Computing and Informatics, Vol. 37, 2018, 1209–1230, doi: 10.4149/cai_2018_5_1209

EXTRA: EXPERTISE-BOOSTED MODEL FOR TRUST-BASED RECOMMENDATION SYSTEM BASED ON SUPERVISED RANDOM WALK

Farshad Bakhshandegan Moghaddam

AIFB, Karlsruhe Institute of Technology
KIT, Karlsruhe, Germany
e-mail: farshad.bakhshandegan-moghaddam@kit.edu

Bahram Sadeghi Bigham*

Department of Computer Science and Information Technology
Institute for Advanced Studies in Basic Sciences, Zanjan, Iran
e-mail: b_sadeghi_b@iasbs.ac.ir

Abstract. The quality of recommendations based on any class of recommender systems may become poor if no or low quality data has been provided by users. This is a situation known as *Cold Start* problem, which typically happens when a new user registers to the system and no preference data is available for that user. Trust-Aware Recommendation Systems can be considered as a solution for the cold start problem. In these systems, the trust between users plays an import role for making recommendations. However, most of the Trust-Aware RSs consider trust as a context independent phenomenon which means if user a trusts user b to the degree k then user a trusts user b to the degree k in all the concepts. However, in reality, trust is context dependent and user a can trust user b in context X but not in Y. Moreover, most of the trust-aware RSs do not consider an expertise concept for users and all the users are considered as same in the recommendation process. In this paper we proposed a novel approach for detecting expert users just based on their ratings (unlike previous systems which consider the separate profile and extra information for each user to find an expert). In this model a supervised random walk is exploited to search the trust network for finding experts. Empirical experiments on the Epinions dataset shows that EXTRA can outperform previous models in terms of accuracy and coverage.

Keywords: Recommendation systems, trust, supervised random walk, expertise

* Corresponding author

1 INTRODUCTION

With the rapidly growing amount of information available on the Internet and availability of large amounts of data, finding the information that exactly matches the user's requirements is very difficult. This problem which is known as "Information Overloading" has led to a system which can automatically select the most relevant information and suggest them to users. These systems which are known as recommender systems, find their way to E-Commerce; e.g. there are popular recommender systems for movies[1], books[2], and so on.

The general task of a *Recommender System (RS)* is to find a list of items which a user may like to find them in the future. Generally, there are two entities which play the main roles in recommender systems, i.e. users and items. Each user can rate a list of items. So, the input of a RS is a User-Item $n \times m$ matrix (n users and m items). This matrix usually is very sparse and the main task of the recommender system is to predict the rating for user u on a non-rated item i or to generally recommend some items for the given user u based on the ratings that already exist. In RS literature, the user u is called "active user" and the item i is called "target item".

One of the most popular and yet effective techniques which are used in recommender systems is the Collaborative Filtering (CF) method [1]. In this method, the system tries to find the same users with the same preferences and aggregate their opinion for predicting the rating for the user u on an item i. The intuition behind the CF is: people often get the best recommendation from someone with a similar taste. Computing the similarity between users is the main bottleneck of this approach. To calculate the similarity of user i and j, most of the similarity functions needs common items which are rated by both user i and j. However, since most users cannot rate (and even are not interested in rating) many items, finding two users with commonly rated items is not easy. Moreover, it is also very challenging to calculate the similarity for users who are new to the system (i.e., cold start users), as they have not rated a significant number of items and hence cannot be properly linked with other similar users.

With the advent of social networks, the new techniques have emerged in recommender systems which are called "trust-aware recommendation systems". In these systems, the inputs is a trust network (a social network which expresses how much the members of the community trust each other) as well as the User-Item matrix. The system uses information about users' profiles and relationships between them to predict the rating of the active user for the target item. In these systems, the trust value between users is exploited instead of the similarity value. However, the trust network is also very sparse and we need a mechanism to calculate the trust value between two unconnected users.

[1] http://www.netflix.com

[2] http://www.amazon.com

Many models have emerged so far for calculating the trust value between two unconnected users and incorporating this value in the recommendation phase [2, 3, 4, 5]. The experimental results show that using trust-aware recommendation systems does not significantly alter the accuracy of the system but improves its coverage [2, 3, 6]. However, due to the complex nature of the trust, it is very difficult to calculate the exact trust value between two arbitrary users. Moreover, all the proposed methods for calculating the trust consider the trust as a context-independent phenomenon. It means that if user a trusts user b to the degree k then user a trusts user b to the degree k in all the concepts. However, in the real-world scenarios trust is context dependent and user a can trust user b in concept X but not in Y.

In this paper, we present a model for mining the trust network based on the supervised random walk which, firstly, rectifies the need of calculating the exact trust value between two users; secondly, considers the trust as a context-dependent phenomenon; and thirdly, exploits the expertise of the users to calculate the recommendation. In most of the previous methods, which have been introduced for finding experts in social networks [7, 8, 9], each user has his/her own profile and system; by comparing the profiles (usually by using text-mining techniques) he/she is able to find experts in the context X. In this paper, we propose a new method for finding experts without the need of any metadata for users, and just based on users ratings.

The rest of the paper is organized as follows. We begin our discussion by reviewing the related work in Section 2. Section 3 describes the problem statements. We discuss the details of our proposed model in Section 4. In Section 5 we introduce some properties of our model. The results of an empirical analysis are presented in Section 6 and in Section 7 we conclude the paper and introduce some directions for future research.

2 RELATED WORK

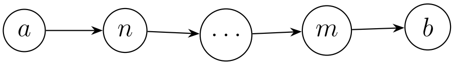
In this section, several major approaches for recommender systems, especially for collaborative filtering and trust-aware recommendation systems, will be reviewed. Two types of collaborative filtering approaches are widely studied: model-based [10, 11] and memory-based [2, 3, 5, 12]. In the model-based approaches, the system tries to learn a model which captures the rating behavior of users. Examples of model-based approaches include the clustering model [13], aspect models [14, 15, 16] and the latent factor model [17]. In the model-based approaches, training datasets are used to train a model. Although the training phase in this approach is very time consuming, it can be performed as an offline preprocess. In these systems, the prediction phase is very fast.

Unlike model-based systems, no learning phase exists in the memory-based systems. The most analyzed examples of memory-based collaborative filtering include user-based approaches [18, 19, 20, 13] and item-based approaches [21, 22, 12]. User-based approaches predict the ratings of active users based on the ratings of similar users, and item-based approaches predict the ratings of active users based on the computed information of items similar to those chosen by the active user. In the memory-based approach, the system stores the entire User-Item matrix in the memory and tries to use it for prediction phase. In these systems, there is no learning phase but the prediction phase is very time-consuming.

Trust metrics could primarily be divided into ones with a global and ones with a local scope. Global trust metrics can take into account all users and trust relationships between them when computing a trust estimation. Examples can be found in [23, 24, 25] which borrow their ideas from the renowned PageRank algorithm [26]. However, the local trust metric only relies on a part of the trust network, hence taking into account the personal bias. The intuition behind the local trust metrics is that user a may trusts user b but user c may not trust user b [27, 28].

[3] deduced some form of trust graph properties from real networks and proposed an algorithm that has been called TidalTrust [3]. TidalTrust is a modified breadth-first search and looks at the trust values along paths connecting two users. Basically, it tries to find all the raters which are in the shortest path from the source user. Finally returns the aggregated rating of users weighted by the trust value. For calculating the trust value, the source user a asks from its neighbors about the trust value of the user b and finally aggregates responses, weighted by trust values of itself and its direct neighbors.

Massa in [2] introduced MoleTrust. The idea used in the MoleTrust algorithm is similar to the TidalTrust algorithm. MoleTrust also uses breadth-first search. However, it considers maximum-depth for the algorithm and just considers all raters up to a given threshold. The trust value from the source user a to the sink user b is the aggregation of trust values between a and users directly connected to b weighted by the trust values [6]. Figure 1 explains the differences between these two methods.



TidalTrust: Trust(a,b) = Trust(a,n)× Trust(n,b)

MoleTrust: Trust(a,b) = Trust(a,m)× Trust(m,b)

Figure 1. A trust network between user a and b and difference between Tidal Trust and Mole Trust

TrustWalker model has been introduced in [6] as a random walk method to combine the trust-aware approach and the item-based CF approach for predicting the rating of a single item. TrustWalker considers not only the ratings of the target item but also similar items. Research by Jamli et al. [6] proved that TrustWalker outperforms both TidalTrust and MoleTrust models which were introduced before.

More information about the trust-aware recommender systems can be found in [32, 33, 34].

3 PROBLEM STATEMENT

In this section we will formalize the recommendation problem. Typically in a recommendation system there is a set of users $U = \{u_1, \dots, u_N\}$ and a set of items $I = \{I_1, \dots, I_M\}$. Each user can rate a subset of I. This subset is denoted by $RI_u = \{r_{u,i_1}, \dots, r_{u,i_k}\}$. RI can be null (something that occurs in many datasets). The rating of the user u on the item i is denoted by $r_{u,i}$. The $r_{u,i}$ can be classified as unary (like), binary (thumbs up/down) or often a scalar range of [m,M]. Usually m represents that the user is completely unsatisfied with the item, and M represents that the user is completely satisfied with the item. Also, in a trust-aware system, there is a trust network among users. We show this network by TN. $t_{a,b}$ denotes the trust value between users a and b. In binary trust networks (such as Amazon, Epinion and eBay), $t_{a,b}$ can be $[0, 1]$. Zero means no trust and one means full trust. We define $N_a = \{b \in U | t_{a,b} = 1\}$ where N_a denotes the set of users directly trusted by a and being neighbors of a. The trust network will currently be outlined as a graph $G = \langle V, E \rangle$ where V represents the users and $E = \{(a, b) | a \in V \& b \in N_a\}$ represents the edges. Each node corresponds to a user and each edge corresponds to a trust statement.

The recommender system is responsible for predicting the rating of an active user $u \in U$ on the target item $i \in I$. The predicted rating is denoted by $P_{u,i}$. Traditional collaborative filtering systems use similar users ratings for obtaining $P_{u,i}$. These systems use a similarity function such as Pearson Correlation, Cosine or Adjusted Cosine for finding similar users. However, as mentioned in Section 1, if two users do not have any rated items in common, the system will fail to calculate the similarity value.

In some of the trust-aware recommender systems techniques, there is no need for calculating the similarity between two users (as the trust value is used instead of the similarity value). Equations (1) and (2) show the difference between these two approaches:

$$P_{u,i} = \frac{\sum_{v \in U, i \in RI_v}^{K} sim(u, v) \times r_{v,i}}{\sum_{v \in U, i \in RI_v}^{K} |sim(u, v)|}, \tag{1}$$

$$P_{u,i} = \frac{\sum_{v \in U, i \in RI_v}^{K} t_{u,v} \times r_{v,i}}{\sum_{v \in U, i \in RI_v}^{K} |t_{u,v}|}. \tag{2}$$

Here, $sim(u, v)$ is the degree of similarity between user u and v and $t_{u,v}$ is the degree of trust between two users. Trust-aware recommender systems have adequate performance due to two important principles of sociology. These principles are *Selection* and *Social Influence* effect. The selection means that people prefer to relate to people with similar preferences, and the social influence means that related people in a social network influence each other to become more similar [29, 6].

Various experiments show that using the trust network for recommendation does not improve the accuracy but it can dramatically increase the coverage. Coverage is the percentage of $\langle u; i\rangle$ pairs which system can compute $P_{u,i}$ for that. Moreover, using the trust network protects the system against attacks like fake profiles due to faked profiles are not being trusted.

4 EXTRA MODEL

The main issues with the current trust-aware systems are:

- Need of having an accurate model to calculate the trust value between two arbitrary users u and v ($t_{u,v}$).
- Unable to capture the real world scenarios due to considering the trust as a context-independent phenomenon.

Moreover, another challenge is to decide how far to go in exploring the trust network. The further we go, the more likely to find raters, but the less trustworthy their ratings become. The random walk model which has been introduced in the [6] can prepare a good trade-off between the precision and the coverage. Although the random walk which we use in EXTRA is identical to the random walk which has been introduced in [6], however, our random walk model differs from that in two ways. First, our random walk is supervised while [6] uses an ordinary one. Second, the stopping criteria of our random walk is different from [6]. The main motivation for proposing this model is that all the trusted users cannot necessarily produce a dependable opinion about the target item i, and the reliability of an opinion must be evaluated based on how much a user u is expert in the field of item i. Based on this fact, the expert user's opinion must be weighted much more than an ordinary user. In this paper, we proposed a new method for finding how much a user u is expert in the item i, solely based on the user u ratings.

To recommend a rating for an active user u_0 on the target item i, we perform random walks on the trust network and select one of the direct neighbors of $u_0(v)$. It must be noted that the probability of selecting each neighbor is different from another and this probability depends on the level of expertise of each neighbor in the item i. The details of how much a user is expert will be discussed later in this section. Each random walk returns a value. We perform several random walks, and the aggregation of all values, returned by different random walks, are considered as the predicted rating $P_{u,i}$.

4.1 Who Is Expert?

Most of the models which exploit expertise concept, consider a profile for their users [7, 8, 9]. This profile encompasses the abilities and interests of each user. When the system is faced with a question (query), it tries to find a user with the highest level of expertise in the question by using text mining techniques (most of the profiles are in the text format) [9]. Also, in [7] authors considered some specific fields and users must promulgate their level of expertise with respect to any of the fields to the system. Then the entered question is converted to a field vector by the system and comparison of the question vector and the user vector will reveal who is an expert about the question.

To the best of our knowledge there is no system which uses just ratings for defining expertise and finding experts. The authors in [30] showed that there is a relation between the rating similarity and the content similarity. They used MovieLens[3] (1 M) dataset and exploited Wikipedia website for extracting the content of each movie in this dataset. Experimental results showed that the correlation between the content of items and item ratings is 14 %. This statement can be interpreted in the following way:

> If two products have the same rating pattern then their content will be the same with probability of 0.14.

We use this fact to define expertise and present the following definitions:

1. A user is expert in an item i, if he/she rates this item or rates large number of similar products to the item i.
2. A user is expert in an item i, if he/she rates this item or rates a very similar item to the item i.

We consider a user as a fully expert user on an item i if he/she rates item i. The reason is in trust-aware recommender systems, for predicting $P_{u,i}$ the system looks for users who rate the item i in the trust network. For modeling these definitions we need a function to calculate the similarity of 2 items. In this paper, the Pearson Correlation function is used for calculating the similarity between two items:

$$corr(i,j) = \frac{\sum_{u \in CU_{i,j}} (r_{u,i} - \mu_u)(r_{u,j} - \mu_u)}{\sqrt{\sum_{u \in CU_{i,j}} (r_{u,i} - \mu_u)^2} \sqrt{\sum_{u \in CU_{i,j}} (r_{u,j} - \mu_u)^2}}. \tag{3}$$

$CU_{i,j}$ is the set of common users who have rated both items i and j, and μ_u denotes the average of ratings expressed by u. $corr(i,j)$ denotes the correlation of items i, j. Values of this function are in the range $[-1,1]$. Based on the results of [30], if two items have a Pearson correlation to the degree p, then with the probability $0.14 \times p$ their contents will be the same. Obviously this value is very small, however, our experiments show that exploiting this small value increases the accuracy and coverage of recommender systems. We should note that, if p is a positive number, then $0.14 \times p$ will present how similar are two items and, on the contrary, if p is a negative number, then the $0.14 \times |p|$ will present how dissimilar are two items. Now we can formally define the expertise concept.

Definition 1. Considering the number of items which have been rated by user u and have positive correlation with the target item i as expertise of user u in the item i.

[3] `http://grouplens.org/datasets/movielens/`

$$\psi(u,i) = \begin{cases} 1, & \text{if } r_{u,i} \neq \emptyset, \\ \frac{|\{j \in RI_u | Corr(i,j) > 0\}|}{\sum_{\forall v \in N_u} |\{j \in RI_v | Corr(i,j) > 0\}|}, & O.W. \end{cases} \tag{4}$$

In this equation $\psi(u,i)$ denotes the expertise of user u in the item i and RI_u denotes all the items which have been rated by u.

Definition 2. Considering the maximum similarity of items which have been rated by user u and the target item i as the expertise of user u in the item i.

$$\psi(u,i) = \begin{cases} 1, & \text{if } r_{u,i} \neq \emptyset, \\ \max Corr(i,j) : \forall j \in RI_u, & O.W. \end{cases} \tag{5}$$

The values of Definition 2 are in the range $[0,1]$, but the values of Definition 1 must be normalized to be usable as a probability for the random walk.

4.2 Random Walk

In this section we describe one step of a random walk and show how the probability of each edge is calculated. Although the trust network which is used in this paper is a binary network, but we exploit a supervised random walk where the probability of each edge is different from another.

We perform the random walk from the source user u_0 (line 3 of Figure 2). At first step, we have to select one of the directly trusted neighbors of u_0 (line 6 of Figure 2). But first, the probability of edges must be calculated (line 5 of Figure 2). In this model, instead of selecting neighbors uniformly, we assign the expertise of each neighbor in item i as the edge probabilities. This makes it possible to visit the experts in item i more than other users. We define S_u as a random variable for selecting a user v from N_u:

$$p(S_u = v) = \frac{\psi(v,i)}{\sum_{w \in N_u} \psi(v,i)}. \tag{6}$$

Here, N_u is the set of all the neighbors of u and $\psi(u,i)$ denotes the expertise of user u in the item i.

Let us assume the user u is selected. If u already has the rating on the target item i (u is fully expert in the item i), then we stop the process and return $r_{u,i}$ as a result (lines 7–9 of Figure 2). However, if u does not have a rating on i, then we have two options:

- With probability α stop the random walk and return the weighted mean of ratings of items which are rated by u (lines 12–14 of Figure 2).
- With probability $1-\alpha$ continue the random walk and select one of the direct neighbors of u (lines 16–17 of Figure 2).

α is the probability of stopping the random walk at the node u. As mentioned earlier, by going far from the active user, the data will be less trustworthy. Thus the number of random walk steps (k) must be involved in the calculation of α to avoid and penalize long random walks. Sigmoid function can satisfy this constraint [6]. This function gives values 1 for big values of k, and 0 for small values of k. Thus, α is defined by Equation (7):

$$\alpha = \psi(u, i) \times \frac{1}{1 + e^{-k}}. \tag{7}$$

As each edge has a probability, there is a chance for a single random walk to continue forever. Therefore, we consider a threshold for going far from the active user (line 16 of Figure 2). We set this threshold to 6 based on the idea of "six-degrees of separation" [6, 31] and terminate the random walk when $k > 6$.

4.3 Termination Condition

Due to the probabilistic nature of a Random Walk (RW), the result of one RW cannot be accurate. Several random walks must be performed to produce a more reliable prediction. In this model, we use the TrustWalker termination condition model which uses the variances of obtained result of several random walks to stop the overall algorithm. As stated in [6], the variance of the results of all the random walks is used as follows:

$$\sigma^2 = \frac{\sum_{i=1}^{T} (r_i - \mu)^2}{T}. \tag{8}$$

Here, r_i is the result of i^{th} random walk, and μ denotes the average of the ratings returned by random walks. T is the number of random walks which is performed to compute the prediction. Since ratings are in a finite range of $[1, 5]$, it has been proved in [6] that σ^2 converges to a constant value. Moreover, ${\sigma_i}^2$ is defined as the variance of the results of the first i random walks. So EXTRA terminates if $|{\sigma_{i+1}}^2 - \sigma_i^2| \leq \epsilon$ (line 4 of Figure 2). In this algorithm, we consider the threshold of 100 for the maximum number of unsuccessful random walks, and after that, we consider the pair $\langle user, item \rangle$ as non-covered [6] (lines 18–19 of Figure 2). The quality of the output depends on the value of ϵ. The smaller ϵ the more accurate results. But we should note that by setting ϵ to small values, the running time of the algorithm will be increased. Thus, we set $\epsilon = 0.001$ for the termination condition (the reason will be explained in Section 6). The complete algorithm of the EXTRA model can be studied in Figure 2.

5 PROPERTIES OF EXTRA MODEL

The EXTRA model (same as TrustWalker) has the potential to be converted to the previous models and this shows the generality of this model. If $\alpha = 1$ for all $u \in U$, then our random walk will never start, and it will return weighted average of the

```
Initial $\sigma^2$, $\epsilon$, Max_Steps
for All U $\in$ Users do
Active User = U;
    while $\sigma^2 > \epsilon$ do
        Assign a value to edges base on users expertise;
        S = Select one of the Active User neighbors      $(S \in N_{ActiveUser})$;
        if S has a rating for target item then
            Return the rating value;
        end if
        Calculate $\alpha$ by Equation (7);
        R = Random Number;
        if R< $\alpha$ then
            SI = weighted-mean of S' ratings list;
            Return SI;
        else
            if Steps < Max_Steps then
                Continue the Random Walk;
            else
                Return "Cannot cover";
            end if
        end if
        Calculate $\sigma^2$;
    end while
end for
```

Figure 2. The EXTRA algorithm

ratings on items in RI_{u_0}. This is the same as the result of Item-based collaborative filtering proposed in [12].

On the other hand, if we consider that the expertise of all users is equal ($\forall u, v \in U$, $\psi(u,i) = \psi(v,i)$) and change the termination condition as $\alpha = 1$ if $\mathrm{r}_{u,i} \neq \emptyset$, then all random walks will continue until they have found a rating for the exact target item i. These changes will convert the EXTRA to existing methods which are explained in [3, 2].

Moreover, if we consider that the expertise of all users is equal ($\forall u, v \in U$, $\psi(u,i) = \psi(v,i)$) and change the termination condition as using maximum similarity instead of ψ $\left(\alpha = \max sim(i,j) \times \frac{1}{1+e^{-\frac{k}{2}}}\right)$, then the EXTRA will be converted to the TrustWalker model [6].

6 EXPERIMENTS

In this section the result of our experiments is reported. In our experiments the EXTRA algorithm is compared against some trust-aware methods such as Trust-

Walker [6] and pure random walk and some fundamental similarity based recommendation methods such as User-Based Collaborative Filtering [1] and Item-Based Collaborative Filtering [12].

6.1 Dataset Description and Experimental Design

Most of the datasets in the recommender system have no social network among users. We choose Epinions[4] as the data source for our experiments which has both a trust network and ratings expressed by users. Epinions.com is a consumers opinion site where users can review items and also assign numeric ratings in the range of [1–5]. Every member of Epinions maintains a "trust" list which presents a network of trust relationships between users, and a "block (distrust)" list which presents a network of distrust relationships. Inserting a user in the Web of Trust equals to issuing a trust statement with value 1 while inserting a user in the Block List equals to issuing a trust statement of value 0.

The dataset used in our experiments consists of 49 290 users who rated a total number of 139 738 different items at least once. The total number of reviews is 664 824. The total number of issued trust statements is 487 181. The rating matrix sparsity is 99.99135 %. In this dataset 52.82 % of users are cold start users (users with less than 5 ratings (same as [22]) which is a huge portion of users. So, considering the performance of recommendation for cold start users is very important. Table 1 shows the details of this dataset. Figure 3 shows the rating distribution over users in Epinion dataset. It is notable that 5 is the dominated rating in this dataset.

Number of users	49 290
Number of items	139 738
Number of trust statements	487 181
Number of reviews	664 824
Average number of out-degree	9.8768
Average number of item each user rate	13.4883
Number of users who do not have any neighbor	15 347
Number of users who rate less than 5 items	26 037

Table 1. Epinions dataset features

The main parameters of the EXTRA algorithm are the number of steps for random walks and the ϵ. Based on Milgram experiment [31], we set 6 for the longest random walks. However, choosing the ϵ is an important part in the EXTRA algorithm. The smaller ϵ we choose, the higher precision we achieve, but the time complexity increases dramatically. So, different values of ϵ were tested with EXTRA-Num (all the values have been normalized to $[0, 1]$). We considered the number of steps as 2. Based on Table 2 and Figure 4, we set $\epsilon = 0.01$, because in this case the $\frac{Precision}{TimeComplexity}$ is maximal.

[4] `http://www.trustlet.org/wiki/Downloaded_Epinions_dataset`

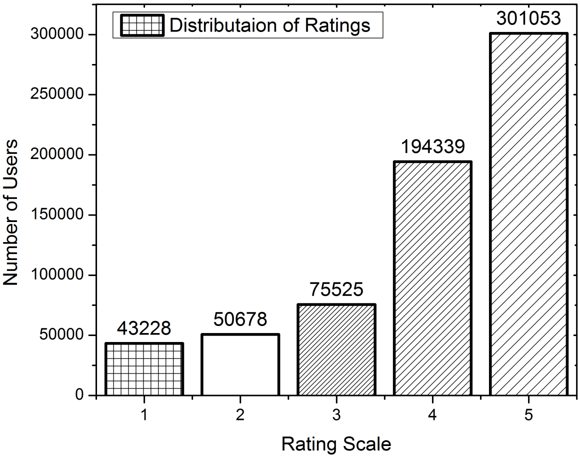


Figure 3. Rating distribution of Epinions dataset

ϵ	100	10	1	0.1	0.01	0.001
$Precision$	0.60	0.61	0.62	0.70	0.75	0.76
$Time(Sec)$	25 002	25 920	27 012	30 121	30 310	41 002
$\frac{Precision}{Time}$	2.42×10^{-5}	2.35×10^{-5}	2.31×10^{-5}	2.32×10^{-5}	2.47×10^{-5}	1.85×10^{-5}
$\frac{Precision}{Time}$ $(Normalized)$	0.90	0.79	0.71	0.75	1	0

Table 2. Results for different values of ϵ

We implemented all methods in MATLAB 7.10. We used an Intel Core i5 2.4 GHz CPU with 4 GB RAM to run our experiments on a Windows 7 system.

In this article we will review the results of the various models as follows:

TrustWalker Model: This method is a random walk method based on the trust and the item similarity [6].

Random Walk (pure): This method is an extension of TrustWalker which stops the random walks whenever finds a user with rated target item.

User-Based Collaborative Filtering: This method is the classic user-based collaborative filtering method, which makes prediction based on the user similarity [1].

Item-Based Collaborative Filtering: This method is the classic item-based collaborative filtering method, which makes prediction based on the item similarity [12].

EXTRA-Num: This method in the proposed model which uses Definition 1 for the calculation of user expertise.

EXTRA-Max: This method in the proposed model which uses Definition 2 for the calculation of user expertise.

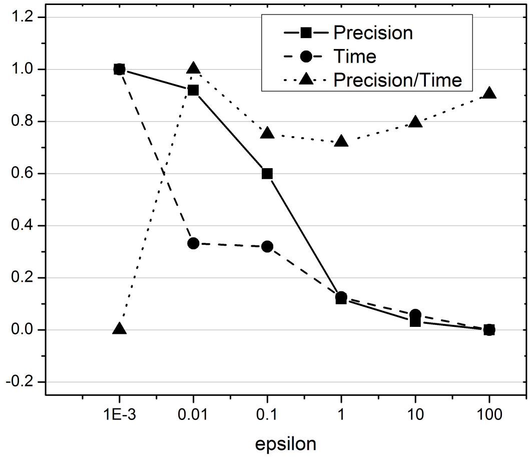


Figure 4. Results for different values of the ϵ

6.2 Accuracy Metrics

We use one-leave-out method to evaluate the recommender system. In this technique, one of the ratings has been masked and the system ties to predict it using the trust network [3, 2, 12]. Finally calculated value and the masked value are compared. For measuring the prediction quality we use the Root Mean Squared Error (RMSE) metric.

$$RMSE = \sqrt{\frac{\sum_{(u,i)} \left(r_{u,i} - P_{u,i}\right)^2}{N}} \tag{9}$$

where $r_{u,i}$ denotes the rating of the user u on an item i, $P_{u,i}$ denotes the rating of the user u on an item i as predicted by the system, and N denotes the number of pairs (u,i) which u has a rating on i. The smaller the value of RMSE, the more precise the recommendation is.

Another important measure is the coverage. We define a coverage metric to measure the percentage of pairs of *(user,item)* for which system can predict a rating.

To have a single evaluation metric and be able to compare the results with the previous models, we can combine RMSE and coverage. F-score can be used for this purpose. However, first, RMSE should be converted to a precision in the range of $[0, 1]$. As the values of ratings are in the range of [1–5], we define the precision as follows:

$$Precision = 1 - \frac{RMSE}{max\ error} \tag{10}$$

where *max error* is $5 - 1 = 4$. We define F-score as follows:

$$F\text{-}score = \frac{2 \times Precision \times Coverage}{Precision + Coverage}. \tag{11}$$

6.3 Results

In this section, we summarize and discuss the most important results of the presented experiments. Since more than half of the users are cold start users, considering the performance of the recommendation for cold start users is very important. Table 3 shows the RMSE, Coverage, and F-score for cold start users. We consider the maximum step of the random walk to 6 for all the algorithms. It should be mentioned that because of the probabilistic nature of the random walk, all the methods which exploited random walk were performed 5 times and an average of the results has been reported. Another point is, as the EXTRA algorithm is a greedy method which works locally and helps random walks to choose the best option for the next step, we set $\psi(u, i) = \varepsilon$ in case $\psi(u, i) = 0$, in order to give a chance to non-expert users to be selected and help EXTRA to escape from the local optimums.

As shown in Table 3, the proposed EXTRA model outperforms other methods according to each three criteria for the cold start users.

Notice that most of the methods which exploit trust network outperform basic methods according to RMSE and coverage. Random Walk (pure) is the only method that has lower coverage than others. Because it tries to find a user who has a rating for the target item, and as it just considers limited steps, finding a user with rated target item is not easy. Among all these methods (except EXTRA) TrustWalker has better RMSE and better coverage. The EXTRA-Num model outperforms TrustWalker by exploiting supervised random walk and considering the rating of expert users. Figures 5 to 8 compare the results of different methods according to the RMSE by considering the step of random walks.

As shown in Figures 5 to 8, there is no significant change in the RMSE after step 2 in the models which use the trust network. It indicates that considering the friends and a friend of friends is enough for the recommendation process and there is no need to consider further users. Considering more users not only does not change the RMSE, it also increases the running time of the system.

The results for all users are shown in Table 4. We observe the similar relative performance of all methods as for cold start users. It should be noted that all methods perform significantly better over all users since there is more information available than cold start users. EXTRA-Num outperforms all other methods in terms of RMSE, Coverage, and F-score.

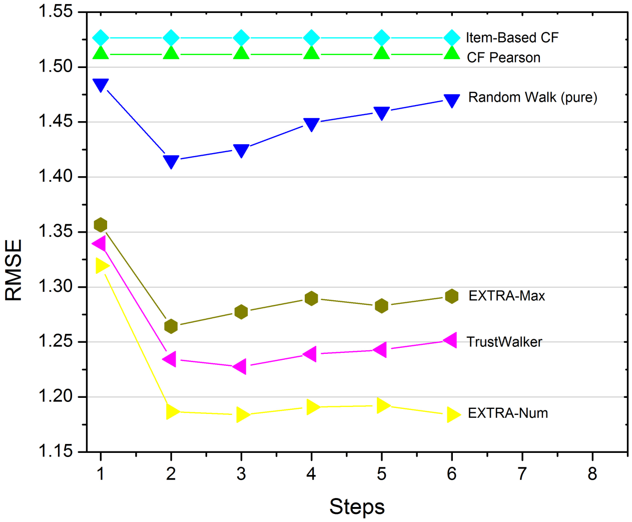


Figure 5. Comparison of RMSE for different methods on cold start users

Model	**RMSE**	**Coverage**	**F-score**
CF Pearson	1.51158	21.79916	0.32285
Random Walk (pure)	1.41527	17.25901	0.27242
Item-based CF	1.52669	28.17851	0.38714
TrustWalker	1.23438	68.62673	0.68883
EXTRA-Num	1.18678	74.76602	0.7248
EXTRA-Max	1.26438	74.21729	0.71185

Table 3. Results for cold start users

The results show that EXTRA-Num can better capture the expertise concept than EXTRA-Max. That is due to the fact that in Definition 2 we consider the maximum similarity of items as the user's expertise, however, imagine there is a user who rates 100 items which 99 of them have a negative correlation with target item but 1 has correlation 1. So, based on Definition 2, we consider this user as a fully expert user in item i, while it is not true.

In summary, EXTRA substantially not only improves the coverage of existing trust-aware approaches, it also improves the precision. This improvement is achieved by considering ratings of users which are expert in the target item, because ratings which are expressed by experts are more reliable than ratings expressed by ordinary users. For example, EXTRA-Num's RMSE is 1.18678 for cold start users while TrustWalker approaches have RMSE of 1.23438. Also, in the case of all

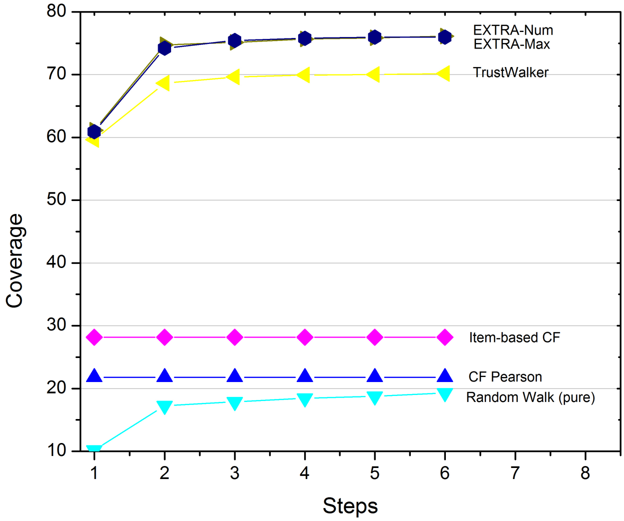


Figure 6. Comparison of Coverage for different methods on cold start users

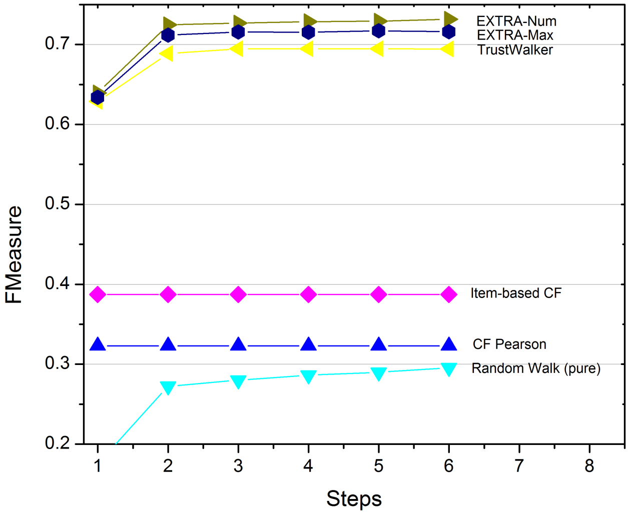


Figure 7. Comparison of F-score for different methods on cold start users

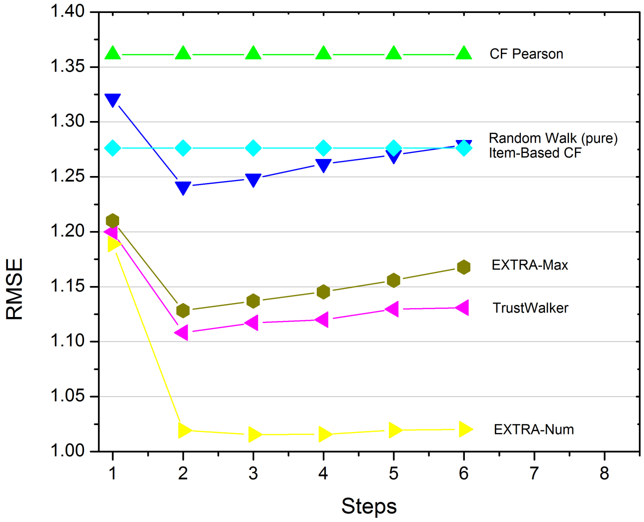


Figure 8. Comparison of RMSE for different methods on all users

Model	RMSE	Coverage	F-score
CF Pearson	1.36135	71.03196	0.68405
Random Walk (pure)	1.24128	60.3375	0.64365
Item-based CF	1.2762	70.0998	0.69083
TrustWalker	1.10822	92.8000	0.81274
EXTRA-Num	1.0193	94.76837	0.83432
EXTRA-Max	1.12837	94.76752	0.81694

Table 4. Results for all users

users, EXTRA-Num's RMSE is 1.0193 while TrustWalker approaches have RMSE of 1.10822. The 9 % RMSE reductions is an evidence for the significant improvement of accuracy of the trust-aware recommender systems.

7 CONCLUSIONS

In this paper, we presented a new approach for enhancing trust-aware recommender systems by exploiting expert users. We introduced a mechanism for finding experts just based on the ratings they make on items and consider just these users for prediction in the recommendation process.

The evaluation on the Epinions dataset showed that EXTRA outperforms both basic and pure trust-aware methods especially in terms of RMSE and coverage. Also, we showed that considering the friends and friends of a friend is adequate

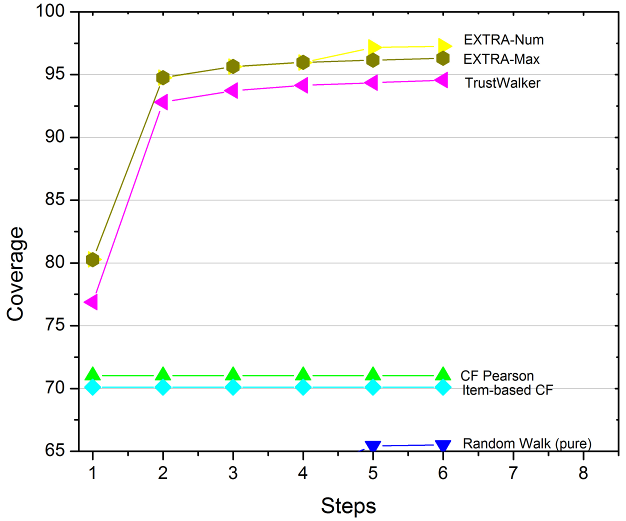


Figure 9. Comparison of Coverage for different methods on all users

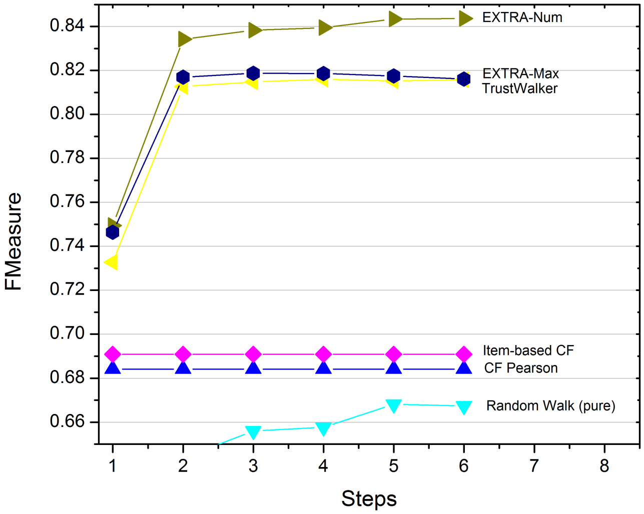


Figure 10. Comparison of F-score for different methods on all users

in trust-aware recommender systems and there is no need to go deeper into trust networks.

There are various ways to extend the EXTRA method for future works. This method can be evaluated against other datasets such as Flixter[5] with different properties. Flixster is a social networking service in which users can rate movies and create a social network. It is bigger and denser than the Epinions dataset. Moreover, defining expertise in a way that properly captures the context of expertise in real-world scenarios is another future work. In this paper, we assumed that the ratings are stored in a centralized repository. So, we could easily calculate the similarity between the two items for finding expert users. However, decentralized applications can have multiple repositories. So calculating the similarity of items in distributed repositories is another future work.

REFERENCES

[1] Goldberg, D.—Nichols, D.—Oki, B. M.—Terry, D.: Using Collaborative Filtering to Weave an Information Tapestry. Communications of the ACM, Vol. 35, No. 12, 1992, doi: 10.1145/138859.138867.

[2] Massa, P.—Avesani, P.: Trust-Aware Recommender Systems. ACM Conference on Recommender Systems (RecSys '07), USA, 2007, pp. 17–24, doi: 10.1145/1297231.1297235.

[3] Golbeck, J. A.: Computing and Applying Trust in Web-Based Social Networks. Ph.D. thesis, University of Maryland at College Park, 2005.

[4] Levien, R.—Aiken, A.: Advogato's Trust Metric. Available at: `http://advogato.org/trust-metric.html`, 2002.

[5] Ziegler, C.-N.: Towards Decentralized Recommender Systems. Ph.D. thesis, University of Freiburg, 2005.

[6] Jamali, M.—Ester, M.: TrustWalker: A Random Walk Model for Combining Trust-Based and Item-Based Recommendation. Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09), 2009, pp. 397–406, doi: 10.1145/1557019.1557067.

[7] Trias Mansilla, A.—de la Rosa Esteva, J. L.: Propagation of Question Waves by Means of Trust in a Social Network. In: Christiansen, H., De Tré, G., Yazici, A., Zadrozny, S., Andreasen, T., Larsen, H. L. (Eds.): Flexible Query Answering Systems (FQAS 2011). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7022, 2011, pp. 186–197.

[8] Shah, P.: Expert Finding Using Social Networking. Master's Thesis, San Jose State University, 2009.

[9] Davoodi, E.—Afsharchi, M.—Kianmehr, K.: A Social Network-Based Approach to Expert Recommendation System. In: Corchado, E., Snášel, V., Abraham, A., Woźniak, M., Graña, M., Cho, S. B. (Eds.): Hybrid Artificial Intelligent

[5] `https://www.flixster.com/`

Systems (HAIS 2012). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 7208, 2012, pp. 91–102.

[10] Rettinger, A.—Nickles, M.—Tresp, V.: A Statistical Relational Model for Trust Learning. Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems – Volume 2 (AAMAS '08), 2008, pp. 763–770.

[11] Ma, H.—Yang, H.—Lyu, M. R.—King, I.: SoRec: Social Recommendation Using Probabilistic Matrix Factorization. Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM '08), 2008, pp. 931–940, doi: 10.1145/1458082.1458205.

[12] Sarwar, B.—Karypis, G.—Konstan, J.—Riedl, J.: Item-Based Collaborative Filtering Recommendation Algorithms. Proceedings of the 10th International Conference on World Wide Web (WWW '01), 2001, pp. 285–295, doi: 10.1145/371920.372071.

[13] Xue, G.-R.—Lin, C.—Yang, Q.—Xi, W.—Zeng, H.-J.—Yu, Y.—Chen, Z.: Scalable Collaborative Filtering Using Cluster-Based Smoothing. Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05), 2005, pp. 114–121, doi: 10.1145/1076034.1076056.

[14] Hofmann, T.: Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis. Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '03), 2003, pp. 259–266, doi: 10.1145/860435.860483.

[15] Hofmann, T.: Latent Semantic Models for Collaborative Filtering. ACM Transactions on Information Systems, Vol. 22, 2004, No. 1, pp. 89–115, doi: 10.1145/963770.963774.

[16] Si, L.—Jin, R.: Flexible Mixture Model for Collaborative Filtering. Proceedings of the 20th International Conference on Machine Learning (ICML '03), 2003, pp. 704–711.

[17] Canny, J.: Collaborative Filtering with Privacy via Factor Analysis. Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '02), 2002, pp. 238–245, doi: 10.1145/564376.564419.

[18] Breese, J. S.—Heckerman, D.—Kadie, C.: Empirical Analysis of Predictive Algorithms for Collaborative Filtering. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI '98), 1998, pp. 43–52.

[19] Herlocker, J. L.—Konstan, J. A.—Borchers, A.—Riedl, J.: An Algorithmic Framework for Performing Collaborative Filtering. Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99), 1999, pp. 230–237, doi: 10.1145/312624.312682.

[20] Jin, R.—Chai, J. Y.—Si, L.: An Automatic Weighting Scheme for Collaborative Filtering. Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '04), 2004, pp. 337–344, doi: 10.1145/1008992.1009051.

[21] Deshpande, M.—Karypis, G.: Item-Based Top-N Recommendation Algorithms. ACM Transactions on Information Systems, Vol. 22, 2004, No. 1, pp. 143–177, doi: 10.1145/963770.963776.

[22] Linden, G.—Smith, B.—York, J.: Amazon.com Recommendations: Item-to-Item Collaborative Filtering. IEEE Internet Computing, Vol. 7, 2003, No. 1, pp. 76–80.

[23] Kamvar, S.—Schlosser, M.—Garcia-Molina, H.: The EigenTrust Algorithm for Reputation Management in P2P Networks. Proceedings of the Twelfth International Conference on World Wide Web (WWW '03), 2003, pp. 640–651, doi: 10.1145/775152.775242.

[24] Guha, R.: Open Rating Systems. Technical report, Stanford Knowledge Systems Laboratory, Stanford, 2003.

[25] Richardson, M.—Agrawal, R.—Domingos, P.: Trust Management for the Semantic Web. In: Fensel, D., Sycara, K., Mylopoulos, J. (Eds.): The Semantic Web (ISWC 2003). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2870, 2003, pp. 351–368.

[26] Page, L.—Brin, S.—Motwani, R.—Winograd, T.: The Pagerank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.

[27] Levien, R.—Aiken, A.: An Attack-Resistant, Scalable Name Service. Draft Submission to the Fourth International Conference on Financial Cryptography, 2000.

[28] Golbeck, J.—Parsia, B.—Hendler, J.: Trust Networks on the Semantic Web. In: Klusch, M., Omicini, A., Ossowski, S., Laamanen, H. (Eds.): Cooperative Information Agents VII (CIA 2003). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 2782, 2003, pp. 238–249.

[29] Wasserman, S.—Faust, K.: Social Network Analysis. Cambridge University Press, 1994, doi: 10.1017/CBO9780511815478.

[30] Baskaya, O.—Aytekin, T.: How Similar is Rating Similarity to Content Similarity? Workshop on Recommendation Utility Evaluation: Beyond RMSE (RUE 2012) held in conjunction with ACM RecSys 2012, Dublin, Ireland, 2012, pp. 27–29.

[31] Milgram, S.: The Small World Problem. Psychology Today, Vol. 1, 1967, No. 1, pp. 61–67.

[32] Ozsoy, M. G.—Polat, F. S.: Trust Based Recommendation Systems. Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM '13), 2013, pp. 1267–1274, doi: 10.1145/2492517.2500276.

[33] Baby, B.—Murali, S.: A Survey on Trust Based Recommendation Systems. International Research Journal of Engineering and Technology (IRJET), Vol. 3, 2016, No. 10, pp. 1021–1023.

[34] Afef, S.—Brahmi, Z.—Gammoudi, M. M.: Trust-Based Recommender Systems: An Overview. 27th IBIMA Conference, Milan, Italy, 2016.

Farshad Bakhshandegan Moghaddam is computer science Ph.D. candidate at Karlsruhe Institute of Technology (KIT). His research interests include various aspects of recommender systems, semantic web and natural language processing. He is currently focusing on word and network embedding models and their usage in recommendation systems. Prior to KIT, he was fortunate to be advised by Bahram Sadeghi as Master student at the Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan, Iran.

Bahram Sadeghi Bigham is Associate Professor in computer sciences at the Institute for Advanced Studies in Basic Sciences (IASBS). His research interests are in the areas of robot motion planning, computational geometry, algorithms, AI and medical applications. Prior to arriving at IASBS, he worked as Postdoctoral Fellow at the University of Cardiff in the School of Computer Science. In June 2008, he completed his Ph.D. at Amirkabir University of Technology (Tehran Polytechnic), where he also completed an M.Sc. in 2000. His B.Sc. is from University of Birjand in mathematics.