Link Prediction in Stochastic Social Networks: Learning Automata Approach

Behnaz Moradabadi and Mohammad Reza Meybodi
Department of Computer Engineering
Amirkabir University of Technology
Tehran, Iran
moradabadi@aut.ac.ir
mmeybodi@aut.ac.ir

Abstract. Link prediction is a main social network challenge that uses the network structure to predict future links. The common link prediction approaches to predict hidden links use a static graph representation where a snapshot of the network is analyzed to find hidden or future links. For example, similarity metric based link predictions are a common traditional approach that calculates the similarity metric for each non-connected link and sort the links based on their similarity metrics and label the links with higher similarity scores as the future links. Because people activities in social networks are dynamic and uncertainty, and the structure of the networks change over time, using deterministic graphs for modeling and analysis of the social network may not be appropriate. This paper proposes a new link prediction method based on learning automata for stochastic social networks. In a stochastic social network, the weights associated with the links are random variables. To do this, we first redefine some of the similarity metrics for link prediction in stochastic graphs and then propose a method based on learning automata to calculate the distribution of the proposed similarity metrics assuming that the probability distributions of the link weights are unknown. Also, the proposed method has capability to use in online stochastic social networks where the social network changes online and the future links must be predicted. To evaluate the proposed method we use different synthetic stochastic social networks and present that the stochastic link prediction achieves better results in comparison to the classical link prediction algorithm in the stochastic social networks.

Keywords: Stochastic Social Network, Link Prediction, Learning Automata.

1 Introduction

The predicting linkage between data objects is an interesting task in data mining research area. For examples predict web hyperlink creation, genetic prediction, protein-protein interactions, and the record linkage problem. In link prediction problem the data are represented with a network/graph representation. These data can be visualized as graphs, where a vertex corresponds to a person and a link represents some form of association between the corresponding persons (1) (2). The concept of a link in a social network usually is a common interest of the corresponding social network. All link prediction methods address the following question: "Given a pair of nodes u and v in the current social network, how likely it is that u will interact with v in the future?" (1). Link prediction can be applied in very areas. For example, in the Internet and web science applications, it can be used to find automatic web hyperlink creation (3) and website hyperlink prediction (4). In e-commerce, link prediction can be used to produce recommendation systems (5) (6). It also has various applications in other scientific disciplines: in the bibliography, library science for deduplication (7), record linkage (8); in Bioinformatics, for protein-protein interaction (PPI) prediction (9), and in security applications to identify hidden groups of terrorists and criminals (2).

Most of the previous link prediction methods have been proposed based on a static network representation, where a snapshot of the network structure is available and the goal is to predict the future links. In such a static network, each link occurrence is represented by a one-time event and all interest is only the existence of the link. For example, one may be interested to know whether a customer will purchase a product in the future or whether an author will ever collaborate with another author in the future. But in many applications, the social networks are really online, non-deterministic and unpredictable, and the structure of the network and its parameters change over time; so using deterministic social network models with fixed values for links are restrictive in solving real social network problems. In other words, link prediction methods based on the static graph representation fail when the social network has online and non-deterministic behavior. One of the solutions to overcome this problem is link prediction using online and stochastic social networks in which each link is a random variable. In (10) Rezvanian proposed the stochastic social network and showed that the stochastic social network with a random variable for each link is a good approach to overcome this problem. By choosing a stochastic social network, they redefined some measures of social networks such as degree distribution, betweenness, cluster coefficient and strongness measures. Also, it is be noticed that some problems in stochastic networks were also proposed by other studies: path in the stochastic network (11), cover in the stochastic network (12), clique in the stochastic network (13), and spanning tree in the stochastic network (14). So in this paper, we propose a learning automata based link prediction in stochastic social networks, called SLP, to predict the link occurrence using the uncertainty of data. Also the proposed method can be extended as an online link prediction to use in the online stochastic social networks where the links can be added or removed over time and the future links must be predicted again.

Learning Automaton (LA) is an adaptive decision-making unit that tries to learn the optimal action from a set of allowable actions by interacting with a random environment (15). In each step, it selects an action from its action set. The action selection in the LA is based on a probability distribution over the action set. The selected action is applied to the environment and then a reinforcement signal is produced by the environment. LA updates the

probability distribution of its actions according to both reinforcement signal and a learning algorithm and again chooses an action. These steps are repeated until LA converges to some action.

This paper tries to propose a link prediction method based on similarity metrics in stochastic networks, in which each link has a random variable with an unknown probability distribution function. Traditionally in similarity based link prediction, the starting point is to extract the values/scores of a chosen similarity metric that represent the proximity of the pairs of nodes. Then, the pairs of non-connected nodes are at first ranked according to a chosen metric (for instance the number of common neighbors). After that, the top L ranked pairs are assigned as predicted links. To put it another way, it is always assumed that the links have the highest scores are most likely to occur. To obtain the goal of this paper, after present a brief overview of recent studies for link prediction in a social network, we first redefine some similarity metrics for stochastic graphs and then design a learning automata based algorithm for calculating the similarity metric under the situation that the probability distribution function of the weight of each link is unknown. In other words, the proposed algorithm tries to estimate the probability distribution function of the similarity metric using learning automata and sampling. The process of sampling from the links of the graph is guided by the aid of learning automata in such a way that the number of samples needed to be taken from the links of the stochastic graph and the computational complexity for estimating the similarity metrics probability distributions be reduced as much as possible. Similar to traditional similarity methods that the output of link prediction is the set of links with higher similarity, the output of the proposed link prediction is also a set of stochastic links with their similarity probability distribution functions. So, to determine a link's existence, we should sample from its similarity probability distribution function and based on this sample and a predefined threshold we make the decision about the link existence. This threshold is chosen based on empirical results and experts. It is also should be noticed that the proposed method can be applied in online stochastic social networks. In order to evaluate the performance of the proposed algorithm in stochastic social networks, several experiments are conducted using different synthetic stochastic social networks. Experimental results show that the stochastic models achieve better link prediction performance in comparison common stochastic link prediction method.

The rest of the paper is organized as follows. Section 2 dedicated to material and methods including a brief introduction of link prediction problem, an overview of recent works about dynamics of user behaviors in social network and a brief description of learning automata theory. In section 3, the proposed similarity metrics for stochastic graphs are described. Section 4 introduces the proposed link prediction method for stochastic social networks based on learning automata. Section 5 presents the simulation results and finally section 6 summarizes the main conclusion of the proposed method.

2 Material and Methods

In this section, to provide the necessary background, we present a brief description of link prediction. We also briefly review the studies performed by the researchers about the distributions of user behaviors in social networks. At the end of this section, learning automata and variable action set learning automata are introduced.

2.1 Link Prediction

A classic definition of the link prediction problem is expressed by: "Given a snapshot of a social network at time t, we seek to accurately predict the links that will be added to the network during the interval from time *t* to a given future time *t+1*" (16). The most widespread approach to the problem is to calculate the topological/structural patterns of the social network of interest (17). Different metrics to describe node pairs have already been adopted in previous studies (18) and they explore the structural patterns of the network and commonly provide a degree of proximity/similarity between the nodes. There are many similarity metrics include (16): 1) Local similarity metrics that only use the local information of a link to calculate similarity metric: Common Neighbors, Salton Index, Jaccard Index, Hub Depressed Index, Hub Promoted Index, Leicht-Holme Newman Index (LHN1), Preferential Attachment Index, Adamic-Adar Index and Resource Allocation Index, 2) Global similarity metrics that can use all information in the network to calculate the similarity metric between two nodes: Katz Index, Leicht-Holme-Newman Index (LHN2), Matrix Forest Index (MFI) and 3) Quasi-local metrics that do not require global topological information but use more information than local indices: Local Path Index, Local Random Walk, Superposed Random Walk, Average Commute Time, Cos+, random walk with restart, SimRank, Resource Allocation index and Local Path index. As previously mentioned, the starting point in these approaches is to extract the values/scores of different metrics that represent the proximity of the pairs of nodes. Then, the pairs of non-connected nodes are at first ranked according to a chosen metric (for instance the number of common neighbors) (16). After that, the top L ranked pairs are assigned as predicted links. To put it another way, it is always assumed that the links have the highest scores are most likely to occur. In the following we briefly present the eight commonly similarity measures that we use them to redefine stochastic similarity metrics:

1) Common Neighborhood: In this measure, two nodes v_i and v_j are more likely to have a link if they have many common neighbors. This score is defined as

$$CN(v_i, v_j) = |\Gamma(v_i) \cap \Gamma(v_j)|$$

where $\Gamma(x)$ denotes neighbors of node x.

2) Salton: This score is defined as

$$Salton(v_i, v_j) = \frac{|\Gamma(v_i) \cap \Gamma(v_j)|}{\sqrt{|\Gamma(v_i)| \times |\Gamma(v_j)|}}$$

3) Jaccard Index (19): This index was proposed by Jaccard and it is defined as:

$$Jaccard(v_i, v_j) = \frac{|\Gamma(v_i) \cap \Gamma(v_j)|}{|\Gamma(v_i) \cup \Gamma(v_j)|}$$

4) Preferential Attachment (PA) (20): The preferential attachment (PA) algorithm is based on the preferential attachment phenomena rule (21) that is discovered in a variety of social networks. In this method the link score is set to be the product of the degrees of the involved nodes and it is defined as follows:

$$PA(v_i, v_j) = |\Gamma(v_i)| \times |\Gamma(v_j)|$$

5) Adamic-Adar Index (AA) (22): This index is an extension of common neighborhood method such that the less-connected neighbors have more weight and it is defined as:

$$AA(v_i, v_j) = \sum_{z \in \Gamma(v_i) \cap \Gamma(v_j)} \frac{1}{\log(\Gamma(z))}$$

6) Resource Allocation Index (RA): This index and Adamic-Adar Coefficient have similar formulas but they come from different motivations. RA is based on physical processes of resource allocation and can be applied to networks formed by airports (flow of aircraft and passengers) or networks formed by electric power stations and is defined as:

$$RA(v_i, v_j) = \sum_{z \in \Gamma(v_i) \cap \Gamma(v_j)} \frac{1}{|\Gamma(z)|}$$

7) Katz Index: In this metric, a similarity is defined as the sum of a number of paths with different lengths such that shorter paths have more weights. It is defined by the following equation:

$$Katz(v_i, v_j) = \sum_{l=1}^{\infty} \beta^l . |Path(v_i, v_j)^{<l>}|$$

Where $|Path(x, y)^{<l>}|$ is the number of paths between v_i and v_j with length l. It is also shown that the Katz metric can be calculated based on the following equation:

$$Katz = (I - \beta A)^{-1} - I$$

8) LP Index: This index is a restricted version of Katz metric such that only paths in length 1 and 2 is considered. This metric has a lower computational complexity in comparison to Katz and it is defined as the following:

$$LP\ Index(v_i, v_j) = A^2 + \epsilon A^3$$

For more information about other similarity metrics please refer to Liben- Nowell and Kleinberg (16). In the rest of this section, we go on to review recently proposed link prediction methods:

In (23) a new data mining process, called Interaction Prediction, is proposed for the link prediction problem in dynamic networks. Interaction Prediction uses dynamic social networks time series forecast in addition to similarity metrics and network community structure to predict the links that may appear in the future. The proposed method focuses on the links between the communities and their experiments on real-world social networks show that the Interaction Prediction outputs interesting results in link prediction.

The authors of (24) have provided a link prediction method that uses the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) to determine the importance of each similarity metric in a linear format. They used a large dynamic social network and their method presents fast convergence and high accuracy for link prediction problem. The authors of (25) proposed a new link prediction using information theory and mutual information of network structure (MI-LP). They compared the model with six typical prediction methods on ten networks. Their results show two good features: improving the link prediction accuracy and the reasonable computational complexity. In (26) a new method called Multivariate Time Series Link Prediction is proposed for link prediction in evolving networks. This method combines temporal evolution of the network in addition to node similarities and

node connectivity information. This method uses links and a calculated similarity metric for each time. The authors finally compare different similarity metrics in their experiments.

In (27) a weighted approach for modeling the occurrence of time is used to generate a different similarity metric and in our previous research we have proposed a new link prediction method based on temporal similarity metrics and Continuous Action set Learning Automata (CALA) (28). The proposed method takes advantage of using different similarity metrics as well as different time periods. In the proposed algorithm, we have modeled the link prediction problem as a noisy optimization problem and use a team of CALAs to solve the noisy optimization problem. The obtained link prediction results show satisfactory of the proposed method for some social network data sets. Finally in (29), to predict the future links we have proposed a new link prediction which uses both fuzzy social network and DLA (FLP-DLA). In the proposed method we first create a fuzzy social network where each link has a fuzzy strength and then use this network to predict future links using distributed learning automata based on graph navigation and the fuzzy strength of links. In order to examine the result of the proposed method, we conducted some evaluations and compared our results with the ones achieved by other link prediction techniques. In general, the experiments showed that the proposed approach performs better than other strategies.

2.2 User Behavior in Social Networks

Since generating and sharing information through social networks are done by users, studying how users behave and interact with their friends in online social networks play a significant role in social network analysis. Based on valuable research in social network dynamics, now we can produce synthetic graphs using computer programs that are similar to real networks in order to study the structures and dynamics of networks and their user activities in a systematic manner. In recent years, several research works regarding user activity modeling, characterizing the distribution of user activities and statistical observations from users and in the rest of this section, we brief review some of them. Reference (30) analyzed the interactions among Facebook users. They found that the distribution of the number of messages sent by a user follows a power-law distribution. Reference (31) studied thousands of blogs and millions of posts, they showed that the popularity of posts drops with a Power-law distribution. Leskovec et al. in (32), also focused directly on the microscopic node behavior of four large social network datasets (Flickr, Delicious, Yahoo Answers, and LinkedIn) and observed that user's lifetime follows an exponential distribution. Nazir et al. in (33) showed that the popularity of the applications follows a power-law distribution with an exponential decay. Reference (34) analyzed the patterns of user content generation in online social networks and they found that the overall user lifetime follows the stretched exponential distribution. Kwak et al. in (35) collect user profiles, social relations, trending topics and tweets on Twitter and they found the distribution of the users in a Retweet tree follows a power-law distribution. Gyarmati et al. in (36) analyzed the degree distribution of nodes for online sessions of users in Bebo, MySpace, Netlog and they revealed that the time spent online by users follows a Weibull distribution. Galuba et al. in (37) analyzed the information propagation in Twitter and they found that posting frequencies followed a power-low distribution across users and URLs. Yan et al. in (38) studied the interval time distribution of people posting in microblogs systems and found that it follows a power-law distribution. Liu et al. in (39) found that the number of daily active users using some applications in Facebook follows a power-law distribution. Vongsingthong et al. in (40) tracked the activities of some users in a Facebook

through a questionnaire and they found questioning on products follows a Power-law distribution. Bild et al. in (41), studied empirically aggregate user behavior on Twitter and they found that the lifetime tweet distribution is a type-II discrete Weibull stemming from a Power-law hazard function. For more information about related studies please refer to (10).

2.3 Learning Automata

A learning automaton (42) is an agent that can make a decision and learn how to choose the true decision by interacting with a random environment iteratively. Each LA has a probability distribution over its finite action-set and at each iteration it selects an action based on the corresponding probability and sends the action to the random environment. The environment sends a reinforcement signal to the LA based on the evaluation of the input action. The action probability vector of the LA is updated based on the given reinforcement signal from the random environment. The goal of a learning automaton is to find the optimal action such that the average penalty received from the environment is minimized as much as possible. The environment can be described by a triple $E \equiv \{\alpha, \beta, c\}$, where $\alpha \equiv \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ represents the finite set of the inputs, $\beta \equiv \{\beta_1, \beta_2, \dots, \beta_m\}$ denotes the set of the values that can be taken by the reinforcement signal, and $c \equiv \{c_1, c_2, \dots, c_m\}$ denotes the set of the penalty probabilities, where the element c_i is associated with the given action α_i. If the penalty probabilities are constant, the random environment is said to be a stationary random environment, and if they vary with the time, the environment is called a non-stationary environment. The random environments based on their reinforcement signal β can be classified into P-model, Q-model and S-model. The reinforcement signal can only take two binary values 0 and 1 in P-model environments, take one of a finite number of values in the interval $[0, 1]$ in Q-model environment, and take a value in the interval $[a, b]$ in S-model. The relationship between the learning automaton and its random environment has been shown in Figure. 1 (42). Learning automata can be classified into two main families (42): fixed structure learning automata and variable structure learning automata. Variable structure learning automata are represented by a triple $\langle \beta, \alpha, T \rangle$, where β is the set of inputs, α is the set of actions, and T is the learning algorithm. The learning algorithm is a recurrence relation which is used to modify the action probability vector. Various learning algorithms have been proposed. But because we have S-model learning automata in the proposed algorithm we use the following learning algorithm: Let α, be the action chosen at step n from the probability distribution p. The linear reward-inaction algorithm is one of the learning schemas and its recurrence equation for updating action probability vector p is defined as the following equation:

$$p_i(n+1) = p_i(n) + a.\big(1 - \beta(n)\big).\big(1 - p_i(n)\big) - b.\beta(n).p_i(n) \qquad (1)$$

$$p_j(n+1) = p_j(n) + a.\big(1 - \beta(n)\big).p_j(n) + \frac{b.\beta(n)}{r-1} - b.\beta(n).p_i(n) \quad j \neq i$$

Where a and b denote the reward and penalty parameters and determine the amount of increases and decreases of the action probabilities, respectively.

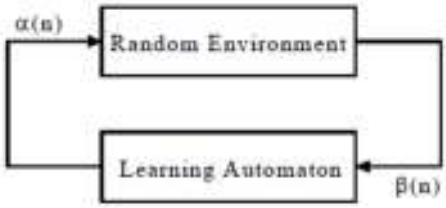

Figure 1. The relationship between a learning automata and its random environment

Learning automata have been found to be useful in systems where incomplete information about the environment, wherein the system operates, exists and recently, several learning automata based approaches have been presented for improving the performance of many applications (43) (44) (45).

3 Proposed Similarity Metrics in Stochastic Graphs

Generally, a stochastic graph G can be described by a triple $< V, G, W >$, where $V = \{v_1, v_2, \dots, v_n\}$ is the set of nodes, $E = \{e_{ij}\} \subseteq V \times V$ is the set of links , and W is a matrix in which w_{ij} is a random variable associated to link e_{ij} if such a link exists. In this section we redefine some similarity metrics for link prediction problem in stochastic network as the following:

1) Stochastic Common Neighborhood: In this measure, the common neighborhood random variable is a random variable that represents the sum of shared stochastic weights that the two nodes v_i and v_j have in their common neighbors and it is defined by:

$$CN(v_i, v_j) = \sum_{\{k|e_{ik}\in E, e_{jk}\in E\}} \min(\overline{w}_{ik}, \overline{w}_{jk})$$

where $\Gamma(x)$ denotes neighbors of node x.

2) Stochastic Salton: This score in stochastic social networks is defined as the following random variable:

$$Salton(v_i, v_j) = \frac{\sum_{\{k|e_{ik}\in E, e_{jk}\in E\}} \min(\overline{w}_{ik}, \overline{w}_{jk})}{\sqrt{\sum_{\{k|e_{ik}\in E\}} \overline{w}_{ik} \times \sum_{\{l|e_{jl}\in E\}} \overline{w}_{jl}}}$$

3) Stochastic Jaccard Index: This index in stochastic social networks is defined as the following random variable:

$$Jaccard(v_i, v_j) = \frac{\sum_{\{k|e_{ik}\in E, e_{jk}\in E\}} \min(\overline{w}_{ik}, \overline{w}_{jk})}{\sum_{\{k|e_{ik}\in E, e_{jk}\in E\}} \max(\overline{w}_{ik}, \overline{w}_{jk})}$$

4) Stochastic Preferential Attachment: The preferential attachment (PA) random variable in the stochastic network is defined as follows:

$$PA(v_i, v_j) = \sum_{\{k|e_{ik}\in E\}} \overline{w}_{ik} \times \sum_{\{l|e_{jl}\in E\}} \overline{w}_{jl}$$

5) Stochastic Adamic-Adar Index (22): This index is an extension of common neighborhood method and in stochastic network is defined as the following random variable:

$$AA(v_i, v_j) = \sum_{\{k|e_{ik}\in E, e_{jk}\in E\}} \frac{1}{\log\left(\sum_{\{z|e_{zk}\in E\}} \overline{w}_{zk}\right)}$$

6) Resource Allocation Index: this index in the stochastic social network is defined as the following random variable:

$$RA(v_i, v_j) = \sum_{\{k|e_{ik}\in E, e_{jk}\in E\}} \frac{1}{\sum_{\{z|e_{zk}\in E\}} \overline{w}_{zk}}$$

7) Stochastic Katz Index: In this metric, the similarity is defined as the sum of the stochastic paths weight between v_i and v_j with different lengths such that shorter paths have more weights and it is defined as the following random variable:

$$Katz(v_i, v_j) = \sum_{l=1}^{\infty} \beta^l \times \sum_{\{e_{zk}\in \text{path}(v_i, v_j)^{<l>}\}} \overline{w}_{zk}$$

Where $\text{path}(v_i, v_j)^{<l>}$ is any path between v_i and v_j with length l.

8) Stochastic LP Index: This index is a restricted version of Katz metric such that only paths in length 2 and 3 is considered. This metric in the stochastic social network is a random variable and it is defined as the following:

$$LP - Index(v_i, v_j) = \sum_{l=2}^{3} \beta^l \times \sum_{\{e_{zk}\in \text{path}(v_i, v_j)^{<l>}\}} \overline{w}_{zk}$$

4 Proposed Link Prediction Method in Stochastic Graphs

In the previous section, we redefined some similarity metrics for stochastic graphs. In this section, a link prediction method based on learning automata (SLP) is proposed for stochastic social networks under the situation that the probability distribution functions of the weights associated with the links of the graph are unknown. The proposed algorithms take advantage of using learning automata to estimate the distribution of some chosen similarity metric in order to predict future links. In the algorithm after initialization phase, it iterates the sampling phase, updating phase and feedback phase until one of the stopping conditions are met. Similar to traditional similarity methods that the output of link prediction is the set of links with higher similarity, the output of the proposed link prediction is also a set of stochastic links with their similarity probability distribution functions. So, to determine a link's existence, we should sample from its similarity probability distribution function and based on this sample and a predefined threshold we make the decision about the link existence. This threshold is chosen based on empirical results and experts. The proposed method also can have an additional phase called changes phase that can

be used in online stochastic social networks to predict the future links after a change in the network has occurred. The details of initialization of the proposed method, sampling phase, updating phase, feedback phase, stopping phase and changes phase are given below.

4.1 Initialization

Let $G(V,E,W)$ be the input stochastic graph, where $V=\{v_1,v_2,\dots,v_n\}$ is the set of nodes, $E=\{e_{ij}\}\subseteq V\times V$ is the set of links, and $W=\{w_{ij}\}$ is the set of random variables each of which is associated with a link weight of the input stochastic graph. It is assumed that the weight of each edge is a positive random variable with an unknown probability distribution function. The proposed algorithm uses two sets of learning automata LA_{Links} and LA_{Tests}:

1) LA_{Links} is a set of LAs for each of existed links in the network and tries to estimate the importance of sampling from the corresponding links in calculation of similarity metrics. Each LA_L in LA_{Links} have two actions: $\{\alpha_L^1=$ "take sample", $\alpha_L^2=$ "do not take sample"$\}$. Let $p_{L_i}=\{p_{L_i}^1,p_{L_i}^2\}$ be the action probability vector of learning automaton LA_{L_i}, and initialized equally $p_{L_i}^1=p_{L_i}^2=1/2$ for all $v_i\in V$. The process of taking samples from a link is determined by the learning automaton assigned to that link. So this set of LAs is learned in a such way that the number of samples of the network that is required for calculating the similarity metric be reduced as much as possible. At the beginning of the algorithm, the weight of each link is initialized with some random samples in order to provide a coarse estimate of the weight of that link.
2) LA_{Tests} is a set of LAs for each of test links that must be predicted and tries to estimate if the distribution of the similarity metric corresponding to the test link must be updated or not. Each LA_T in LA_{Tests} also has two actions: $\{\alpha_T^1=$ "update the similarity metric probability distribution", $\alpha_T^2=$"do not update the similarity metric probability distribution"$\}$. Let $p_{T_i}=\{p_{T_i}^1,p_{T_i}^2\}$ be the action probability vector of learning automaton LA_{T_i}, and initialized equally $p_{T_i}^1=p_{T_i}^2=1/2$ for all $t_i\in T$ (Test set). The action of each LA_T in LA_{Tests} determines that if we should update the similarity metric probability distribution of the corresponding link or not. So the LA_{Tests} is guided such that the computational complexity of calculating similarity metrics is decreased as much as possible.

4.2 Sampling Phase: Sampling the stochastic social network

In each iteration of the proposed method, in the first phase, each LA_L in the LA_{Links} chooses an action in a parallel manner. The action of each LA_L determines that we should sample a new weight or use the previous weight as the weight of corresponding link in the current iteration. If the chosen action be $\alpha_L^1=$ "take sample", then we sample a new weight for the corresponding link and if it be $\alpha_L^2=$ "do not take sample" then we use the weight of previous iteration as the weight of corresponding link in the current iteration. So this set of LAs is learned in a such way that the number of samples of the network that is required for calculating final similarity metrics probability distributions be reduced as much as possible. In other words in SLP the sampling process implemented by learning automata aims to take more samples from the promising region of the graph, the regions that reflects higher rate of

activities, instead of walking around and taking unnecessary samples from non-promising regions of the graph. Now we have a weighted social network and the first phase of the SLP is finished here.

4.3 Updating Phase: Update the similarity metrics probability distributions

In this phase, we try to update the probability distribution of the similarity metrics using the weighted graph that is generated in the previous phase. To do this, each LA_T in the LA_{Tests} selects a new action. If the action of the LA_T be α_T^1= "update the similarity metric probability distribution", the calculation of the new estimates for the probability distribution of the similarity metric for the corresponding link are then performed. But if the chosen action be α_T^2 ="do not update the similarity metric probability distribution"} then we do not update the similarity metric probability distribution of the corresponding link and we use its previous probability distribution as its current probability distribution.

4.4 Feedback Phase: Update the action probabilities of LAs

The goal of this phase is to reward or penalize each LA in the network to update their action probability distributions. To do this first we calculate the distance between the current similarity metric probability distribution and the one obtained in the previous iteration for every test link i and called it d_i. To calculate the distance we use Skew divergence distance that is described in the experiment setting. This metric is chosen with empirical experiments and based on the trade-off of computational complexity and accuracy. Now we use this value to generate reinforcement signal for every LA in the network based on the following equations:

$$\beta_{T_i} = d_i \qquad for\ every\ i\ \in\ LA_{Test},$$

$$\beta_{L_j} = \frac{\sum_{i \in m} d_i}{|m|} \qquad for\ every\ i\ \in\ LA_{Links},$$

where m is the set of test links that use link j to calculate their similarity metrics. Finally we update each LA according to the learning algorithm and the generated reinforcement signal. In other words for each LA_{T_i} in LA_{Tests} set we use d_i as its reinforcement signal such that if d_i is low then the action α_T^2 of the corresponding LA is rewarded proportional to the amount of difference and penalized otherwise. Also for each LA_{L_i} in LA_{Links} we use the average reinforcement signal of the test links that use the corresponding link to calculate their similarity metrics. So the proposed algorithm tries to investigate the sample of links that are more important to generate new information about similarity metrics by using LA_{Links} and reduce the cost of calculating similarity metrics in each iteration by using LA_{Tests}.

4.5 Stopping Phase

In the proposed algorithm sampling, updating and feedback phases is repeated until the average of entropy of probability vector of learning automata reaches a predefined value T_{min} or the maximum number of iteration, K, is reached to a threshold k. The information entropy of a learning automaton with *r* actions can be defined as follows (46):

$$H = -\sum_i^r p_i . \log(p_i) \qquad (2)$$

where p, is the probability of choosing ith action of a learning automaton. The entropy for a learning

automaton has maximum value of one when all the actions have equal probabilities of choosing and has minimum value of zero when the action probability vector is a unit vector. After stopping the SLP, we have a similarity probability distribution functions for each test link as the output of the proposed method. Now because of the similarity metrics are random variable, to determine a link's existence, we should sample from its similarity probability distribution function and based on this sample and a predefined threshold we make the decision about the link existence. This threshold is chosen based on empirical results and experts.

4.6 Changes Phase

This phase of SLP is proposed to adapt the SLP in online stochastic social networks. In the online stochastic social networks the links can be occurred or removed over the time and the future links must be predicted again. In the proposed algorithm after the stopping phase is finished, we have a link prediction result for the structure of the network until time T. Now for adding the online link prediction capability to the SLP, assume that in time T+1 we have two available actions: a link is added to the network or a link is removed from the network. In the following we proposed two scenarios to handle these actions for reconsidering future links:

1) A new link is added to the network: in this case for the new link l we consider a new LA_{Link} learning automata, LA_l, to determine the importance of the link l' sampling in the prediction task. Also we reset every LA_t of the LA_{Test} set in the neighbors of link l with a special depth d to the initial configuration. Then we repeat the learning procedure for the set of LA_l and the LA_t in order to reconsider the link prediction result.
2) A new link is removed from the network: in this case for the removed link l we only reset every LA_t of the LA_{Test} set in the neighbors of link l with a special depth d to initial configuration. Then we repeat the learning procedure for the set of LA_t in order to reconsider the link prediction result.

This online learning procedure reduces the computational complexity of calculating link prediction by using local learning instead of considering total network structure. The pseudo-code of the proposed SLP is shown in Algotrithm 1.

Algorithm 1: Pseudo code of the proposed algorithm (SLP)

Let t be the iteration counter and initially set to **0.**

Let K and T_{min} be the maximum number of iterations and the entropy threshold, respectively.

Let a, b be the reward and punishment rate to update learning automaton in the learning algorithm.

Set LA_{Link} be the set of learning automata, one learning automaton for each link in the network.

Set LA_{Test} be the set of learning automata, one learning automaton for each link that must be predicted.

Set the initial action probability distribution for each learning automaton in LA_{Link} and LA_{Test} using $p^1_{L_i} = p^2_{L_i} = \frac{1}{2}, p^1_{T_i} = p^2_{T_i} = 1/2$.

while $t<K$ **or** $P<T_{min}$ **do**

// Sampling Phase

Select action for each learning automaton LA_{L_i} in the LA_{Link} set based on their action probability distributions.

For each LA_{L_i} that chooses the action α_L^1= "take sample" we sample a new weight for the corresponding link as the weight of current iteration.

For each LA_{L_i} that chooses the action α_L^2= "do not take sample" we use the previous weight of the corresponding link as the weight of current iteration.

// Updating Phase

Select action for each learning automaton LA_T in the LA_{Test} set based on their action probability distributions.

For each LA_{T_i} that chooses the action α_T^1= "update the similarity metric probability distribution" we update the similarity metric probability distribution using the stochastic similarity metrics proposed in section 3 and the generated weights in the previous phase.

For each LA_{T_i} that chooses the action α_T^2= "do not update the similarity metric probability distribution" we use the previous similarity metric probability distribution as the current probability distribution.

// Feedback Phase

For each LA_{T_i} we calculate the skew divergence distance d_i.

For each LA_{T_i} we set $\beta_{T_i} = d_i$ and update the action probability distribution function of LA_{T_i} based on β_{T_i} and equation (1).

For each LA_{L_i} we set $\beta_{L_j} = \frac{\sum_{i \in m} d_i}{|m|}$ and update the action probability distribution function of LA_{L_i} based on β_{L_i} and equation (1).

//**Calculate LAs Information Entropy**

Set P = Calculate average entropy of all LAs in the network using equation (2).

Set t = t + 1

end while

For each test link j

Output the similarity metric probability distribution function as its output.

---

5 Experimental Results

In this section, in order to evaluate the performance of the SLP, several computer experiments are conducted on different synthetic stochastic graphs and the proposed method has been compared in term of performance and accuracy. In the rest of this section, we first give the experiments materials that we used in our experiments and then give a set of four experiments.

5.1 Experiment Materials

This sub-section describe the method and materials that we used in our experiments. To do this first we review the methods we used to produce synthetic graphs and then present the evaluation parameters and metrics that we used them in our experiments.

5.1.1 Random Methods to Generate Synthetic Graphs

This sub-section presents the three methods we used them to generate synthetic graphs:

- Barabasi-Albert model (BA model) as a scale-free network with heavy-tailed degree distribution and parameters $N \in \{2000, 5000, 10000\}$ and $m_0 = m = 5$.
- Watts-Strogatz model (WS model) as a small world network is which reflect a common property of many real networks such as short average path length with parameters $N \in \{2000, 5000, 10000\}$, $k = 4$ and $p = 0.2$.
- Erdds-Renyi model (ER model) with parameters $N \in \{2000, , 5000, 10000\}$ and $p = 0.2$.

Also, we assume each link has a random weight variable with an exponential distribution whose mean is selected randomly from set $p \in \{1, 1.5, 2, 2.5\}$. This setting is adopted from the empirical experimtns (47).

5.1.2 Evaluation Metrics

Here we briefly review four common distance measures that try to estimate the distance of two distributions:

1) Kolmogorov-Smirnov distance statistic (48): for estimate the distance between two cumulative distribution functions (CDFs) and it is in the interval [0 1] and values closer to 0 means more similarity and lower difference. This metric is defined as the follow:

$$\mathrm{KS(P, Q)} = \max_x |P(x) - Q(x)|$$

where P and Q are two CDFs of original and estimated data, respectively, and x represents the range of the random variable.

2) Skew divergence (48): for estimate the distance between two probability distribution functions (PDFs) and defined as follows:

$$\mathrm{SD(P, Q, \alpha)} = D[\alpha \mathrm{P} + (1 - \alpha)\mathrm{Q} || \alpha \mathrm{Q} + (1 - \alpha)\mathrm{P}]$$

where D is the Kiillback-Leibler (KL) divergence, which measures the similarity between two PDFs *P* and *O* that do not have continuous support over the full range of values and *a=0.99*. The KL divergence is defined as follows:

$$\mathrm{KL(P(x)||Q(x))} = \sum_x p(x) \log \frac{P(x)}{Q(x)}$$

3) Pearson's correlation coefficient (49): to measure the similarity between the estimated parameter values and the original parameters values and it is in the interval [0 1] and values closer to 1 mean more similarity. This distance defined as follow:

$$\mathrm{PCC(P, Q)} = \frac{n \sum_i p_i q_i - \sum_i p_i \sum_i q_i}{\sqrt{n \sum_i p_i^2 - (\sum_i p_i)^2} - \sqrt{n \sum_i q_i^2 - (\sum_i q_i)^2}}$$

where p_i, and q_i, are the values of the original parameter *P* and estimated parameter *O* and n is the number of parameters.

4) Normalized L1 distance (50): to measure the distance between two positive *m*-dimensional real vectors *P*

(original distribution) and Q (estimated distribution) and defined as below:

$$L_1(\mathrm{P},\mathrm{Q}) = \frac{1}{n}\sum_i \frac{|p_i - q_i|}{p_i}$$

5.1.3 Experiment Settings

In the proposed method for stopping criteria we used the following parameters: $T_{min} = 0.05$, $k = n \times 50$. Also for the learning automata parameters we used $a = 0.05, b = 0.01$. It also should be noted that these parameters are obtained based on empirical experiments. Also, all the results reported in following experiments are based on the averages taken over 30 runs.

5.2 Experiments

In this section a set of experiments is conducted as the follow: in the first experiment, we evaluate the performance of the proposed method in term of stochastic evaluation metrics. In the second experiment, we report the final probability distribution of some random nodes in an instance run to get an idea about the output of the SLP. In the third experiment we analyze the number of samples that is taken by the proposed LA to reach a certain accuracy and its execution time. Finally in the fourth experiment we test the proposed SLP using three different learning parameters to obtain the best learning parameter.

5.2.1 Experiment 1

In this experiment, we try to compare the proposed method to SSM (sampling stochastic method) in term of stochastic evaluation metrics. The SSM for link prediction based on some chosen similarity metric operates as follow: in each iteration, the stochastic graph is sampled one time and then it calculates the similarity metric for each test link and updates the distributions of the similarity metrics of the test links. Finally, it outputs the distribution of similarity metrics as the output of link prediction. For this experimentation, different synthetic stochastic graphs (BA, WS, and ER) with size from 1000 to 10000 are used and the results of this experimentation for each type of network are averages taken over different used sizes. To do this we consider the different distribution distance metrics: Kolmogorov-Smimov (KS) distance, skew divergence (SD), normalized L distance (ND) and Pearson's correlation coefficient (PCC) for estimated similarity metrics including: Stochastic Common Neighborhood (SCN), Stochastic Salton (SS), Stochastic Jaccard Index (SJ), Stochastic Preferential Attachment (SPA), Stochastic Adamic-Adar Index (SAA), Resource Allocation Index (SRA), Stochastic Katz Index (SKatz), and Stochastic LP Index (SLP) to compare the proposed algorithm with the SSM. The average distance of different similarity metrics is given in figures 2 through 4 for different synthetic stochastic graphs. From the figures 2 through 4, our proposed method (SLP) outperforms SSM for all type of datasets, and for all of the proposed similarity metrics. Based on the learning ability of our algorithm to sample important regions, it is not surprising that the proposed algorithm does better in comparison to the blind SSM algorithm.

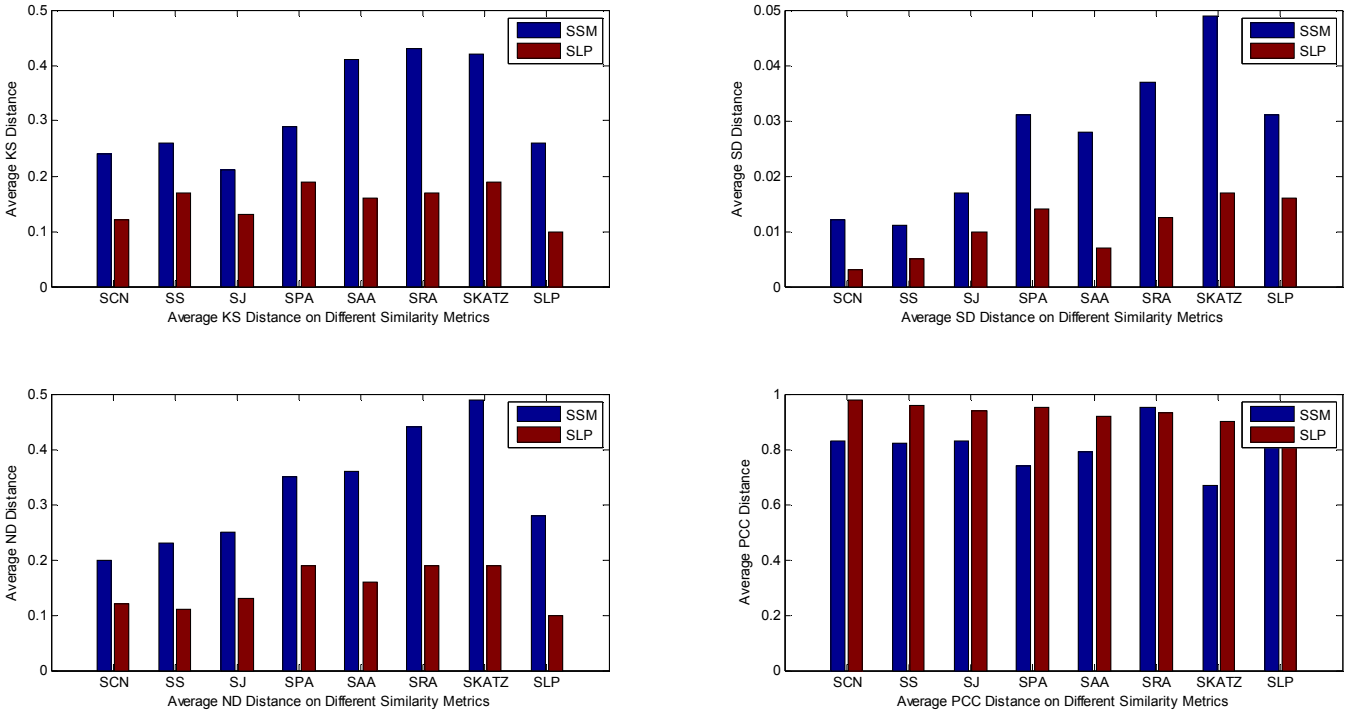

Figure 2. Comparing average distance metrics of different similarity metrics for synthetic BA graph

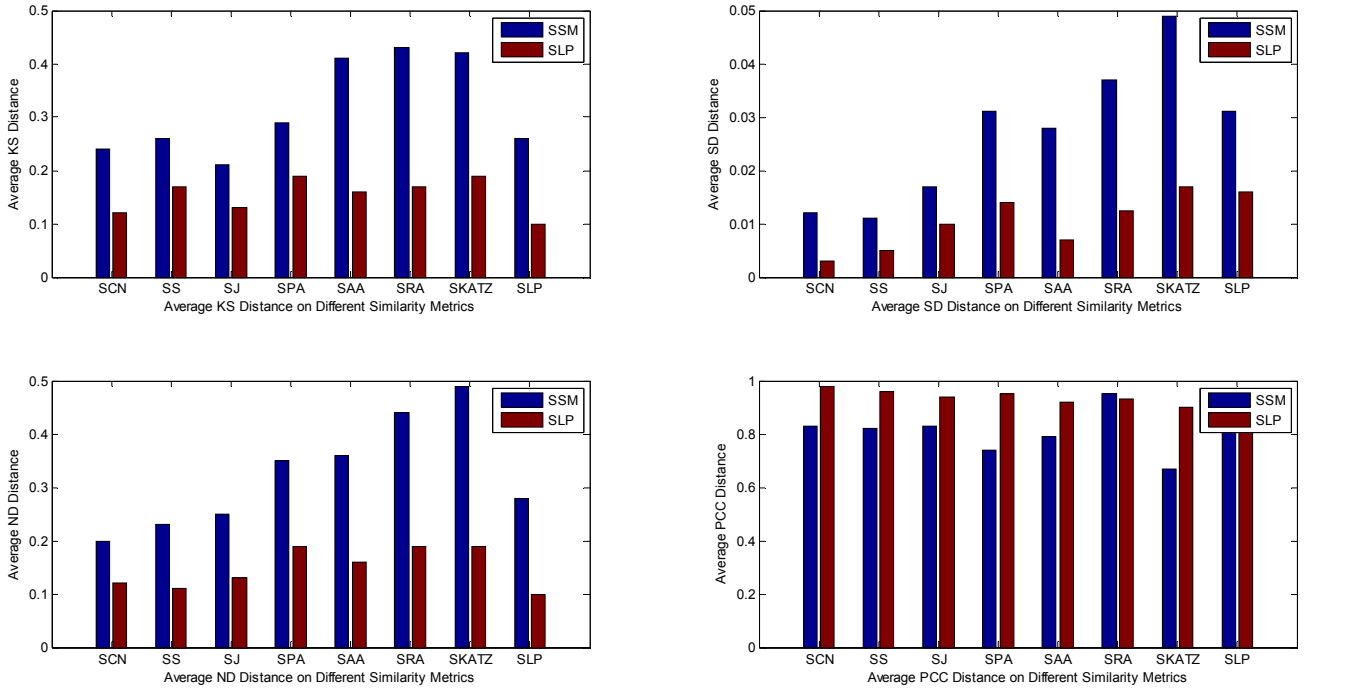

Figure 3. Comparing average distance metrics of different similarity metrics for synthetic WS graphs

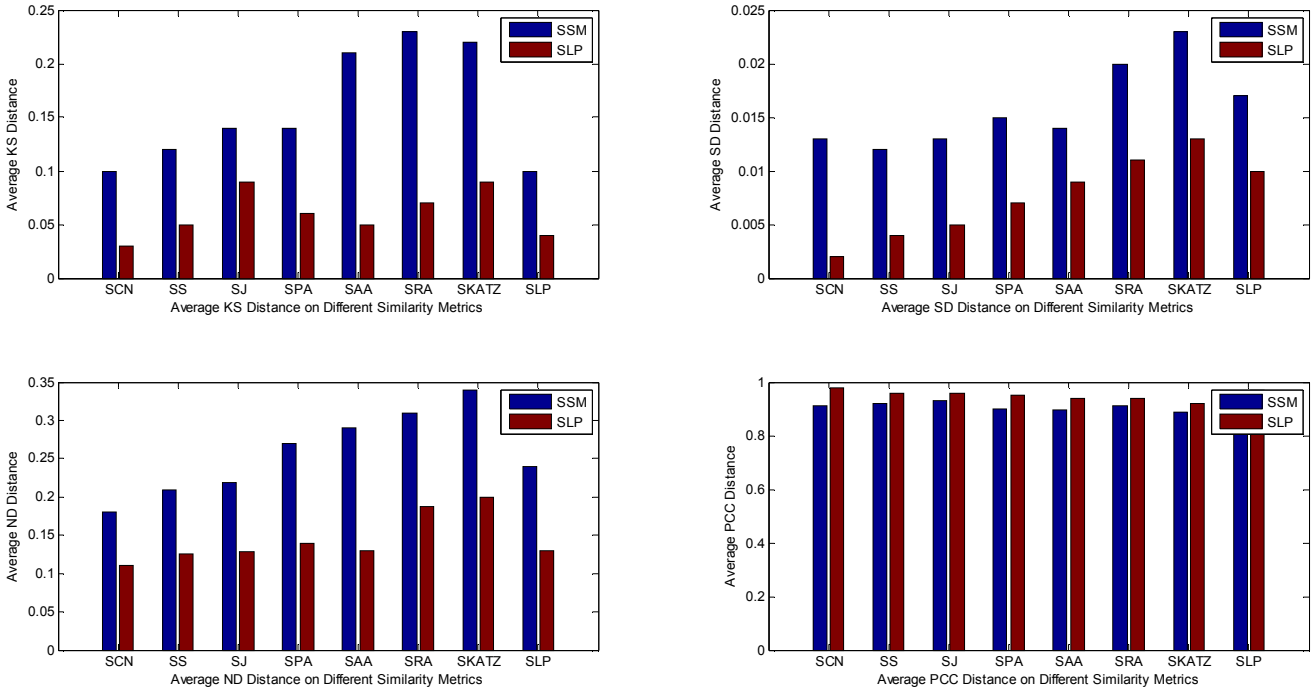

Figure 4. Comparing average distance metrics of different similarity metrics for synthetic ER graphs

5.2.2 Experiment 2

In this experiment, we choose three random test nodes in BA graphs and report the final probability distribution of each similarity metric of the corresponding node in an instant run. These reports are presented in figures 5 through 7. Also we try to present the fitted distribution for each link using non-parametric kernel method. So the figures 8 through 10 show the histogram of each random link with fitted distribution using non-parametric kernel method. This experiment is conducted here to get an idea of the final output of the SLP.

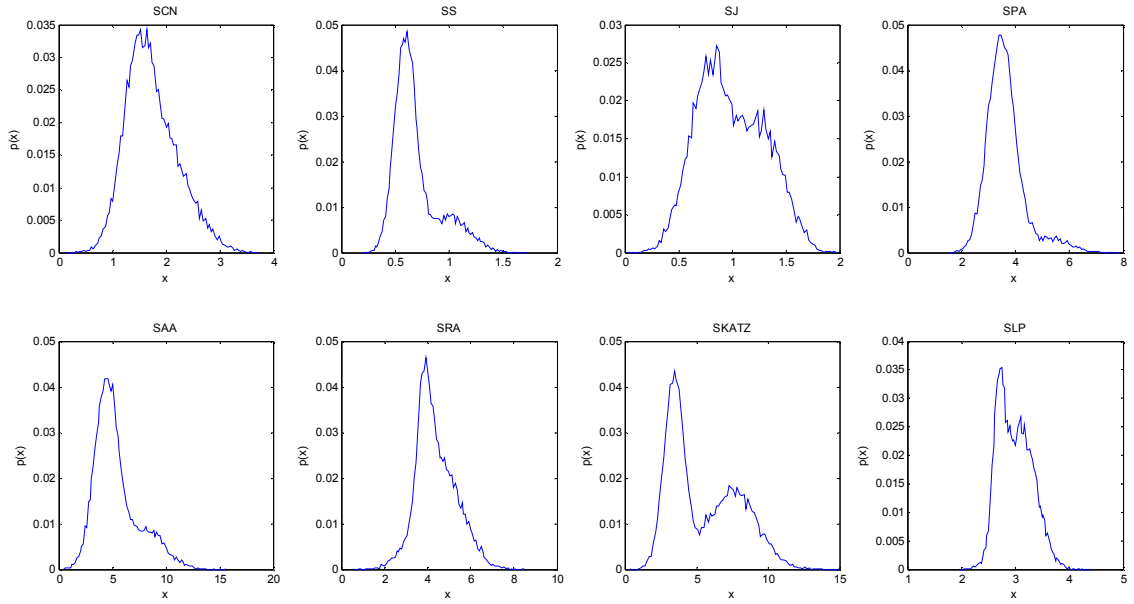

Figure 5. Different similarity metrics probability distributions of the random test node 1 in an instant run

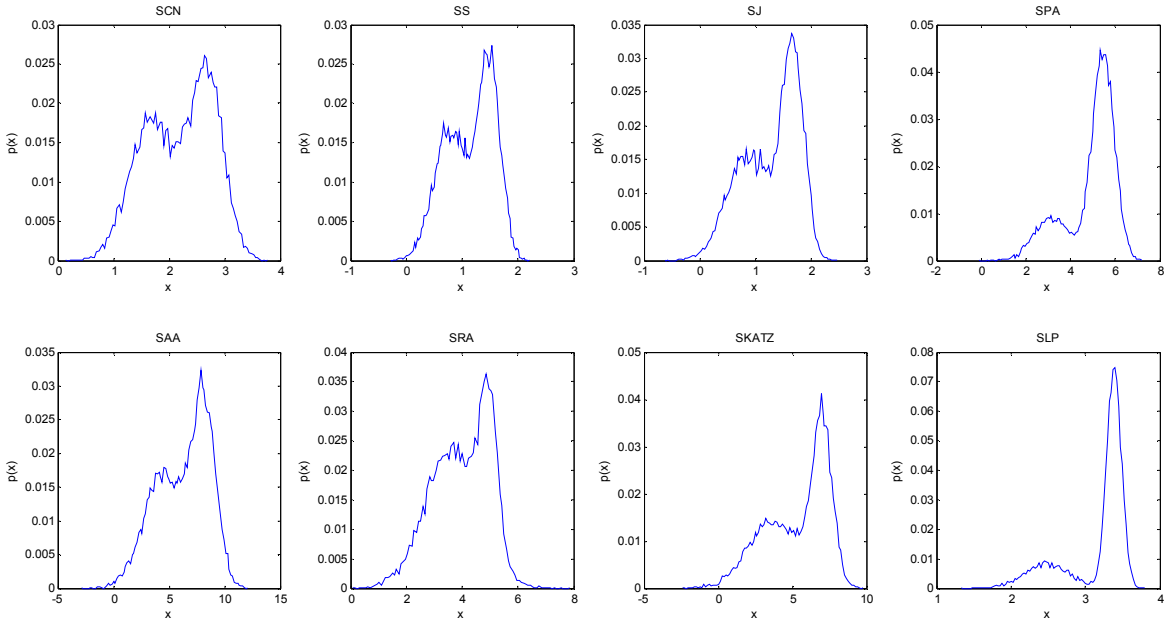

Figure 6. Different similarity metrics probability distributions of the random test node 2 in an instant run

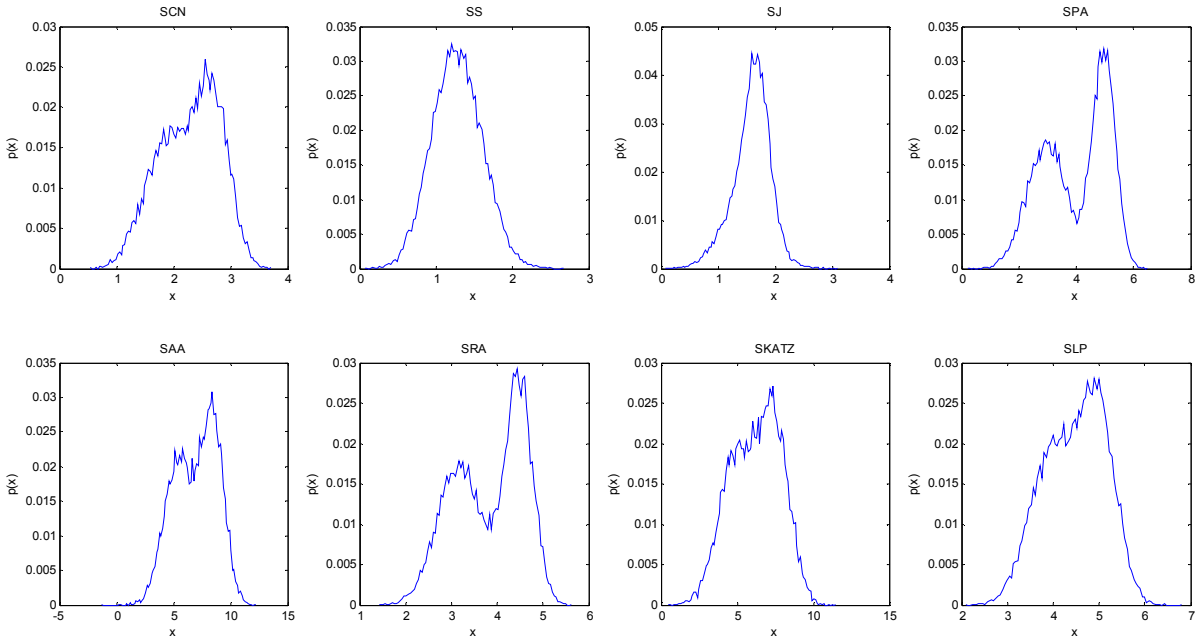

Figure 7. Different similarity metrics probability distributions of the random test node 3 in an instant run

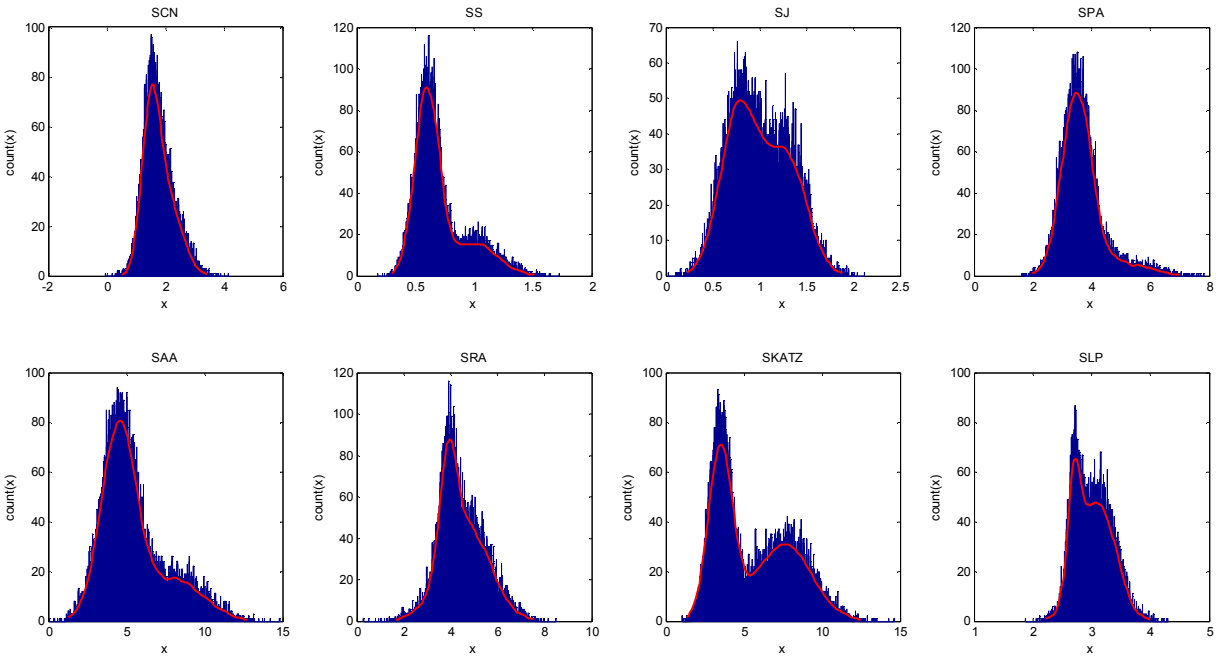

Figure 8. Different similarity metrics histogram with a fitted distribution using kernel method for node 1

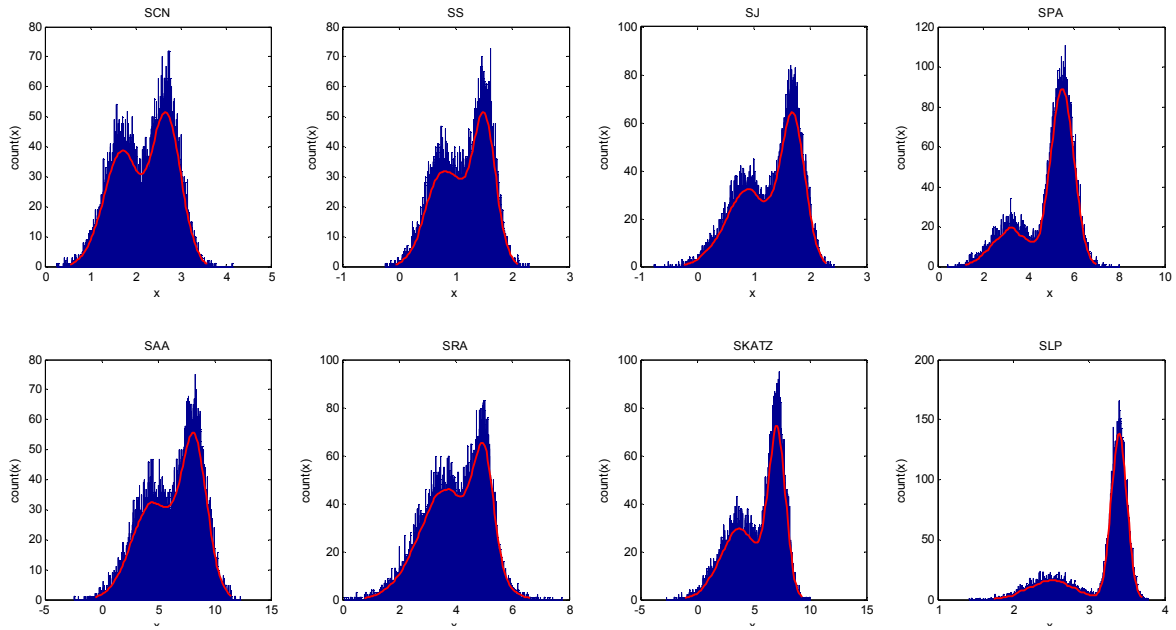

Figure 9. Different similarity metrics histogram with a fitted distribution using kernel method for node 2

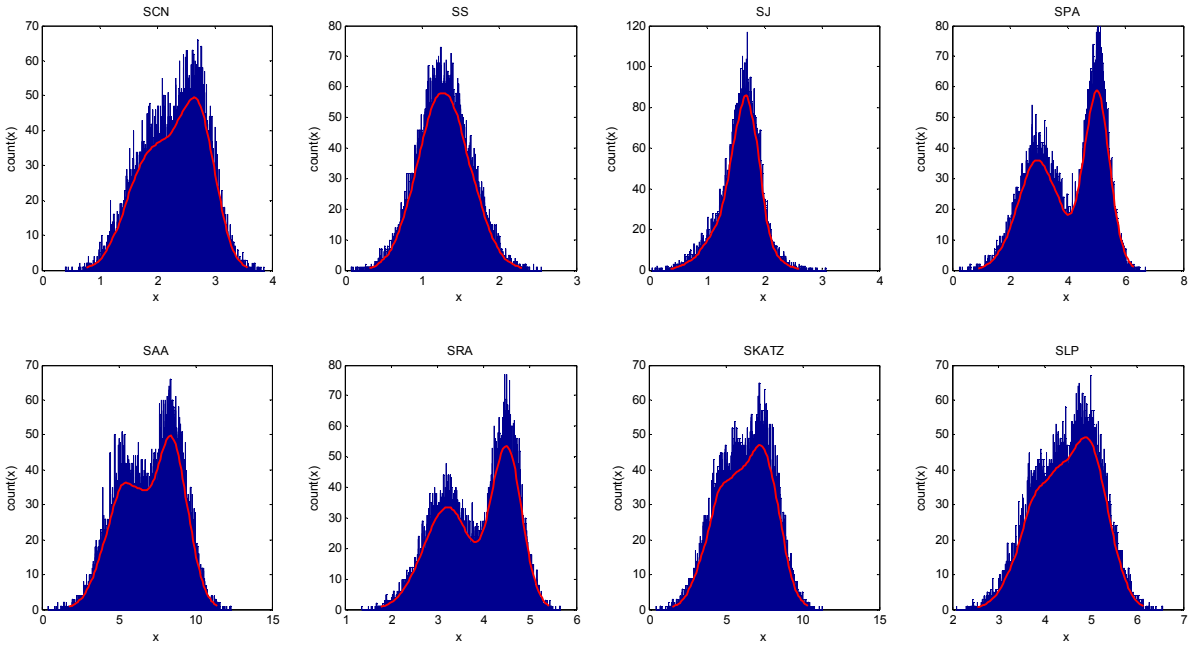

Figure 10. Different similarity metrics histogram with a fitted distribution using kernel method for node 3

5.2.3 Experiment 3

In this experiment we try to compare the number of samples that is taken by the proposed LA with the number of samples taken by SSM to reach a certain accuracy. To do this we consider three accuracy values: {0.65, 0.80,

0.95} and run the SLP and SSM on different synthetic networks BA, WS, and ER with different sizes. We report the average of required samples of the SLP and SSM for each synthetic graphs that is obtained using 30 instant runs (10 instant runs for each size of {2000, 5000, 10000}) in Table 1. From the results it can be found the SLP decreases the rate of required samples by 50% which shows that the proposed SLP is completely better than SSM method. Also to evaluate the performance of the LA_{Tests} set, we compare execution time of the proposed SLP with the SSM and the SLP-Without-LA_{Tests} (SLP that don't have LA_{Tests} set and do not learn anything about if we should calculate and update the similarity metrics probability distributions in each iteration or not) and report the average of execution time over 30 instant runs in Table 2 for different sizes of synthetic graphs and accuracy level 0.95. From the results reported here we see that using LA_{Tests} completely decreases the computational complexity of calculating similarity metric probability distributions.

Table 1. Comparison of the number of samples required by SLP and SSM based on different synthetic graphs and different accuracy levels

Method/Graph	SLP Accuracy = 0.65	SSM Accuracy = 0.65	SLP Accuracy = 0.80	SSM Accuracy = 0.80	SLP Accuracy = 0.95	SSM Accuracy = 0.95
BA-Graph	$\mathbf{7.84 \times 10^4 \pm 9.7 \times 10^2}$	$19.25 \times 10^4 \pm 31 \times 10^2$	$\mathbf{12.87 \times 10^4 \pm 11 \times 10^2}$	$24.12 \times 10^4 \pm 45 \times 10^2$	$\mathbf{16.93 \times 10^4 \pm 15 \times 10^2}$	$35 \times 10^4 \pm 61 \times 10^2$
WS-Graph	$\mathbf{7.12 \times 10^4 \pm 8.8 \times 10^2}$	$21.41 \times 10^4 \pm 45 \times 10^2$	$\mathbf{12.14 \times 10^4 \pm 10 \times 10^2}$	$26.11 \times 10^4 \pm 49 \times 10^2$	$\mathbf{15.55 \times 10^4 \pm 14 \times 10^2}$	$39 \times 10^4 \pm 58 \times 10^2$
ER-Graph	$\mathbf{7.99 \times 10^4 \pm 6.9 \times 10^2}$	$20.89 \times 10^4 \pm 24 \times 10^2$	$\mathbf{13.02 \times 10^4 \pm 9.1 \times 10^2}$	$24.82 \times 10^4 \pm 52 \times 10^2$	$\mathbf{16.12 \times 10^4 \pm 16 \times 10^2}$	$38 \times 10^4 \pm 63 \times 10^2$

Table 2. Average execution time (in minutes) of the SLP, SLP-Without LA_{Tests}, and the SSM based on different synthetic graphs

Method/Graph	SLP	SLP-Without LA_{Tests}	SSM
BA-2000	$\mathbf{10.52 \pm 2.10}$	20.45 ± 3.40	45.10 ± 5.51
BA-5000	$\mathbf{14.13 \pm 2.33}$	26.22 ± 3.23	60.47 ± 6.47
BA-10000	$\mathbf{18.39 \pm 3.45}$	33.29 ± 4.20	91.42 ± 9.30
WS-2000	$\mathbf{11.41 \pm 1.58}$	21.21 ± 3.10	42.17 ± 4.26
WS-5000	$\mathbf{15.46 \pm 2.58}$	27.01 ± 3.40	62.13 ± 6.59
WS-10000	$\mathbf{20.00 \pm 3.51}$	35.18 ± 4.37	93.39 ± 9.51
ER-2000	$\mathbf{10.35 \pm 2.23}$	20.48 ± 3.22	44.24 ± 5.10
ER-5000	$\mathbf{14.40 \pm 3.01}$	27.32 ± 3.58	34.11 ± 7.15
ER-10000	$\mathbf{21.12 \pm 3.40}$	34.50 ± 4.51	95.02 ± 10.03

5.2.4 Experiment 4

In this experiment, we evaluate the convergence behavior of the SLP using information entropy and three different LA configurations. To do this we using the three following configurations and calculate the average information entropy of the probability vector of the LAs: $a = 0.05, b = 0.05$, $a = 0.05, b = 0.01$, $a = 0.01, b = 0.01$ So, Figure 11 plots the average information entropy taken over LAs versus iteration number for three kinds of synthetic stochastic graphs (BA-2000, BA-5000, BA-10000; WS-2000, WS-5000, WS-10000; ER-2000, ER-5000, ER-10000). These figures show that the average entropy of SLP decreases generally where the algorithm proceeds. Also based on these figures we can conclude that the configuration $a = 0.05, b = 0.01$ has the best convergence results and for this reason we use this configuration for the other experiments as mentioned before.

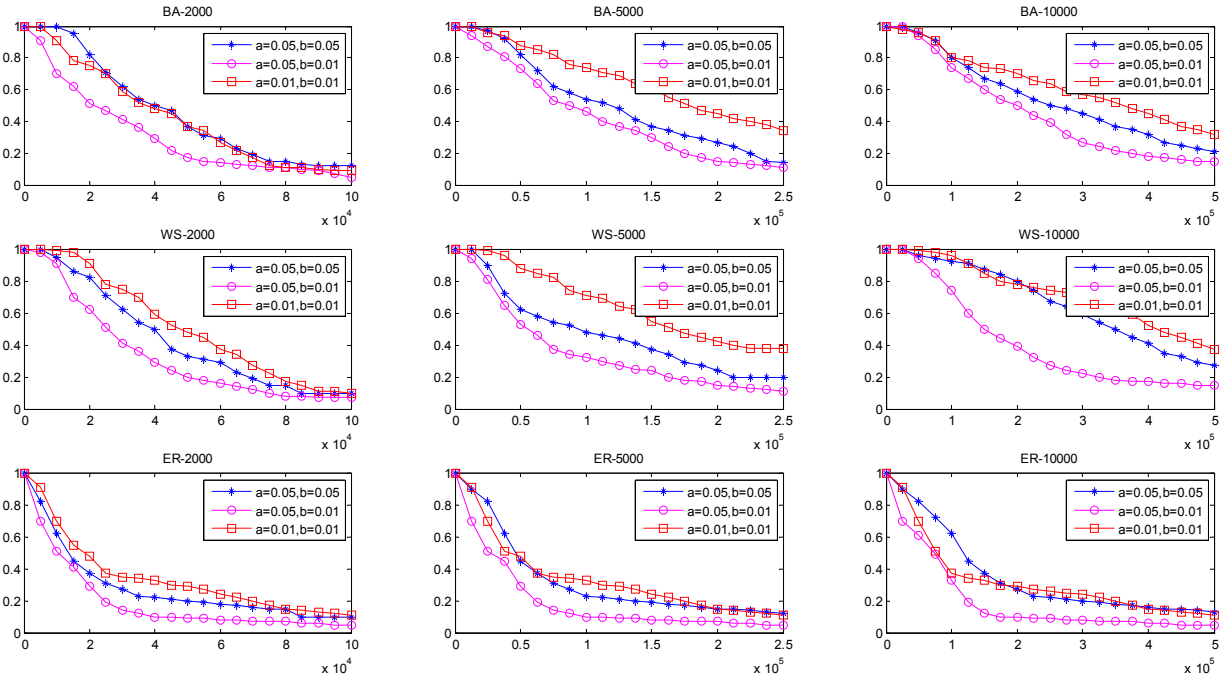

Figure 11. The convergence diagram of the proposed SLP using different LA configurations

6 Conclusion

Because of people activities in social networks are dynamic and uncertainty, and the structure of the networks change over time, deterministic graphs may not be appropriate for modeling and analysis of the social network. The one of solutions is using stochastic social networks as a model to simulate the dynamics and uncertainty of the social networks. This paper tries to present a new similarity based link prediction method for stochastic social networks to overcome the problem of traditional link prediction that only uses a static snapshot of the graph to predict future links. The proposed method takes advantage of using learning automata to estimate the distribution of some chosen similarity metric in such a way that the number of required samples and also the computational complexity be decreased as much as possible. Also, the proposed method has capability to use in online stochastic social networks where the social network changes online and the future links must be predicted again. We consider different synthetic stochastic social networks as the test bench mark and conduct some experiments. The experiments showed that the proposed stochastic link prediction improves the link prediction accuracy and the computational complexity in the stochastic networks.

References

1. Lü, Linyuan, and Tao Zhou. "Link prediction in complex networks: A survey." Physica A: Statistical Mechanics and its Applications 390.6 (2011): 1150-1170.

2. Al Hasan, Mohammad, and Mohammed J. Zaki. "A survey of link prediction in social networks." Social network data analytics. Springer US, 2011. 243-275.

3. Adafre, Sisay Fissaha, and Maarten de Rijke. "Discovering missing links in Wikipedia." Proceedings of the 3rd international workshop on Link discovery. ACM, 2005.

4. Zhu, Jianhan, Jun Hong, and John G. Hughes. "Using Markov models for web site link prediction." Proceedings of the thirteenth ACM conference on Hypertext and hypermedia. ACM, 2002.

5. Li, Xin, and Hsinchun Chen. "Recommendation as link prediction: a graph kernel-based machine learning approach." Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries. ACM, 2009.

6. Huang, Zan, Xin Li, and Hsinchun Chen. "Link prediction approach to collaborative filtering." Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries. ACM, 2005.

7. Malin, Bradley, Edoardo Airoldi, and Kathleen M. Carley. "A network analysis model for disambiguation of names in lists." Computational & Mathematical Organization Theory 11.2 (2005): 119-139.

8. Elmagarmid, Ahmed K., Panagiotis G. Ipeirotis, and Vassilios S. Verykios. "Duplicate record detection: A survey." IEEE Transactions on knowledge and data engineering 19.1 (2007): 1-16.

9. Freschi, Valerio. "A graph-based semi-supervised algorithm for protein function prediction from interaction maps." International Conference on Learning and Intelligent Optimization. Springer Berlin Heidelberg, 2009.

10. Rezvanian, A., & Meybodi, M. R. (2016). Stochastic graph as a model for social networks. Computers in Human Behavior, 64, 621–640.

11. Beigy, H., & Meybodi, M. R. (2006). Utilizing distributed learning automata to solve stochastic shortest path problems. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 14(05), 591–615.

12. Rezvanian, A., & Meybodi, M. R. (2015). Finding minimum vertex covering in stochastic graphs: a learning automata approach. Cybernetics and Systems, 46(8), 698–727.

13. Rezvanian, A., & Meybodi, M. R. (2015). Finding maximum clique in stochastic graphs using distributed learning automata. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 23(01), 1–31.

14. Torkestani, J. A., & Meybodi, M. R. (2012). A learning automata-based heuristic algorithm for solving the minimum spanning tree problem in stochastic graphs. The Journal of Supercomputing, 59(2), 1035–1054.

15. Thathachar, Mandayam AL, and P. Shanti Sastry. "Varieties of learning automata: an overview." IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 32.6 (2002): 711-722.

16. Al Hasan, M., & Zaki, M. J. (2011). A survey of link prediction in social networks. In Social network data analytics (pp. 243–275). Springer.

17. Murata, T., & Moriyasu, S. (2008). Link prediction based on structural properties of online social networks. New Generation Computing, 26(3), 245–257.

18. Al Hasan, M., Chaoji, V., Salem, S., & Zaki, M. (2006). Link prediction using supervised learning. In SDM'06: Workshop on Link Analysis, Counter-terrorism and Security.

19. Gerard Salton and Michael J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, Inc., New York, NY, USA, 1986.

20. Barabási, Albert-László, and Réka Albert. "Emergence of scaling in random networks." science 286.5439 (1999): 509-512.

21. Newman, Mark EJ. "Clustering and preferential attachment in growing networks." Physical Review E 64.2 (2001): 025102.

22. Adamic, Lada A., and Eytan Adar. "Friends and neighbors on the web." Social networks 25.3 (2003): 211-230.

23. Rossetti, G., Guidotti, R., Pennacchioli, D., Pedreschi, D., & Giannotti, F. (2015). Interaction Prediction in Dynamic Networks exploiting Community Discovery. In Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Ana. s and Mining, 2015, 2015, pp.553-558 : s.n.

24. Bliss, C. A., Frank, M. R., Danforth, C. M., & Dodds, P. S. (2014). An evolutionary algorithm approach to link prediction in dynamic social networks. Journal of Computational Science, 5(5), 750–764.

25. Tan, Fei, Yongxiang Xia, and Boyao Zhu. "Link prediction in complex networks: a mutual information perspective." PloS one 9.9 (2014): e107056.

26. Ozcan, A., & Oguducu, S. G. (2015). Multivariate temporal Link Prediction in evolving social networks. In Computer and Information Science (ICIS), 2015 IEEE/ACIS 14th International Conference on (pp. 185–190).

27. S. Huang, Y. Tang, F. Tang, and J. Li, "Link prediction based on time-varied weight in co-authorship network," in Computer Supported Cooperative Work in Design (CSCWD), Proceedings of the 2014 IEEE 18th International Conference on, 2014, pp. 706–709.

28. Moradabadi, B., & Meybodi, M. R. (2016). Link prediction based on temporal similarity metrics using continuous action set learning automata. Physica A: Statistical Mechanics and Its Applications, 460, 361–373.

29. Moradabadi, B., & Meybodi, M. R. (2016). Link Prediction in Fuzzy Social Networks using Distributed Learning Automata. Applied Intelligence (Under Published).

30. Golder, S. A., Wilkinson, D. M., & Huberman, B. A. (2007). Rhythms of social interaction: Messaging within a massive online network. In Communities and technologies 2007 (pp. 41–66). Springer.

31. Leskovec, J., McGlohon, M., Faloutsos, C., Glance, N. S., & Hurst, M. (2007). Patterns of Cascading behavior in large blog graphs. In SDM (Vol. 7, pp. 551–556).

32. Leskovec, J., Backstrom, L., Kumar, R., & Tomkins, A. (2008). Microscopic evolution of social networks. In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 462–470).

33. Nazir, A., Raza, S., & Chuah, C.-N. (2008). Unveiling facebook: a measurement study of social network based applications. In Proceedings of the 8th ACM SIGCOMM conference on Internet measurement (pp. 43–56).

34. Guo, L., Tan, E., Chen, S., Zhang, X., & Zhao, Y. E. (2009). Analyzing patterns of user content generation in online social networks. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 369–378).

35. Kwak, H., Lee, C., Park, H., & Moon, S. (2010). What is Twitter, a social network or a news media? In Proceedings of the 19th international conference on World wide web (pp. 591–600).

36. Gyarmati, L., & Trinh, T. A. (2010). Measuring user behavior in online social networks. IEEE Network, 24(5), 26–31.

37. Galuba, W., Aberer, K., Chakraborty, D., Despotovic, Z., & Kellerer, W. (2010). Outtweeting the twitterers-predicting information cascades in microblogs. WOSN, 10, 3–11.

38. Yan, Q., Wu, L., & Zheng, L. (2013). Social network based microblog user behavior analysis. Physica A: Statistical Mechanics and Its Applications, 392(7), 1712–1723.

39. Liu, H., Nazir, A., Joung, J., & Chuah, C.-N. (2013). Modeling/predicting the evolution trend of osn-based applications. In Proceedings of the 22nd international conference on World Wide Web (pp. 771–780).

40. Vongsingthong, S., Boonkrong, S., Kubek, M., & Unger, H. (2015). On the Distributions of User Behaviors in Complex Online Social Networks. In Recent Advances in Information and Communication Technology 2015 (pp. 237–246). Springer.

41. Bild, D. R., Liu, Y., Dick, R. P., Mao, Z. M., & Wallach, D. S. (2015). Aggregate characterization of user behavior in twitter and analysis of the retweet graph. ACM Transactions on Internet Technology (TOIT), 15(1), 4.

42. Thathachar, Mandayam AL, and Pidaparty S. Sastry. Networks of learning automata: Techniques for online stochastic optimization. Springer, 2003.

43. Rezvanian, A., & Meybodi, M. R. (2010). An adaptive mutation operator for artificial immune network using learning automata in dynamic environments. In Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on (pp. 479–483).

44. Rezvanian, A., & Meybodi, M. R. (2010). LACAIS: learning automata based cooperative artificial immune system for function optimization. In Contemporary Computing (pp. 64–75). Springer.

45. Rezvanian, A., Rahmati, M., & Meybodi, M. R. (2014). Sampling from complex networks using distributed learning automata. Physica A: Statistical Mechanics and Its Applications, 396, 224–234.

46. Mousavian, A., Rezvanian, A., & Meybodi, M. R. (2014). Cellular learning automata based algorithm for solving minimum vertex cover problem. In 2014 22nd Iranian conference on electrical engineering (ICEE) (pp. 996–1000).

47. Bild, D. R., Liu, Y., Dick, R. P., Mao, Z. M., & Wallach, D. S. (2015). Aggregate characterization of user behavior in twitter and analysis of the retweet graph. ACM Transactions on Internet Technology (TOIT), 15(1), 4.

48. Jalali, Z. S., Rezvanian, A., & Meybodi, M. R. (2016). Social network sampling using spanning trees. International Journal of Modern Physics C, 27(05), 1650052.

49. Luo, P., Li, Y., Wu, C., & Zhang, G. (2015). Toward cost-efficient sampling methods. International Journal of Modern Physics C, 26(05), 1550050.

50. Jalali, Z. S., Rezvanian, A., & Meybodi, M. R. (2015). A two-phase sampling algorithm for social networks. In 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI) (pp. 1165–1169).