

MRRA: A New Approach for Movie Rating Recommendation

Chiraz Trabelsi¹✉ and Gabriella Pasi²

¹ Faculté des Sciences de Tunis, LIPAH-LR 11ES14,
Université de Tunis El Manar, 2092 Tunis, Tunisia
`chiraz.trabelsi@fst.rnu.tn`

² DISCo Viale Sarca, Università degli Studi di Milano-Bicocca,
336 - Edificio U14, 20126 Milano, Italy
`pasi@disco.unimib.it`

Abstract. Nowadays, Movie constitutes a predominant form of entertainment in human life. Most video websites such as YouTube and a number of social networks allow users to freely assign a rate to watched or bought videos or movies. In this paper, we introduce a new movie rating recommendation approach, called MRRA, based on the exploitation of the Hidden Markov Model (HMM). Specifically, we extend the HMM to include user's rating profiles, formally represented as triadic concepts. Triadic concepts are exploited for providing important hidden correlations between rates, movies and users. Carried out experiments using a benchmark movie dataset revealed that the proposed movie rating recommendation approach outperforms conventional techniques.

Keywords: User's rating profile model · Triadic analysis · Rate recommendation · Hidden Markov Model

1 Context and Motivations

The ability of recommender systems to generate direct connections between users and items that represent matches in interests and preferences makes them an important tool for alleviating information overload for Web users. They are becoming increasingly important in the success of electronic commerce, and being used in most video websites such as YouTube and Hulu and a number of social networks allow users rate on videos or movies. In a general way, a recommendation system constructs items' profiles, and users' profiles based on their previous recorded behaviour. Thereafter it makes a prediction on the given rating by a certain user on a certain item which he/she has not yet evaluated. Based on the prediction, the system makes recommendations. Various techniques for recommendation generation have been proposed and widely deployed in commercial environments, among which collaborative filtering (*CF*) methods still represents the most commonly adopted technique in crafting academic and commercial [1,5,9] recommender systems. Its basic idea refers to making recommendations

based upon ratings that users have assigned to products. Ratings can either be explicit, *i.e.*, by having the user state his opinion about a given product, or implicit, when the mere act of purchasing or mentioning of an item counts as an expression of appreciation. While implicit ratings are generally more facile to collect, their usage implies adding noise to the collected information.

The information domain for a collaborative filtering system consists of users which have expressed preferences for various items. A preference expressed by a user for an item is called a rating and is frequently represented as a (User, Item, Rating) triple. These ratings can take many forms, depending on the system in question. Some systems use real- or integer-valued rating scales such as 0–5 stars, while others use binary or ternary (like/dislike) scales. *CF* algorithms involve then matching the ratings of a current user for items, *e.g.*, movies or books, with those of similar users in order to produce recommendations for items not yet rated or seen by an active user. Pearson Correlation and Vector Similarity [8] are two most common measures for finding the user similarities. Later several researchers have proposed different other measures for calculating user similarities. Weighted average of the most similar users' ratings for the test items are output as the predicted rating. There are several algorithms that use probabilistic graphical models for solving the task of rating prediction [6, 17]. Matrix factorization algorithms have also been widely popular. These algorithms model both the user and items as vectors in a low dimensional feature space. Representation of the user and items in the joint feature space is then used to compute the predicted ratings [4].

Regarding those aforementioned approaches, we introduce in this paper a new approach, called MRRA¹, for movie rating recommendation. MRRA is based on the use of the Hidden Markov Model (HMM). In fact, contrary to existing approaches dedicated to movie rate recommendation, neither a given user nor the co-occurrences of movies or rates values are handled for rate recommendation. We only consider the movie, *i.e.*, a movie to be rated, as input and by matching the movie to its corresponding context according to HMM states [14], we estimate a rate value candidate, that represents the most probable user's rating profile. In fact, HMMs have been successfully applied in many prediction field especially for users' web search query prediction [3, 7]. Therefore, in this paper, we introduce a novel Hidden Markov Model (HMM) based approach, to handle two main challenges addressing movie rate recommendation problem: (i) Using the three-dimensional structure of the movie rating database for identifying and representing users' rating profiles; and (ii) Exploiting users' rating profiles for predicting users' next rate value which could/should be applied by the users to a particular movie.

From the moment that the usage of rates values assigned by users sharing similar interests tends to converge to a shared behavior [10], then we firstly propose, to define a user rating profile as an implicit shared conceptualization formally sketched by a triadic concept. Indeed, triadic concepts allow grouping semantically related movies that take into account the users' rating behavior.

¹ MRRA is the acronym of Movie Rating Recommendation Approach.

Therefore, instead of using matrix based models or co-occurrence techniques, we use an algorithm, called TRICONS [16], to mine users' rating profiles from a movie database. Moreover, on the contrary of the previous approaches which consider a 2-dimensional pair relations, missing by the way a part of the total interaction between the three dimensions, *i.e.*, user, rate and resource, we introduce a unified framework to concurrently model the three dimensions handled by a HMM [14]. Indeed, we propose to exploit the HMMs prediction capabilities [3, 7] to model the whole rating process as a sequence of (auto)-transitions between states. Hence, each rating profile, represented as triadic concept, can be defined as a state of the HMM, and the rate value and evaluated movies as observations generated by the state.

The remainder of the paper is organized as follows. In Sect. 2, we introduce our approach for Movie Rating Recommendation consisting of two phases: the model-building phase (Sect. 2.1) and the exploitation phase (Sect. 2.2). The experimental study of our approach is illustrated in Sect. 3. Section 4 concludes this paper.

2 MRRA for Movie Rating Recommendation

In this section, we present our recommendation approach, called *MRRA*, which aims to effectively assigning the right rate value to a particular movie. *MRRA* is able to generate recommendations in constant time and performs a triadic concept analysis to mine users' rating profiles. The triadic concepts can be used as an access structure for providing important hidden correlations between rates, movies and users. In order to achieve these goals, the proposed *MRRA* approach performs two main phases: the model-building phase and the exploitation phase.

2.1 The Model-Building Phase

MRRA starts by learning users' rating behavior by identifying users' rating profiles behind the assigned rates. Considered as a tripartite graph of users, ratings and movies, the users' movies rates assignments can be, formally, represented as a triadic context [12].

The model-building phase performs concurrently by retrieving user's movies sequences from a given movie database, *e.g.*, an example of a collection of users' movies rates assignments \mathcal{S}^∇ with $\mathcal{U} = \{u_1, u_2, u_3, u_4\}$, $\mathcal{M} = \{m_1, m_2, m_3, m_4, m_5\}$ and $\mathcal{R} = \{r_1, r_2, r_3\}$. Each triple from \mathcal{S}^∇ represents a triadic relationship between a user belonging to \mathcal{U} , a rate from \mathcal{R} and a movie belonging to \mathcal{M} , and mining users' rating profiles. The results of these previously steps, are then used for the HMM training. Each step in the building phase is described below.

Step 1: User's Movies Ratings Extraction: This step aims to determine for each user u_i the sequences SR_i of his similarly rated movies. It proceeds by firstly collect users' assignments S_i defined as follows:

Table 1. An example of users' ratings assignments.

$S_1 := \{\{m_{1,1}, m_{1,2}, m_{1,3}\}, r_{1,1}\}; \{\{m_{1,4}\}, r_{1,2}\}$
$S_2 := \{\{m_{2,3}, m_{2,4}\}, r_{2,3}\}$
$S_3 := \{\{m_{3,2}, m_{3,3}\}, r_{3,4}\}; \{\{m_{3,4}, m_{3,5}\}, r_{3,6}\}$

$$S_i := \{\{\text{User movies } m_{S_i,p}\}, r_{S_i,j}\}.$$

with $r_{S_i,j} :=$ The rate value j assigned by the user u_i in the post S_i , $m_{S_i,p} :=$ The p ordered movie rated in S_i .

Table 1 reports an example of users' ratings assignments. For example, the user assignments S_2 , highlights that the user u_2 has assigned the rate value $r_{2,3}$ to the two movies $m_{2,3}$ and $m_{2,4}$. Once the users ratings assignments are collected, we generate user's movies sequences by keeping, for each user, the sequences of movies related to his assignments and discard useless information. An example of user's movies sequences associated to Table 1 is given in the following: $SL_1: ((m_{1,1}, m_{1,2}, m_{1,3}); (m_{1,1}, m_{1,4}))$, $SL_2: (m_{2,3}, m_{2,4})$, $SL_3: ((m_{3,2}, m_{3,3}); (m_{3,4}, m_{3,5}))$ where SL_i , describes movies rating sequences of the user u_i .

Step 2: Users' Rating Profiles Mining: The second step of the model-building phase step is to mine the users' rating profiles, formally represented by triadic concepts. Let us firstly recall in the following the main definition related to a triadic concept [11] that will be used in the remainder.

Definition 1 (TRIADIC CONCEPT). *A triadic concept (tri-concept for short) of a collection of users' movies rates assignments $\mathcal{S} := (\mathcal{U}, \mathcal{R}, \mathcal{M}, \mathcal{G})$ is a triple $(\mathcal{U}_1, \mathcal{R}_1, \mathcal{M}_1)$ with $\mathcal{U}_1 \subseteq \mathcal{U}$, $\mathcal{R}_1 \subseteq \mathcal{R}$, and $\mathcal{M}_1 \subseteq \mathcal{M}$ with $\mathcal{U} \times \mathcal{R} \times \mathcal{M} \subseteq \mathcal{G}$ such that the triple $(\mathcal{U}_1, \mathcal{R}_1, \mathcal{M}_1)$ is maximal.*

Consequently, a rating profile can be formally represented, in $\mathcal{S} = (\mathcal{U}, \mathcal{R}, \mathcal{M}, \mathcal{G})$, as a triadic concept $\mathcal{RP} = (U', R', M')$ where $U' \subseteq \mathcal{U}$, $R' \subseteq \mathcal{R}$, and $M' \subseteq \mathcal{M}$ with $U' \times R' \times M' \subseteq \mathcal{G}$. The users' rating profiles are therefore obtained by invoking the TRICONS algorithm [16] on the collection \mathcal{S} . Roughly speaking, the rating profile $RP_1 = \{(u_1, u_2, u_3), (m_1, m_2), (r_4)\}$ highlights that the community of users (u_1, u_2, u_3) share the same rating behaviour on the movies (m_1, m_2) assigned by r_4 .

Given the users' movies sequences and the users' rating profiles, previously extracted, MRRA proceeds in the next step with the HMM training.

Step 3: HMM Training Sequences Extraction: During this last step of the model-building phase, MRRA trains the HMM. Actually, in a HMM, there are two types of states: the observable states and the hidden ones [14]. Thereby, we define users' movies sequences as the observable states in the HMM, whereas the hidden states are modeled by the users' rating profiles.

Hence, given the set of hidden states $St = \{st_1, \dots, st_{ns}\}$, we denote the set of distinct rates values as $\mathcal{R} = \{r_1, \dots, r_{nr}\}$, the set of movies $\mathcal{M} = \{m_1, \dots, m_{nm}\}$

and the set of users $\mathcal{U} = \{u_1, \dots, u_{nu}\}$, where ns is the number of states of the model, nr is the total number of rates, nm is the total number of movies, nu is the number of users, and SL_i is a state sequence. Our HMM denoted $\lambda = (A, B, B', \pi)$, is a probabilistic model defined as follows:

- $\pi = [\dots \pi_i \dots]$, the initial state probability, where $\pi_i = P(st_i)$ is the probability that a state st_i occurs as the first element of a state sequence SL_i .
- $B = [\dots b_j(r) \dots]$, the rate emission probability distribution, where $b_j(r) = P(r | st_j)$, denotes the probability that a user, currently at a state st_j , assigned a rate r .
- $B' = [\dots b_k(m) \dots]$, the movie emission probability distribution, where $b_k(m) = P(m | st_k)$, denotes the probability that a user, currently at a state st_j , rates the movie m .
- $A = [\dots a_{ij} \dots]$, the transition probability, where $a_{ij} = P(st_j | st_i)$ represents the transition probability from a state st_i to another one st_j .

Once the HMM is formalized, we proceed with learning its parameters (A, B, B', π) . This is done by computing the four sets of the HMM parameters: the initial state probabilities $\{P(st_i)\}$, the rate emission probabilities $\{P(r | st_j)\}$, the movie emission probabilities $\{P(m | st_k)\}$, and the transition probabilities $\{P(st_j | st_i)\}$. Hence, inspired from [3], we compute these sets as follows:

1. $\pi_i = P(st_i) = \frac{|\varphi(st_j)|}{|SL_c|}$ with:
 - $SL_c = \cup_{i \in 1, \dots, t} \{E_i\}$ = total set of candidate states sequences to which could be matched a sequence of movies, where E_i denotes the set of candidate states that could match a movie from a given sequence of movies.
 - $\varphi(st_j)$ = set of states sequences in SL_c starting from st_j .
2. $b_j(r) = P(r | st_j) = \frac{\sum_{m \in \mathcal{M}_j} Count(m, r)}{\sum_{r \in \mathcal{R}_j} \sum_{m \in \mathcal{M}_j} Count(m, r)}$.
3. $b_k(m) = P(m | st_k) = \frac{\sum_{r \in \mathcal{R}_k} Count(m, r)}{\sum_{r \in \mathcal{R}_k} \sum_{m \in \mathcal{M}_k} Count(m, r)}$, where $Count(m, r)$ = number of times the movie m is assigned the rate r in the collection.
4. $a_{i,j} = P(st_j | st_i) = \frac{CS(st_i, st_j)}{NC}$ with:
 - NC = the number of occurrences of st_j in SL_c .
 - $CS(st_i, st_j)$ = the number of times the state st_i is followed by the state st_j in SL_c .

2.2 The Exploitation Phase

The exploitation phase aims to identifying the movie's context and then predict the next rate which would/could be used by an active user according to the next HMM state. Actually, during the rating process of a movie m , two consecutive stages are considered: (i) Matching the current movie m to its related context according to HMM states; and then (ii) Predicting the next HMM state which represents the most similar rating profile. The prediction process proceeds by identifying the most likely HMM state, *i.e.*, rating profile, st_{MS} to which m

could better belong. This is done by computing, for each HMM state, the value of the quantity $Mat_i = \pi_i \times b_i(m)$, where π_i is the initial probability of the state st_i and $b_i(m)$ is the emission probability of m at st_i . Therefore, the state with the highest value, *i.e.*, st_{MS} , of Mat_i will define the context of m . Once the context is found, the rate, with the highest probability, belonging to the rating profile represented by the state st_{MS} is recommended. The prediction of a similar user's rating profile is then performed, by looking for the next state s_{NextMS} of st_{MS} . This is obtained by calculating the index value $NextMS$ as follows: $NextMS = \operatorname{argmax}_j \{a_{\{MS,j\}}\}$, where st_j is a successor of st_{MS} in the HMM. Hence, the state s_{NextMS} represents the most probably rating profile that could match the user behaviour in rating the movie m .

Table 2. An example of a HMM states.

$st_1 = \{\{m_1, m_2, m_3\}, \{r_4\}, \{u_1, u_2, u_3, u_4\}\}$
$st_2 = \{\{m_4, m_3\}, \{r_2\}, \{u_3, u_6\}\}$
$st_3 = \{\{m_4, m_7\}, \{r_1\}, \{u_4, u_7\}\}$
$st_4 = \{\{m_5, m_6\}, \{r_1\}, \{u_4, u_5, u_6\}\}$
$st_5 = \{\{m_8, m_9\}, \{r_3\}, \{u_5, u_6\}\}$

Illustrative Example: Consider a HMM with five states described in Table 2, *i.e.*, $\{st_1, st_2, st_3, st_4, st_5\}$, where each state denotes a user rating profile, *i.e.*, RP_1, RP_2, RP_3, RP_4 and RP_5 , extracted by the algorithm TRICONS from a sample taken from the MovieLens dataset (*cf.*, Sect. 3).

Each rating profile is represented by a triplet, *i.e.*, the set of all movies similarly rated by a set of users. The corresponding transition matrix A , and the distributions of the different probabilities of observation (of movies and rates) are obtained by calculating probabilities as described in Sect. 2. Suppose that the generated HMM with five states $\{st_1, st_2, st_3, st_4, st_5\}$ has a transition probability matrix as follows:

$$A = \begin{pmatrix} 0.4 & 0.6 & 0.0 & 0.0 & 0.0 \\ 0.2 & 0.5 & 0.3 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.3 & 0.2 & 0.5 \\ 0.0 & 0.0 & 0.0 & 0.3 & 0.7 \\ 0.0 & 0.0 & 0.0 & 0.6 & 0.4 \end{pmatrix}$$

And let us assume that $\pi = (0.2 \ 0.2 \ 0.2 \ 0.2 \ 0.2)$. Hence, considering the rating represented by the state st_1 , for example, to predict the rate of the movie m_8 , the prediction process starts with finding the context of the movie m_8 by looking for the most likely HMM state to which the movie m_8 could better belong. This is obtained by computing for each of the five states, the quantity $Mat_i = \pi_i \times b_i(m_8)$ including:

$$Mat_1 = \pi_1 \times b_1(m8) = 0;$$

$$Mat_2 = Mat_3 = Mat_4 = 0 \text{ and}$$

$Mat_5 = \pi_5 \times b_5(m8) = 0.2 \times 0.4 = \mathbf{0.08}$. Consequently, st_5 is the state which has the highest probability to represent the users' rating profile for the resource $m8$. Thus the candidate rate value $r3$ is recommended. Furthermore, possible states transitions from st_5 are either st_4 or st_5 . Thus, the corresponding candidate rate to be assigned to the potential next movie, after the $m8$'s movie, are computed by the following formula, $argmax_j \{a_{5,j} \times b_j(m)\}$ with $j \in \{4, 5\}$, *i.e.*, possible state transition from st_5 and m is a movie belonging to the rating profile represented by st_4 or st_5 states.

3 Experiments

In this section we report results of the experimental evaluation of our proposed approach. We describe the data set used, the baselines description, as well as performance measures we consider appropriate for the task of predicting the rating given a user u and a movie m .

3.1 Dataset Description

We conducted experiments on a benchmark dataset: the MovieLens 100K². MovieLens 100K is a movie rating dataset consisting of 100,000 ratings (1–5) from 943 users on 1682 movies, and each user has rated at least 20 movies.

3.2 Baselines Description

To confirm the validity of our results, we compare them with the results obtained using three recommendation approaches. We describe in the following the specific setting used to run them.

Item-Based K Nearest Neighbor Algorithm (IB-KNN) [15]. In order to determine the rating of User u on Movie m , we find other movies that are similar to Movie m , and based on User u 's ratings on those similar movies we infer his rating on Movie m . Thus, the implemented IB-KNN algorithm performs the following generic pattern:

- Compute the similarity of movie a and movie b . As in [15], we use the adjusted cosine similarity between Movie a and b :

$$sim(a, b) = \frac{\sum_{u \in U(a) \cap U(b)} (r_{a,u} - \bar{r}_u)(r_{b,u} - \bar{r}_u)}{\sqrt{\sum_{u \in U(a) \cap U(b)} (r_{a,u} - \bar{r}_u)^2 \sum_{u \in U(a) \cap U(b)} (r_{b,u} - \bar{r}_u)^2}} \quad (1)$$

where $r_{a,u}$ is User u 's rating on Movie a , \bar{r}_u is User u 's average rating, $U(a)$ is the set of users that have rated Movie a and $U(a) \cap U(b)$ is the set of users that have rated both Movie a and b .

² <http://grouplens.org/datasets/movieLens/>.

- Select a set of K most similar movies to the target movie and generate a predicted value of user u 's rating. Hence, KNN finds the nearest K neighbors of each movie under the above defined similarity function.
- Generate a predicted value $P_{m,u}$ of user u 's rating by using the weighted means as follows:

$$P_{m,u} = \frac{\sum_{u \in N_u^K(m)} sim(m, j) r_{j,u}}{\sum_{u \in N_u^K(m)} |sim(m, j)|} \quad (2)$$

where $N_u^K(m) = \{j: j \text{ belongs to the } K \text{ most similar movies to Movie } m \text{ and User } u \text{ has rated } j\}$, and $sim(m, j)$ is the adjusted cosine similarity defined in (1), $r_{j,u}$ represent the existent ratings of User u on Movie j .

Content Based Filtering (CBF) [13]. In CBF approaches, the recommended movies are those having similar features to the ones that the user have already rates. Hence, the implemented CBF, variants pursue the following generic algorithm:

- Compute features similarities $Sim(m, x_i)$ between the candidate movie m and all the other movies x_i based on their sets of features $\mathbf{f}_i = \{f_{i,1}, \dots, f_{i,n}\}$, *i.e.*, genre, subject matter, actors and director.
- Select S_m , the set of the k most similar movies to the candidate movie m .
- Generate a predicted value $P_{m,u}$ based on the ratings assigned by the user u to movies in S_m (c.f. Eqs. 1 and 2).

The User-Centred Collaborative Filtering Approach (UC-CF) [9]. UC-CF is based on the assumption that users with similar preferences will rate movies similarly. Thus, the implemented variants of UC-CF proceeds as following:

- Compute $Sim(u_a, u_i) = Sim(\mathbf{r}_a, \mathbf{r}_i)$, the similarities between the active user u_a and all the other users u_i based on their common ratings \mathbf{r}_a and \mathbf{r}_i assigned to the same movies.
- Select S_{ua} , the set of the k most similar users to the active user u_a .
- Estimate \bar{r}_{ac} using the Mean rating estimator on the set of ratings assigned by S_{ua} to the movie m_c . The Mean rating estimator is implemented for the recommendation algorithms in order to estimate the rating r_{ac} that the active user u_a would assign to a candidate movie m_c . Let N be the number of existing ratings $r_{i,j}$ such as $u_i \in S_{ua}$ and $m_i \in S_{m_c}$. The mean estimator is as follows:

$$\hat{r}_{ac} = \frac{1}{N} \sum_{u_i \in S_{ua}} \sum_{m_i \in S_{m_c}} r_{i,j}$$

3.3 Effectiveness Evaluation

We used a supervised learning process to assess the performances of our *MRRA* approach vs. those of IB-KNN, CBF and UC-CF on the MovieLens dataset as our training and testing set. Specifically, we randomly selected 60 rated movies to use as training set, and 40 as a test set. The training set is used to estimate the model while the test set is used for the evaluation. Hence, the main task of our approach is to predict for each user’s movie, picked from the test dataset, its related rate. In order to analyze the accuracy of our approach we adopted the common Information Retrieval evaluation measures, namely Recall and Precision that produces scores ranging from 0 to 1 (100%) [2]. Therefore, if we suppose that for a user rate value r , the set of movies actually assigned by the value r is \mathcal{R}_m , *i.e.*, as the ground truth, and the set of the predicted rates values is T_r , then the measures of *recall* and *precision* are given as follows: $Recall = \frac{|\{\mathcal{R}_m\} \cap \{T_r\}|}{|\{T_r\}|}$, $Precision = \frac{|\{\mathcal{R}_m\} \cap \{T_r\}|}{|\{\mathcal{R}_m\}|}$. We report in the following results averaged over all movies from the test set. Ten-fold cross evaluation method was used while the number of nearest neighbors was fixed at 20.

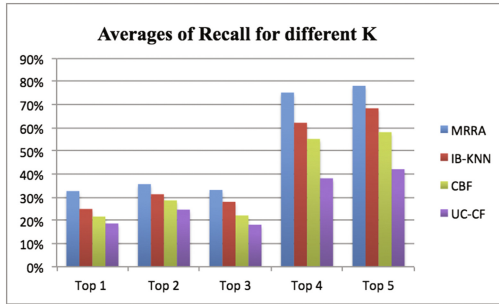


Fig. 1. Averages of recall on the MovieLens dataset.

Figure 1, depicts averages of *recall* for different values of K , *i.e.*, the number of predicted rates values, ranging from 1 to 5. Thus, according to the sketched histograms, we can point out that our approach sharply outperforms those of IB-KNN, CBF and UC-CF. In fact, the *Recall* values of the three baselines are significantly lower than those achieved by our approach. Furthermore, the average *Recall* of *MRRA* achieves high percentage for higher value of K . Indeed, for $K = 5$, the average *Recall* is equal to 78.16%, showing an increase of 45.44% compared to the average *Recall* for $K = 1$. In this case, for a higher value of K , *i.e.*, $K = 5$, by matching the current movie with its corresponding context, our approach can produce almost all of the ratings that are likely to be assigned by the user on the current movie.

Besides, according to Fig. 2, the percentage of *Precision* of our approach outperforms the two baselines over the MovieLens dataset. Indeed, our approach achieves the best results when we choose the value of K around 5. In fact, for

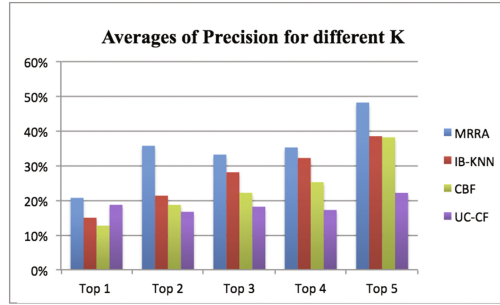


Fig. 2. Averages of precision on the MovieLens dataset.

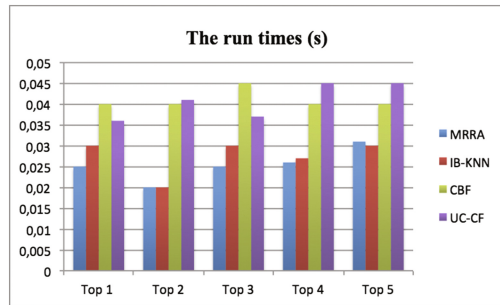


Fig. 3. The run times (s) of MRRA with different values of K vs. those of IB-KNN CBF and UC-CF on the MovieLens dataset

$K = 5$, the mean precision, is equal to 48.16%. Whereas, for $K = 1$, it has an average of 20.72% showing a drop of the rating prediction accuracy around 40% *vs.* an exceeding about 16% against the IB-KNN baseline. It should be pointed out that the performances of UC-CF approach are the lowest. This is because the user-based data appear to be sparser. Indeed, it is very unlikely that a movie has only been rated by 1 or 2 users, and highly possible that a user only rates 1 or 2 movies. Interestingly enough, these results highlight that our approach can better improve rate prediction accuracy, and thus rate recommendation, even for a high number of movies. Moreover, our approach achieves a good coverage, since it produces predictions for 76% of rates assigned to movies contained in the test dataset. Our approach successfully captures the relationship between users, movies and the rates values. From the result, we can see that our approach can generate better performance than those of three well known approaches. It can reach a highest Recall at certain K and this is greatly due to the highly independence of our approach on the characteristic of the dataset. Since no pre-processing stage is made on the data before performing the different phases.

3.4 Online Evaluation

We present in Fig. 3 the runtime of our approach. Since it is hard to measure the exact runtime of the four approaches, we simulated their online execution among the MovieLens dataset with different values of K , *i.e.*, the number of predicted rates, ranging from 1 to 5. With respect to Fig. 3, the maximum value of run time of our approach is about $0.031(s)$, whereas the minimum value is around $0.02(s)$ which is efficient and satisfiable compared to the run times achieved by the baselines approaches.

4 Conclusion

In this paper, we have introduced a new approach, called *MARRA*, for movie rating recommendation. The contributions of *MARRA* are twofold. First, we have presented a representation for collaborative filtering tasks that allows the use of the three dimensional structure of movie rating databases. We hope that this will lead to further analysis of the suitability of learning algorithms on such databases. Second, we have shown that exploiting HMM and triadic concept analysis can lead to improved predictive performance. In a set of experiments with MovieLens database we have shown that *MARRA* outperforms three baselines approaches. Future experiments will reveal if further performance improvements can be achieved through the addition of unlabeled training data. We believe that additional knowledge about the similarity of users and items can be gained through the analysis of textual descriptions of movies. Our long-term goal of this work is to combine *MARRA* approach and content-based filtering techniques. Similarity between users could then be influenced by similarity between features of rated movies. We also plan to address the challenges of the big data era, the efficiency of developing recommender system approaches must be improved.

References

1. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* **17**, 734–749 (2005)
2. Baeza-Yates, R., Berthier, R.N.: *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston (1999)
3. Cao, H., Jiang, D., Pei, J., Chen, E., Li, H.: Towards context-aware search by learning a very large variable length hidden Markov model from search logs. In: *Proceedings of the 18th International World Wide Web Conference*, pp. 191–200, Spain (2009)
4. Desarkar, M.S., Saxena, R., Sarkar, S.: Preference relation based matrix factorization for recommender systems. In: Masthoff, J., Mobasher, B., Desmarais, M.C., Nkambou, R. (eds.) *UMAP 2012*. LNCS, vol. 7379, pp. 63–75. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-31454-4_6](https://doi.org/10.1007/978-3-642-31454-4_6)
5. Deshpande, M., Karypis, G.: A survey of collaborative filtering techniques. *ACM Trans. Inf. Syst.* **22**(1), 143–177 (2004)

6. Harvey, M., Carman, M.J., Ruthven, I., Crestani, F.: Bayesian latent variable models for collaborative item rating prediction. In: Proceedings of 20th ACM International Conference on Information and Knowledge Management, pp. 699–708. ACM (2011)
7. He, Q., Jiang, D., Liao, Z., Hoi, S.C.H., Chang, K., Lim, E., Li, H.: Web query recommendation via sequential query prediction. In: Proceedings of the IEEE International Conference on Data Engineering, pp. 1443–1454. IEEE, USA (2009)
8. Herlocker, J., Konstan, J.A., Riedl, J.: An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Inf. Retr.* **5**(4), 287–310 (2002)
9. Herlocker, L., Konstan, A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Theoretical Models, pp. 230–237. ACM (1999)
10. Hotho, A.: Data mining on folksonomies. In: Armano, G., de Gemmis, M., Semeraro, G., Vargiu, E. (eds.) *Intelligent Information Access. Studies in Computational Intelligence*, vol. 301, pp. 57–82. Springer, Heidelberg (2010)
11. Jäschke, R., Hotho, A., Schmitz, C., Ganter, B., Stumme, G.: Discovering shared conceptualizations in folksonomies. *Web Semant.* **6**, 38–53 (2008)
12. Lehmann, F., Wille, R.: A triadic approach to formal concept analysis. In: Ellis, G., Levinson, R., Rich, W., Sowa, J.F. (eds.) *ICCS-ConceptStruct 1995. LNCS*, vol. 954, pp. 32–43. Springer, Heidelberg (1995). doi:[10.1007/3-540-60161-9_27](https://doi.org/10.1007/3-540-60161-9_27)
13. Pazzani, M.J.: A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.* **13**(5), 393–408 (1999)
14. Rabiner, L.: A tutorial on hidden Markov models and selected applications in speech recognition. *IEEE* **77**(2), 257–286 (1989)
15. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of 10th International Conference on World Wide Web, pp. 285–295. ACM (2001)
16. Trabelsi, C., Jelassi, N., Ben Yahia, S.: Scalable mining of frequent tri-concepts from folksonomies. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) *PAKDD 2012. LNCS*, vol. 7302, pp. 231–242. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-30220-6_20](https://doi.org/10.1007/978-3-642-30220-6_20)
17. Yoo, J., Choi, S.: Bayesian matrix co-factorization: variational algorithm and cramer-rao bound. In: Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M. (eds.) *ECML PKDD 2011. LNCS*, vol. 6913, pp. 537–552. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-23808-6_35](https://doi.org/10.1007/978-3-642-23808-6_35)