Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090558

A Novel Non-Negative Matrix Factorization Method for Recommender Systems

Mehdi Hosseinzadeh Aghdam, Morteza Analoui and Peyman Kabiri*

School of Computer Engineering, Iran University of Science and Technology, Tehran, 16846-13114, Iran.

Received: 10 Feb. 2015, Revised: 11 May 2015, Accepted: 13 May 2015
Published online: 1 Sep. 2015

Abstract: Recommender systems collect various kinds of data to create their recommendations. Collaborative filtering is a common technique in this area. This technique gathers and analyzes information on users preferences, and then estimates what users will like based on their similarity to other users. However, most of current collaborative filtering approaches have faced two problems: sparsity and scalability. This paper proposes a novel method by applying non-negative matrix factorization, which alleviates these problems via matrix factorization and similarity. Non-negative matrix factorization attempts to find two non-negative matrices whose product can well approximate the original matrix. It also imposes non-negative constraints on the latent factors. The proposed method presents novel update rules to learn the latent factors for predicting unknown rating. Unlike most of collaborative filtering methods, the proposed method can predict all the unknown ratings. It is easily implemented and its computational complexity is very low. Empirical studies on MovieLens and Book-Crossing datasets display that the proposed method is more tolerant against the problems of sparsity and scalability, and obtains good results.

Keywords: recommender system, collaborative filtering, non-negative matrix factorization, latent factors

1 Introduction

Collaborative Filtering (CF) is a natural choice for designing Recommender Systems (RSs) since it provides recommendations by centrally analyzing the user-item rating matrix alone [1]. CF can generate user specific recommendations based on historical user preferences. Inside a CF, the user interests on involved items (e.g., films, books, purchase records, etc.) are quantized into a user-item rating matrix, where high ratings denote strong preferences. So the problem of CF can be considered as the problem of missing data estimation, in which the main task is to predict the unknown user-item pairs based on known entries with minimum accumulative error [2]. CF supposes that users sharing the same ratings on past items likely to agree on new items. Research on CF can be grouped into two categories: memory-based and model-based [3, 4].

Memory-based methods compute similarities between users or between items and apply them to recognize the top most similar neighbors. Then the unknown rating is predicted by combining the known rating of the neighbors. However, there exist three essential challenges for memory-based approaches. The first one is sparsity. These methods rely on exact matches of two user-item vectors, which cause the methods to sacrifice RS coverage and accuracy. More specifically, since the correlation coefficient is only defined between users who have rated at least two items in common, or the items which have been unrated, then many pairs of users-items will have no correlation at all. As a consequence, memory-based RSs cannot precisely determine the neighborhood and recognize the items to recommend, which will surely lead to poor recommendations. The second one is scalability. In reality, both the number of users and items can be quite large. This may slow down the recommendation process significantly since memory-based methods will need too much computation in this case [5]. The third one is cold start. When the rating matrix is sparse, two users or items are unlikely to have common ratings, and consequently, memory-based methods will predict ratings using a very limited number of neighbors. Moreover, similarity weights may be computed using only a small number of ratings, resulting in biased recommendations. This is aggravated by the fact that users or items newly added to the RS may have no ratings at all, this problem termed as cold start. Cold start can be assumed as a sub problem of

* Corresponding author e-mail: analoui@iust.ac.ir

coverage because it evaluates the system coverage over a particular set of users and items [6].

Different from memory-based methods, model-based methods require establishing a model using training instances that can estimate the unknown ratings of a user. For example, decision tree, aspect models [7], clustering models [8], latent factor models [9], Bayesian network [10] and dimension reduction approaches [11] are model-based methods. However, creating a model and keeping it up to date are often time-consuming since there are usually many free parameters to tune [12].

The Matrix Factorization (MF) based methods have become popular for CF, due to the high accuracy and scalability. A low-rank MF method begins with the assumption that there exist a small number of latent factors (features) that can explain the rating behavior of users. Users and items are displayed as feature vectors in this latent space, where similarity between a user-item pair represents the tendency of the user to rate that item. While exact interpretability is never easy with latent feature models, when the items are movies, one can imagine that latent factors implicitly capture features such as personal information for users (e.g., age, gender and occupation) or information about the movies (e.g., genre, actors and directors) [13]. In a typical MF method to CF, a user and an item are displayed as unknown feature vectors whose dimensions are considered as latent features. These feature vectors are learnt so that inner products estimate the known ratings with respect to some cost function. Once the features are learnt, they prepare estimation for unknown ratings which may then be used for producing recommendations. Various methods differ in the approximation measures or the cost function they employ, and variants may be derived by adding different types of regularization to avoid overfitting [13].

Much of the prior work in these contexts has explored unconstrained SVD-like factorizations, while this paper focuses on the use of Non-negative Matrix Factorizations (NMF) [14]. NMF imposes non-negative constraints on the latent features. The non-negative constraints lead to a parts-based representation because they let only additive, not subtractive, combinations [15]. NMF with generalized KL-divergence cost function is equivalent to Probabilistic Latent Semantic Analysis which has previously been used for CF tasks [16,17]. This paper proposes a novel NMF method to learn the latent factors of users and items and estimate the unknown ratings using these latent features. Because this method is very easy to implement and use, we have found it very useful in RSs [18].

The rest of this paper is organized as follows. Section 2 explains RSs methods. The proposed NMF-based method is described in Section 3. Section 4 reports computational experiments. It also includes a brief discussion of the results obtained and finally we conclude the paper in the last section.

2 Background and related work

Recommender systems are software tools that collect various kinds of data in order to create their recommendations. Data is primarily about the items to propose and the users who will receive these recommendations. The recommendation problem truly arises as an independent field of research in the mid 1990s. It has deeper roots in several other areas like information retrieval and cognitive science. Methods for this problem are normally grouped in two categories: content-based and collaborative filtering methods [6].

The core of content-based (cognitive) methods is to recognize the common attributes of items that have received a favorable rating from a user, and recommend to user new items that share these attributes. In content-based RSs, rich information explaining the nature of each item is assumed to be achievable in the form of a feature vector. The user profiles can then be used to recommend new items to a user, by proposing the item whose feature vector is most similar to the profile vector of user, for example, using the cosine similarity or the minimum description length [19]. This method can also be used to estimate the rating of a user for a new item. Bayesian methods using content information have also been proposed to estimate ratings [20].

RSs based purely on content generally suffer from the problems of limited content analysis and over-specialization [6]. Limited content analysis emerges from the fact that the system may have only a limited amount of information on its users or the content of its items. There are many reasons for this lack of information. For example, because of privacy issues a user may not provide personal information, or the exact content of items may be difficult or costly to get for some kinds of items, such as music or images. Finally, the content of an item is often inadequate to determine its characteristic. Over-specialization is a result of the way in which content-based systems recommend new items, where the estimated rating of a user for an item is high if this item is similar to the ones liked by this user. Solutions suggested for this problem include adding some randomness or filtering out items that are too similar [21].

In contrast to content-based methods, collaborative filtering (social) methods depend on the ratings of a user as well as those of other users in the system [6]. CF methods overcome some of the limitations of content-based methods. For example, items for which the content is not available or difficult to get can still be recommended to users through the feedback of other users. In addition, CF recommendations are based on the quality of items as rated by neighbors, instead of depending on content. CF methods can recommend items with very different content, as long as other users have already shown preference for these different items. CF methods can be grouped in the two general categories of memory-based and model-based methods.

In memory-based (neighborhood-based or heuristic-based) [6], the ratings collected in the system are directly used to estimate ratings for new items. Memory-based methods can be implemented in two ways known as user-based or item-based recommendation. User-based methods, such as GroupLens, Bellcore video, and Ringo, evaluate the preference of a user for an item using the ratings for this item by other users that have similar rating patterns. The neighbors of a user are typically the users whose ratings on the items rated by both users are most correlated to those of user. Item-based methods estimate the rating of a user for an item based on the ratings of user for items similar to this item. In Item-based methods, two items are similar if several users of the system have rated these items in a similar way.

Unlike memory-based methods, which use the stored ratings directly in the estimation, model-based methods use these ratings to learn a predictive model. The main idea is to model the user-item interactions with factors displaying latent features of the users and items in the system. This model is then trained using the available information, and later used to estimate ratings of users for new items. Model-based methods for recommendation are numerous and include Latent Semantic Analysis [17], Latent Dirichlet Allocation [22], Maximum Entropy [23], Bayesian Clustering [10], Support Vector Machines and Singular Value Decomposition [6].

According to recent progress on RSs, one most successful type of method to CF is based on MF [24,25]. MF based recommenders work by converting both users and items into the same latent feature space, determining each entity with a feature vector inferred from the available ratings, and then generating predictions for unknown ratings using the inner products of the corresponding vector pairs. The earliest work of this type is reported by Sarwar et al. employing the Singular Value Decomposition (SVD) [26]. More recently, several MF methods have been successfully applied to implementing RSs, including the probabilistic latent semantic analysis [17], the Maximum Margin MF [27], and the Expectation Maximization for MF [28]. During the Netflix Prize, Brandyn Webb reported the Regularized Matrix Factorization (RMF), which is accurate, highly efficient and easy to implement. Inspired by RMF, many researchers have further investigated MF based methods. They have presented sophisticated MF based CF recommenders [6].

3 The proposed Non-negative matrix factorization-based method for RSs

Non-negative matrix factorization has been investigated by many researchers, but it has achieved popularity through the researches of Lee and Seung reported in Nature and NIPS [15,18]. In order to the argument that the non-negativity is critical in human perception they presented simple methods for finding non-negative representations of non-negative data. The basic NMF problem can be considered as follows: Given a non-negative matrix $R \in \Re_+^{n\times m}(R \geq 0)$ and a rank k, find two non-negative matrices $U \in \Re_+^{n\times k}$ and $I \in \Re_+^{m\times k}$ which factorize R as well as possible ($k \leq min(n,m)$), that is:

$$R \approx UI^T \tag{1}$$

It can be changed column by column as $r \approx Ui^T$ where r and i^T are the corresponding columns of R and I^T. In other words, each vector r is estimated by a linear combination of the columns of U, weighted by the factors of i^T. Therefore U can be considered as containing a basis that is optimized for the linear estimation of the data in R. Since relatively few basis vectors are used to display many vectors, acceptable estimation can only be obtained if the basis vectors find structure that is latent in the data.

The non-negative constraints on U and I only allow additive combinations among different bases. This is the main difference between NMF and the other MF methods, e.g., SVD. Unlike SVD, no subtractions can happen in NMF. As result, it is believed that NMF can learn a parts-based representation [18]. The benefits of this parts-based representation have been seen in real-world applications such as face analysis, gene expression analysis and document clustering [29].

3.1 Problem definition

There are a set of users $\{u_1,u_2,...,u_n\}$ and a set of items $\{i_1,i_2,...,i_m\}$ in RSs. The ratings given by users on items are given in a rating matrix $R_{n\times m}$. In this matrix, R_{ui} indicates the rating of user u on item i. R_{ui} be any real number, but often ratings are integers in the range [1..5]. The task of a RS is as follows: Given a user v and an item j for which R_{vj} is unknown, estimate the rating for v on item j using matrix R. This paper applies NMF to learn the latent features of users and items and estimate the unknown ratings using these latent features. Let $U_{n\times k}$ and $I_{m\times k}$ be latent user and item factor matrices, with row vectors U_u and I_i representing k-dimensional user-specific and item-specific latent feature vectors of user u and item i, respectively. The proposed method attempts to learn these latent features and exploit them for recommendation.

3.2 Initialization

The results and convergence supported by NMF methods usually highly depend on initialization. So, it is important to have efficient and consistent strategies for initializing matrices U and I. On the other hand, the efficiency of many NMF methods is affected by the selection of the starting matrices. Poor initializations often yield a slow convergence, and in certain instances may lead even to an

incorrect or irrelevant solution. The problem of initialization matrices becomes even more complicated for large NMF problems and when certain constraints are applied on the factored matrices involved [14].

The proposed method uses the similarity weights for initializing matrices U and I. The similarity weights between users and between items forms two matrices $SU_{n\times n}$ and $SI_{m\times m}$, respectively. SU_{uv} indicates the similarity weight between users u and v; also, SI_{ij} denotes the similarity weight between items i and j. One of the common similarity measures used in RSs is Pearsons Correlation Coefficient (PCC) in which the similarity weight of users u and v, given the rating matrix R, is computed as follow [19]:

$$sim(u,v)=\frac{\sum_{i\in P}(R_{ui}-\bar{R}_u)(R_{vi}-\bar{R}_v)}{\sqrt{\sum_{i\in P}(R_{ui}-\bar{R}_u)^2}\sqrt{\sum_{i\in P}(R_{vi}-\bar{R}_v)^2}} \quad (2)$$

where $\bar{R}_u$ denotes the average rating of user u and P indicates the set of items that are rated by both users u and v. The other common similarity metric is Adjusted Cosine. Considering H is the set of users that rated both items i and j, the Adjusted Cosine measure is then defined as follows:

$$sim(i,j)=\frac{\sum_{u\in H}(R_{ui}-\bar{R}_u)(R_{uj}-\bar{R}_u)}{\sqrt{\sum_{u\in H}(R_{ui}-\bar{R}_u)^2}\sqrt{\sum_{u\in H}(R_{uj}-\bar{R}_u)^2}}. \quad (3)$$

The equations (2) and (3) are used for formation matrixes SU and SI, respectively.

3.3 Cost function

In order to estimate factor matrices U and I in the NMF, we need to consider the cost function to quantify a difference between the matrix R and the approximate NMF model matrix $\hat{R}=UI^T$. The choice of the cost function mostly depends on the probability distribution of data. The simple way, use Frobenius-norm measure:

$$D_F(R\,\|\,UI^T)=\left\|R-UI^T\right\|_F^2=\sum_{u,i}(R_{ui}-\sum_{f=1}^{k}U_{uf}I_{fi}^T)^2 \quad (4)$$

This is lower bounded by zero, and clearly disappears if and only if $R=UI^T$. This is also termed as the squared Euclidean distance. Another useful measure is:

$$D_{KL}(R\,\|\,UI^T)=\sum_{u,i}(R_{ui}ln\frac{R_{ui}}{\left[UI^T\right]_{ui}}-R_{ui}+\left[UI^T\right]_{ui}) \quad (5)$$

Like the Frobenius-norm this is also lower bounded by zero, and disappears if and only if $R=UI^T$. But it cannot be termed a distance, because it is not symmetric in R and UI^T, so we will refer to it as the divergence of R from UI^T. The objective of the cost functions in equations (4) and (5) is the minimization of the difference between matrix R and matrix UI^T with respect to U and I, subject to the constrains $U,I\geq 0$. Therefore, we can use either Euclidean distance or divergence. In this paper, the divergence cost function is used for the proposed method.

The above cost functions are convex with respect to either the entries of the matrix U or the matrix I, but not both. Therefore, it is impossible to solve this problem in the sense of finding global minimum. However, there are many methods from numerical optimization that can be applied to discover local minimum. Gradient descent is the simplest method to implement, but convergence can be slow. Other techniques like conjugate gradient have faster convergence, at least in the neighborhood of local minimum, but are more complex to implement than gradient descent.

3.4 The proposed algorithm

In this paper, the update rules are derived from equation (5) by using gradient descent. Gradient descent is based on the observation that if the multivariable function $F(x)$ is defined and differentiable in a neighborhood of a point A, then $F(x)$ decreases fastest if one goes from A in the direction of the negative gradient of F at A. It follows that, if $B\leftarrow A-\eta\nabla F(A)$ for η small enough, then $F(A)\geq F(B)$. The value of the learning rate η can be changed at every iteration [30]. The simple update rules for U and I that reduce the divergence can be written as:

$$U_{uf}\leftarrow U_{uf}+\eta_{uf}[\sum_{\mu=1}^{m}I_{f\mu}^T\frac{R_{u\mu}}{(UI^T)_{u\mu}}-\sum_{\mu=1}^{m}I_{f\mu}^T] \quad (6)$$

$$I_{fi}^T\leftarrow I_{fi}^T+\eta_{fi}[\sum_{\mu=1}^{n}U_{\mu f}\frac{R_{\mu i}}{(UI^T)_{\mu i}}-\sum_{\mu=1}^{n}U_{\mu f}]. \quad (7)$$

If η are all set equal to some small positive number, this is equal to conventional gradient descent. As long as the learning rate is small enough, these update rules should reduce $D_{KL}(R\,\|\,UI^T)$. Now if we set:

$$\eta_{uf}=\frac{U_{uf}}{\sum_{\mu=1}^{m}I_{f\mu}^T} \quad (8)$$

$$\eta_{fi}=\frac{I_{fi}^T}{\sum_{\mu=1}^{n}U_{\mu f}} \quad (9)$$

Then we obtain the update rule for U and I as:

$$U_{uf}\leftarrow U_{uf}\times\frac{\sum_{\mu=1}^{m}I_{f\mu}^TR_{u\mu}/(UI^T)_{u\mu}}{\sum_{\mu=1}^{m}I_{f\mu}^T} \quad (10)$$

$$I_{fi}^{T} \leftarrow I_{fi}^{T} \times \frac{\sum_{\mu=1}^{n} U_{\mu f} R_{\mu i} / (UI^{T})_{\mu i}}{\sum_{\mu=1}^{n} U_{\mu f}}. \qquad (11)$$

These update rules are a good compromise between speed and ease of implementation to predict the unknown ratings. This paper applies these update rules to estimate factor matrices U and I. The divergence D_{KL} is non-increasing under these update rules and this divergence is invariant under these update rules if and only if U and I are at a stationary point of the divergence. The proof of this theorem is reported in Lee and Seungs publication [18].

The Proposed NMF-based Method for RS

input: sparse rating matrix $R_{n\times m}$
α, β
max-iter
output: full rating matrix $\hat{R}_{n\times m}$

begin
Initializing:
generate two non-negative random matrixes $U_{n\times k}$ and $I_{m\times k}$
compute similarity matrix users $SU_{n\times n}$ by using equation (2)
compute similarity matrix items $SI_{m\times m}$ by using equation (3)

for each user u=1,...,n **do**
  for each user v=1,...,n **do**
    if similarity $(u, v) > \alpha$ **do**
      $U_u = U_v$
    end
  end
end

for each item i=1,...,m **do**
  for each item j=1,...,m **do**
    if similarity $(i, j) > \beta$ **do**
      $I_i = I_j$
    end
  end
end

Training:
for $iter$=1,...,$max\text{-}iter$ **do**
  for each user u=1,...,n **do**
    for each feature f=1,...,k **do**
      update U_{uf} using equation (10)
    end
  end
  for each item i=1,...,m **do**
    for each feature f=1,...,k **do**
      update I_{fi}^{T} using equation (11)
    end
  end
  $\hat{R} = UI^{T}$
  if $\hat{R} = R$ **then**
    break and **return** $\hat{R}$
  end
end
return $\hat{R}$
end

At each iteration of the proposed method, the new values of U or I are found by multiplying the current values by some features that depend on the quality of the predictions in equation (1). The quality of the prediction gets better monotonically with the application of these multiplicative update rules. In other words, the repetition of the update rules is guaranteed to converge to a locally optimal matrix factorization. Fig. (1) illustrates this process by providing simple example.

The rating matrix that RSs operate on must thus be retrained and kept up-to-date, in order to maintain high

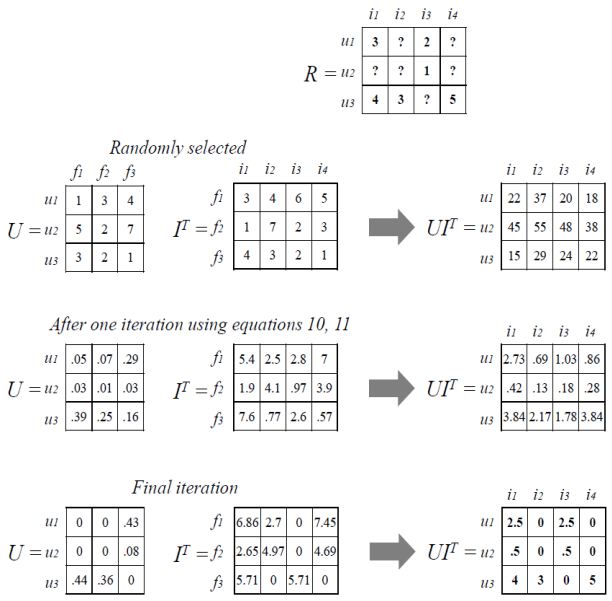

Fig. 1: A simple example for the proposed algorithm.

prediction accuracy. The drawback of MF methods is that once the matrices are calculated, the model is static. For real world applications, updating a model is crucial. Particularly, when ratings on new users or new items come in, updating the feature vectors is important. When a new rate comes in, the proposed method updates only the feature vectors related to the new rate. This operation has a very low complexity $O(kn + km)$. On the other hand, both empirical and theoretical results for runtime complexity of the proposed method, makes it feasible for huge datasets.

4 Experiments

This paper conducted a series of experiments to examine the effectiveness of the proposed method for CF in terms of scalability and recommendation quality. This paper has used two popular datasets to evaluate the performance of the proposed method. The following sections describe the datasets and implementation results.

4.1 Datasets

There are several types of datasets for RS. This paper conducted a set of experiments with real usage data on the MovieLens datasets [31] and Book-Crossing dataset [32]. GroupLens Research has gathered and made available rating datasets from the MovieLens website. This dataset consists of 100,000 ratings on an integer scale from 1 to 5 given to 1642 movies by 943 users, where each user has rated at least 20 movies.

The Book-Crossing dataset was collected by Cai-Nicolas Ziegler in a four weeks crawl from the Book-Crossing community. The Book-Crossing dataset is extremely sparse. We edit the dataset in order to achieve more meaningful results from CF methods when estimating recommendations. Hence, the book titles with fewer than 20 ratings are deleted, along with all ratings referring to them. Only users with at least 20 ratings each were kept. The resulting datasets dimensions were considerably more moderate, 3156 users and 7485 books. The edited dataset consists of 253902 book ratings on an integer scale from 0 to 10.

4.2 Evaluation metrics

There are several kinds of measures for evaluating the performance of CF approaches [5,33]. This paper uses two popular metrics, the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE), to measure the estimation quality. The metrics MAE is defined as [6]:

$$MAE = \frac{1}{R_{test}} \sum_{u,i} \left| R_{ui} - \hat{R}_{ui} \right| \tag{12}$$

Where $\hat{R}_{ui}$ denotes the rating user u gave to item i as predicted by a method, and R_{test} denotes the number of tested ratings. The metrics RMSE is defined as [6]:

$$RMSE = \sqrt{\frac{1}{R_{test}} \sum_{u,i} \left(R_{ui} - \hat{R}_{ui} \right)^2} \tag{13}$$

From the definitions, a smaller MAE or RMSE value means a better performance. The MAE and RMSE do not fully express the usefulness of the RS. We also need to consider the effectiveness of the RS by computing the total coverage of the system. Coverage is the measure of the percentage of items for which a RS can provide predictions [34]. A RS may not be able to create predictions on every item. For example, if a user has rated very few items, or if an item has been rated by very few users. A RS which has high prediction accuracy, but only on a small set of items, would not be very useful. Computing coverage will give further insight into the effectiveness of the RS. There are several ways to compute coverage [34], this paper calculates coverage as number of items for which the RS can create estimates, over the total number of item predictions that are requested.

$$coverage = \frac{|R_i | R_i \in S|}{S} \tag{14}$$

Where R_i denotes the estimate that the RS created on item i, and S denotes the set of items for which the RS is creating an estimation. Accuracy and coverage are two metrics that must be considered together; a RS can only be useful if both accuracy and coverage are high.

4.3 Experimental results

This section focuses on comparing the performance of the proposed method against other approaches. The tested models include the proposed NMF-based method and a number of other approaches. All tested models were implemented on MATLAB R2011b. All experiments were conducted on a Microsoft Windows with two 2.67 GHz dual-core CPUs and 4 GB RAM. Various values were tested for the parameters of the proposed method. The results display that the highest performance is obtained by setting the parameters to values as follow: $\alpha = \beta = 0.2$ the rank of matrix R is 500 ($k = 500$) and the maximum number of iterations is 20 (*max-iter*=20). These values were empirically determined in our preliminary experiments; but we make no claim that these are optimal values.

4.3.1 Performance comparison with other methods

In order to show the effectiveness of the proposed recommendation method, this paper compares the reported results against the recommendation results of the following methods:

User Mean: This method uses the mean value of every user to predict the missing values.

Item Mean: This method utilizes the mean value of every item to predict the missing values.

User-based Collaborative Filtering, PCC: This method predicts the rating of a user for a new item using the ratings given to item by users most similar to the user.

Item-based Collaborative Filtering: This method computes the similarity between two items by comparing ratings made by the same user on these items.

In this work, we applied the 5-fold cross validation in our experiments. Each fold contains 80% of data as the training set and the remaining 20% as the test data. Analyzing the MAE and RMSE shown in tables (1) and (3), we see that on average, the proposed method obtained a higher accuracy value than the other methods.

As mentioned earlier, in the MovieLens and Book-Crossing datasets, each user has rated at least 20 items. As a result, the User Mean method has 100% coverage but it is not true in all datasets with cold start problem. The proposed method has 100% coverage in all datasets.

If the standard deviation is high, the User Mean and Item Mean methods have poor performance. The standard deviation of rating for each user and each item is higher than 1 on the MovieLens dataset. Similarly, the standard deviation of rating for each user and each item is higher than 3 on the Book-Crossing dataset.

To graphically illustrate the progress of the proposed method as it searches for solutions, we take number of iteration as the horizontal coordinate and the MAE/RMSE measure as the vertical coordinate. This should illustrate the process of improvement of the proposed method as

Table 1: The performance of the proposed method on the MovieLens dataset.

	User Mean		**Item Mean**		**User-based CF**		**Item-based CF**		**Proposed Method**	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Fold-1	0.8502	1.0630	0.8285	1.0361	0.7129	0.9986	0.7074	0.9922	0.7090	0.9942
Fold-2	0.8383	1.0467	0.8210	1.0314	0.7016	0.9908	0.6993	0.9849	0.6933	0.9794
Fold-3	0.8265	1.0329	0.8135	1.0242	0.7035	0.9897	0.6967	0.9814	0.6964	0.9816
Fold-4	0.8308	1.0367	0.8124	1.0194	0.7017	0.9863	0.6990	0.9831	0.6898	0.9765
Fold-5	0.8350	1.0393	0.8187	1.0281	0.7039	0.9824	0.7026	0.9817	0.6978	0.9794
AVG	0.8362	1.0437	0.8188	1.0278	0.7047	0.9896	0.7010	0.9847	0.6973	0.9822

Table 2: The coverage of the proposed method on the MovieLens dataset.

	User Mean	**Item Mean**	**User-based CF**	**Item-based CF**	**Proposed Method**
Fold-1	100	96.79	96.43	97.80	100
Fold-2	100	96.91	96.73	97.86	100
Fold-3	100	95.90	96.37	97.80	100
Fold-4	100	96.91	97.32	98.27	100
Fold-5	100	96.73	96.73	97.74	100
AVG	100	96.65	96.72	97.89	100

Table 3: The performance of the proposed method on the Book-Crossing dataset.

	User Mean		**Item Mean**		**User-based CF**		**Item-based CF**		**Proposed Method**	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Fold-1	5.7119	6.6013	5.7657	6.5367	5.6537	6.5600	5.7109	6.5627	2.3530	4.3422
Fold-2	5.7108	6.5987	5.7785	6.5493	5.6563	6.5601	5.7236	6.5745	2.3599	4.3500
Fold-3	5.7262	6.6096	5.7844	6.5525	5.6730	6.5745	5.7334	6.5803	2.3383	4.3213
Fold-4	5.7318	6.6152	5.7944	6.5552	5.6642	6.5614	5.7405	6.5833	2.3304	4.3192
Fold-5	5.7329	6.6190	5.7712	6.5381	5.6798	6.5834	5.7217	6.5691	2.3545	4.3514
AVG	5.7227	6.6088	5.77888	6.5464	5.6654	6.5679	5.7260	6.5740	2.3472	4.3368

Table 4: The coverage of the proposed method on the Book-Crossing dataset.

	User Mean	**Item Mean**	**User-based CF**	**Item-based CF**	**Proposed Method**
Fold-1	100	99.06	72.89	75.06	100
Fold-2	100	99.14	72.94	75.23	100
Fold-3	100	99.07	72.93	75.05	100
Fold-4	100	99.19	73.10	75.45	100
Fold-5	100	99.07	72.77	75.30	100
AVG	100	99.11	72.93	75.22	100

Table 6: Comparing results from the proposed method against the results listed in [3, 12].

	ML_100			**ML_200**			**ML_300**		
	Given5	Given10	Given20	Given5	Given10	Given20	Given5	Given10	Given20
PCC	0.874	0.836	0.818	0.859	0.829	0.813	0.849	0.841	0.820
PD	0.849	0.817	0.808	0.836	0.815	0.792	0.827	0.815	0.789
AM	0.963	0.922	0.887	0.849	0.837	0.815	0.820	0.822	0.796
MMMF	0.945	0.861	0.803	0.930	0.849	0.786	0.929	0.843	0.773
CBCF	0.924	0.896	0.890	0.908	0.879	0.852	0.847	0.846	0.821
SCBPCC	0.848	0.819	0.789	0.831	0.813	0.784	0.822	0.810	0.778
SF2	0.847	0.774	0.791	0.827	0.773	0.783	0.804	0.761	0.769
CFONMTF	0.838	0.801	0.804	0.827	0.791	0.787	0.801	0.780	0.782
Proposed Method	0.803	0.789	0.767	0.777	0.757	0.736	0.764	0.747	0.728

Table 5: The standard deviation of MovieLens and Book-Crossing datasets.

	MovieLens Dataset	Book-Crossing Dataset
Fold-1	1.1186	3.7290
Fold-2	1.1244	3.7288
Fold-3	1.1292	3.7303
Fold-4	1.1287	3.7328
Fold-5	1.1274	3.7282
AVG	1.1257	3.7298

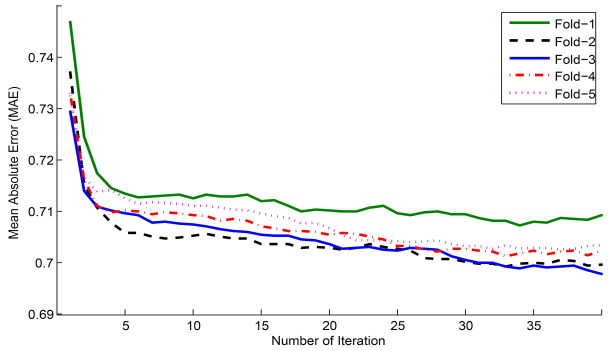

Fig. 2: Mean absolute error vs. the number of iterations.

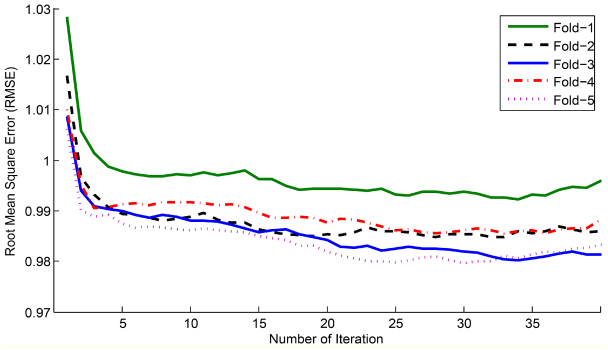

Fig. 3: Root mean square error vs. the number of iterations.

the number of iteration increase. Figs. (2) and (3) show the MAE and RMSE measures on MovieLens dataset as we change the number of iteration. For conveniently comparing with other CF methods reported in [3,12], for MovieLens dataset, we also extracted a subset of 500 users with more than 40 ratings. The first 100, 200 and 300 users in the MovieLens dataset are selected into three different training user sets, which are indicated as ML_100, ML_200 and ML_300 respectively. But for different training sizes, the test user set is fixed, i.e. the last 200 users. In our experiments, the available ratings of each test user are equally split into an observed set and a held out set. The observed ratings are used to estimate the held out ratings. Furthermore, we randomly chose 5, 10 and 20 items rated by test users in the observed set, which were termed Given5, Given10, and Given20 respectively.

As mentioned above, the proposed method could alleviate two fundamental problems: data sparsity and scalability. In order to show the performance of the proposed method to CF, we compare the proposed method versus the state-of-art methods: Pearson Correlation Coefficient (PCC) [10], Personality Diagnosis (PD) [35], Aspect Model (AM) [7], Cluster based collaborative filtering (CBCF) [8], Scalable Cluster-Based Pearson Correlation Coefficient (SCBPCC) [12], similarity fusion (SF2) [36] and Collaborative Filtering using Orthogonal Nonnegative Matrix Tri-Factorization (CFONMTF) [3].

A comparison between the test results for the proposed method versus other methods tested on the MovieLens dataset are presented in Table (6). It can be stated that all the methods tested on this dataset offered an acceptable level of performance but the proposed method outperforms all the other methods, and is just a little worse than SF2 in Given10. SF2 suffers from the scalability problem while the proposed method successfully resolves scalability problem. Hereby, it can be said that the overall performance of the proposed method is the best, considering the balance between computation efficiency and prediction accuracy.

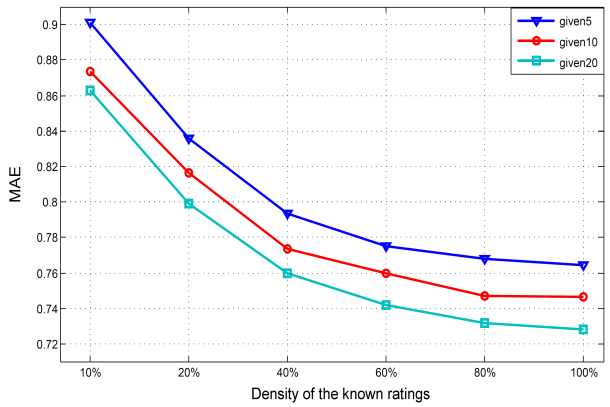

Fig. 4: The performance of the proposed method on different density of ML_300 dataset.

4.3.2 Sparsity

The sparsity of a rating matrix can have an important impact on the performance of CF. To evaluate the performance of the proposed method, this paper conducted an experiment to simulate the phenomenon of the sparseness of rating matrix and compare the performance about two methods: SF2 and CFONMTF. This paper empirically analyzes how MAE evolves with the density of rating matrix. Fig. (4) shows the

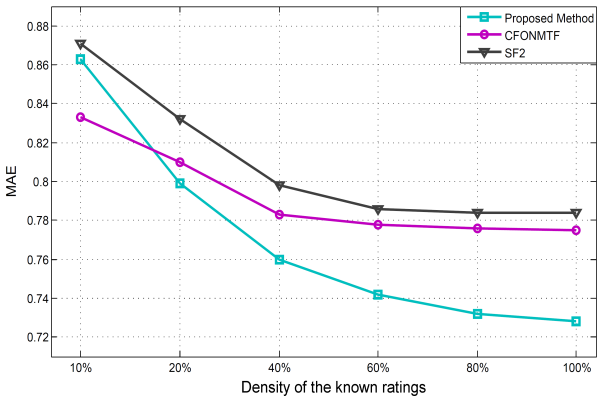

Fig. 5: Comparison of the proposed method with the other methods.

performances of the proposed method when the sparsity of the dataset ML_300 varies. We randomly selected the 10%, 20%, 40%, 60%, 80%, and 100% of the known ratings from the whole dataset to represent different degrees of sparsity of the rating matrix.

The results display that indeed the sparsity has a great effect on the performance of different methods. When the rating matrix becomes dense, all methods tend to achieve higher performance. As seen from Fig. (5), the MAE curve of the proposed method is below that of the other methods, which means that the sparseness has the least impact on the proposed method.

4.3.3 Scalability

Collaborative filtering methods are usually designed to apply on very large datasets. For that reason, the scalability of the approaches is crucial. Assuming the average number of items that are rated by two specific users is $\bar{p}(\bar{p} << m)$, and the average number of users that rated two specific items is $h(h << n)$. The time complexity of the proposed method consists of two parts: initialization and training time complexity. In the initialization step, the complexity of computing the similarity matrixes is $O(n^2\bar{p} + m^2\bar{h})$, and the complexity of equalization of the feature vectors is $O(n^2 + m^2)$. In training step, the most significant complexity is in updating the feature vectors. The complexity of learning each user feature vector and each item feature vector are $O(km)$ and $O(kn)$, respectively. Therefore, the complexity of the training step is $O(max - iter \times (nkm + mkn))$. We can rewrite this as $O(max - iter \times nkm)$, which denotes that the computational complexity of the proposed method is linear with respect to either the number of users or items. This complexity analysis shows that the proposed method is very efficient and can scale to very large datasets.

The experiments performed on the MovieLens and Book-Crossing datasets showed that the proposed methods tolerance to sparsity. When the rating matrix is quite sparse, we cannot get sufficient similar ratings by similar users or similar items. This yields in the poor recommendations of CF methods based on memory-based alone. The proposed method supports the complementary information and improves the prediction accuracy when only sparse data is available. Also, we can see that, compared with other methods, the proposed method is quicker in prediction of unknown ratings. In general, it can predict all of the unknown ratings within tens of iterations while most other CF methods have failed to predict the all unknown ratings. The proposed method comprises a very simple concept, and the ideas can be implemented in a few lines of computer code. It needs only primitive mathematical operators, and has low requirements in term of memory. Also, the proposed method using factorizing the rating matrix overcomes the scalability problem. The experiments prove that the proposed method is scalable to large dataset.

4.3.4 Stability

This section concentrates on stability of recommendation methods, which measures the consistency of RS predictions. Stability has a positive effect on the users' view to accept recommendations. In this paper, stability is measured using following steps that already reported by Adomavicius and Zhang [37].

1) Train the RS based on the known ratings and predict all the unknown ratings.

2) Select and add a subset of the predicted ratings as the new incoming ratings to the original dataset.

3) Re-train the RS based on the new data, and make new predictions for unknown ratings.

4) Compare predictions from Steps 1 and 3 and compute stability by using Mean Absolute Shift (MAS) or Root Mean Squared Shift (RMSS).

$$MAS = \frac{1}{R_{test2}} \sum_{(u,i)\in R_{test2}} |R_{ui1} - R_{ui2}| \qquad (15)$$

Where R_{ui1} and R_{ui2} denote the ratings user u gave to item i as predicted by a method in steps 1 and 3, respectively. R_{test2} denotes the number of tested ratings in step 3. RMSS is defined as:

$$RMSS = \sqrt{\frac{1}{R_{test2}} \sum_{(u,i)\in R_{test2}} (R_{ui1} - R_{ui2})^2} \qquad (16)$$

In order to compare the proposed method with other methods, this paper has considered four methods that are explained earlier on the MovieLens dataset. The total number of ratings is 100,000 in this dataset. For initial stability computations, we used 80000 ratings as the input

Table 7: Stability of the proposed method on the MovieLens dataset.

	MAS	RMSS
User Mean	0.0405	0.0615
Item Mean	0.0211	0.0426
User-based CF	0.2025	0.6110
Item-based CF	0.0789	0.2818
Proposed Method	0.0248	0.2120

to the RSs in order to predict the remaining 20000 unknown ratings. From these predicted ratings, we drew a random sample of 5000 ratings and treated them as new incoming ratings, and the remaining 15000 ratings were used to compute the prediction shift, as described above. In order to obtain robust empirical results, we ran the experiments three times for each method and reported the average stability of the three runs in Table (7).

Both User Mean and Item Mean had lower prediction shift rather than other methods. In other words, adding new ratings does not significantly change the user/item average. Furthermore, while the proposed method, User-based CF and Item-based CF were comparable in term of predictive accuracy, the proposed method which is based on the global optimization approach, represented higher stability than the memory-based neighborhood methods (User-based CF and Item-based CF) that are based on the local, nearest neighbor heuristics.

5 Conclusions and future work

Recommender Systems (RSs) intend to estimate what the most suitable items are, based on the users preferences and limitations. In order to complete such a computational task, RSs gather user preferences, which are either explicitly expressed as ratings for items, or are concluded by interpreting user behaviors. This paper focuses on the non-negative matrix factorization in RSs. The basic idea is to assume that there exist a latent low dimensional representation of users and items where users and items can be modeled accurately. For instance, the rating that a user gives to a movie might be imagined to be dependent on few latent features such as the users taste across different movie genres. MF methods are a class of latent feature models that try to find weighted low rank approximations to the rating matrix, where weights are used to hold out missing ratings.

The accuracy of memory-based approaches suffers from the lack of available ratings. Sparsity is a problem common to most RSs due to the fact that users usually rate only a small number of the available items [6]. Data sparsity has an important effect on the performance of CF methods. As RSs are designed to help users navigate in huge datasets, one of the goals of the designers of such RSs is to scale up to real datasets. With the growth of the dataset, many methods are either slowed down or require additional resources such as computation power or memory. Therefore, dimensionality reduction comes in naturally. The proposed method greatly mitigates two essential challenges: sparsity and scalability by applying the NMF. Empirical studies verified that the proposed method effectively improves the prediction accuracy of CF and resolves the sparsity and scalability challenges. From the experiment we have concluded that application of the update rules in equations (10) and (11) are guaranteed to find at least a locally optimal solution. The update rules themselves are extremely easy to implement computationally, and will hopefully be utilized by others for a wide range of applications.

As for the future work, intention is to combine the proposed method with clustering to improve the performance of RS. Clustering similar users is recently attracting a lot of attention and many researches in this area have been reported in literature. As mentioned already, the scalability of the RSs is vital. Our intension is to cluster similar users based upon self-similarity to increase the scalability of the proposed method and then apply NMF in each cluster instead of all users.

Acknowledgement

This research was supported by the Iran Telecommunication Research Center (ITRC).

References

[1] G. Adomavicius and A. Tuzhilin, Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering, **6**, 734749 (2005).
[2] X. Luo, H. Liu, G. Gou, Y. Xia, and Q. Zhu, A parallel matrix factorization based recommender by alternating stochastic gradient decent, Engineering Applications of Artificial Intelligence, **7**, 1403-1412 (2012).
[3] G. Chen, F. Wang, and C. Zhang, Collaborative Filtering Using Orthogonal Nonnegative Matrix Tri-factorization, Information Processing and Management: an International Journal, **3**, 368-379 (2009).
[4] K. Christidis and G. Mentzas, A topic-based recommender system for electronic marketplace platforms, Expert Systems with Applications, **11**, 4370-4379 (2013).
[5] B. Sarwar, G. Karypis, j. Konstan, and J. Riedl, Item-based collaborative filtering recommendation algorithms, Proceedings of the 10th international conference on World Wide Web, New York, NY, USA, 285-295 (2001).
[6] F. Ricci, L. Rokach, and B. Shapira, Recommender Systems Handbook, Springer, (2011).
[7] T. Hofmann and J. Puzicha, Latent class models for collaborative filtering, Proceedings of the 16th international joint conference on Artificial intelligence, San Francisco, CA, USA, 688-693 (1999).
[8] L.H. Ungar and D.P. Foster, Clustering methods for collaborative filtering, Proceedings of the Workshop on Recommendation Systems, Menlo Park, CA, 1-16 (1998).

[9] J. Canny, Collaborative filtering with privacy via factor analysis, Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, 238-245 (2002).
[10] J.S. Breese, D. Heckerman, and C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence, San Francisco, CA, USA, 43-52 (1998).
[11] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, Eigentaste: a constant time collaborative filtering algorithm, Information Retrieval, **2**, 133-151 (2001).
[12] G.R. Xue et al., Scalable collaborative filtering using cluster-based smoothing, Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, 114-121 (2005).
[13] V. Sindhwani, S.S. Bucak, J. Hu, and A. Mojsilovic, A Family of Non-negative Matrix Factorizations for One-Class Collaborative Filtering Problems, RecSys, New York, USA, 2009.
[14] A. Cichocki, R. Zdunek, A. Phan, and S. Amari, Nonnegative Matrix and Tensor Factorizations: Applications to Exploratory Multi-way Data Analysis and Blind Source Separation, John Wiley & Sons, 2009.
[15] D.D. Lee and H.S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature, **401**, 788-791 (1999).
[16] C. Ding, T. Li, and W. Peng, On the equivalence between non-negative matrix factorization and probabilistic latent semantic indexing, Computational Statistics and Data Analysis, **8**, 39133927 (2008).
[17] T. Hofmann, Latent semantic analysis for collaborative filtering, ACM Transactions on Information Systems, **1**, 89 - 115 (2004).
[18] D.D. Lee and H.S. Seung, Algorithms for Non-negative Matrix Factorization, NIPS, 2001.
[19] D. Jannach, M. Zanker, A. Felfernig, and G. Friedrich, Recommender Systems An Introduction. New York, USA: Cambridge University Press, 2011.
[20] K. Lang, News Weeder: Learning to filter netnews, 12th International Conference on Machine Learning, San Mateo, USA, 331339 (1995).
[21] D. Billsus and M. J. Pazzani, User modeling for adaptive news access, User Modeling and User-Adapted Interaction, **2-3**, 147180 (2000).
[22] D. M. Blei, A.Y. Ng, and M.I. Jordan, Latent dirichlet allocation, Journal of Machine Learning Research, **3**, 9931022 (2003).
[23] C.L. Zitnick and T. Kanade, Maximum entropy for collaborative filtering, Proceedings of the 20th conference on Uncertainty in artificial intelligence, Arlington, Virginia, USA, 636643 (2004).
[24] Y. Koren, R. Bell, and C. Volinsky, Matrix factorization techniques for recommender systems, IEEE Computer, **8**, 30-37 (2009).
[25] M.H. Aghdam, M. Analoui, and P. Kabiri, Application of Nonnegative Matrix Factorization in Recommender Systems, 6'th International Symposium on Telecommunications, Tehran, 873 - 876 (2012).
[26] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, Application of dimensionality reduction in recommender systems - a case study, in ACM Web KDD, 2000.
[27] N. Srebro, J.D. M. Rennie, and T.S. Jaakola, Maximum-margin matrix factorization, Advances in Neural Information Processing Systems, 2005.
[28] M. Kurucz, A. Benczur, and K. Csalogany, Methods for large scale svd with missing values, Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California , USA, 31-38 (2007).
[29] D. Cai, X. He, J. Han, and T.S. Huang, Graph Regularized Non-negative Matrix Factorization for Data Representation, IEEE Transactions on Pattern Analysis and Machine Intelligence, **8**, 1548-1560 (2011).
[30] J. Kivinen and M. Warmuth, Additive versus exponentiated gradient updates for linear prediction, Journal of Information and Computation, **1**, 164 (1997).
[31] MovieLens datasets available at: http://www.grouplens.org/node/12, as visited on July 2012.
[32] Book-Crossing dataset available at: http://www.informatik.uni-freiburg.de/~cziegler/BX/, as visited on April 2013.
[33] F. Hernndez del Olmo and E. Gaudioso, Evaluation of recommender systems: A new approach, Expert Systems with Applications, **3**, 790-804 (2008).
[34] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl, Evaluating collaborative filtering recommender systems, ACM Transactions on Information Systems, **1**, 5-53 (2004).
[35] D.M. Pennock, E. Horvitz, S. Lawrence, and C.L. Giles, Collaborative Filtering by Personality Diagnosis: A Hybrid Memory and Model-Based Approach, Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, San Francisco, CA, USA, 473-480 (2000).
[36] J. Wang, A.P. de Vries, and M.J.T. Reinders, Unifying user-based and item-based collaborative filtering approaches by similarity fusion, Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, New York, NY, USA, 501-508 (2006).
[37] G. Adomavicius and J. Zhang, On the Stability of Recommendation Algorithms, Proceedings of the fourth ACM conference on Recommender systems, Barcelona, Spain, 47-54 (2010).

Mehdi Hosseinzadeh Aghdam is a Ph.D. candidate in the computer engineering department at the Iran University of Science and Technology. He graduated with a master's in computer engineering Artificial Intelligence from University of Isfahan (UI) in 2008. At UI, he worked on swarm intelligence-based method for feature selection. His main research interests are: Data Mining (Information Retrieval, Recommendation Systems), Computational Intelligence, Pattern Recognition and Social Network Analysis.

Morteza AnaLoui is an associate university professor in the school of computer engineering and the director of Computer Network Research Lab in Iran University of Science and Technology (IUST). He received a B.A. degree in electrical engineering from IUST in 1983 and a Ph.D. degree in electrical engineering from Okayama University, Japan, in 1989. His research interests include Networking (Virtual Social, Virtual Economic and Delivery), Artificial Intelligence, and Persian Computing. Dr. AnaLoui has authored over 150 publications, including journal articles, book chapters, and conference papers.

Peyman Kabiri is assistant professor of computer engineering at the Iran University of Science and Technology (IUST). He received a B.Eng. degree in computer hardware engineering from IUST in 1992 and the M.Sc. and Ph.D. degrees in computer science at the Nottingham Trent University, UK, in 1996 and 2000 respectively. He is the director of the Intelligent Automation Laboratory at the Department of Computer Engineering-IUST. Dr. Kabiri has authored a number of publications including journal articles, book chapters, and conference papers. He is editor of a book in intrusion detection work area.