# Link Prediction in Weighted Social Networks using Learning Automata

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 an important task in Social Network Analysis. The present paper addresses predicting the emergence of future relationships among nodes in a social network. Our study focuses on a strategy of learning automata for link prediction in weighted social networks. In this paper, we try to estimate the weight of each test link directly from the weights information in the network. To do so, we take advantage of using learning automata, intelligent tools that try to learn the optimal action based on reinforcement signals. In the method proposed here, there exist one learning automata for each test link that must be predicted and each learning automata tries to learn the true weight of the corresponding link based on the weight of links in the current network. All learning automata iteratively select their action as the weight of corresponding links. The set of learning automata actions will then be used to calculate the weight of training links and each learning automata will be rewarded or punished according to its influence upon the true weight estimating of the training set. A final prediction is then performed based on the estimated weights. Our preliminary link prediction experiments with co-authorship and email networks have provided satisfactory results when weights are considered.

**Keywords:** Social Network, Link Prediction, Weighted Network, Learning Automata.

## I. Introduction

The advancement of the internet has provided better chances of collaboration and interaction among people and organizations. The advancement has paved the way for the emergence of social networks over the internet which is nowadays very popular. A social network can be formally shown as a graph, where the vertices represent people or organizations, and the connecting edges indicate social connections. Social Network Analysis (SNA) is a vast area of research dealing with techniques and strategies for the study of social networks (1). The analysis and knowledge of networks widely employed to understand the behavior of a community (1) (2). SNA gives us opportunities and benefits in different areas like marketing, economics, health, sociology and safety (2). Link prediction is one of the main tasks undertaken by SNA. The task is concerned with the problem of predicting the prospective existence of relationships among nodes in a network, based on patterns observed in the existing nodes and relations. Link prediction can help us make out the mechanisms that trigger the evolution in a social network and it can be applied to many application areas. For instance, in the area of Internet and web science, it can be used in tasks such as automatic web hyper-link creation (3) as well as web site hyper-link prediction (4). In e-commerce, one of the most prevalent usages of link prediction is to build recommendation systems (5) (6). It also finds various applications in other scientific fields. For example in bibliography and library science, it can be tapped for de-duplication (7) as well as record linkage (8). In bioinformatics, nevertheless, it has been used in protein-protein interaction (PPI) prediction (9). In security-related areas, it can be applied to identify hidden groups of terrorists and criminals (2). The literature shows various strategies and approaches to treating this problem (10) (11). In general, the most widely used techniques are based on one of three approaches, namely: structural measures or patterns in the network; the similarity between nodes (content and/or semantics of the nodes); probabilistic models. These approaches will be briefly explained in section II.

Weighted networks are a kind of social network in which each link has a weight that indicates the strength of the corresponding link (1). Link prediction in such networks is required to adapt the current methods such as adopting the similarity metric based link prediction to consider weights in the network. But in this area, there are some researches that show the strong links are important in link prediction (12). On the other hand, there are studies that show weak links are important in the link prediction (13). So, in this research, we will try to estimate the weight of each test link directly from the links weight information in the network. To do so, we use learning automata, intelligent tools that try to learn the optimal action based on the reinforcement signal.

A learning automata (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 (14). Within each step, an LA selects an action from its action-set. Action selection in the learning automata is based on a probability distribution over the action set. The selected action is applied to the environment, after which a reinforcement signal is produced by the environment. The learning automata update the probability distribution of its actions according

to both reinforcement signal and a learning algorithm. Then it will choose an action once more. These steps are repeated until the automata converge to some action.
In the proposed method there exist one learning automata for each link that must be predicted, and each LA tries to learn the true weight of the corresponding link according to the current network's links weight information. We also partition the network links in two sets: the training set that we use for training LAs, and the test set that must be predicted. In each iteration of the proposed algorithm, each LA chooses a weight as its action. After choosing actions, we will have a weighed network of the test links. Now, we define some metrics to calculate the weight of the training set using these new weights. After calculating the weight of the training set from the weights of the test set, we generate a reinforcement signal for each LA based on its influence on the true weight estimation of the training set, and each LA updates its action probability distribution according to its reinforcement signal. After estimating the weight of test links, we sort them by their weights and predict the existence/absence of each link based on its weight. Our experiments demonstrate that link prediction in the proposed method out-performs other link prediction methods.
Section II briefly discusses the link prediction problem and related works in the weighted network. Section III will then present the used weighted similarity metrics for calculating the weight of links from the weights information in the network. Section IV reviews learning automata briefly. After that, section V leads through the proposed algorithm and procedures for weighted link prediction problem based on learning automata. Section VI brings the experiments and obtained results to a conclusion.

## II. 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 edges that will be added to the network during the interval from time t to a given future time t+1" (1). The most widespread approach to the problem is to explore the topological/structural patterns from the social network of interest (15), (16). Different metrics to describe node pairs have already been adopted in previous works, including for example the number of common neighbors, the path distance between the two nodes, Jaccard's coefficient, and the Adamic-Adar coefficient, among others (17), (1), (16). Such metrics explore structural patterns of the network and commonly provide a degree of proximity/similarity between the nodes. The used metrics can be either local (limited to the direct neighbors of nodes) or global (covering the entire network). As previously mentioned, the starting point in these techniques is to extract the values/scores of different metrics that represent the proximity of pairs of nodes. Then, the obtained data are processed to build a model which can predict the hidden links or links that will appear in the future.
The node-wise similarity based approach searches appropriate measures of similarity between two nodes according to the content and/or semantics they present (17). Each node on the network can be represented as a vector of features. The more similar two nodes are in terms of their particular attributes, the more likely they are to relate. Cosine coefficient, mutual information, and Dice coefficient are examples of techniques used in this approach.
The approaches that are based on probabilistic models aim to learn the best probabilistic model that abstracts the network information. The basic idea here is to create the model through a set of parameters $\theta$, given the observed social network G= (V, E). The existence of the link between the pair of nodes x and y is estimated by the conditional probability P(e(x,y)| $\theta$). This approach examines the elements of the network with the help of relational data models and it enables us to encapsulate relevant information from nodes relationships. Relational Markov Networks and Relational Bayesian Networks are two examples of models dealt with in this approach.
With regard to link prediction considering structural patterns, pairs of non-connected nodes are at first ranked according to a chosen metric (for instance, the number of common neighbors) (15), (16). Then, the top L ranked pairs are assigned as predicted links. To put it another way, it is always assumed that links that have the highest scores are most likely to occur.
We should underline here that earlier works in supervised link prediction consider metrics computed for unweighted social networks. In this type of network, the strength of relationships is not taken into account (only their existence is considered) (17). In the rest of this section, we go on to review related link prediction methods in weighted social networks:
In (18) a supervised machine learning strategy for link prediction in the weighted network is proposed. The method uses link weights that express the "strength" of relationships. Here, the results of supervised prediction on a co-authorship network revealed satisfactory results when weights were taken into account.
Reference (12) indicates that link prediction based on graph proximity measures fits open and dynamic online social networks. It proffers new weighted graph proximity measures for link prediction of social networks. The method relies on an assumption that proximities between nodes would be better estimated by using both graph proximity measures and the weights of existing links in a social network. By taking into consideration the weights of links, link prediction performance is improved via previous proximity measures.
Paper (19) has studied the effect of using weight information when recovering missing edges in a network following the framework of (11). The researchers have observed that the application of a Poisson-based model on a binary network does not hamper the structure modeling. Using Poisson-based models for weighted networks, and for binary versions of the same networks, they observed that weight information did not improve link prediction. They further witnessed that complex and flexible models in general performed better than simpler models regardless of the available information (i.e., a fraction of edges treated as missing). When predicting the weights of the missing edges, the researchers in the said study saw that complex model overfits to the edges, resulting in a poor recovery of the true edge-weight. Also, there are some relevant works about the prediction in the weighted social networks: In (20) the dynamics properties

of mobile calling patterns and some social characteristics are studied based on a large mobile call duration network where the weights of the links are call durations. They found that the stronger ties have lower call duration; the average call duration get shorter when the end point of call have more common neighbors; the opinion leaders have shorter call duration and the social balance tends to shorter call duration. Based on these facts they proposed a probabilistic model to predict the call duration and they compared their methods with some based methods. In (21) an obvious method to infer the tie strength between the users using bipartite event and people network is proposed. They modeled the characterizations of functions that could serve as a measure of tie strength. They showed that for the applications where the ranking of the tie strength is important, the axioms are equivalent to a natural partial order; and presented that to settle on a particular measure, a non-obvious decision about extending this partial order to a total order which is best left to the particular application is needed. They evaluated the method and showed the coverage of the axioms through the use of Kendall's Tau correlation. In (22) an unsupervised model to estimate link strength from the interaction activity and the user similarity is proposed. This method is based on a link-based latent variable model, along with a coordinate ascent optimization procedure for the inference. The authors used Facebook and LinkedIn datasets to evaluate their methods and showed that the proposed method improves the autocorrelation and the classification performance. The result of this method is a set of link strengths and so a weighted social network that can be used in link prediction tasks. In paper (13), they applied three local similarity indices, Common Neighbor, Adamic-Adar index and Resource Allocation index with the consideration of weights. They were surprised to see that the precision of weighted indices turns out even worse than their corresponding unweighted versions. These unexpected observations reminded them of Weak Ties Theory which claims that links with small weights play yet an important role in social networks. The extensive experimental study has shown that weak ties play a significant role in the link prediction problem, and that to emphasize the contribution of weak ties can improve predicting accuracy to a high degree. Finally, in (23) they have introduced a new link prediction method called Low Rank (LR) using robust principal component analysis. In their method, the adjacency matrix of the target network is decomposed into a low-rank matrix which can be regarded as the backbone of network containing the true links and sparse matrix consisting the corrupted or spurious links in the network. Link prediction, actually, can be regarded as matrix completion problem from the corrupted or incomplete adjacency matrix. By solving the optimization problem, they obtained the low-rank matrix which later on plays a role as score matrix illustrating the possible connectivity between each pair of vertices. Then to show that their method can also deal with weighted network, they have compared their method with other weighted-based algorithms and showed the superiority of the proposed method.

## III. Weighted Similarity Metrics

In the present section, we will review the weighted similarity metrics proposed by (13) to estimate the weight of an edge based on the weights information in the network. These methods inherit from popular similarity scores such as Common Neighborhood (CN), Jaccard Index (JC), Preferential Attachment (PA) and Adamic-Adar (AA). To do this, let $\Gamma(x)$ be the set of neighbors of node x in the social network, $|\Gamma(x)|$ be the degree (number of neighbors) of node x and w(x, y) be the link weight between nodes x and y. Also, it should be noticed that we consider undirected graphs and do not consider self-connections; so, w(x, y) = w(y, x). In the following we review the weighted similarity metrics:

1. Common Neighbors (CN)

The CN measure for unweighted networks is defined as the number of nodes with a direct relationship with both evaluated nodes x and y:

$$CN(x,y) = |\Gamma(x) \cap \Gamma(y)| \qquad (1)$$

The CN measure is one of the most widely adopted metrics in link prediction, mainly for its simplicity (2). Also, it is intuitive because it is expected that a high number of common neighbors make easier future contacts between two nodes. Now to estimate weight based on CN measure, the WCN measure is defined as:

$$WCN(x,y) = \frac{\sum_{z\in|\Gamma(x)\cap\Gamma(y)|} \mathrm{w(x,z)} + \mathrm{w(y,z)}}{|\Gamma(x) \cap \Gamma(y)|} \qquad (2)$$

2. Jaccard's Coefficient (JC)

The JC measure is well explored in Data Mining. It assumes higher values for pairs of nodes that share a higher proportion of common neighbors relative to the total number of neighbors they have. For unweighted networks, the JC measure is defined as (24):

$$JC(x,y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \qquad (3)$$

To calculate weight from this similarity metric, the JC coefficient can be extended as:

$$WJC(x,y) = \frac{\sum_{z\in|\Gamma(x)\cap\Gamma(y)|} \mathrm{w(x,z)} + \mathrm{w(y,z)}}{\sum_{z'\in|\Gamma(x)|} \mathrm{w(x,z')} + \sum_{z''\in|\Gamma(y)|} \mathrm{w(y,z'')}} \qquad (4)$$

3. Preferential Attachment (PA)

The PA measure assumes that the probability that a new link is created from a node x is proportional to the node degree $|\Gamma(x)|$ (which means the nodes that currently have a high number of relationships tend to create more links in the future). Barabasi (25) and Newman (26) have proposed that the probability of a future link between a pair of nodes could be expressed by the product of their number of collaborators. For unweighted networks, the PA measure is given by:

$$PA(x,y) = |\Gamma(x)| * |\Gamma(y)| \tag{5}$$

For weighted networks, the PA measure can be extended as:

$$WPA(x,y) = \sum_{a\in\Gamma(x)} \mathrm{w(a,x)} * \sum_{b\in\Gamma(y)} \mathrm{w(b,y)} \tag{6}$$

4. Adamic-Adar Coefficient (AA)

The AA measure for unweighted networks is defined as:

$$AA(x,y) = \sum\nolimits_{z\in\Gamma(x)\,\cap\Gamma(y)} \frac{1}{\log(\Gamma(z))} \tag{7}$$

Adamic and Adar (27) formulated this metric related to Jaccard's coefficient. It defines a higher importance to the common neighbors which have fewer neighbors. Hence, it measures the relationship between a common neighbor and the evaluated pair of nodes. The AA measure is extended for weighted networks as:

$$WAA(x,y) = \sum\nolimits_{z\in\Gamma(x)\,\cap\Gamma(y)} \frac{\mathrm{w(x,z)} + \mathrm{w(y,z)}}{\log(1 + \sum_{c\in|\Gamma(z)|} \mathrm{w(z,c)})} \tag{8}$$

## IV. Learning Automata

Learning automata (LA) are a type of decision-making units that try to learn the optimal action from the set of possible actions by interaction with an unknown stochastic environment (28) (29). In each iteration, the LA selects an action from its action probability distribution and sends it to the random environment (30) (31) (32). The random environment evaluates the selected action and generates a stochastic response to the LA. This stochastic response is called reinforcement signal. Then the LA updates its action probability distribution using the reinforcement signal and a learning algorithm. This tool has many applications in different areas such as optimization tasks (33) (34), capacity assignment problems (35) (36) (37), graph problems (38) (39) (40), artificial intelligence (41)0020 (42) (43), social area (44) (45) (46) and other applications (47) (48) (49) (50).

In learning automata the environment can be described by a triple $\mathrm{E} = \{\alpha, \beta, \mathrm{c}\}$ where $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ denotes the finite set of possible actions for each learning automata, $\beta = \{\beta_1, \beta_2, \dots, \beta_m\}$ represents the set of values that can be taken by the reinforcement signal, and $\mathrm{c} = \{\mathrm{c}_1, \mathrm{c}_2, \dots, \mathrm{c}_r\}$ represents the set of penalty probabilities, where $\mathrm{c_i}$ is associated with the given action $\alpha_\mathrm{i}$. Based on the penalty probabilities of the environment are constant or varied with time, the random environment is called stationary environment and non-stationary environment, respectively.

Also, the random environment based on its possible values could be classified into three classes: P-, Q-, and S-model environments. The reinforcement signal in P-model environments has only two values {0, 1} while in Q-model is bounded in the interval [0,1] and in S-model environments, it is generated from a continuous random variable.

The LA based on its possible actions can be partitioned into two main classes: FALA and CALA (14). In FALA, the action set is finite and the action probability distribution of a FALA with r actions is defined by an r-dimensional probability distribution. In contrast, in CALA the possible actions is a set of real values and CALA uses a probability distribution function to display its actions probability. The relationship between the learning automata and its random environment has been shown in Figure 1. In the following of this section, we introduce the CALA that is used in this paper:

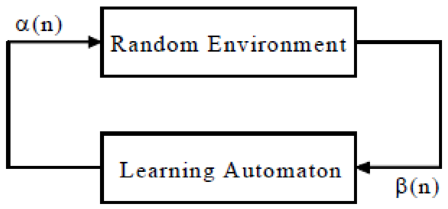


Figure 1. The relationship between a learning automata and its random environment

In (33), a CALA is given, in which the action probability distribution at instant n is a normal distribution with mean $\mu_n$ and standard deviation $\sigma_n$. At each instant, the CALA choose an action according to its probability distribution function and updates its action

probability distribution using the reinforcement signal that is generated from the environment by updating $\mu_n$ and $\sigma_n$. Since CALA has no knowledge of the reinforcement signal $\beta(.)$, the objective of automata is to identify the optimal action, which results in the minimum value of $\beta(.)$. This is achieved by the learning algorithm that updates the action probability distribution using the reinforcement signal that is sent from the random environment. The general learning algorithm of CALA is described as follows:
Consider a function $f: \Re \to \Re$, such that only noisy values of f($\alpha$) are available for measurement of any $\alpha$. Noisy values also correspond to $\beta(\alpha)$ and we have $f(\alpha) = [\beta(\alpha)|\alpha]$. Hence the optimal action of the CALA is an $\alpha$ that minimizes $f(\alpha)$ and $[\beta(\alpha)|\alpha]$. To do this at instance k, the CALA chooses action $\alpha_k$ according to the normal distribution N($\mu_k$, $\sigma_k$). Then the CALA obtains two responses from the environment for two action $\mu_k$and $\alpha_k$. Let these reinforcement signals be $\beta_{\mu_k}$, $\beta_{\alpha_k}$, respectively. Then $\mu_k$ and $\sigma_k$ are updated according to the following equations:

$$\mu_{k+1} = \mu_k + \lambda \frac{\beta_{\alpha_k} - \beta_{\mu_k}}{\phi(\sigma_k)} \frac{\alpha_k - \mu_k}{\phi(\sigma_k)} \tag{11}$$

$$\sigma_{k+1} = \sigma_k + \lambda \frac{\beta_{\alpha_k} - \beta_{\mu_k}}{\phi(\sigma_k)} \left[ \left( \frac{\alpha_k - \mu_k}{\phi(\sigma_k)} \right)^2 - 1 \right] - \lambda K(\sigma_k - \sigma_L) \tag{12}$$

where

$$\phi(\sigma) = \begin{cases} \sigma_L & for\ \sigma \leq \sigma_L \\ \sigma & for\ \sigma > \sigma_L > 0 \end{cases} \tag{13}$$

And λ is the learning parameter for controlling step size $0 < \lambda < 1$, K is a large positive constant and $\sigma_L$ is the lower bound on σ. The iteration continue until $\mu_k$ does not change appreciably and $\sigma_k$ is close to $\sigma_L$.
The idea behind the updating rule given by (1) and (2) is as following: If $\sigma_k$ get 'better' response from the environment, then $\mu_k$ move towards $\alpha_k$. Otherwise, it is moved away from $\alpha_k$. For updating $\sigma_k$, whenever an action $\alpha_k$ is away from $\mu_k$ by more than one standard deviation and it improves the reinforcement signal or when the action selection within one standard deviation from the current mean is worse, we increase the variance; otherwise we decrease it.

## V. The Proposed Weighted Link Prediction

This section describes the proposed algorithm, CALA-WLP, for weighted link prediction problem based on learning automata. Within the proposed method, there is one CALA for each link that must be predicted. Each CALA attempts to learn the true weight of the corresponding link according to the links weight information in the current network. To do so, we partition the network links in two sets: the training set that we use for training CALAs, and the test set which has to be predicted. In each iteration of the proposed algorithm, each CALA chooses a weight as its action. After all the CALAs choose their actions, we will have a weighed network of the test links. We now use these weights to calculate the weights of k percentage links of training set using one of the scores that are introduced in the previous sections. In the proposed algorithm we choose k percentage links of the training set using a random order. After calculating the weights of the training set, we will generate a reinforcement signal to each CALA based on its influence on the true weight estimation of the training set. Here, each CALA updates its action probability distribution according to its reinforcement signal. This procedure is repeated until the action of each CALA converges to some value, which is used as the weight of the test link. Finally, we sort the test links based on their respective weights and predict the existence or lack of each test link based on its weights. The obtained weights of test links are really the influence of each test link in the true reconstruction of the original weighted network. Contrary to other weighted link predictions some of which show that the links with higher weights are important in the final prediction and some show that the links with lower weights are important, the method hereby proposed is independent of whether higher or lower weights in the network are important. That is because it tries to estimate the influence of each test link to reconstruct the original weighted social network. Our preliminary results from link prediction on some co-authorship and email networks proved satisfactory when weights were considered. The experiments demonstrate that the performance of the link prediction in the weighted social network is better than that in a social network without weights. In the rest of this section, we will first describe the main procedure of the proposed algorithm and then provide an example of the proposed method.

### 1. Main Procedure

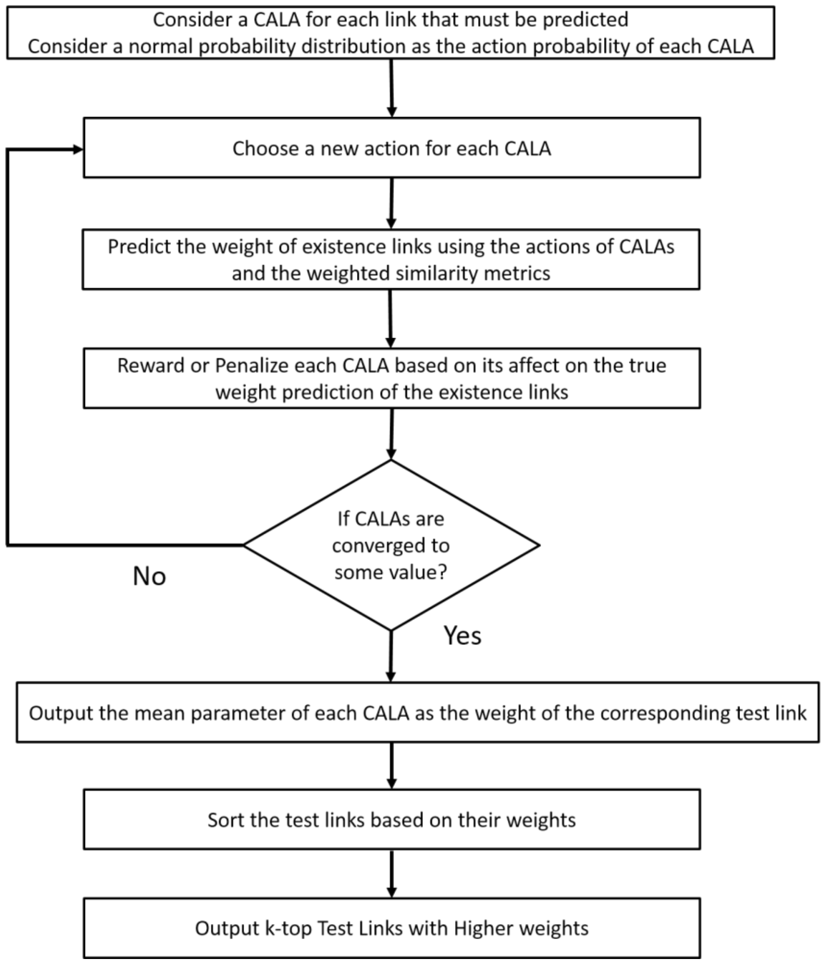
As we said before, for each test link in the proposed algorithm there is one CALA and each $CALA_j$ tries to find the optimal action through a normal distribution $N_j(\mu, \sigma)$. In each iteration k of the proposed algorithm, in the action selection step, each $CALA_j$ chooses its action based on its normal distribution, $N_j(\mu_k, \sigma_k)$. Action $\alpha_j$ is used as the weight of the corresponding link in the network. After actions selection phase, the actions are evaluated and each CALA updates its probability distribution according to some reinforcement signals. Now, in order to generate the reinforcement signal to $\mathrm{CALA_j}$ in iteration k, we calculate two reinforcement signals: one is for the chosen action $\alpha_j(k)$, $\beta_{\alpha_j}(k)$, and the other is for the mean parameter $\mu_j(k)$,$\beta_{\mu_j}(k)$. To do so, we have a weighted network that is created according to the chosen action of CALAs. Also, we have a training set which is a set of real links with some weight value. Now, we use the weight information of test set to calculate the weight of k percentage of real links in training set using one of the scores that are introduced in the previous sections, such as: WCN, WJC, WPA. The percentage of training links that we use them for the weight estimation is considered as a parameter and it is studied in the experiment section. The final reinforcement signals for each $\mathrm{CALA_j}$ are calculated according to the following equations:

$$\beta_{\alpha_j}(k) = \sum_{l=0}^{i} (w'_l(\alpha_j(k), \alpha_{-j}(k))) - w_l \tag{14}$$

$$\beta_{\mu_j}(k) = \sum_{l=0}^{i} (w'_l(\mu_j(k), \alpha_{-j}(k))) - w_l \tag{15}$$

Where i is the number of real links whose weights we calculate, and $w'_l(\alpha_j(k), \alpha_{-j}(k))$ is the estimated weight of link l that is calculated with reference to the chosen action of $\mathrm{CALA_j}$ in addition to the chosen action of other CALAs, $\alpha_{-j}(k)$. Also, $w'_l(\mu_j(k), \alpha_{-j}(k))$ is the estimated weight of link l by using the current mean of $\mathrm{CALA_j}$, $\mu_j(k)$, in addition to the chosen action of other CALAs, $\alpha_{-j}(k)$. Finally $w_j$ is the real weight of link j. Now, each $\mathrm{CALA_j}$ updates its $\mu_j(k)$, $\sigma_j(k)$ based on $\beta_{\alpha_j}(k)$, $\beta_{\mu_j}(k)$ according to equation (11), (12). In other words, in each iteration, each CALA chooses its action. These actions are evaluated based on a training set in terms of reinforcement signal, and the parameters of each CALA are updated by using the reinforcement signal. The procedure is repeated until the value of each CALA converges to some value.

After running the training phase, the test phase will generate the final output prediction of the proposed algorithm. To do that, we sort the test links based on the estimated weights and also use a threshold to classify the test links into two groups: existence links and non-existence links. The diagram of the proposed link prediction method is present in figure 2.



Consider a CALA for each link that must be predicted
Consider a normal probability distribution as the action probability of each CALA
Choose a new action for each CALA
Predict the weight of existence links using the actions of CALAs and the weighted similarity metrics
Reward or Penalize each CALA based on its affect on the true weight prediction of the existence links
If CALAs are converged to some value?
No
Yes
Output the mean parameter of each CALA as the weight of the corresponding test link
Sort the test links based on their weights
Output k-top Test Links with Higher weights

Figure 2. Diagram of the Proposed Link Prediction Method

2. Example

In this sub-section we give an example of the proposed link prediction on a weighted network: Consider figure 3: Sub-figure (a) shows a weighted network with each link has a weight value. Sub-figure (b) shows the possible test links in the network and their CALA parameters. As we mentioned in the previous section, for each test link we have a CALA, with a normal distribution as its parameter. Then each CALA chooses an action based on its normal distribution. The chosen actions of CALAs are shown is sub-figure (c). Now we have a new weighted network which we use to estimate the weight of original links based on some weight scores such as WCN, WJC, or WRA. Sub-figure (d) shows the weights of original links that are calculated based on some similarity metric. Sub-figure (e) also shows that each CALA updates its normal distribution based on its influence on the true weight estimation of the original network. Then each CALA chooses an action again such as sub-figure (c) illustrates, and the procedure is repeated until the action of each CALA converges to some value as shown in (f). Then we sort the test links based on their weights and predict the existence or not the existence of each test link based on its weights as it shown in figure (g).

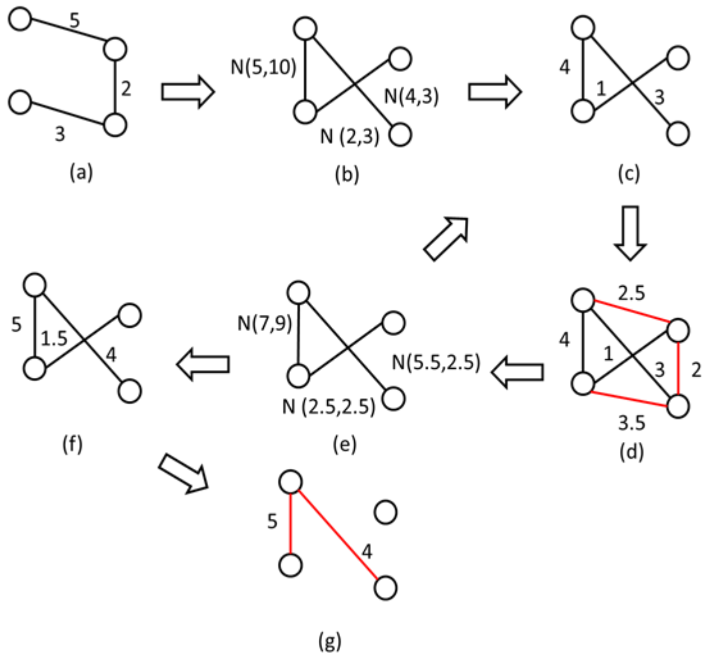


Figure 3. An Example of the Proposed Weighted Link Prediction Method

## VI. Experiment Results

Here, some computer experiments have been conducted to test the proposed algorithm in terms of performance and accuracy. In these experiments, we use the quality of solutions as well as the convergence rate of the proposed algorithm as criteria for performance. In the rest of this section, we will first present the data set we used in our experiments, and then give a set of experiments.

### 1. Data Set

In this sub-section, we describe the social network's data used in our experiments. For the experiments developed in this work, we consider the following two groups of networks:

- Co-authorship Networks: A type of social network in which the nodes represent the authors and two authors are connected if they have collaborated on a paper. Collaboration network is often used to understand the topology and dynamics of complex networks. In this study, we have adopted three co-authorship networks from three sections of Arxiv[1] and extracted data from the years 1993 to 2003

[1]http://www.arxiv.org

for all the data sets. The first network is composed by authors that collaborated in theoretical high-energy physics[2] (hep-th). The second network is formed by authors who published papers in the high-energy physics[3] (hep-ph) and the third is sampled from collaboration in Astro Physics[4] (Astro-ph). In these data sets, if author i co-authored a paper with author j, the graph contains an undirected edge from i to j. If the paper is co-authored by k authors, this generates a completely connected (sub) graph on k nodes.

- Email Communication Networks: A type of social network whose nodes are email addresses and if an address i sends at least one email to address j, the graph contains an undirected edge from i to j. In our experiment, we use two email communication data sets: Enron email communication network[5] and Eu-All email communication network[6]. Enron email communication network includes a data set around half million emails that are public according to the Federal Energy Regulatory Commission and we extract data from May 1999 through May 2002 (36 months). Also, Eu-All email communication network was extracted using email data from a large European research institution and we extracted data from October 2003 to May 2005 (18 months). The network specification of each data set is presented in Table 1. Since these networks are highly sparse, to make computation feasible we have reduced the number of candidate pairs by choosing only the ones that have at least two connections on the network.

Table 1. Network Size in terms of Nodes and Edges

| Data Set | Nodes | Edges | Description |
|---|---|---|---|
| Hep-th | 9,877 | 51,971 | Collaboration network of Arxiv High Energy Physics Theory |
| Hep-ph | 12,008 | 237,010 | Collaboration network of Arxiv High Energy Physics |
| Astro-ph | 18,772 | 396,160 | Collaboration network of Arxiv Astro Physics |
| Email-Enron | 36,692 | 367,662 | Email communication network from Enron |
| Email-EuAll | 265,214 | 420,045 | Email network from a EU research institution |

Now to create a weighted network using these social networks we use the following strategy: Build a weighted version of the network in which each link between a pair of nodes is weighted by the total number of occurred events between the two corresponding nodes.

## 2. Evaluation Metrics

This sub-section introduces the two common evaluation metrics that we use in our experiments:

1. AUC Metric (51): If we rank the entire non-existent links according to their scores, the AUC metric can be interpreted as the probability that a random missing link has a higher score than a random non-existent link. In the algorithmic implementation, at each time we usually pick a missing link and a nonexistent link in a random fashion and compare their scores. If among n independent comparisons there are n′ times when missing links have a higher score and n" times when they have the same score, the AUC value is:

$$\mathrm{AUC} = \frac{n' + 0.5n''}{n} \tag{21}$$

If the AUC has a value greater than 0.5, it is better than the random link prediction algorithm; and the farther from 0.5, the more accurate the algorithm.

2. Precision (51): If we predict L links to be connected and Lr links from L links are right, the Precision is defined as:

$$\mathrm{Precision} = \frac{Lr}{L} \tag{22}$$

[2] http://arxiv.org/archive/hep-th
[3] http://arxiv.org/archive/hep-ph
[4] http://arxiv.org/archive/Astro-ph
[5] http://www.cs.cmu.edu/~enron/
[6] http://snap.stanford.edu/data/email-EuAll.html

Clearly, higher precision means higher prediction accuracy.

3. Experiment 1

This experiment evaluates the proposed CALA-WLP accuracy. To do so, for collaboration networks (Hep-th, Hep-ph and Astro-ph), we consider the data from 1993 to 2002 as the training data (each year as a time period) and year 2003 as test data. Also for email networks (Enron and EuAll), we consider the first 70% available months as the training data (each month as a time period) and the 30% remaining months as the test data. For all conducted experiments, the initial parameters of each CALA-WLP j, $(\mu_0(j), \sigma_0(j))$ are chosen randomly, parameter $\lambda$ is set to 0.005, k is set to 70% and $\sigma_L$ is set to $10^{-3}$. Also, we run the proposed algorithm with different similarity metrics (CN, JI, AA, PA) which were described in previous sections. So, we call the proposed algorithm CALA-WLP-XX if the proposed algorithm uses similarity metric XX in its procedure. In order to improve the comparison of the proposed algorithm, we choose a set of algorithms in three categories:

1) Similarity-based algorithms: this group of algorithms contains a set of common similarity metrics such as CN, Jaccard, PA, AA, Katz and LP (51). In KATZ metric, a similarity is defined as the sum of paths with different lengths such that shorter paths have greater weights. Also, the LP index is a restricted version of Katz metric that only considers paths in lengths 1 and 2. For more i nformation about other similarity metrics please refer to Liben-Nowell and Kleinberg (1).
2) Supervised algorithms: in this group of algorithms, we choose Interaction Prediction (IP) (52), CMA-ES (53), Neighbor Commu nities (NC) (54), and Likelihood based Link Prediction (LLP) (55) algorithms. IP predicts future interactions through combining dynamic social networks analysis, time series forecast, feature selection such as similarity metrics, as well as network communit y structure (52). CMA-ES uses Covariance Matrix Adaptation Evolution Strategy (CMA-ES) to optimize weights which are use d in a linear combination of sixteen neighborhood and node similarity indices (53). The LLP method uses a framework where a network's probability is calculated according to a predefined structural Hamiltonian, and a non-observed link is scored by the co nditional probability of adding the link to the observed network. Finally, the NC method proposes a network-structural similarit y index and a link prediction method based on the neighbor communities using the probabilities of the possible situations in whi ch the two nodes are linked to the same community.
3) Weighted link prediction algorithms: in this group of algorithms two weighted algorithms, Sup-WLP (18) and Weak-WLP (13), and LR (23) are chosen to be compared with the proposed algorithm.

The parameters of the used algorithms are borrowed from their references. Tables 2 and 3 present the average AUC and precision scores based on the 10-fold cross-validation method, respectively. The results reported here (tables 2 and 3) demonstrate that the proposed CALA-WLP-XX is able to achieve an AUC and Precision measures that are significantly better than the XX method which does not consider weights. Also, the results are better than Katz, LP. So, we can conclude that the proposed algorithm outperforms all considered static methods which do not consider weights. Also, in comparison to the IP, CMA-ES, LLP, and NC, we see that the proposed algorithm achieves a better result in AUC and precision measures. Therefore, we also conclude that CALA-WLP is superior to recent unweighted link prediction algorithms and does not do worse than other recent them. Finally, from the result reported here we can see that the proposed algorithm quite better in AUC and precision measures than Sup-WLP and Weak-WLP which use the weight information of the social graph. These results show that CALA-WLP does not depend on whether strong links or weak links are important. Also the proposed method is superior to the LR method because it directly tries to use the weight information as an important information in the networks comparing to the LR method. So, the results of CALA-WLP suggest that the prediction is better with consideration of weight information in the prediction task using our policy.

Table 2. AUC Measures of Proposed CALA-WLP and other Link Prediction Methods

| Method/Data Set | Hep-th | Hep-ph | Astro-ph | Enron | EuAll |
|---|---|---|---|---|---|
| CN | 0.7945 | 0.7025 | 0.6791 | 0.8123 | 0.6643 |
| Salton | 0.7850 | 0.6854 | 0.6441 | 0.8087 | 0.6285 |
| Jaccard | 0.6438 | 0.6026 | 0.5719 | 0.7010 | 0.6259 |
| PA | 0.6400 | 0.6101 | 0.5574 | 0.6743 | 0.6049 |
| AA | 0.7562 | 0.7109 | 0.6840 | 0.8045 | 0.6097 |
| Katz | 0.8487 | 0.8611 | 0.7486 | 0.8896 | 0.7008 |
| LP | 0.8305 | 0.8128 | 0.7115 | 0.8542 | 0.6905 |
| IP | 0.8525 | 0.8548 | 0.7328 | 0.8836 | 0.7376 |
| CMA-ES | 0.8462 | 0.8501 | 0.7241 | 0.8601 | 0.7243 |
| LLP | 0.8759 | 0.8549 | 0.7798 | 0.8851 | 0.7721 |
| NC | 0.8421 | 0.8247 | 0.7415 | 0.8497 | 0.7084 |
| LR | 0.9145 | 08678 | 0.7814 | 0.8927 | 0.7641 |
| Sup-WLP-CN | 0.8502 | 0.8541 | 0.7701 | 0.6317 | 0.6162 |

| | | | | | |
|---|---|---|---|---|---|
| Weak-WLP-CN | 0.6723 | 0.5536 | 0.6412 | 0.8212 | 0.7315 |
| CALA-WLP-CN | 0.9215 | 0.8719 | 0.7847 | 0.9023 | 0.7761 |
| Sup-WLP-JI | 0.8461 | 0.8270 | 0.7503 | 0.6032 | 0.5843 |
| Weak-WLP-JI | 0.6723 | 0.5536 | 0.6412 | 0.8212 | 0.7315 |
| CALA-WLP-JI | 0.9042 | 0.8562 | 0.7602 | 0.8839 | 0.7593 |
| Sup-WLP-PA | 0.8637 | 0.8431 | 0.7699 | 0.6310 | 0.6134 |
| Weak-WLP-PA | 0.6661 | 0.5481 | 0.6400 | 0.8021 | 0.7142 |
| CALA-WLP-PA | **0.9301** | **0.8756** | **0.7900** | **0.9139** | **0.7803** |
| Sup-WLP-AA | 0.8315 | 0.8194 | 0.7465 | 0.6023 | 0.5731 |
| Weak-WLP-AA | 0.6502 | 0.5391 | 0.6286 | 0.8016 | 0.7072 |
| CALA-WLP-AA | 0.8992 | 0.8432 | 0.7493 | 0.8497 | 0.7396 |

Table 3. Precision Measures of Proposed CALA-WLP and other Link Prediction Methods

| Method/Data Set | Hep-th | Hep-ph | Astro-ph | Enron | EuAll |
|---|---|---|---|---|---|
| CN | 0.5421 | 0.4532 | 0.4291 | 0.5620 | 0.4196 |
| Salton | 0.5395 | 0.4260 | 0.4021 | 0.5582 | 0.3758 |
| Jaccard | 0.4856 | 0.3690 | 0.3620 | 0.4529 | 0.3795 |
| PA | 0.4821 | 0.3699 | 0.3052 | 0.4283 | 0.3593 |
| AA | 0.5027 | 0.4549 | 0.4503 | 0.5598 | 0.3500 |
| Katz | 0.6135 | 0.6296 | 0.5049 | 0.6473 | 0.4503 |
| LP | 0.5703 | 0.5831 | 0.4611 | 0.6184 | 0.4319 |
| IP | 0.5853 | 0.6086 | 0.5025 | 0.6204 | 0.4702 |
| CMA-ES | 0.6032 | 0.6140 | 0.5000 | 0.6360 | 0.4821 |
| LLP | 0.6051 | 0.5941 | 0.5081 | 0.6647 | 0.6475 |
| NC | 0.5694 | 0.4972 | 0.4595 | 0.6175 | 0.4589 |
| LR | 0.6071 | 0.6051 | 0.5147 | 0.6718 | 0.6500 |
| Sup-WLP-CN | 0.5732 | 0.5631 | 0.4392 | 0.4293 | 0.5039 |
| Weak-WLP-CN | 0.4250 | 0.4004 | 0.3930 | 0.6103 | 0.6294 |
| CALA-WLP-CN | 0.6246 | 0.6176 | 0.5594 | 0.7049 | 0.6792 |
| Sup-WLP-JI | 0.5995 | 0.5794 | 0.4703 | 0.4013 | 0.4849 |
| Weak-WLP-JI | 0.5013 | 0.4829 | 0.4103 | 0.6305 | 0.6375 |
| CALA-WLP-JI | 0.6042 | 0.5934 | 0.5692 | 0.6893 | 0.6485 |
| Sup-WLP-PA | 0.5737 | 0.3951 | 0.3492 | 0.4509 | 0.5195 |
| Weak-WLP-PA | 0.4258 | 0.3905 | 0.3602 | 0.6059 | 0.6199 |
| CALA-WLP-PA | **0.6399** | **0.6298** | **0.5703** | **0.7194** | **0.6894** |
| Sup-WLP-AA | 0.6016 | 0.5896 | 0.4893 | 0.4054 | 0.5023 |
| Weak-WLP-AA | 0.4193 | 0.3902 | 0.4294 | 0.6025 | 0.6312 |
| CALA-WLP-AA | 0.6185 | 0.6032 | 0.5603 | 0.6394 | 0.6325 |

## 4. Experiment 2

In this experiment, we compare the topological features of the predicted network which is obtained from prediction result with the original social network. The predicted network is a network n random predicted edges, where n is the number of edges in the original network. So it has the same number of edges as the original network. So, the goal of this experiment is to compare the topological features of the original network such as the number of connected components, efficiency, and clustering coefficient with the predicted network. If the values of these features in both the original and predicted network are similar, we can conclude that the proposed algorithm works well. For each dataset, we test this experiment on its largest connected component. Table 4 summarizes the topological features of the original and predicted network for the largest component of the used data sets. In the Table4 $NUMC_X$ is the number of the connected components in network X and the size of the largest one. For example, 1222/2 means that this network has 2 connected components and the largest one contains1222 nodes. In this table, $e_X$ is the efficiency of the network X, $C_X$ is clustering coefficient and $K_X$ is the average degree of the network. From the obtained result we can see that the topological features in the original network and the predicted network are approximately similar which confirms the efficiency of the proposed algorithm. Also, from the results of the previous section and the reported topological features of the original network, we can conclude that the proposed algorithm does better in the networks with higher cluster coefficient and higher average degree.

Table 4. The result of topological features in the original network and predicted network

| Feature/Data Set | Hep-th | Hep-ph | Astro-ph | Enron | EuAll |
|---|---|---|---|---|---|
| $NumC_{Original}$ | 4042/1 | 7031/1 | 9032/1 | 14704/1 | 18046/1 |
| $NumC_{Predicted}$ | 3703/1 | 6260/1 | 7901/1 | 11651/1 | 13893/1 |
| $E_{Original}$ | 0.185 | 0.131 | 0.115 | 0.073 | 0.054 |
| $E_{Predicted}$ | 0.151 | 0.102 | 0.084 | 0.042 | 0.025 |
| $C_{Orginal}$ | 0.764 | 0.683 | 0.581 | 0.236 | 0.145 |
| $C_{Predicted}$ | 0.734 | 0.654 | 0.552 | 0.204 | 0.101 |
| $K_{Orginal}$ | 4.247 | 4.021 | 2.538 | 2.531 | 1.032 |
| $K_{Predicted}$ | 3.521 | 3.184 | 2.128 | 2.035 | 0.912 |

5. Experiment 3

In this experiment, we study the method that we choose k percentage of training links in the learning phase. To choose k links there are three suggested methods:

1. Using k percentage training links using random order
2. Using k percentage training links with the lowest weights
3. Using k percentage training links with the highest weights

In table 5 we compare CALA-WLP AUC measure using three selection methods, where the reported value is the mean of CALA-WLP-XX for all four used similarity metrics. From the results reported here, we can conclude that random selection is the best method where we don't have any special information whether lower or higher weights are important for the prediction task.

Table 5. The comparison of selection methods on the performance of the CALA-WLP

| Method/Data Set | Hep-th | Hep-ph | Astro-ph | Enron | EuAll |
|---|---|---|---|---|---|
| CALA-WLP Random | **0.9201** | **0.8628** | **0.7620** | **0.8934** | **0.7689** |
| CALA-WLP Ascending | 0.7263 | 0.7598 | 0.6351 | 0.7388 | 0.6101 |
| CALA-WLP Descending | 0.7361 | 0.7001 | 0.6624 | 0.6843 | 0.6390 |

6. Experiment 4

This experiment compares the impact of parameter k which determines the percentage of training links that are chosen for CALAs evaluation. To do this, parameter k is varied from the set $\{35\%, 70\%, 100\%\}$ and the performance of the proposed algorithm is calculated for the Hep-th data set. Figure 4 presents the results of this experiment. This figure shows that the best value of parameter k is about70%; because when we choose a small amount of training set, the proposed method underfits data and results in poor accuracy. Also, when we use all training sets of the social network, the proposed method overfits data and now again results in low accuracy. So the best choice for parameter k is about 70%.

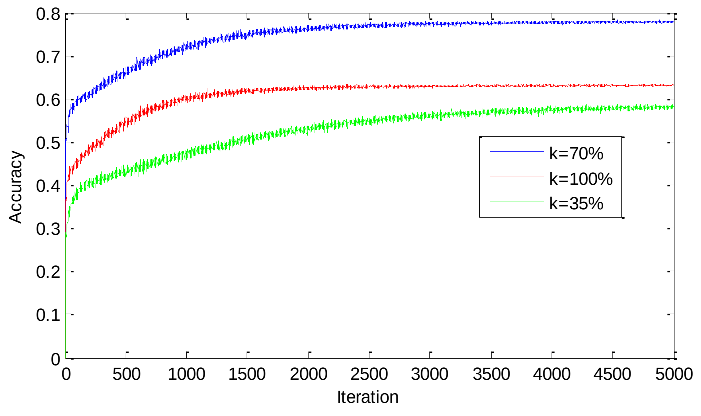


Figure 4. Study of parameter k on the CALA-WLP

7. Experiment 5

This experiment tries to improve the convergence rate of the proposed method using defining CALA state as the following: each CALA has two states, On the state and Off state. In On state, the CALA chooses an action and updates its action probability. The On state describes that the corresponding CALA mean is not good enough and it must choose action again; but in Off state, the CALA does not perform any operation of choosing an action and updating its probability distribution because the mean of CALA is good enough and there is no need to perform the learning phase repeatedly. To transfer this idea to the proposed algorithm, we use the following procedure: in the beginning of the proposed algorithm the state of each CALA is set to On and so all CALA perform learning phase. After generating reinforcement signals for each CALA, we normalize the $\beta_{\mu_j}(k)$ signals for each CALA$_j$. Now if $\beta_{\mu_j}(k)$ for CALA$_j$ is larger than 0.9 and $\beta_{\mu_j}(k) > \beta_{\alpha_j}(k)$, we set the state of CALA$_k$ to Off and choose the mean parameter value of the CALA$_k$ as its action forever. This means that CALA$_j$ has a higher reinforcement signal for its mean in comparison to the chosen action$\alpha_j$ and overall the mean reinforcement signal is good in comparison to all CALAs in the network. So we assume this CALA is converged to its mean.To do this, we compare the performance of proposed algorithm on Hep-th data set with and without suggested procedure. Figure 5. presents the results of this experiment. This figure shows that the final accuracy of the proposed algorithm does not change overall; only if we use the mentioned procedure, we have a faster convergence rate in comparison to not using the mentioned procedure.

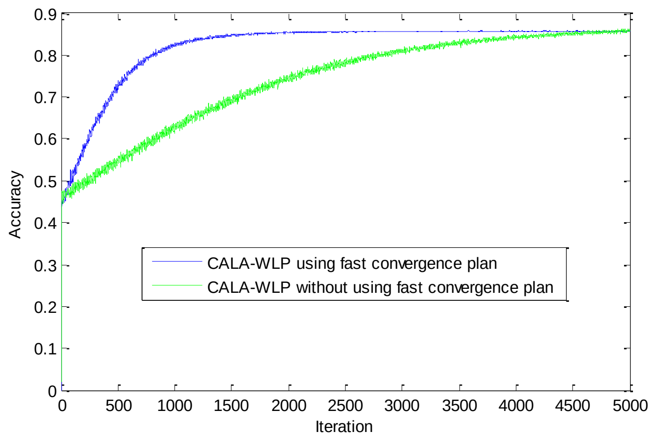


Figure 5. Study of using fast convergence plan

## VII. Conclusion

This paper presents a new weighted link prediction method which uses learning automata to predict the occurrence or non-occurrence of each link using the weight information of the current network. This research uses continuous action set learning automata as a means to learn the optimal weight of each test link. In the proposed method, there is one CALA for each test link that must be predicted and each CALA tries to learn the true weight of the corresponding link based on the weight information in the current network. All learning automata iteratively choose the weight of their corresponding link as their action. The set of learning automata actions is used to calculate the weight of training set. Each learning automata is rewarded or punished according to its influence on the true weight estimating of the training set. The final prediction is performed based on the estimated weights. The experimental results reported here show that the proposed algorithm is superior to other link prediction methods.

## References

1. D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," J. Am. Soc. Inf. Sci. Technol., vol. 58, no. 7, pp. 1019–1031, 2007.

2. M. Al Hasan and M. J. Zaki, "A survey of link prediction in social networks," in Social network data analytics, Springer, 2011, pp. 243–275.

3. S. F. Adafre and M. de Rijke, "Discovering missing links in Wikipedia," in Proceedings of the 3rd international workshop on Link discovery, 2005, pp. 90–97.

4. J. Zhu, J. Hong, and J. G. Hughes, “Using Markov models for web site link prediction,” in Proceedings of the thirteenth ACM conference on Hypertext and hypermedia, 2002, pp. 169–170.

5. X. Li and H. Chen, “Recommendation as link prediction: a graph kernel-based machine learning approach,” in Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries, 2009, pp. 213–216.

6. Z. Huang, X. Li, and H. Chen, “Link prediction approach to collaborative filtering,” in Proceedings of the 5th ACM/IEEE-CS joint conference on Digital libraries, 2005, pp. 141–142.

7. B. Malin, E. Airoldi, and K. M. Carley, “A network analysis model for disambiguation of names in lists,” Comput. Math. Organ. Theory, vol. 11, no. 2, pp. 119–139, 2005.

8. A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios, “Duplicate record detection: A survey,” Knowl. Data Eng. IEEE Trans., vol. 19, no. 1, pp. 1–16, 2007.

9. V. Freschi, “A graph-based semi-supervised algorithm for protein function prediction from interaction maps,” in Learning and Intelligent Optimization, Springer, 2009, pp. 249–258.

10. E. W. Xiang, “A survey on link prediction models for social network data,” Sci. Technol., 2008.

11. C. Wang, V. Satuluri, and S. Parthasarathy, “Local probabilistic models for link prediction,” in Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on, 2007, pp. 322–331.

12. T. Murata and S. Moriyasu, “Link Prediction of Social Networks Based on Weighted Proximity Measures,” IEEE/WIC/ACM Int. Conf. Web Intell., 2007, pp. 85–88.

13. L. Lü and T. Zhou, “Role of weak ties in link prediction of complex networks,” in Proceedings of the 1st ACM international workshop on Complex networks meet information & knowledge management, 2009, pp. 55–58.

14. M. A. L. Thathachar and P. S. Sastry, Networks of learning automata: Techniques for online stochastic optimization. Springer Science & Business Media, 2011.

15. Z. Huang, “Link prediction based on graph topology: The predictive value of generalized clustering coefficient,” Available SSRN 1634014, 2010.

16. T. Murata and S. Moriyasu, “Link prediction based on structural properties of online social networks,” New Generation Computing, vol. 26, no. 3, pp. 245–257, 2008.

17. M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, “Link prediction using supervised learning,” in SDM’06: Workshop on Link Analysis, Counter-terrorism and Security, 2006.

18. H. R. De Sá and R. B. C. Prudêncio, “Supervised link prediction in weighted networks,” in Neural Networks (IJCNN), The 2011 International Joint Conference on, 2011, pp. 2281–2288.

19. D. K. Wind and M. Morup, “Link prediction in weighted networks,” 2012 IEEE Int. Work. Mach. Learn. Signal Process., 2012, pp. 1–6.

20. Dong, Y., Tang, J., Lou, T., Wu, B., & Chawla, N. V. (2013). How long will she call me? distribution, social theory and duration prediction. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 16–31).

21. Gupte, M., & Eliassi-Rad, T. (2012). Measuring tie strength in implicit social networks. In Proceedings of the 4th Annual ACM Web Science Conference (pp. 109–118).

22. Xiang, R., Neville, J., & Rogati, M. (2010). Modeling relationship strength in online social networks. In Proceedings of the 19th international conference on World wide web (pp. 981–990).

23. Pech, R., Hao, D., Pan, L., Cheng, H., & Zhou, T. (2017). Link prediction via matrix completion. EPL (Europhysics Letters), 117(3), 38002.

24. P. Jaccard, "Etude comparative de la distribution florale dans une portion des Alpes et du Jura," Impr. Corbaz, 1901.

25. A.L. Barabási and R. Albert, "Emergence of scaling in random networks," Science Journal, vol. 286, no. 5439, pp. 509–512, 1999.

26. M. E. J. Newman, "Clustering and preferential attachment in growing networks," Phys. Rev. E, vol. 64, no. 2, p. 25102, 2001.

27. L. A. Adamic and E. Adar, "Friends and neighbors on the web," Soc. Networks, vol. 25, no. 3, pp. 211–230, 2003.

28. M. Thathachar and P. S. Sastry, "Varieties of learning automata: an overview," Syst. Man, Cybern. Part B Cybern. IEEE Trans., vol. 32, no. 6, pp. 711–722, 2002.

29. K. S. Narendra and M. A. L. Thathachar, Learning automata: an introduction. Courier Corporation, 2012.

30. M. L Thathachar and B. R. Harita, "Learning automata with changing number of actions," Syst. Man Cybern. IEEE Trans., vol. 17, no. 6, pp. 1095–1100, 1987.

31. M. A. L. Thathachar and P. S. Sastry, "A hierarchical system of learning automata that can learn die globally optimal path," Inf. Sci. (Ny)., vol. 42, no. 2, pp. 143–166, 1987.

32. M. Thathachar and V. V Phansalkar, "Convergence of teams and hierarchies of learning automata in connectionist systems," Syst. Man Cybern. IEEE Trans., vol. 25, no. 11, pp. 1459–1469, 1995.

33. G. Santharam, P. S. Sastry, and M. A. L. Thathachar, "Continuous action set learning automata for stochastic optimization," J. Franklin Inst., vol. 331, no. 5, pp. 607–628, 1994.

34. H. Beigy and M. R. Meybodi, "A new continuous action-set learning automata for function optimization," J. Franklin Inst., vol. 343, no. 1, pp. 27–47, 2006.

35. B. J. Oommen and T. D. Roberts, "Continuous learning automata solutions to the capacity assignment problem," Comput. IEEE Trans., vol. 49, no. 6, pp. 608–620, 2000.

36. M. Jahanshahi, M. Dehghan, and M. R. Meybodi, "On channel assignment and multicast routing in multi--channel multi--radio wireless mesh networks," Int. J. Ad Hoc Ubiquitous Comput., vol. 12, no. 4, pp. 225–244, 2013.

37. M. Jahanshahi, M. Dehghan, and M. R. Meybodi, "A mathematical formulation for joint channel assignment and multicast routing in multi-channel multi-radio wireless mesh networks," J. Netw. Comput. Appl., vol. 34, no. 6, pp. 1869–1882, 2011.

38. J. A. Torkestani and M. R. Meybodi, "Finding minimum weight connected dominating set in stochastic graph based on learning automata," Inf. Sci. (Ny)., vol. 200, pp. 57–77, 2012.

39. J. A. Torkestani and M. R. Meybodi, "A learning automata-based heuristic algorithm for solving the minimum spanning tree problem in stochastic graphs," J. Supercomput., vol. 59, no. 2, pp. 1035–1054, 2012.

40. J. A. Torkestani and M. R. Meybodi, "Learning automata-based algorithms for finding minimum weakly connected dominating set in stochastic graphs," Int. J. Uncertainty, Fuzziness Knowledge-Based Syst., vol. 18, no. 06, pp. 721–758, 2010.

41. J. K. Kordestani, A. Ahmadi, and M. R. Meybodi, "An improved Differential Evolution algorithm using learning automata and population topologies," Appl. Intell., vol. 41, no. 4, pp. 1150–1169, 2014.

42. D. Yazdani, B. Nasiri, A. Sepas-Moghaddam, M. Meybodi, and M. Akbarzadeh-Totonchi, "mNAFSA: a novel approach for optimization in dynamic environments with global changes," Swarm Evol. Comput., vol. 18, pp. 38–53, 2014.

43. A. Alizadegan, M. R. Meybodi, and B. Asady, "A Novel Hybrid Artificial Bee Colony Algorithm and Differential Evolution for Unconstrained Optimization Problems," Adv. Comput. Sci. Eng., vol. 8, no. 1, pp. 45–56, 2012.

44. M. Soleimani-pouri, A. Rezvanian, and M. R. Meybodi, "An ant based particle swarm optimization algorithm for maximum clique problem in social networks," in State of the art applications of social network analysis, Springer, 2014, pp. 295–304.

45. M. Soleimani-Pouri, A. Rezvanian, and M. R. Meybodi, "Finding a maximum clique using ant colony optimization and particle swarm optimization in social networks," in Proceedings of the 2012 international conference on advances in social networks analys.

46. A. Rezvanian, M. Rahmati, and M. R. Meybodi, "Sampling from complex networks using distributed learning automata," Phys. A Stat. Mech. its Appl., vol. 396, pp. 224–234, 2014.

47. M. S. Obaidat, G. I. Papadimitriou, A. S. Pomportsis, and H. S. Laskaridis, "Learning automata-based bus arbitration for shared-medium ATM switches," Syst. Man, Cybern. Part B Cybern. IEEE Trans., vol. 32, no. 6, pp. 815–820, 2002.

48. H. Mostafaei, M. Esnaashari, and M. R. Meybodi, "A coverage monitoring algorithm based on learning automata for wireless sensor networks," arXiv Prepr. arXiv1409.1515, 2014.

49. S. M. Safavi, M. R. Meybodi, and M. Esnaashari, "Learning automata based face-aware Mobicast," Wirel. Pers. Commun., vol. 77, no. 3, pp. 1923–1933, 2014.

50. A. Safari Mamaghani, K. Asghari, and M. R. Meybodi, "Designing a New Structure Based on Learning Automata to Improve Evolutionary Algorithms (With Considering Some Case Study Problems)," J. Adv. Comput. Res., vol. 4, no. 3, pp. 1–24, 2013.

51. L. Lü and T. Zhou, "Link prediction in complex networks: A survey," Phys. A Stat. Mech. its Appl., vol. 390, no. 6, pp. 1150–1170, 2011.

52. G. Rossetti, R. Guidotti, D. Pennacchioli, "Interaction Prediction in Dynamic Networks exploiting Community Discovery," in Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis, 2015, pp.553-558.

53. C. A. Bliss, M. R. Frank, C. M. Danforth, and P. S. Dodds, "An evolutionary algorithm approach to link prediction in dynamic social networks," J. Comput. Sci., vol. 5, no. 5, pp. 750–764, 2014.

54. Xie, Z., Dong, E., Li, J., Kong, D., & Wu, N. (2014). Potential links by neighbor communities. Physica A: Statistical Mechanics and Its Applications, 406, 244–252.

55. Pan, L., Zhou, T., Lü, L., & Hu, C.-K. (2016). Predicting missing links and identifying spurious links via likelihood analysis. Scientific Reports, 6.