Knowl Inf Syst
DOI 10.1007/s10115-014-0776-5

REGULAR PAPER

Extended feature combination model for recommendations in location-based mobile services

Masoud Sattari · Ismail Hakki Toroslu · Pinar Karagoz · Panagiotis Symeonidis · Yannis Manolopoulos

Received: 7 September 2013 / Revised: 22 May 2014 / Accepted: 27 July 2014

Abstract With the increasing availability of location-based services, location-based social networks and smart phones, standard rating schema of recommender systems that involve user and item dimensions is extended to three-dimensional (3-D) schema involving context information. Although there are models proposed for dealing with data in this form, the problem of combining it with additional features and constructing a general model suitable for different forms of recommendation system techniques has not been fully explored. This work proposes a technique to reduce 3-D rating data into 2-D for two reasons: employing already developed efficient methods for 2-D on a 3-D data and expanding it with additional features, which are usually 2-D also, if it is necessary. Our experiments show that this reduction is effective. The proposed 2-D model supports content-based, collaborative filtering and hybrid recommendation approaches effectively, whereas we have achieved the best accuracy results for pure collaborative filtering recommendation model. Since our method was built on efficient singular value decomposition-based dimension reduction idea, it also works very efficiently, and in our experiments, we have obtained better run-time results than standard methods developed for 3-D data using higher-order singular value decomposition.

M. Sattari · I. H. Toroslu · P. Karagoz (✉)
Middle East Technical University, Ankara, Turkey
e-mail: karagoz@ceng.metu.edu.tr

M. Sattari
e-mail: msattari@ceng.metu.edu.tr

I. H. Toroslu
e-mail: toroslu@ceng.metu.edu.tr

P. Symeonidis · Y. Manolopoulos
Aristotle University, Thessaloníki, Greece

P. Symeonidis
e-mail: symeon@csd.auth.gr

Y. Manolopoulos
e-mail: manolopo@csd.auth.gr

Keywords Feature combination · Extended feature combination · Location-based social networks (LBSN) · Recommendation · Tensor · Singular value decomposition (SVD) · Higher-order singular value decomposition (HOSVD)

1 Introduction

Increasing use of mobile devices and new technologies in mobile communication enabled the collection of data involving context such as location and time, which affected the recommender systems as well. In a typical collaborative filtering recommendation model, the input is the rating matrix corresponding to users and items. However, in many real-world data, there are additional features that could be useful. Mainly, the following extensions are possible:

- In recent applications, the rating data include additional dimensions to user and item in the form of context information mostly involving time and location. Typically, the context is time, or place, or both. This addition changes the main data structure of the problem from 2-D to 3-D (or even higher). Some recent studies on this direction are [1,2].
- In some applications, there are additional data about the objects of the dimensions that could be useful. Usually, these data represent the relationships, more specifically similarities, among the objects of the dimensions. Indeed, when there are no rating data, only these similarities are used that could be interpreted as a form of content-based filtering. Recent studies on extending rating data with additional features for standard 2-D rating matrix can be found in [33,41].
- There are also some applications (hybrid models) with both multidimensional rating data and additional features' data on the objects of the dimensions [38,40].

The main objective of this paper is to develop a uniform model suitable for all the models explained above. To achieve this, multidimensional rating data (we will focus on 3-D form) will be reduced to 2-D form, which can potentially be extended with the additional features.

The extension of 2-D rating matrix with additional features has been introduced in [33]. It has also been shown that the approach is quite effective and outperforms other more complicated solutions, such as [41]. There are studies that specifically attempt to deal with 3-D rating data input [37,40]. In this work, we show that 3-D rating matrix can be reduced to three 2-D matrices with a very little information loss. Our evaluations show that our reduction-based model outperforms 3-D-based model inspired from [37].

Some methods use ratings that are given by users to find similar objects (generally users and items) within the system to fulfill the recommendation [11,18]. Some others incorporate the context-based information about users and items to catch a probable similarity between them, which is used to find like-minded users and similar items [38]. Furthermore, some methods try to combine several sources of data with different concepts to improve the precision of the final results [35]. The latter one is indeed the approach we have extended in our previous work in [33], and in this paper, we aim to further extend the given approach.

Main contributions of this paper are as follows:

- Defining a 2-D matrix structure for rating data that could be constructed from 3-D or higher-order rating data.
- Introducing a general data combination approach that could be used for content-based recommendation by using similarity matrices on objects of the different dimensions, and combining these matrices with rating matrices of collaborative filtering.

The rest of paper is organized as follows. In Sect. 2, problem definition is given. In Sect. 3, an overview of dimensionality reduction for two-dimensional matrix and tensor (third-order or higher dimensional matrix) [21] is presented. Extended feature combination (EFC) method and some examples that present different models of EFC are discussed in Sect. 4. Performed experiments and evaluating the results are shown in Sect. 5, and finally, a conclusion is given in Sect. 6.

2 Problem definition

In recommender systems' domain, conventionally 'user–item' matrix is used to denote and keep the rating given by a user for an item. However, considering the characteristics of the data and the field in which recommender system is used, these objects and even their dimensions may change. Social activity recommendation is one of the domains in which the types of objects are changed, and even another dimension is added to the original data. That is, we have user, location and activity as objects, and high-dimensional data structure (tensor) [21] is used to handle the existent ternary relation among them. In social activity recommendation, beside the original user–location–activity rating matrix, we might have access to different informative resources of data. As we have already explained in [33], it is possible to acquire several data and merge them into a single integrated matrix and propagate the effect of additional data to the core part of the matrix. Observed results for this model show that the matrix merging can influence the process of recommendation and, actually, improves the final results.

In this work, we extend the model given in [33] by exploiting even more data and merge them to create a single matrix, which further is utilized as an input to the recommendation model. One of the problems in construction of integrated matrix is to maintain its structure. That means we have to combine additional data so that, in addition to making use of its informative content, we should keep the structure of final matrix in a rectangular shape. In order to maintain its structure, we might need to insert *zero value* to the portion where its corresponding data are not available.

In practice, the main data are a 3-D tensor $\mathcal{A}$ having user, location and activity in each dimension. Each entry of this tensor contains a rating value corresponding to user–location–activity triple. The first step of this work is to reduce these 3-D data into a 2-D matrix form. In order to generate 2-D matrices, namely activity–user, location–user and location–activity matrices from 3-D tensor $\mathcal{A}$, it should be aggregated over location, activity and user dimensions, respectively. For instance, to obtain location–activity matrix, the original tensor is aggregated over user's dimension. To calculate the rating value of an entry in location–activity matrix, we use mean value of ratings for all users who have rated activity a in location l. Note that, the mean value is calculated over the entries that have nonzero rating values.

In the model which is introduced in Fig. 1, we propose to utilize location–location, activity–activity and user–user similarity matrices as well as activity–user, location–activity and location–user data which have been obtained from the 3-D tensor $\mathcal{A}$.

2.1 User–User similarity matrix

Similarity among users has a significant effect on the result of neighborhood-based recommendation systems such as social networks as reported in [3]. In addition, several distance metrics have been proposed in the literature such as *Graph Distance*, *Common Neighbors*, *Jaccard Coefficient, Pearson Correlation* and *Adamic & Adar* [13]. The impact of some of them on recommendation systems has been discussed in [6] and [31] as well. In this study, User–User similarity matrix has been constructed by using the friendship network among the users. We have conducted tests using several distance metrics. However, Jaccard Coefficient which is defined in Eq. (1) produced the best result for our data sets.

$$Jaccard's\ Coefficient\,(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \tag{1}$$

In this definition, $\Gamma(x)$ and $\Gamma(y)$ are first level neighbors of x and y. Figure 2 shows a small sample of friendship network out of 149 users' friendship graph.

In order to find the similarity between User1 (u_1) and User2 (u_2) in this friendship network, we first find neighbors of each user and then calculate *Jaccard* similarity as given in below calculations:

$$\Gamma\,(u_1) = \{u_3, u_5, u_6\} \quad \Gamma\,(u_2) = \{u_3, u_4\}$$
$$similarity\,(u_1, u_2) = \frac{|\{u_3, u_5, u_6\} \cap \{u_3, u_4\}|}{|\{u_3, u_5, u_6\} \cup \{u_3, u_4\}|} = 0.25$$

2.2 Activity–Activity similarity matrix

As also stated in various resources in the literature [41], activities that we do in normal life are not independent and have a relation. A simple method to find the correlation between

Fig. 1 Proposed integrated matrix (M) model

	Activity	**Location**	**User**
Activity	Activity Activity ***A***		
Location	Location Activity ***B***	Location Location ***C***	
User	User Activity ***D***	User Location ***E***	User User ***F***

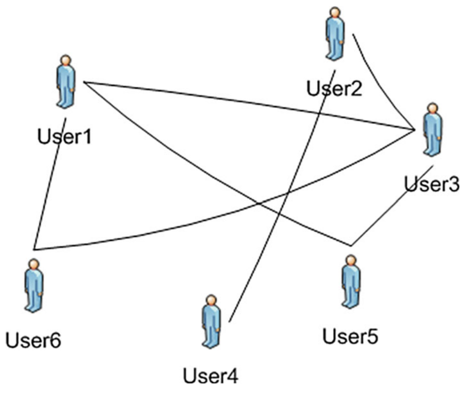


Fig. 2 Users' friendship network

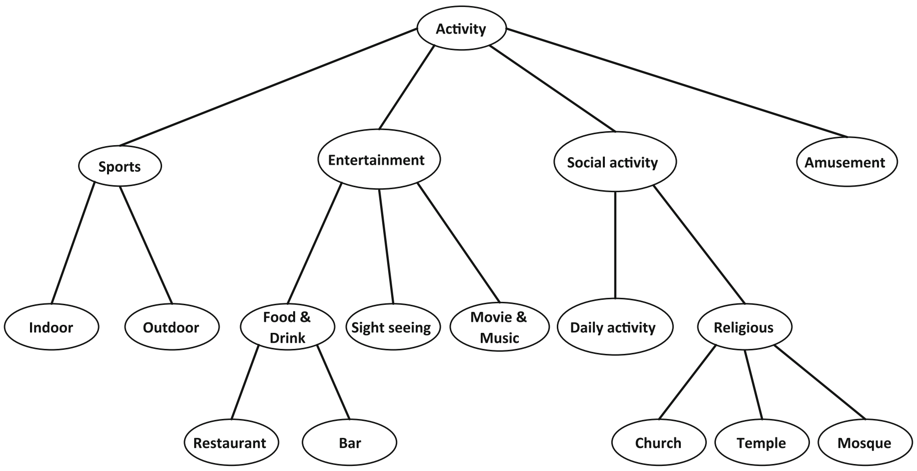


Fig. 3 Simple tree structure of activities

Table 1 Sample location table

Location name	X coordinate	Y coordinate	Address	District	City
Starbucks	22.95	40.63	Egnatias 123	City Center	Thessaloniki
Goodys	22.93	40.63	Dodekanisou 7	Vardaris	Thessaloniki
McDonalds	23.73	37.97	Ermou 2	Sintagma Square	Athens
Hard Rock Cafe	23.73	37.97	Filellinwn 18	–	Athens

activities is proposed in [40], such that Web is searched for any two activities and from retrieved total and related Web sites, similarity between these activities is calculated. Another method, which may help us to find this relation, is using activity ontology. In this method, each activity is denoted as a vertex and a relation between two vertices is shown by an edge. However, the conceptual structure may be a simple tree structure as shown in Fig. 3 or may be obtained from available ontologies in the Web. *WordNet* is a large lexical database of English nouns, verbs, adjectives and adverbs, which labels the semantic relations among words [27], and it can be used as an ontology for activities in our approach. In order to find similarity between words from WordNet, several distance metrics have been proposed in the literature [16,17,22,23,30]. In this work, we utilize these metrics to find similarity between activities.

2.3 Location–Location similarity matrix

In social activity recommendation systems and in location-based social networks, coordinates of visited locations are kept in terms of geographical altitude and longitude. Further information such as location address, district and city may be available as well. In order to find the similarity between two locations, it is sufficient to use simple Euclidian distance calculation. Table 1 shows sample location information from our data set.

3 Preliminaries in dimensionality reduction

In this section, we give an overview of background techniques that are utilized later in our approach.

3.1 Singular value decomposition (SVD) overview

Singular Value Decomposition (SVD) [21] is a well-known matrix factorization technique that factorizes an $m \times n$ matrix R into three matrices as given in Eq. (2).

$$R_{m \times n} = U_{m \times m} S_{m \times n} V^T_{n \times n} \tag{2}$$

In this equation, U (left singular vector) and V (right singular vector) are two orthogonal matrices of size $m \times m$ and $n \times n$, respectively, and V^T is transpose of V. The eigenvectors of RR^T and $R^T R$ make up columns of U and V correspondingly. Also, the singular values in S are square roots of eigenvalues of RR^T or $R^T R$ [25,32].

S is a matrix of size $m \times n$ having all singular values of matrix R as its diagonal entries. All the entries of S are real numbers (if matrix R is a real matrix) and stored in decreasing order of their magnitude. SVD provides the best lower-rank approximations of the original matrix R in terms of difference in *Frobenius* norm [12].

We utilize SVD in recommender systems to perform two different tasks: First, we use it to capture latent relations among users and items that allow us to compute the predicted likeliness of a certain product by a customer. Second, we use SVD to produce a *low-dimensional* representation of the original user–item space and then compute neighborhood in the reduced space [32]. In general, SVD is used for dimensionality reduction, so that, with selecting the k largest values of S and k columns of U and V (or k rows of V^T) and by multiplying them, respectively, we can represent matrix R with reduced-rank matrix R^k, which has rank of k ($k \leq Rank(R)$) as it is shown in Eq. (3).

$$R^k = U_{m \times k} \quad S_{k \times k} \quad V^T_{k \times n} \tag{3}$$

However, dimensionality reduction is a data lossy approach. It means that the more we reduce the dimensionality, the bigger the portion of original data that is lost. It is possible to determine the percentage of loss in advance. According to study in [5], to maintain p percent of original data, we can select k highest singular values from matrix S regarding the relations in Eq. (4).

$$p = \frac{\sum_{i=1}^{k} S_i}{\sum_{all} S} \times 100 \tag{4}$$

For instance, if singular values of a matrix are $S = \{10, 7.2, 5.4, 3.2, 3.1, 2.2, 1.9, 1.3, 0.8, 0.4, 0.1\}$; in order to keep 80 % of original data, we should reduce the rank of matrix to $k = 5$ since:

$$\frac{\sum_{i=1}^{5} S_i}{\sum_{all} S} = \frac{28.9}{35.6} \times 100 \approx 80$$

For simplicity, in the rest of the paper we show the reduced-rank components with U_k, S_k and V_k^T.

3.2 Tensor-based recommendation

A *Tensor* is a multidimensional matrix which can be used as a rating storage in recommendation systems. In this work, the dimensions of 3-order (3-D) tensor corresponds to user,

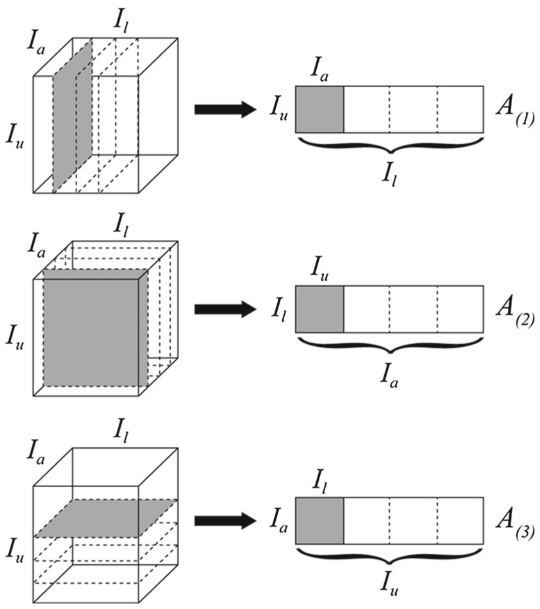


Fig. 4 Unfolding of 3-order tensor [37]

location and activity, respectively. Each entry of form (i, j, k) is a numerical value which shows the rating that is given by user i for performing activity k at location j. The high-order singular value decomposition (HOSVD) [20] is generalized to apply SVD to tensors. Cutting-edge research is conducted by Symeonidis et al. [37] and Marinho et al. [24], which propose a unified social tagging recommendation system model exploiting HOSVD.

3.2.1 Dimensionality reduction in tensor

Initially, the method in [37] starts the tensor reduction process by giving matrix-shape representation of rating tensor $\mathcal{A}$. Applying *unfolding* to $\mathcal{A}$ yields three matrices A_1, A_2 and A_3, which are defined as follows:

$$A_1 \in R^{I_u \times I_l I_a}, \quad A_2 \in R^{I_l \times I_u I_a}, \quad A_3 \in R^{I_u I_l \times I_a}$$

Unfolding is a method of representing a flat mode of tensor $\mathcal{A}$ by rearranging each slice (page) of $\mathcal{A}$ as a column of corresponding flat mode in a specific dimension as shown in Fig. 4. Each of A_1, A_2, and A_3 is called 1-mode, 2-mode and 3-mode matrix unfolding of $\mathcal{A}$.

In the next step, regular SVD is applied to each of unfolded matrices (A_1, A_2, A_3), which is shown in Eq. (5).

$$A_1 = U^{(1)}.S^{(1)}.(V^{(1)})^T, \qquad A_2 = U^{(2)}.S^{(2)}.(V^{(2)})^T, \qquad A_3 = U^{(3)}.S^{(3)}.(V^{(3)})^T \tag{5}$$

Despite regular SVD in which we select k singular values to reduce the rank of matrix, for each mode of tensor we should determine three distinct parameters k_1, k_2 and k_3 since SVD is applied to A_1, A_2 and A_3 independently. The parameters k_1, k_2 and k_3 are empirically chosen by preserving a percentage of information of the original matrices $S^{(1)}$, $S^{(2)}$ and $S^{(3)}$ after appropriate tuning [37]. As we discussed in previous sections, we can determine the percentage that we desire to maintain from original data. In [37], Symeonidis et al. mention that the percentage is usually set to be 50 % of each unfolded matrix.

After the rank of left singular vectors is determined for all three unfolded matrices, construction of the core tensor S is started. Core tensor acts as an intermediate, which can control the interaction between user, location and activity. Core tensor S is constructed as shown in Eq. (6).

$$S = \mathcal{A} \times_1 (U_{k1}^{(1)})^T \times_2 (U_{k2}^{(2)})^T \times_3 (U_{k3}^{(3)})^T \tag{6}$$

In this equation, $\mathcal{A}$ is the original tensor, and $U_{k1}^{(1)}$, $U_{k2}^{(2)}$ and $U_{k3}^{(3)}$ are reduced matrices of left singular vectors in Eq. (5) by rank of k_1, k_2 and k_3, respectively.

In addition, $\times_n$ is defined as the n-mode product of an N-order tensor $\mathcal{A} \in R^{I1 \times I2 \times \ldots \times IN}$ by a matrix$U \in R^{Jn \times In}$. The result of product is an $I_1 \times I_2 \times \ldots \times I_{n-1} \times J_n \times J_{n+1} \times \ldots \times I_N$-tensor whose entries are defined as presented in Eq. (7).

$$(A \times_n U)_{i_1 i_2 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 i_2 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} u_{j_n i_n} \tag{7}$$

Finally, tensor $\hat{\mathcal{A}}$ is constructed from product of core tensor S and dimensionally reduced left singular vectors of each unfolding mode as demonstrated in Eq. (8).

$$\hat{\mathcal{A}} = S \times_1 U_{k1}^{(1)} \times_2 U_{k2}^{(2)} \times_3 U_{k3}^{(3)} \tag{8}$$

As discussed in [37], authors expect to extract latent semantic relation from $\hat{\mathcal{A}}$ as well as the value of missing entries in $\mathcal{A}$. A visual description of HOSVD is depicted in Fig. 5.

As an example, we have a sample rating tensor T containing 5 users, 5 locations and 4 activities as illustrated in Fig. 6. This running example is used throughout the paper.

In HOSVD, the process begins by producing unfolding matrices of T, and then, SVD is applied to each mode which results the matrices that are shown in Eq. (5). Applying HOSVD to this tensor, according to Fig. 5 and reducing to rank 2, yields three matrices $U_2^{(1)}$, $U_2^{(2)}$, $U_2^{(3)}$ (subscript 2 shows the rank of each matrix) and a core tensor C which are shown in Figs. 7 and 8.

3.2.2 *Recommendation steps for tensor-based method*

Tensor $\mathcal{A}$ presents the associations among the entities and serves as rating data model that is used during the recommendation. In order to predict the rating value that is given by user i for performing activity k in location j, firstly, we find most similar neighbors of user i from reduced tensor. Since each of $U_{k1}^{(1)}$, $U_{k2}^{(2)}$ and $U_{k3}^{(3)}$ corresponds to user, location and activity, we use them to find similarity among users, locations and activities, respectively. Cosine similarity is used as the metric to accomplish this task. Secondly, we freeze the indices of location and activity and retrieve the rating values given by similar neighbors of user i for

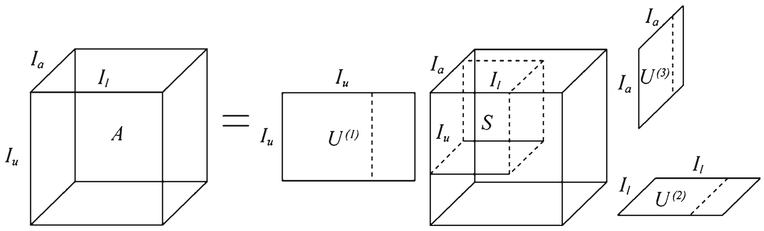


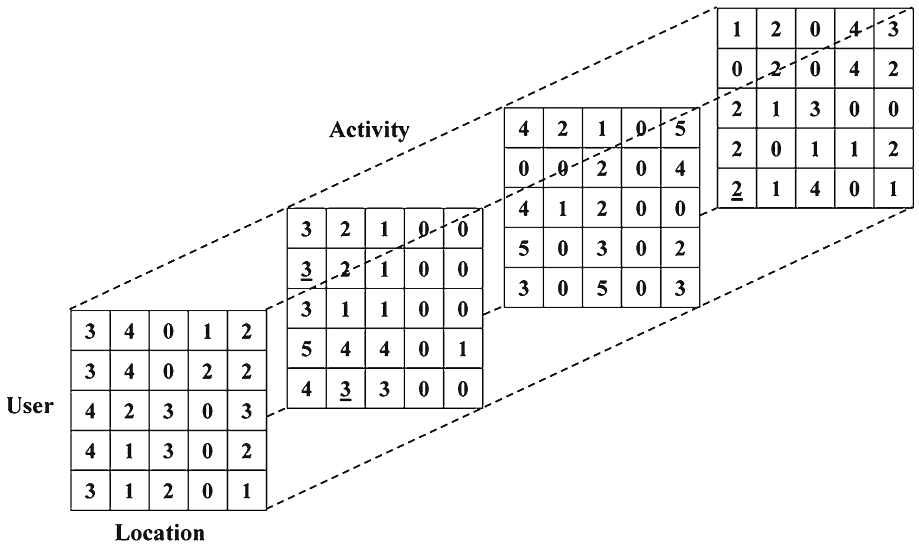
Fig. 5 Visualization of tensor reduction using HOSVD [37]

doing activity k in location j. Finally, we combine the rating values of each neighbor utilizing Eq. (11), whose details are given in the next subsection, to produce a partial prediction from user. Likewise, the routine of index freezing and exploring the similar neighbors' rating is applied to location and activity.

As it is discussed in the previous subsection, applying HOSVD to tensor $\mathcal{A}$ yields three matrices $U^{(1)}$, $U^{(2)}$ and $U^{(3)}$. Considering the relation among entries, $U^{(1)}$ is the left singular vectors matrix of A_1 that is the result of aggregation on original tensor over location and activity. In other words, each row of $U^{(1)}$ is a vector for observation of user having location and activity together as the second dimension. This fact also holds for $U^{(2)}$ and $U^{(3)}$ where the former one is observations of location regarding user and activity and the latter one is observations of activities regarding user and location. This fact allows us to use these three matrices as a source to find similarity among entries. For example, choosing $U^{(1)}$ and any similarity metric (Euclidean, Cosine…), we can construct User–User similarity matrix.

3.2.3 Rating prediction

In the prediction process, once the neighbors are determined, the neighbors' ratings must be combined to produce the target prediction. A trivial method uses an average neighbors'



3	4	0	1	2
3	4	0	2	2
4	2	3	0	3
4	1	3	0	2
3	1	2	0	1

3	2	1	0	0
3	2	1	0	0
3	1	1	0	0
5	4	4	0	1
4	3	3	0	0

4	2	1	0	5
0	0	2	0	4
4	1	2	0	0
5	0	3	0	2
3	0	5	0	3

1	2	0	4	3
0	2	0	4	2
2	1	3	0	0
2	0	1	1	2
2	1	4	0	1

Fig. 6 Sample rating tensor (T)

0.46	0.57
0.38	0.55
0.37	-0.34
0.53	-0.31
0.47	-0.40

(a) User ($U_2^{(1)}$)

0.70	-0.16
0.39	0.36
0.46	-0.56
0.09	0.57
0.37	0.46

(b) Location ($U_2^{(2)}$)

0.56	0.13
0.48	0.68
0.57	-0.29
0.36	-0.65

(c) Activity ($U_2^{(3)}$)

Fig. 7 Components of tensor, **a** user ($U_2^{(1)}$), **b** location ($U_2^{(2)}$), **c** activity ($U_2^{(3)}$)

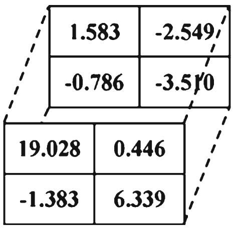


Fig. 8 Core tensor C

ratings so that rating of user a for item $b(P_{a,b})$ is calculated using ratings given by n neighbors of a for item $b(P_{j,b})$ as shown in Eq. (9).

$$P_{a,b} = \frac{\sum_{j=1}^{n} P_{j,b}}{n} \tag{9}$$

However, this is nonpersonalized method which does not consider the correlation among users, location or activity. In order to utilize the correlation among users, when it is available, in a better way, use of weighted average is proposed in [34], which takes similarity into consideration as given in Eq. (10). In this formula, $w_{a,j}$ is the similarity among user a and its neighbor j.

$$P_{a,b} = \frac{\sum_{j=1}^{n} P_{j,b}.w_{a,j}}{\sum_{j=1}^{n} w_{a,j}} \tag{10}$$

Nevertheless, there might be subjective factors in real-world problems. There are some users that always give high scores to whatever they rate for. In contrast, some other users tend to give very low scores. Therefore, the deviation of ratings from rating mean of each entry should be considered as an effective factor to prediction step. In [14], deviation from average rating of all entries (either user or activity or location) is calculated and incorporated in the prediction as given in Eq. (11) in which $\bar{r}_a$ and $\bar{r}_j$ are rating means of user a and its neighbor user j, respectively.

$$P_{a,b} = \bar{r}_a + \frac{\sum_{j=1}^{n} (P_{j,b} - \bar{r}_j).w_{a,j}}{\sum_{j=1}^{n} w_{a,j}} \tag{11}$$

We have made experiments with all three equations discussed above. As the result of the analysis, we employ Eq. (11) to combine the results obtained from each neighborhood, since it produced the best accuracy result.

Indeed, usually more than one prediction is made for each entry. Therefore, the final prediction for a given entry in tensor T can be calculated using a weight factor and p predictions as follows:

$$\hat{T}\left(u_i, l_j, a_k\right) = \frac{\boldsymbol{W.Prediction}}{\sum_{i=1}^{p} w_i} \tag{12}$$

In this equation, $\boldsymbol{W}$ is weight vector, $\boldsymbol{Prediction}$ is vector of predictions and the operator is matrix dot product. Also, $\hat{T}$ demonstrates the predicted value for (u_i, l_j, a_k) in tensor T.

Fig. 9 Rating mean of users, locations, and activities

	U1	U2	U3	U4	U5
Users rating mean	2.53	2.58	2.31	2.67	2.57

	L1	L2	L3	L4	L5
Locations rating mean	3.22	2.06	2.44	2.4	2.36

	A1	A2	A3	A4
Activities rating mean	2.5	2.57	3.07	2.11

(a)

Sorted similarity values	U1	U2	U3	U4	U5
	0.99	0.99	0.99	0.98	0.99
	0.15	0.08	0.98	0.98	0.98
	0	0	0	0.15	0
	-0.03	-0.09	-0.06	0.08	-0.03
	-0.06	-0.13	-0.13	0	-0.09

(b)

L1	L2	L3	L4	L5
0.79	0.99	0.77	0.86	0.99
0.57	0.78	0	0.78	0.86
0.45	0.57	-0.05	0	0.45
0	0	-0.19	-0.05	0
-0.06	-0.05	-0.66	-0.66	-0.19

(c)

A1	A2	A3	A4
0.76	0.75	0.82	0.82
0.75	0.13	0.75	0.26
0.26	0	0.13	0
0	-0.43	0	-0.43

Fig. 10 Similarity values of HOSVD, **a** sorted value of similarity from U_1, **b** sorted value of similarity from U_2, **c** sorted value of similarity from U_3

3.2.4 Example for tensor-based recommendation system

Considering the sample rating tensor T in Fig. 6, we explain the procedure of predicting the values of randomly chosen entries $a_1 = (2, 1, 2)$, $a_2 = (5, 2, 2)$ and $a_3 = (5, 1, 4)$. Before applying HOSVD, the original values of a_1, a_2 and a_3 are replaced with zero. We also need to know rating mean for each user, location and activity since it is needed in prediction process as shown in Eq. (11). For instance, to compute rating mean for all user, we first find the entries with nonzero ratings in sample tensor T. Then, we sum up T over second and third dimensions and divide the result by number of entries that are greater than zero. The rating mean of users, locations and activities for T is presented in Fig. 9. In this figure, U1, L1 and A1 correspond to user1, location1 and activity1.

In order to find similarity between entries of matrices $U_2^{(1)}$, $U_2^{(2)}$ and $U_2^{(3)}$, cosine similarity is used as the distance metric. The similarity matrices are shown in Fig. 10. In each of Fig. 10a–c, columns represent users, locations and activities, respectively. Entries in each column show sorted similarity values of the item in that column to another items. Corresponding ID's for items (users, locations and activities) are given in the index matrices of Fig. 11. Note that, similarity between each item and itself is equal to 1 (distance between each item and itself is 0), and in order to show a correct relation among similar items, the values altered so that each item is the least similar item to itself.

In the first step of prediction, for randomly selected entry $a_1 = (2, 1, 2)$, we keep values of location1 and activity2, and then, we find the most similar user (the size of neighborhood in this sample is 1) to user2 from column U2 of Fig. 11a (column U2 shows similar users to user2 sorted from the most similar to the least similar) which is user1. Its corresponding similarity value is 0.99 according to column U2 of Fig. 10a. The rating value of (1, 1, 2) with regard to sample tensor T is equal to 3. To accomplish the prediction, we also need to know

ID's of sorted values	U1	U2	U3	U4	U5
	2	1	5	5	3
	4	4	4	3	4
	1	2	3	1	5
	5	5	1	2	1
	3	3	2	4	2

(a)

L1	L2	L3	L4	L5
3	5	1	5	2
2	4	3	2	4
5	1	2	4	1
1	2	5	1	5
4	3	4	3	3

(b)

A1	A2	A3	A4
3	1	4	3
2	3	1	1
4	2	2	4
1	4	3	2

(c)

Fig. 11 Similarity indices of HOSVD **a** sorted index of similarity from U_1, **b** sorted index of similarity from U_2, **c** sorted index of similarity from U_3

rating mean of users 2 and 1, which is 2.58 and 2.53 as computed in Fig. 9. Using Eq. (11), first prediction can be calculated as:

$$P_{User} = 2.58 + \frac{(3 - 2.53) * 0.99}{0.99} = 3.05$$

In the second step, we maintain the values of user2 and activity2 and then find the most similar location to location1 referring to column L1 of Fig. 11b (column L1 shows similar locations to location1 sorted from the most similar to the least similar) which in this case is location3 with similarity value of 0.79 as presented in column L1 of Fig. 10b. The value of $T(2, 3, 2)$ is 1, and rating mean of locations 1 and 3, according to Fig. 9, is 3.22 and 2.44. Therefore, the prediction is calculated as follows:

$$P_{Location} = 3.22 + \frac{(1 - 2.44) * 0.79}{0.79} = 1.79$$

Finally, we freeze values of user2 and location1 and then find the most similar activity to activity2 using column A2 of Fig. 11c which is activity1 with similarity value of 0.75. The value of $T(2, 1, 1)$ is 3, and rating mean of activities 2 and 1, according to Fig. 9, is 2.57 and 2.5. Final prediction is computed as:

$$P_{Activity} = 2.57 + \frac{(3 - 2.5) * 0.75}{0.75} = 3.07$$

At the end, utilizing the formula introduced in Eq. (12), the final prediction for $a_1 = (2, 1, 2)$ can be calculated as:

$$\hat{T}\,(2, 1, 2) = \frac{[1\ 1\ 1]\,.\,[3.05\ 1.79\ 3.07]^T}{3} = 2.64$$

4 Proposed method: extended feature combination (EFC)

When SVD is applied to a single user–item matrix, it produces left singular vectors U and right singular vectors V matrices. Each row of U represents a user object, and each row of V represents an item object which both of them together with S (singular values) can be used to reduce the dimension of original matrix and finding the similarity within users and items as well. In this work, we intend to make use of additional data and propagate their influences to other data matrices in order to improve the accuracy of recommendation.

Generally speaking, the basic step in most of recommendation systems begins by predicting unknown values in rating matrix. Afterward, item recommendation is performed

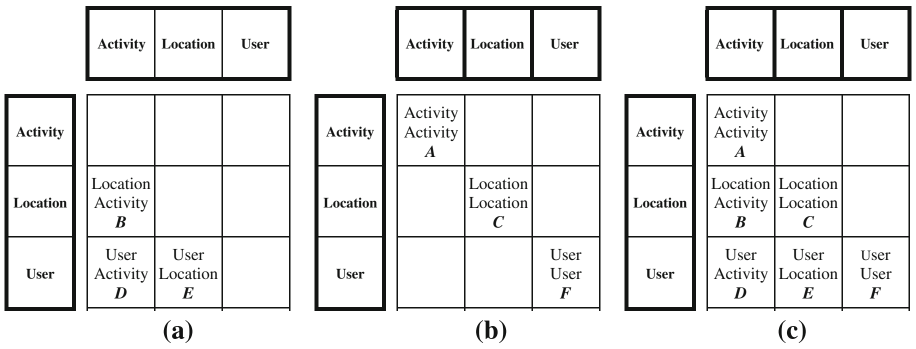


Fig. 12 Extended feature combination (EFC) models overview (empty submatrices contain zero values), **a** model 1: collaborative filtering recommendation system, **b** model 2: content-based recommendation system, **c** model 3: hybrid recommendation system

according to some threshold of estimated rating. Earlier we mentioned that, for a given user–location–activity entry (u_i, l_j, a_k) in rating tensor T, HOSVD-based technique finds similar entries to u_i, l_j and a_k from reduced-rank matrices and then refers to original tensor to find corresponding rating values. In EFC, we propose to find similarities from two-dimensional integrated model. To do so, we firstly construct the model from available matrices (details are given below in Sect. 4.1). Secondly, we apply SVD to extract the similarity matrices, and at the end, we calculate the value of missing entry using similarity matrices and referring to original rating tensor.

4.1 Model construction

Integrated matrix M as introduced in Fig. 1 can be used to define three different models as shown in Fig. 12. They are as follows:

- By using the three 2-D matrices (B, D, E) obtained from the original tensor data $\mathcal{A}$, it is possible to construct purely collaborative filtering recommendation system (Fig. 12a).
- Likewise, by using three similarity matrices (A, C, F) calculated from feature data of users, locations and activities, we may construct a content-based recommendation system (Fig. 12b).
- Finally, by combining the former two methods, we define a hybrid recommendation system as shown in Fig. 12c.

Note that, since the values of other matrices B, D and E are in the range of [0, 5] and in order to keep the magnitude of all entries in the same range, we scale all three similarity matrices A, C and E within the range of [0, 5] using min–max normalization method [13].

4.2 Similarity calculation

Once the final integrated matrix has been obtained, we proceed with applying SVD and finding similarity for a given user, location and activity (u_i, l_j, a_k) in rating tensor T. Once SVD is applied to integrated matrix, matrices U, S and V^T are produced. By selecting first r columns of U and first r rows of V^T and also keeping r greatest singular values of S, we may reduce the rank of the integrated matrix to r. However, the value of parameter r practically affects the total accuracy of model and needs to be determined in advance. We can determine the value of r using Eq. (4) in order to maintain 50 % of the original data.

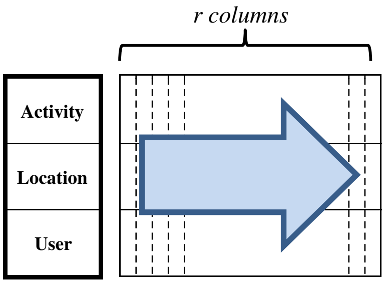


Fig. 13 Reduced-rank visualization of U

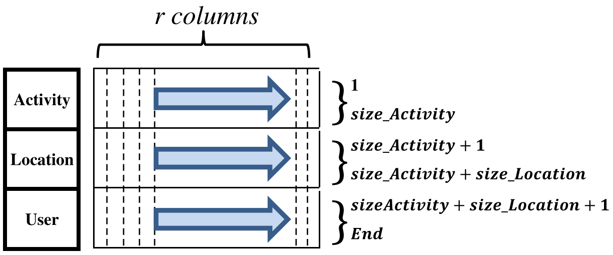


Fig. 14 Visualization of matrix trimming

As it is shown in Fig. 13, we may think about U_r as row vectors in which columns are representing hidden attributes that reflect the latent relations within data set. Hence, we can trim U_r such that only required information can be selected. If we define $size\_Activity$ to be the number of activities in the data set and select first $size\_Activity$ rows of U_r, then we get a matrix that its rows represent activities and its columns show the latent attributes of data set. Furthermore, we call the matrix $U_r\_Activity$ and use it to calculate similarities between a distinct activity and other activities.

We define $size\_Location$ and $size\_User$ to be the numbers of locations and users in our data set. Similarly, by proper trimming of rows in Fig. 13, we get matrices $U_r\_Location$ and U_r_User whose rows represent locations and users correspondingly and columns show the latent attributes of data set. Figure 14 shows the overall schema of trimming process.

Analogously, we can think about V_r^T as column vectors in which rows are representing latent attributes of data set (Fig. 15). By a proper trimming and selecting columns of V_r^T according to size of user, location and activity, we can extract matrices $V_r^T\_Activity$, $V_r^T\_Location$ and $V_r^T\_User$ whose columns are representing activities, locations and users, and rows are representing hidden relations of the data set. Trimming process of V_r^T is very similar to Fig. 14 with a difference that instead of row-wise selection, we trim it column-wise.

So far, by exploiting U_r and V_r^T and performing trimming on them, we have achieved 6 matrices which we can directly use to calculate the required similarity matrices. Cosine similarity is a typical and efficient proximity metric, which has been reported to outperform other measures in domain of recommendation systems, and thus, we utilize it at this step as well [14]. Suppose that, we have chosen U_r_User as the matrix that we want to calculate cosine similarity. Column c_i of the similarity matrix shows the similarity between user u_i and all other users in data set. However, we are not interested in all of those users and seeking for the most similar users to user u_i (it means they have the highest values in column c_i of similarity matrix). In order to find similar users conveniently, we sort each column in

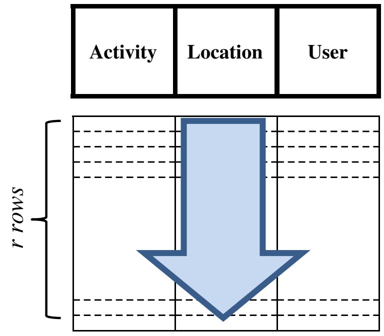


Fig. 15 Reduced-rank visualization of V^T

Table 2 Similarity matrices

	Trimmed matrix	Similarity matrix (sorted)	Comment
1	$U_r\_Activity$	$U\_Activity\_sim$	Activity–Activity similarity produced from U_r
2	$U_r\_Location$	$U\_Location\_sim$	Location–Location similarity produced from U_r
3	U_r_User	U_User_sim	User–User similarity produced from U_r
4	$V_r^T\_Activity$	$V\_Activity\_sim$	Activity–Activity similarity produced from V_r^T
5	$V_r^T\_Location$	$V\_Location\_sim$	Location–Location similarity produced from V_r^T
6	$V_r^T\_User$	V_User_sim	User–User similarity produced from V_r^T

descending order of similarity magnitude and name it U_User_sim, so that the most similar user to u_i stands at the top of column c_i of matrix U_User_sim. We are going to utilize these matrices in the examples below. The trimmed matrices and corresponding sorted similarity matrices together with short comments about each one are listed in Table 2.

4.3 Rating prediction

In order to predict the rating value of an entry (u_i, l_j, a_k) in user–location–activity rating tensor T, in each step, we maintain two indices of the entry and find similar objects of remaining index using information in Table 2.

In the first step, we freeze the index values of location l_j and activity a_k for selected entry (u_i, l_j, a_k). Then, referring to Table 2, we choose similarity matrices at rows 3 and 6 (U_User_sim and V_User_sim). Using U_User_sim we select m_3 similar users to user $u_i(\{u_i^m\})$ so that:

$$\forall u_i^m \in \left\{u_i^1, u_i^2, \ldots, u_i^{m_3}\right\}, \qquad T\left(u_i^m, l_j, a_k\right) \neq 0 \tag{13}$$

Once we have found the value of $T\left(u_i^m, l_j, a_k\right)$ for desired neighborhood $\{1, \ldots, m_3\}$, first prediction is calculated as the weighted average of deviations from the similar neighbor's mean [14]. The formula is just a customized version of Eq. (11).

$$P_{User}^U = \bar{r}_{u_i} + \frac{\sum_{p=1}^{m_3}\left(T\left(u_p, l_j, a_k\right) - \bar{r}_{u_p}\right) * sim\left(u_i, u_p\right)}{\sum_{p=1}^{m_3} sim\left(u_i, u_p\right)} \tag{14}$$

Similarly, utilizing V_User_sim, we select m_6 similar users to user $u_i(\{u_i^m\})$ and compute another prediction by freezing the index values of location l_j and activity a_k.

Table 3 Estimated values and parameters

	Similarity matrix	Estimated value	Size of neighborhood
1	$U\_Activity\_sim$	$P^U_{Activity}$	m_1
2	$U\_Location\_sim$	$P^U_{Location}$	m_2
3	U_User_sim	P^U_{User}	m_3
4	$V\_Activity\_sim$	$P^V_{Activity}$	m_4
5	$V\_Location\_sim$	$P^V_{Location}$	m_5
6	V_User_sim	P^V_{User}	m_6

In the second step, we freeze the index values of user u_i and activity a_k for selected entry (u_i, l_j, a_k). Then, again referring to Table 2, we choose similarity matrices at rows 2 and 5 ($U\_Location\_sim$ and $V\_Location\_sim$). The rest of process is similar to previous step except that the size of neighborhood is m_2 and m_5.

Similarly, in step three, we freeze the index values of user u_i and location l_j for selected entry (u_i, l_j, a_k) and make predictions using similarity among activities. Table 3 shows the estimated values and parameters that are achieved from mentioned three steps.

Similar to HOSVD, the final step combines 6 predictions using Eq. (12) that are calculated from previous steps. We may have a weighted combination method which assigns separate weights for each individual prediction. However, for simplicity, we assign same weights for all predictions.

4.4 Feature combination using collaborative filtering

In this model, we aim to reconstruct a model that utilizes only the data which are extracted from rating tensor T. In other words, the condition is to integrate those matrices from Fig. 12a that reflects the property of users' rating and not any contextual information about users, locations or activities. Main idea for this reconstruction is to include the impact of only rating on prediction, without using the information coming from resources other than users' feedback.

As it is discussed in Sect. 2, we have only three matrices location–activity (B), user–activity (D) and user–location (E), which satisfy the condition above. To accomplish the reconstruction, matrices with all zero values also have to be inserted into the model. Integration of those matrices is depicted in Fig. 12a.

We construct a sample model using the rating matrix of Fig. 6 such that we only insert matrices, which are achieved from rating values into the final model as illustrated in Fig.16. Once the SVD is applied to the model and rank of it is reduced to 2, matrices in rows 2, 3, 4 and 5 of Table 2 and their corresponding similarities in Table 3 are calculated. Figure 17 presents the matrices which are obtained from applying SVD and reduced to the rank of 2 (subscripts in names show the rank of matrices). Remember that we sort the similarity matrices in descending order as are illustrated in Figs. 18 and 19 in forms of indices and values.

To predict the value of the first randomly selected entry $a_1 = (2, 1, 2)$ with rating value of 3, we freeze the index of $location = 1$ and $activity = 2$. Then, referring to column U2 of Fig. 18a (column U2 shows sorted index of similar users of user2 from the most similar to the least similar), we find the most similar user (the size of neighborhood in this case is

	Activity				Location					User				
Activity	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Locations	3.4	3.6	4	1.75	0	0	0	0	0	0	0	0	0	0
	2.4	2.4	1.5	1.5	0	0	0	0	0	0	0	0	0	0
	2.67	2	2.6	2.67	0	0	0	0	0	0	0	0	0	0
	1.5	0	0	3	0	0	0	0	0	0	0	0	0	0
	2	1	3.5	2	0	0	0	0	0	0	0	0	0	0
User	2.5	2	3	2.5	2.75	2.5	1	2.5	3.33	0	0	0	0	0
	2.75	2	3	2.67	3	2.67	1.5	3	2.67	0	0	0	0	0
	3	1.67	2.33	2	3.25	1.25	2.25	0	3	0	0	0	0	0
	2.5	3.5	3.34	1.5	4	2.5	2.75	1	1.75	0	0	0	0	0
	1.75	3.33	3.67	2	3	1.67	3.5	0	1.67	0	0	0	0	0

Fig. 16 Collaborative filtering-based combination model

-0.41	0.25
-0.42	0.26
-0.37	0.17
-0.44	0.13
-0.41	0.05

(a)

-0.41	-0.41
-0.39	-0.39
-0.47	-0.32
-0.29	0.08

(b)

-0.26	-0.67
-0.17	-0.38
-0.11	-0.31
-0.08	-0.06
-0.19	-0.36

(c)

-0.37	0.44
-0.25	0.32
-0.26	0.25
-0.15	0.25
-0.28	0.38

(d)

Fig. 17 Reduced-rank matrices, **a** U_2_User, **b** $V_2^T\_activity$, **c** $U_2\_Location$, **d** $V_2^T\_Location$

Fig. 18 Similarity indices of CF model, **a** sorted index of U_User_sim, **b** sorted index of $V\_Activity\_sim$, **c** sorted index of $U\_Location\_sim$, **d** sorted index of $V\_Location\_sim$

U1	U2	U3	U4	U5
2	1	1	3	4
3	3	2	5	3
4	4	4	1	1
5	5	5	2	2
1	2	3	4	5

(a)

A1	A2	A3	A4
2	1	2	3
3	3	1	2
4	4	4	1
1	2	3	4

(b)

L1	L2	L3	L4	L5
3	1	1	5	2
2	5	2	2	1
5	3	5	1	3
4	4	4	3	4
1	2	3	4	5

(c)

L1	L2	L3	L4	L5
2	5	1	5	2
5	1	2	2	1
3	4	5	1	4
4	3	4	3	3
1	2	3	4	5

(d)

1) to user2 which is user1. According to the sample tensor T, the rating value of new entry (1, 1, 2) is equal to 3. The similarity between user2 and its most similar neighbor (user1) is 0.99 with regard to the corresponding entry in column U2 of Fig. 19a. We also need to know the mean rating of user2 and its similar neighbor user1, which can be computed from sample tensor T. Knowing the mean rates of user2 and user1 are 2.58 and 2.53, first prediction is

U1	U2	U3	U4	U5
0.99	0.99	0.99	0.99	0.98
0.99	0.99	0.99	0.98	0.95
0.96	0.96	0.98	0.96	0.91
0.90	0.90	0.95	0.96	0.90
0	0	0	0	0

(a)

A1	A2	A3	A4
0.99	0.99	0.98	0.65
0.98	0.98	0.98	0.51
0.49	0.51	0.65	0.49
0	0	0	0

(b)

L1	L2	L3	L4	L5
0.99	0.99	0.99	0.99	0.99
0.98	0.99	0.99	0.99	0.95
0.96	0.98	0.96	0.94	0.92
0.94	0.95	0.92	0.90	0.90
0	0	0	0	0

(c)

L1	L2	L3	L4	L5
0.99	0.99	0.99	0.99	0.99
0.99	0.99	0.99	0.99	0.99
0.99	0.99	0.98	0.98	0.99
0.98	0.99	0.96	0.96	0.98
0	0	0	0	0

(d)

Fig. 19 Similarity values of CF model, **a** sorted value of U_User_sim, **b** sorted value of $V\_Activity\_sim$, **c** sorted value of $U\_Location\_sim$, **d** sorted value of $V\_Location\_sim$

calculated as follows:

$$P^{U}_{User} = 2.58 + \frac{(3 - 2.53) * 0.99}{0.99} = 3.05$$

In the second prediction, we freeze the index of $user = 2$ and $activity = 2$ and then find the most similar location to $location = 1$. However, we can find the similarity from two matrices in Fig. 18c, d, which results to calculate two separate predictions. Using column L1 of Fig. 18c (column L1 shows the sorted index of similar locations of location1 from the most similar to the least similar), the most similar location of 1 is location3 with its corresponding similarity value of 0.99 (same entry in column L1 of Fig. 19c). Rating value for the new entry (2, 3, 2) in T is 1. In addition, mean ratings of locations 1 and 3 are computed from T, which is 3.22 and 2.44. Hence, we obtain the prediction as:

$$P^{U}_{Location} = 3.22 + \frac{(1 - 2.44) * 0.99}{0.99} = 1.76$$

Analogously, using column L1 of Fig. 18d, the most similar location is location2 with similarity value of 0.99 with respect to column L1 of Fig. 19d. Rating value of $T(2, 2, 2)$ is equal to 2. Also, mean rating of locations 1 and 2 is 3.22 and 2.06. Related prediction is calculated as follows:

$$P^{V}_{Location} = 3.22 + \frac{(2 - 2.06) * 0.99}{0.99} = 3.16$$

Finally, we keep the index of $user = 2$ and $location = 1$ and find the most similar activity to $activity = 2$ utilizing column A2 of Fig. 18b which is activity1 with corresponding similarity of 0.99. Moreover, $T\ (2, 1, 1) = 3$, mean rating of activities 2 and 1 is 2.56 and 2.5, and the final prediction is:

$$P^{V}_{Activity} = 2.56 + \frac{(3 - 2.5) * 0.99}{0.99} = 3.06$$

The prediction of entry $a_1 = (2, 1, 2)$ is shown by $\hat{T}\,(2, 1, 2)$ and is calculated from partial predictions that are produced from previous steps as formulated in Eq. (12).

$$\hat{T}\,(2, 1, 2) = \frac{[1\ 1\ 1\ 1] \,.\, [3.05\ 1.76\ 3.16\ 3.06]^T}{4} = 2.76$$

4.5 Feature combination using contextual information

In this second model, we are interested in considering the effect of contextual information on accuracy of predicted values as well. In this case, contextual information is such information that is not achieved from rating matrix. In contrast, it is related to contents of users, locations and activities, which reflect their intrinsic characteristic. That information is comprised of *user–user*, *location–location* and *activity–activity* similarity matrices, which were discussed in detail before.

As the model reconstruction in Fig. 12b presents, contextual information is inserted as the main diagonal, and similar to other models, zero is inserted to the other components of the model in order to construct the big integrated matrix. The procedure of finding similarity matrices is very similar to what explained in Sect. 4.2.

As discussed in content-based model definition, in the following example we merely insert similarity matrices into the combination model. Similar to the previous method, in each step index values of two dimensions are frozen and similar entries are found using the third dimension of randomly selected entry from tensor T. Note that, regarding Table 3 and the construction of model for content-based information, we only have three similarity matrices that are available in rows 1, 2 and 3 of Table 3. Calculations are analogous to the CF model, and hence, we only present value of each prediction and final estimation of given random values.

$$\hat{T}\,(2, 1, 2) = \frac{[1\ 1\ 1] \,.\, [2.58\ 3.22\ 2.56]^T}{3} = 2.79$$

4.6 Hybrid feature combination

In previous models, each of integrated data has specific properties which both are informative and reveal us different aspect of data set. Proposed hybrid model corresponds to combining both of the models explained above. In this case, we combine contextual information with the data extracted from rating values.

In this model, we insert location–activity (B), user–activity (D) and user–location (E) matrices into the output from previous step. As Fig. 12c illustrates, we still have three missing submatrices. One solution to complete the missing parts is to use transpose of location–activity (B^T), user–activity (D^T) and user–location (E^T) from model 1 in Fig. 12a. However, if we want to treat fairly, we should not utilize them twice even with different format. Hence, we insert zero into the missing parts of model which completes the matrix and makes it lower triangular.

We demonstrate the hybrid feature combination model with an example. The typical data set which is introduced in Fig. 6 is used in this example. It is a real subtensor of the original data, and the similarity matrices are also available as shown in Fig. 20.

Additionally, the model requires three more partial matrices to complete the final integrated matrix as displayed in Fig. 12c, which can be easily obtained from tensor T as explained in Sect. 2. Obtained matrices are shown in Fig. 21.

Once we get all partial matrices, we proceed with constructing the model according to Fig. 12c. As explained in model construction, similarity matrices are located in main diagonal

1	**0.9**	**0.5**	**0.7**	**0.55**
0.9	**1**	**0.7**	**0.4**	**0.3**
0.5	**0.7**	**1**	**0.3**	**0.1**
0.7	**0.4**	**0.3**	**1**	**0.8**
0.55	**0.3**	**0.1**	**0.8**	**1**

(a)

1	**0.6**	**0.5**	**0.1**	**0.3**
0.6	**1**	**0.7**	**0.75**	**0.65**
0.5	**0.7**	**1**	**0.3**	**0.1**
0.1	**0.75**	**0.3**	**1**	**0**
0.3	**0.65**	**0.1**	**0**	**1**

(b)

1	**0.6**	**0.9**	**0.7**
0.6	**1**	**0.2**	**0**
0.9	**0.2**	**1**	**0.1**
0.7	**0**	**0.1**	**1**

(c)

Fig. 20 Similarity matrices, **a** user–user similarity, **b** location–location similarity, **c** activity–activity similarity

3.4	3.6	4	1.75
2.4	2.4	1.5	1.5
2.67	2	2.6	2.67
1.5	0	0	3
2	1	3.5	2

(a)

2.5	2	3	2.5
2.75	2	3	2.67
3	1.67	2.33	2
2.5	3.5	3.337	1.5
1.75	3.33	3.67	2

(b)

2.75	2.5	1	2.5	3.33
3	2.67	1.5	3	2.67
3.25	1.25	2.25	0	3
4	2.5	2.75	1	1.75
3	1.67	3.5	0	1.67

(c)

Fig. 21 Additional extracted matrices, **a** location–activity, **b** user–activity, **c** user–location

and other matrices are inserted into the lower triangle of model. Applying SVD to the model and reducing its rank to 3 (to keep 50 % of original data) from one side and trimming it together with calculation of cosine similarity from other side yields all six matrices which are shown in second column of Table 2.

In order to make prediction for the value of $a_1 = (2, 1, 2)$, we freeze the index of $location = 1$ and $activity = 2$ and find similar users referring to Fig. 22. As shown in column U2 of Fig. 22a, the most similar user to user2 is user number 3. Additionally, using Fig. 22.d, the most similar user to user2 is also user1. Note that values of parameters m_1 to m_6 are set to 1. According to the U2 column of Fig. 23a, d, both of the similarity values are 0.99. Rating mean of users 2, 3, and 1 is 2.58, 2.31 and 2.53, respectively. Predictions from users are calculated as:

$$P^U_{User} = 2.58 + \frac{(3 - 2.31) * 0.99}{0.99} = 3.28$$

$$P^V_{User} = 2.58 + \frac{(3 - 2.53) * 0.99}{0.99} = 3.05$$

In the second step, we freeze the index of $user = 2$ and $activity = 2$ and find similar locations to $location = 1$. According to part b and e of Fig. 22, the most similar location to location1 is 5 in both parts. Since $T\ (2, 5, 2) = 0$, second similar locations are chosen which are locations 2 and 3, respectively. The calculations are given as follows:

$$P^U_{Location} = 3.22 + \frac{(2 - 2.06) * 0.84}{0.84} = 3.16$$

$$P^V_{Location} = 3.22 + \frac{(1 - 2.44) * 0.75}{0.75} = 1.79$$

Similarly, by freezing the index value of $user = 2$ and $location = 1$, we find the most similar activity to $activity = 2$ which in Fig. 22c is activity1 and in Fig. 22f is activity4. Since $T\ (2, 1, 4) = 0$, second similar activity in Fig. 22f is chosen which is activity1. Hence,

U1	U2	U3	U4	U5
2	3	2	5	4
3	5	5	3	3
5	4	4	2	2
4	1	1	1	1
1	2	3	4	5

(a)

L1	L2	L3	L4	L5
5	3	4	3	1
2	4	2	2	2
3	1	1	1	3
4	5	5	5	4
1	2	3	4	5

(b)

A1	A2	A3	A4
2	1	4	3
4	4	2	2
3	3	1	1
1	2	3	4

(c)

U1	U2	U3	U4	U5
3	1	1	5	4
2	3	2	3	3
4	4	4	1	1
5	5	5	2	2
1	2	3	4	5

(d)

L1	L2	L3	L4	L5
5	4	2	2	1
3	3	4	3	3
2	1	1	1	2
4	5	5	5	4
1	2	3	4	5

(e)

A1	A2	A3	A4
3	4	1	1
4	1	4	3
2	3	2	2
1	2	3	4

(f)

Fig. 22 Indices of similarities for the hybrid model, **a** sorted index of U_User_sim, **b** sorted index of $U\_Location\_sim$, **c** sorted index of $U\_Activity\_sim$, **d** sorted index of V_User_sim, **e** sorted index of $V\_Location\_sim$, **f** sorted index of $V\_Activity\_sim$

U1	U2	U3	U4	U5
0.99	0.99	0.99	0.99	0.99
0.98	0.99	0.99	0.99	0.99
0.97	0.99	0.99	0.99	0.99
0.97	0.99	0.98	0.97	0.97
0	0	0	0	0

(a)

L1	L2	L3	L4	L5
0.99	0.99	0.99	0.99	0.99
0.84	0.99	0.99	0.99	0.95
0.84	0.98	0.96	0.95	0.93
0.81	0.95	0.93	0.91	0.91
0	0	0	0	0

(b)

A1	A2	A3	A4
0.97	0.97	0.99	0.99
0.83	0.94	0.90	0.94
0.77	0.90	0.77	0.83
0	0	0	0

(c)

U1	U2	U3	U4	U5
1	0.99	1	0.99	0.99
1	0.99	0.99	0.99	0.99
0.99	0.99	0.99	0.99	0.99
0.99	0.99	0.99	0.99	0.99
0	0	0	0	0

(d)

L1	L2	L3	L4	L5
0.99	0.99	0.99	0.99	0.99
0.75	0.99	0.99	0.99	0.77
0.75	0.80	0.86	0.79	0.70
0.73	0.70	0.77	0.68	0.68
0	0	0	0	0

(e)

A1	A2	A3	A4
0.99	0.98	0.99	0.99
0.99	0.81	0.99	0.99
0.95	0.81	0.95	0.98
0	0	0	0

(f)

Fig. 23 Values of similarities for the hybrid model, **a** sorted value of U_User_sim, **b** sorted value of $U\_Location\_sim$, **c** sorted value of $U\_Activity\_sim$, **d** sorted value of V_User_sim, **e** sorted value of $V\_Location\_sim$, **f** sorted value of $V\_Activity\_sim$

predictions are computed as follows:

$$P^{U}_{Activiy} = 2.56 + \frac{(3 - 2.5) * 0.97}{0.97} = 3.06$$

$$P^{V}_{Activiy} = 2.56 + \frac{(3 - 2.5) * 0.81}{0.81} = 3.06$$

The final prediction for $a_1 = (2, 1, 2)$, as discussed before, is calculated from the mean of all predictions as follows:

$$\hat{T}\,(2, 1, 2) = \frac{[1\ 1\ 1\ 1\ 1\ 1]\,.\,[3.28\ 3.05\ 3.16\ 1.79\ 3.06\ 3.06]^T}{6} = 2.97$$

5 Experimental results

In this section, we evaluate the results and compare the accuracy of each method in the EFC model for the prediction of unknown rating values. Then, we consider the effect of several parameters on the final results. All the experiments are performed on a 2.53 GHz PC machine with 4 GB of main memory. Program codes are written in MATLAB R2009a [26], and the tensor dimensionality reduction part is implemented using Sandia tensor toolbox [4,19]. For the experimental evaluation of the proposed approach, we have utilized two data sets from location-based social tagging networks, which are briefly described as follows.

In order to evaluate the prediction accuracy of the proposed models, we employ 10-fold cross-validation on our data set [36]. Note that, in this method, each single member of data is set to be only one time in test set and nine times in training set, and at the end, final evaluation metric is the average accuracy over all folds.

5.1 Data sets

For our first experiment, we have used Geosocial2 data set which extracted from a location-based network and is available online.[1] Since the data set is introduced and utilized for the first time in the literature, we aim to give more details about it. The Geosocial2 data set is gathered from ratings that are given by 149 users to 112 different social activities. Ratings are submitted to system for performing the activities in 438 physical locations, which are located in some cities of Greece.

The original data set consists of *Activities*, *Places* (coordinates and name), *Users* (user profiles) and *Paths* (users' friendship network) in form of data matrices, as well as *Check-ins* table which shows the relation between the user, location, activity, rating and a time stamp which contains the time and date of performing an activity. We organize check-ins matrix in a 3-order tensor T, so that each entry $T(x, y, z)$ shows the rating that user x gives to activity z in location y. Since similarity matrices are not directly available, we have to extract them from data set.

In order to examine the functionality and accuracy of proposed approach and observe how each method predicts the values, we have selected another data set named Gowalla[2] [9] that was a location-based social network in which users were able to check in the network from various physical locations. Later, it was reportedly acquired by Facebook on 2011. The system stores user ID, date and time of check-ins, altitude and longitude of the point user checked in and a location ID which is assigned to that coordinates. In addition, the data set also contains an undirected graph of users which shows the friendship relation among the users. To properly utilize the data set, we have to modify it so that it conforms to our system's structure. Since it consists of around 196K users and 6400K check-ins, we may reduce its scale so that its size is similar to Geosocial2 data set. One idea is to filter the data in a way that it covers a small portion of physical location. For instance,

1 http://delab.csd.auth.gr/geosocial2/index2.html.

2 http://snap.stanford.edu/data/loc-gowalla.html.

Paris can be chosen as a target location. Hence, we diminish the scale by choosing the check-in points in which their coordination is within a triangle which represents the city of Paris. Additionally, users are selected so that at least checked in once in mentioned locations.

As we mentioned earlier in previous data set, each entry in user–location–activity table demonstrates a rating value (between 1 and 5) which a user gives for performing a specific activity in a determined location. However, check-ins data only show that a user either checked in a location or not. To construct a 3-D matrix similar to discussed one, we may think user ID and location ID as being our first and second dimension of the matrix. In practice, we do not have any information about performing an activity, and hence, the problem arises when we want to construct the third dimension of data which is activity in user–location–activity. To deal with this problem, one may think that each check-in of a user in a location is performed for doing an activity. Since we do not have any activity category or ID here, we may construct the third dimension to be time slice. That means, we may divide the time interval of 24 hours to slices of 1, 2, 3, … hours and assign each check-in to the slice that it has been occurred. With this definition, we will have 24, 12, 8, … time slices as the third dimension depending on the number of hours in each slice.

The other problem we encounter during diminishing the scale of data is the rating issue. We previously stated that users give their feedback as form of rating values to the system. However, when we are talking about check-ins, there is no feedback and rating from users. Instead, it is a binary state that shows whether a user has checked in a specific location or not. If we consider the frequency of check-ins in a location, human intuition tells us that the place or the activity which is done in that place may have an interest to the user. In other words, by counting the number of check-ins performed by a user in a specific location, we would be able to estimate the level of eagerness of the user. Notice that, the frequency of check-ins in some locations reaches more than 50 times. Hence, we have to define a function which maps the check-in frequency to the range of rating value which is between 1 and 5. Proposed mapping function for a given check-in frequency f_c is given by $CheckinMap(f_c)$, which is described in Eq. (15) as:

$$CheckinMap(f_c) = \begin{cases} 0 & \textit{if } f_c = 0 \\ \lfloor 1 + \ln f_c \rfloor & \textit{if } f_c > 0 \end{cases} \tag{15}$$

This function guaranties that return value of $CheckinMap(f_c)$ for all f_c is member of $\{0, 1, 2, 3, 4, 5\}$, since the maximum value of check-in frequency observed for reduced-scale data set is 76 check-ins. The rest of test scenario is similar to the first data set, and the evaluations are discussed in following subsections.

5.2 Prediction evaluation metrics

In our study, the accuracy of the proposed models is measured by mean absolute error (MAE) [13] and root mean square error (RMSE) [13], which are known statistical evaluation metrics and commonly are used in prediction systems to measure how close forecasts are to the original values [15]. Corresponding formulas are given in Eqs. (16) and (17) in which O and

$E$are observed values and estimated values and n is the cardinality of test set.

$$MAE\,(O,E)=\frac{1}{n}\sum_{i=1}^{n}|o_i-e_i| \tag{16}$$

$$RMSE\,(O,E)=\sqrt{\frac{\sum_{i=1}^{n}(o_i-e_i)^2}{n}} \tag{17}$$

5.3 Neighborhood selection

Selecting proper neighborhood set is a tricky task which potentially influences the accuracy of prediction. One solution is to choose *top-N* neighborhood in which neighbors with the highest similarity to active entry (user, location, or activity) is selected to predict the target rating [29]. However, determining the magnitude of N is crucial since if it is too low, it cannot be a precise prediction and if it is too high, the noise will affect the result negatively.

An alternative solution proposed in [8] chooses neighbors with regard to a threshold of similarity value. For instance, one strategy is to add such entries to neighborhood set that the value of similarity (correlation) is greater than 0.7 to active entry. Another simple method which is mostly utilized in small data sets is to set a threshold to the percent of similarities that are going to determine neighborhood. In this case, for example, we can seek top 10 % of sorted similarity matrix to construct the neighborhood set. When there is no neighbor for the defined boundaries, zero value is set as the prediction value.

In this test, for both data sets we reduce the rank so that 50 % of original data are maintained and Jaccard correlation is opted to calculate User–User similarity from friendship network as well. For the first data set, "*Wu And Palmer*" [39] is chosen as a metric to compute the similarity among activities. All these configurations are maintained same for EFC and HOSVD.

Figure 24 illustrates the influence of neighborhood size on MAE for all methods in Geosocial2 data set. In HOSVD, for a small size of neighborhood MAE is high but as the size is increased, error is decreased and reaches to a minimum value. Collaborative filtering shows an interesting result since it begins with high MAE for small size of neighborhood and soon after it reaches to 5, MAE shows a steady trend to the size of neighborhood.

In the content-based model, in which we use the similarity matrices to construct the model, MAE tends to be steady by changing the size of neighborhood. Finally, Hybrid model shows a slightly fluctuations by increasing the size of neighborhood, and it reaches to the lowest value of MAE in size = 10 and after that point it keeps an ascending move. It can be the influence of extra noise in calculation as the size of neighborhood is increased.

For the Gowalla data set as it shown in Fig. 25, HOSVD and content-based method seems to give the same results. However, collaborative method and hybrid method give better results in comparison to others. All methods have their best results for small size of neighborhood, and as the size is increased, the value of MAE also increases. We believe that sparseness of data set affects the similar neighbors, so that as we search for more similar neighbors, we get more entries with zero ratings which in turn mislead the predicted values.

5.3.1 Other parameters

The rank of SVD or HOSVD defines the amount of original data maintained, which has been explained in the Sect. 3.1. Hence, by changing the amount of the data, we control the rank of the matrices in each model. In our experiments, for Geosocial2 and Gowalla data sets, we set

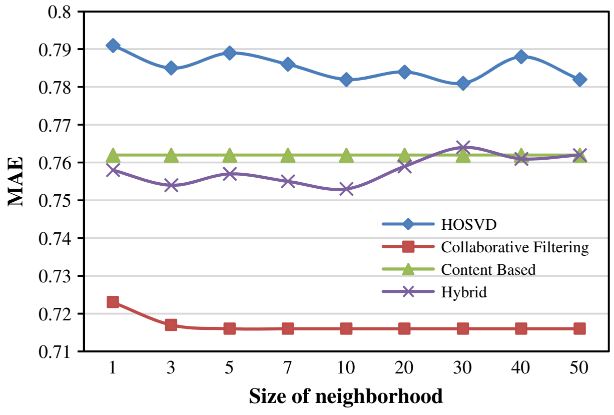


Fig. 24 Neighborhood size versus MAE for Geosocial 2 data set

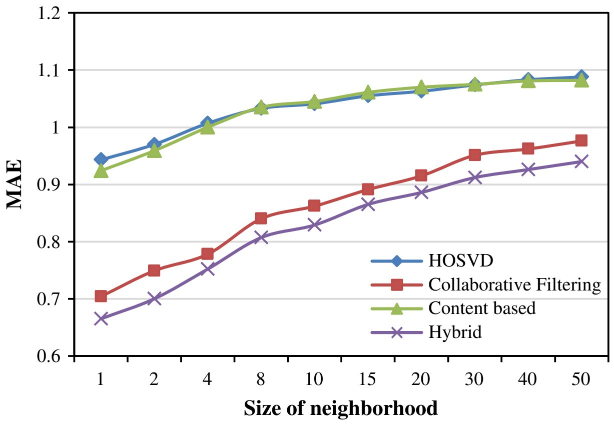


Fig. 25 Neighborhood size versus MAE for Gowalla data set

the neighborhood size to be 10 and 1 correspondingly. In the first data set, "*Wu And Palmer*" is chosen as a metric to compute the similarity among activities.

As illustrated in Figs. 26 and 27, we analyze the effect of the amount of the data maintained to MAE. HOSVD model shows the optimum result at *10 %* of original data and the performance drops as the percent is increased. Together with increasing the percent of original data to maintain, actually some noise is included which also increases MAE.

Collaborative filtering and hybrid models show slightly similar effect to the magnitude of rank. Both start from an initial MAE for lower ranks and after some fluctuation reach to a minimum value of MAE and take an increasing trend as the rank gets larger. Similar to the same trend in size of neighborhood given in Fig. 24, content-based model shows a steady trend and changing the rank of reduced dimension data has no effect on this model.

Earlier we mentioned in Sect. 4.6 that to construct EFC model, we utilize the Activity–Activity similarity matrix. Moreover, there are different metrics that are used to calculate similarity between words within WordNet [27]. Utilizing various metrics yields different results for similarity matrix which in turn influences the efficiency of final predicted results. Note that, this experiment is only applicable to the first data set, since we use time slice

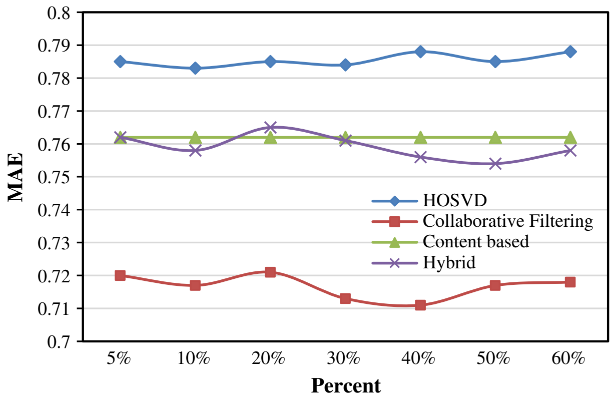


Fig. 26 Percent of the data maintained in dimension reduction versus MAE in Geosocial2 data set

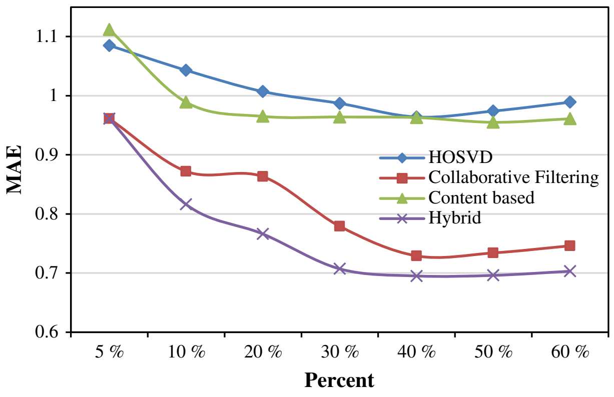


Fig. 27 Percent of the data maintained in dimension reduction versus MAE in Gowalla data set

instead of activity in Gowalla. The size of neighborhood and the percent of data to keep in this data set are 10 and 50 %.

As presented in Fig. 28 for the several distance metrics we experimented with to find similarity among activities, "*Wu And Palmer*" has better results in Hybrid model.

5.4 Recommendation evaluation metrics

In a typical recommendation system, items are recommended to users. Since in our model we have both activities and locations, our system can be used to recommend activities at certain locations, or locations for certain activities. We simply predict the rating value of user–location–activity triple, if it is unknown, and then generate either activity or location recommendation if that rating value is greater than some threshold.

To assess the quality of the recommendations generated by the methods introduced in this paper, we employed precision and recall which are standard and widely used metrics in information retrieval [28]. The way we measure them for each method is as follows:

– First, we randomly pick 10 % of the nonzero entries from rating tensor and set their values to be zero.

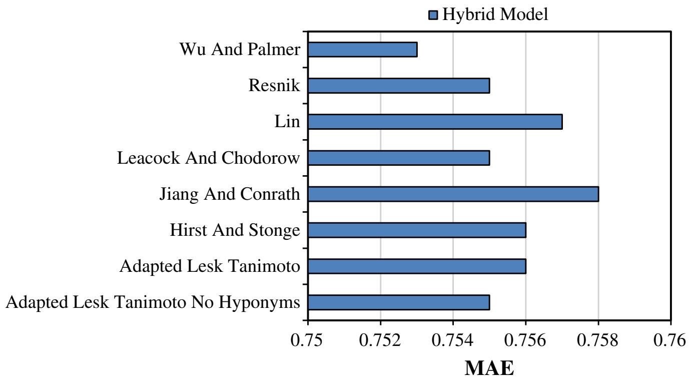


Fig. 28 Distance metrics of WordNet versus MAE in Hybrid model

- Then, we use one of our methods and predict their values.
- For each predicted value, we determine the followings:

I. True positives (TP): if both the original value and its predicted value are greater than the threshold, then that means we have made correct recommendation.
II. False positives (FP): if the original value is less than the threshold, but the predicted value is greater than it, then that means we have made an incorrect recommendation.
III. True negatives (TN): if both the original value and the predicated value are less than the threshold, then that means we did not make recommendation and it was correct.
IV. False negatives (FN): if the original value was greater than the threshold, but the predicted value was less than it, then we have missed the recommendation that we should have done.

Using these values, precision and recall are calculated as combination of TP, FP and FN as given in Eqs. (18) and (19).

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

Additionally, F-measure as a combination of precision and recall [31] can be computed as shown in Eq. (20).

$$F - measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{20}$$

As explained above, recommendation in top-N neighbor systems is determined using a threshold. In this experiment which is performed for Geosocial2 data set, we have examined the values of 2, 3 and 4 for the threshold. The size of neighborhood is chosen to be 10, and similar to other tests, we maintain 50 % of the data after reducing its rank. The impact of each value on precision, recall and F-measure is shown in Figs. 29, 30 and 31.

When the objective is to predict rating values less than or equal to 2, all methods give a reasonable results in terms of recall as seen in Fig. 29. However, to predict the rating values less than or equal to 3, all methods show a decreasing result for recall, but hybrid method outperforms others considering the recall and F-measure in Fig. 30. Once the objective is

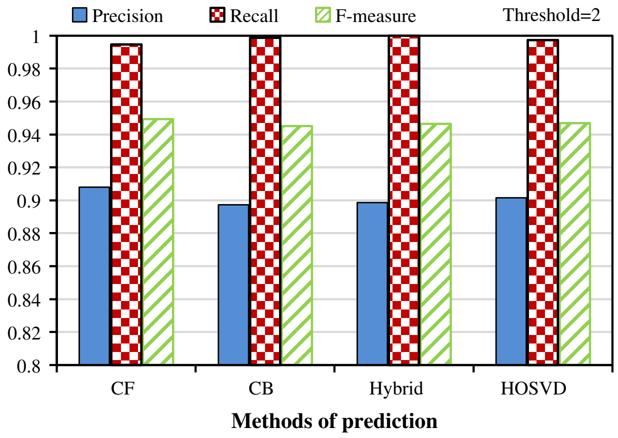


Fig. 29 Impact of threshold on precision and recall (threshold = 2)

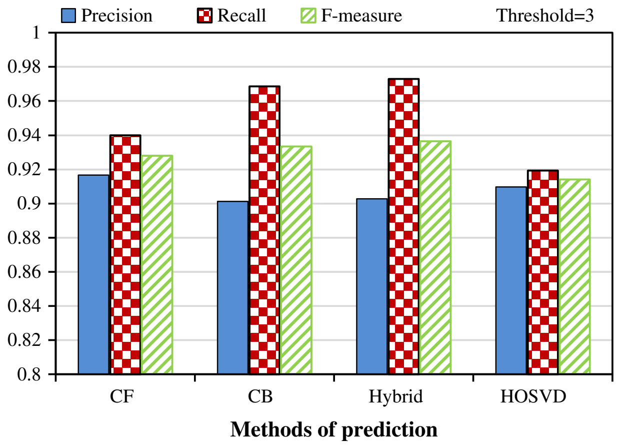


Fig. 30 Impact of threshold on precision and recall (threshold = 3)

to predict higher rating values, hybrid and CB methods give better results for precision, but both have a higher rate of FN as it is presented in Fig. 31.

5.4.1 Overall discussion

Table 4 presents the comparison of MAE, RMSE and execution time for different models we have discussed so far. Running time can be a quantitative factor to assess the response time of each technique. However, it is divided into two parts in which first one is the time of preparation for the task of prediction, and the second one is actual time of prediction. Note that, for each fold of execution in Geosocial2 data set, we randomly select 10 % of nonzero values of rating tensor (which in this work is 82) and replace them with zero. Consequently, it is the time required for predicting all those 82 entries selected randomly from data set. Similarly, in Gowalla data set 250, nonzero entries are selected to be predicted in given time.

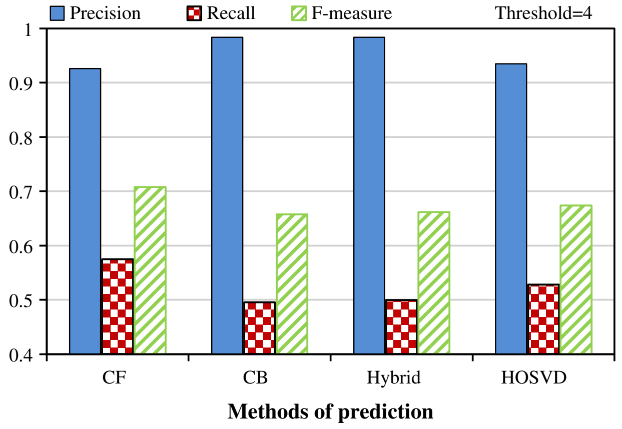


Fig. 31 Impact of threshold on precision and recall (threshold = 4)

Table 4 Comparison between models according to MAE, RMSE, and time of execution

	MAE	RMSE	Preparation time (s)	Prediction time (s)
Geosocial2				
HOSVD	0.767	0.958	0.83	1.12
Collaborative filtering	0.72	0.927	0.43	0.84
Content based	0.764	0.945	0.37	0.78
Hybrid	0.755	0.955	1.12	1.78
Gowalla				
HOSVD	0.941	1.056	0.08	0.17
Collaborative filtering	0.716	0.913	0.12	0.29
Content based	0.924	1.047	0.18	0.24
Hybrid	0.672	0.864	0.21	0.29

We applied paired *t* test [7] on 20 MAE experiment results of HOSVD and hybrid methods on both Geosocial2 and Gowalla data sets. For both of them, the two-tailed P value is less than 0.0001. By conventional criteria, this difference is considered to be statistically significant.

6 Conclusion

In this work, we have introduced a simple model that could be used for nonstandard recommendation systems with multidimensional rating data and other forms of information about the objects of its dimensions. It has been shown that with our technique, 3-D rating data can easily be reduced to 2-D matrix. The resulting 2-D matrix could be extended with additional feature matrices. We have applied this idea on geospatial data with user–activity–location dimensions. In addition to a very sparse 3-D tensor with these three dimensions, we also had three similarity matrices for the objects of each dimension.

Our main idea can be summarized as combining several matrices into single big matrix, applying SVD for dimensionality reduction and finally looking for similar entries for an entry whose rating value is being predicted. Since an entry has three dimensions in 3-D data structure, this similarity search operation has been performed on different domain corresponding to the different parts of the low-rank reduced matrices obtained after SVD, which is the main reason for constructing the results efficiently. Moreover, since the information loss is very little due to 3-D to 2-D reduction process, the accuracy of the prediction is also very high.

HOSVD-based approach has already been effectively used for 3-D rating data. However, our experiments show that pure collaborative filtering on the 2-D reduced model is even more effective and efficient than HOSVD-based solution. Moreover, the model is also suitable for content-based recommendation systems if only similarity matrices are utilized. Although it is not better than collaborative filtering method, the results for content-based method is surprisingly very good as well. Finally, our method allows combining 3-D data together with additional feature matrices very easily. Although we had anticipated even better results by combing these additional features, at least on our data set the accuracy values were obtained between pure collaborative filtering and the content-based recommendation models. As the future work, the performance of the method may be evaluated on several other data sets.

Acknowledgments This work has been partially funded by the Greek GSRT (project number 10TUR/4-3-3) and the Turkish TUBITAK (project number 109E282) national agencies as part of Greek-Turkey 2011-2012 bilateral scientific cooperation.

References

1. Adomavicius G, Sankaranarayanan R, Sen S, Tuzhilin A (2005) Incorporating contextual information in recommender systems using a multidimensional approach. ACM Trans Inf Syst 23(1):103–145
2. Anand SS, Mobasher B (2007) Contextual recommendation. In: Berendt B, Hotho A, Mladenic D, Semeraro G (eds) From web to social web: discovering and deploying user and content profiles. Springer, Berlin, pp 142–160
3. Anderson A, Huttenlocher D, Kleinberg J, Leskovec J (2012) Effects of user similarity in social media. In: Proceedings of the fifth ACM international conference on web search and data mining (WSDM '12). ACM, New York, USA, pp 703–712
4. Bader BW, Kolda TG (2012) MATLAB Tensor Toolbox Version 2.5. http://www.sandia.gov/tgkolda/TensorToolbox/. Accessed 1 May 2013
5. Bardsley E (2002) Lossy compression using SVD. http://edbardsley.org/classes/15-211/lab4/lossy.html. Accessed 1 May 2013
6. Bellogin A, de Vries AP (2013) Understanding similarity metrics in neighbour-based recommender systems. In: Proceedings of the 2013 conference on the theory of information retrieval (ICTIR '13). ACM, New York, USA, pp 48–55
7. Box JF (1987) Guinness, gosset, fisher, and small samples. Stat Sci 2(1):45–52
8. Breese JS, Heckerman D, Kadie C (1998) Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the fourteenth conference on uncertainty in artificial intelligence. Morgan Kaufmann, San Francisco, CA, USA, pp 43–52
9. Cho E, Myers SA, Leskovec J (2011) Friendship and mobility: user movement in location-based social networks. In: Proceedings of the 17th ACM SiGKDD international conference on knowledge discovery and data mining. ACM, New York, USA, pp 1082–1090
10. Fan J, Li R (2003) Local modeling: density estimation and nonparametric regression. In: Fang J, Lu Y (eds) Advanced medical statistics. World Scientific, New Jersey, pp 885–930
11. Foltz PW (1990) Using latent semantic indexing for information filtering. In: Proceedings of the ACM SIGOIS and IEEE CS TC-OA conference on office information systems. ACM, New York, USA, pp 40–47
12. Golub GH, Loan CFV (1996) Matrix computations. Johns Hopkins, Baltimore
13. Han J, Kamber M, Pei J (2011) Data mining: concepts and techniques. Morgan Kaufmann, San Francisco, CA, USA
14. Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '99). ACM, New York, USA, pp 230–237

15. Herlocker JL, Konstan JA, Terveen LG, Riedl JT (2004) Evaluating collaborative filtering recommender systems. ACM Trans Inf Syst 22(1):5–53
16. Hirst G, St-Onge D (1998) Lexical chains as representations of context for the detection and correction of malapropisms. In: Fellbaum C (eds) WordNet: An Electronic Lexical Database. MIT Press, Cambridge, pp 305–332
17. Jiang JJ, Conrath DW (1997) Semantic similarity based on corpus statistics and lexical taxonomy, In: Proceedings of 10th international conference on research in Computational Linguistics (ROCLING'97). Taiwan, pp 19–33
18. Kautz H, Selman B, Shah M (1997) Referral Web: combining social networks and collaborative filtering. Commun ACM 40(3):63–65
19. Kolda TG, Sun J (2008) Scalable tensor decompositions for multi-aspect data mining. In: Proceedings of the 8th IEEE international conference on data mining (ICDM 2008). IEEE Computer Society, Washington, DC, USA, pp 363–372
20. Lathauwer LD, Moor BD, Vandewalle J (2000) A multilinear singular value decomposition. SIAM J Matrix Anal Appl 21(4):1253–1278
21. Lax PD (2007) Linear algebra and its applications. Wiley, New York
22. Leacock C, Chodorow M (1998) Combining local context and WordNet similarity for word sense identification. In: Fellbaum C (eds) WordNet: An Electronic Lexical Database. MIT Press, Cambridge, pp 265–283
23. Lin D (1998) An information-theoretic definition of similarity. In: Proceedings of the 15th international conference on machine learning (ICML'98). Morgan Kaufmann, San Francisco, CA, USA, pp 296–304
24. Marinho LB, Nanopoulos A, Schmidt-Thieme L, Jaschke R, Hotho A, Stumme G, Symeonidis P (2011) Social tagging recommender systems. In: Ricci F, Rokach L, Shapira B, Kantor PB (eds) Recommender systems handbook. Springer US, New York, pp 615–644
25. Massachusetts Institute of Technology (2002) Singular value decomposition (SVD) tutorial. http://web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm. Accessed 1 May 2013
26. MATLAB Release 7.8.0 (R2009a) The MathWorks Inc, Natick, Massachusetts, United States
27. Miller GA (1995) WordNet: a lexical database for english. Commun ACM 38(11):39–41
28. Raghavan V, Bollmann P, Jung GS (1989) A critical investigation of recall and precision as measures of retrieval system performance. ACM Trans Inf Syst 7(3):205–229
29. Resnick P, Iacovou N, Suchak M, Bergstrom P, Riedl J (1994) GroupLens: an open architecture for collaborative filtering of netnews. In: Proceedings of the 1994 ACM conference on computer supported cooperative work (CSCW '94). ACM, New York, NY, USA, pp 175–186
30. Resnik P (1995) Using information content to evaluate semantic similarity in a taxonomy. In: Proceedings of the 14th international joint conference on artificial intelligence. Morgan Kaufmann, San Francisco, CA, USA, pp 448–453
31. Sanchez JL, Serradilla F, Martinez E, Bobadilla J (2008) Choice of metrics used in collaborative filtering and their impact on recommender systems. In: 2nd IEEE international conference on digital ecosystems and technologies (DEST 2008). IEEE, pp 432–436
32. Sarwar BM, Karypis G, Konstan JA, Riedl JT (2000) Application of dimensionality reduction in recommender system—a case study. In: ACM WebKDD workshop. ACM SIGKDD
33. Sattari M, Manguoglu M, Toroslu IH, Symeonidis P, Senkul P, Manolopoulos Y (2012) Geo-activity recommendations by using improved feature combination. In: Proceedings of the 2012 ACM conference on ubiquitous computing (UbiComp '12). ACM, New York, NY, USA, pp 996–1003
34. Shardanand U, Maes P (1995) Social information filtering: algorithms for automating "word of mouth". In: Proceedings of the SIGCHI conference on human factors in computing systems (CHI '95). ACM/Addison-Wesley, New York, NY, USA, pp 210–217
35. Spiegel S, Kunegis J, Li F (2009) Hydra: a hybrid recommender system [cross-linked rating and content information]. In: Proceedings of the 1st ACM international workshop on complex networks meet information; knowledge management (CNIKM '09). ACM, New York, NY, USA, pp 75–80
36. Stone M (1974) Cross-validatory choice and assessment of statistical predictions. J R Stat Soc Ser B (Methodol) 36(2):111–147
37. Symeonidis P, Nanopoulos A, Manolopoulos Y (2010) A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. IEEE Trans Knowl Data Eng 22(2):179–192
38. Woerndl W, Schueller C, Wojtech R (2007) A hybrid recommender system for context-aware recommendations of mobile applications. In: Proceedings of the 2007 IEEE 23rd international conference on data engineering workshop (ICDEW '07). IEEE Computer Society, Washington, DC, USA, pp 871–878

39. Wu Z, Palmer M (1994) Verbs semantics and lexical selection. In: Proceedings of the 32nd annual meeting on association for computational Linguistics (ACL '94). Association for Computational Linguistics, Stroudsburg, PA, USA, pp 133–138
40. Zheng VW, Cao B, Zheng Y, Xie X, Yang Q (2010) Collaborative filtering meets mobile recommendation: a user-centered approach, In: Proceedings of the twenty-fourth AAAI conference on artificial intelligence (AAAI 2010). AAAI Press, pp 236–241
41. Zheng VW, Zheng Y, Xie X, Yang Q (2010) Collaborative location and activity recommendations with GPS history data. In: Proceedings of the 19th international conference on world wide web (WWW '10). ACM, New York, NY, USA, pp 1029–1038

Masoud Sattari is currently a Ph.D. student at the Data Mining Lab of Middle East Technical University (METU), Turkey. He received his master from the same school under the supervisions of the Prof. Ismail H. Toroslu and Prof. Pinar Karagoz in 2013. His research interests include data mining, recommendation systems, and social network analysis.

Ismail Hakki Toroslu is with the Department of Computer Engineering, Middle East Technical University (METU) since 1993. He has received his B.S. and M.S. degrees in computer engineering from METU, Ankara in 1987 and Bilkent University, Ankara in 1989 respectively. Prof. Toroslu received his Ph.D. from the Department of Electrical Engineering and Computer Science at Northwestern University, IL, in 1993. He has also been a visiting professor in the Department of Computer Science at University of Central Florida between 2000 and 2002. His current research interests include data mining, bioinformatics and intelligent data analysis. Prof. Toroslu has published more than 60 technical papers in variety of areas of computer science. His recent publications are mostly on web mining, inductive logic programming, bioinformatics and recommendation systems. Prof. Toroslu has also received IBM Faculty Award in 2010.

Pinar Karagoz is currently Associate Professor in Computer Engineering Department of Middle East Technical University (METU). She received her Ph.D. from the same department in 2003. She worked as a visiting researcher in State University of New York (SUNY) at Stony Brook. Dr. Karagoz is the author of more than 80 publications in several conferences and journals including VLDB, CIKM, KAIS, Information Systems and SIGMOD Record. She has been serving as PC member or reviewer in conferences and journals in her research areas. Her research interests include data mining, web usage mining, semantic web services, web service discovery and composition, workflow modeling and analysis.

Panagiotis Symeonidis received his bachelor degree in applied informatics in 1996, and the MSc degree in information systems in 2004, from Macedonia University, Greece. He received the Ph.D. degree in web mining from Aristotle University of Thessaloniki, Greece, in 2008. Currently, he is working as a postdoctoral researcher at Aristotle University of Thessaloniki, Greece. He is the coauthor of 2 books and more than 35 articles in international journals and conference proceedings. His articles have received more than 850 citations from other scientific publications. He teaches courses on databases, data mining and data warehousing in a postgraduate program in Aristotle University of Thessaloniki. He is also the director of 1st SEK (School Laboratory Center) of Stavroupolis, Thessaloniki. His research interests include web mining (usage mining, content mining and graph mining), information retrieval and filtering, recommender systems, social media in Web 2.0 and time-evolving online social networks.

Yannis Manolopoulos is Professor with the Department of Informatics of Aristotle University of Thessaloniki. He has been with the University of Toronto, the University of Maryland at College Park and the University of Cyprus. He has also served as Rector of the University of Western Macedonia in Greece, Head of his own department, and Vice-Chair of the Greek Computer Society. He has co-authored 5 monographs published by Kluwer and Springer, 8 textbooks in Greek, as well as $\sim$ 300 journal and conference papers on Data Management. He received $>$8,000 citations from $>$1,200 distinct academic institutions and 2 best paper awards from ACM SIGMOD and ECML/PKDD conferences. He has also served as main co-organizer of several major fora, among others: ADBIS'2002, SSTD'2003, SSDBM'2004, ICEIS'2006, EANN'2007, ICANN'2010, AIAI'2012, WISE'2013, CAISE'2014, conferences. He has acted as evaluator for funding agencies in Austria, Canada, Cyprus, Czech Republic, Estonia, EU, Hong-Kong, Georgia, Greece, Israel, Italy and Russia. Currently, he serves in the Editorial Board of The VLDB Journal, The World Wide Web Journal, and The Computer Journal.