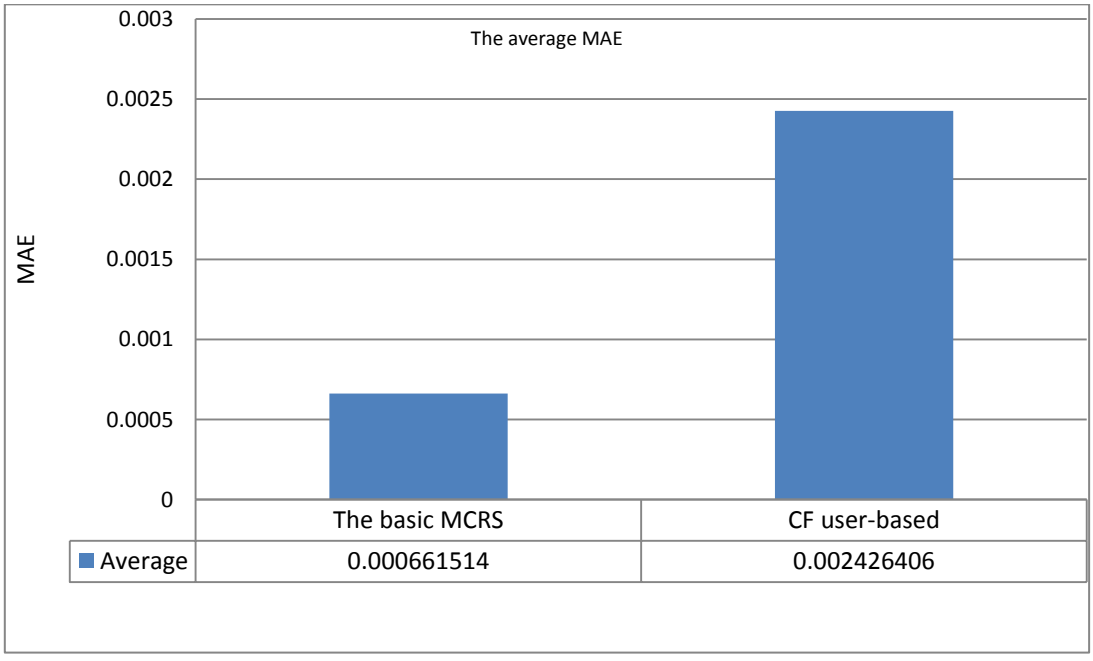

Figure 6 The average MAE of the basic MCRS VS CF user-base algorithm in the case of excluding the active user's accessed items in the recommendation results

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 3, Issue 1, pp: (27-29), Month: January-April 2016, Available at: www.noveltyjournals.com

The mean absolute error (MAE):

On the other hand, the basic MCRS and CF user-based algorithm are compared using the mean absolute error (MAE) which is calculated from the twelve tests results. MCRS has the MAE average (0.000661514); it is less than the CF user-based algorithm by (0.001764892) as represented in Figure 6.

This means that the basic MCRS outperforms the CF user-based algorithm using mean average precision and the mean absolute error (MAE).

IX. CONCLUSION

This paper introduces the general area of recommending items to users, using Collaborative filtering techniques and the general idea of using the time factor to weight ratings of users for items before they are used in the recommendation processes. Then, we find some limitations in these techniques. The first limitation comes from ways of calculating the similarities between users and items. These similarities are based on users' rating for items. These ratings are not accurate; because users' opinions vary with time. Thus, the similarities calculation violates the accuracy of the recommendation. Markov chain techniques have a limitation. They consider the sequences of accessing items. The transition matrix in these techniques is the probabilities of accessing item that follows a sequence of items. They use sets of a of k items, while users can access more than k items. Also users that access less than k items can't benefit from these techniques. We illustrate ways of addressing this limitation and give the motivation of designing new technique that can be used to recommend items to users. The new technique is Markov Chain recommendation system. It based on users rating for items, in the same session or period of time, that taken implicitly from users' preferences. These ratings are used to calculate the transition matrix and the initial matrix. The new technique outperforms the conventional collaborative filtering recommendation system. We use a data set from MovieLens for the evaluation. The evaluation is done using the mean absolute error, and precision and recall.

REFERENCES

[1] I. Guy, N. Zwerdling, I. Ronen, D. Carmel, and E. Uziel, “Social Media Recommendation based on People and Tags,” no. Lc, pp. 194–201, 2010.

[2] R. Burke, “Hybrid Recommender Systems : Survey and Experiments †,” pp. 1–29.

[3] J. a. Konstan and J. Riedl, “Recommender systems: from algorithms to user experience,” User Model. User-adapt. Interact., vol. 22, no. 1–2, pp. 101–123, Mar. 2012.

[4] X. Su and T. M. Khoshgoftaar, “A Survey of Collaborative Filtering Techniques,” Adv. Artif. Intell., vol. 2009, no. Section 3, pp. 1–19, 2009.

[5] A. Ahmed And N. Salim, “Using Trend Analysis And Social Media Features To Enhance Recommendation Systems: A Systematic Literature Review.,” J. Theor. Appl. …, Vol. 55, No. 3, P. 408, 2013.

[6] M. J. Pazzani and D. Billsus, “Content-Based Recommendation Systems,” Springer-Verlag Berlin Heidelb. 2007, no. 4321, pp. 325–341, 2007.

[7] R. Burke, “Hybrid Recommender Systems: Survey and Experiments,” User Model. User-adapt. Interact., vol. 12, no. 4, pp. 331–370, Nov. 2002.

[8] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, “Recommender Systems,” p. 97, Feb. 2012.

[9] M. D. Ekstrand, “Collaborative Filtering Recommender Systems,” Found. Trends® Human–Computer Interact., vol. 4, no. 2, pp. 81–173, 2010.

[10] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel, “Probabilistic memory-based collaborative filtering,” Knowl. Data Eng. IEEE Trans., vol. 16, no. 1, pp. 56–69, 2004.

[11] J. Ben Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative Filtering Recommender Systems,” pp. 291–324.

[12] G. Qian, S. Sural, Y. Gu, and S. Pramanik, "Similarity between Euclidean and cosine angle distance for nearest neighbor queries," in Proceedings of the 2004 ACM symposium on Applied computing - SAC '04, 2004, p. 1232.

[13] H. Parvin, H. Alizadeh, and B. Minaei-bidgoli, "MKNN : Modified K-Nearest Neighbor," in Proceedings of the World Congress on Engineering and Computer Science, WCECS, 2008, pp. 22–25.

[14] G. P. Basharin, A. N. Langville, and V. a. Naumov, "The life and work of A.A. Markov," Linear Algebra Appl., vol. 386, pp. 3–26, Jul. 2004.

[15] D. P. Foster and L. H. Ungar, "Spectral dimensionality reduction for HMMs," Representations, no. 1989, pp. 1–26, Mar. 2012.

[16] D. Stirzaker, "Markov Chains," pp. 396–477, Oct. 2003.

[17] M. Jacobsen, "Markov Chains and Markov Processes," no. 1, pp. 1–3, Sep. 2006.

[18] T. Konstantopoulos, "MARKOV CHAINS AND RANDOM WALKS," 2009.

[19] T. Markov, "FINITE-STATE MARKOV CHAINS," vol. 103.

[20] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-Based Recommender System ∗," vol. 6, pp. 1265–1295, 2005.

[21] X. Wu, Q. Liu, E. Chen, L. He, J. Lv, C. Cao, and G. Hu, "Personalized Next-song Recommendation in Online Karaokes," 2013.

[22] S. Rendle, C. Freudenthaler, and L. Schmidt-thieme, "Factorizing Personalized Markov Chains for Next-Basket Recommendation," 2010.

[23] M. Eirinaki, M. Vazirgiannis, and D. Kapogiannis, "Web path recommendations based on page ranking and Markov models," Proc. seventh ACM Int. Work. Web Inf. data Manag. - WIDM '05, p. 2, 2005.

[24] Y. Huang, T. Huang, K. Wang, and W. Hwang, "A Markov-based Recommendation Model for Exploring the Transfer of Learning on the Web," vol. 12, pp. 144–162, 2009.

[25] F. Fouss, S. Faulkner, M. Kolp, A. Pirotte, and M. Saerens, "MARKOV-CHAIN MODEL," 2001.

[26] M. Munoz-Organero, G. A. Ramiez-Gonza ez, P. J. Munoz-Merino, and C. Delgado Kloos, "A Collaborative Recommender System Based on Space-Time Similarities," Pervasive Comput. IEEE, vol. 9, no. 3, pp. 81–87, 2010.

[27] P. Resnick, P. Bergstrom, and J. Riedl, "GroupLens : An Open Architecture for Collaborative Filtering of Netnews," pp. 175–186, 1994.

[28] G. Qian, S. Sural, Y. Gu, and S. Pramanik, "Similarity between Euclidean and cosine angle distance for nearest neighbor queries," in Proceedings of the 2004 ACM symposium on Applied computing - SAC '04, 2004, p. 1232.

[29] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," Adv. Artif. Intell., vol. 2009, no. Section 3, pp. 1–19, 2009.

[30] Y. Ding and X. Li, "Time weight collaborative filtering," in Proceedings of the 14th ACM international conference on Information and knowledge management, 2005, pp. 485–492.

[31] T. D. Wang, C. Plaisant, B. Shneiderman, N. Spring, D. Roseman, G. Marchand, V. Mukherjee, and M. Smith, "Temporal Summaries: Supporting Temporal Categorical Searching, Aggregation and Comparison," Vis. Comput. Graph. IEEE Trans., vol. 15, no. 6, pp. 1049–1056, 2009.

[32] K. Stefanidis, I. Ntoutsi, K. Nørvåg, and H.-P. Kriegel, "A Framework for Time-Aware Recommendations.," in DEXA (2), 2012, vol. 7447, pp. 329–344.

[33] I. Cantador, P. Brusilovsky, and T. Kuflik, "2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011)," in Proceedings of the 5th ACM conference on Recommender systems, 2011.