The Journal of Supercomputing
https://doi.org/10.1007/s11227-022-04615-z

A new multi-wave continuous action-set cellular learning automata for link prediction problem in weighted multi-layer social networks

Mozhdeh Khaksar Manshad[1] · **Mohammad Reza Meybodi**[2] · **Afshin Salajegheh**[3]

Accepted: 18 May 2022

Abstract

One of the main research areas in social network analysis (SNA) is link prediction (LP). Social networks can be shown as a graph, and LP algorithms predict future links using the previous structure of the social network. A significant part of previous studies focuses on the LP problem with a homogeneous structure. On the contrary, real social networks include some heterogeneous interactions. Also, in the LP problem, the learning process must move on to the next stage with some particular order at the same time as convergence speed increases with no loss of accuracy. Therefore, there is a need for an algorithm that is compatible with heterogeneous social networks. This paper presents a new Continuous Action-Set Cellular Learning Automaton (CACLA), and we call it Multi-Wave CACLA (MWCACLA). MWCACLA has a connected structure, and there is a module of CALAs in each cell where a cell's neighbors are its successors. We show that the model converges upon a stable and compatible configuration. Then we propose an MWCACLA model for considering the LP problem in a weighted multi-layer social network, and the proposed model is called MWCALA-WLP. The MWCALA-WLP merges the gained local information of different CALAs residents in a module related to a test link for estimating that the link will be appeared in the future or not. For the LP problem in the weighted multi-layer social networks, it has been demonstrated that MWCACLA produces much better results than other approaches and has an acceptable convergence rate.

Keywords Link prediction · Social network analysis · Cellular learning automata · Wave propagation

✉ Mozhdeh Khaksar Manshad
Khaksar.mojdeh@yahoo.com

Extended author information available on the last page of the article

Published online: 09 June 2022

1 Introduction

Organizations and people can interact with each other much easier by using the advancement of cyberspace. This advancement provides the facility for creating social networks, which are the most common and popular tools for communication nowadays. Formally, a social network can be shown as a graph, where the nodes illustrate the entities, and the links show the interactions between entities. LP is, by far, one of the most significant tasks in SNA. The LP task uses the existing patterns in the current network structure to predict future communications in the network [1, 2]. Most studies on LP have examined homogeneous social networks, which include only one type of nodes and connections. While in the real world, a challenge is that social networks have a heterogeneous structure, with several different types of nodes connected by several different connections. The link prediction (LP) problem in heterogeneous social networks has been addressed in references [3, 4]. A common way to predict links in heterogeneous networks in the works presented so far is to use heterogeneous links and nodes properties, such as node similarity properties. In these methods, all types of links are considered equally. The main drawbacks of these methods are that the weight of the links is ignored. A type of social network where a number is assigned as a weight to each link and the number indicates the strength of that link is called a weighted social network [5]. In this paper, we try to calculate the score of test links using the weights of various types of real links in the network. To achieve this, we propose a new model of CLA, which is called multi-wave continuous action-set cellular learning automata (MWCACLA).

A Cellular Automaton (CA) [6] is a model made of many simple cells, and they pass through a collection of states based on local interaction. The collection of a cell's neighborhood makes up its local environment, based on which the cell's local rule is defined. There is a limited number of states that each cell can select. A cell's configuration refers to its state as well as those of its neighboring cells. The local rule of CAs shows their evolution through time and indicates CAs' configuration changes in one stage. CAs are suitable for modeling systems made up of simple, connected building blocks with a local connection [7].

Learning automata (LAs) [8] are simple agents that, throughout their learning process, attempt to pick out the optimum action from a limited team of actions according to the environment's response to previous actions. They can be of use in unknown random environments. The capabilities of LAs are known when several LAs interact locally. This local interaction can be shown in various forms, such as trees, meshes, or arrays, based on its application. The LA is divided into two categories based on its possible actions [9]: continuous action-set learning automaton (CALA) and finite action set learning automaton (FALA). In CALA, the actions are a set of real numbers, while in FALA, the actions are selected from a finite set.

Combining CAs with LAs, Reference [10] introduces a new model named cellular learning automata (CLAs). CLAs are distributed computation models that combine learning automata (LAs)' learning capabilities with cellular automata

(CAs)' computation power. This model of automaton outdoes CA for its ability to learn optimal actions. It also surpasses single LA for in it, a set of LAs interacts with each other. CLAs come with many applications in various areas [11], such as computer networks [12], social networks [13–16], evolutionary computation [17], as well as optimization [18, 19] and image processing [20].

Lately, a class of asynchronous CLAs called Wavefront CLAs (WCLAs) [21, 22] has been introduced. WCLAs are different from CLAs in two main ways. One is that they enjoy the connected neighborhood; that is, the neighbors of an LA are defined as successor LAs in the network. The second is that they have the capacity to propagate information. Also, in [21], their steady-state behavior through time has been analyzed and assessed with a minimum spanning tree as well as graph coloring.

In many real-world situations, such as LP in social networks, routing, classification, and channel assignment, the learning process has to be propagated in parallel in several stages with a particular order to increase convergence speed while preserving accuracy. The simple idea for such situations is to use a kind of WCLA in which a multiplicity of waves run parallel to each other, instead of only one wave being active. Therefore, in this paper, we introduce a new algorithm called multi-wave continuous action-set cellular learning automaton (MWCACLA). In this model, instead of one LA, we use a module of CALAs in each cell of a CLA and the neighborhoods of these cells are determined by their successors. Each of the n CALAs in a cell can create a wave and dispatch it to neighbor cells if their selected action differs from their previous actions. These CALAs create n waves running in parallel. The method we introduce in this paper paves the way for choosing more than one real-value action at a time, helping boost convergence speed without losing accuracy. The main idea here is that since in problems, the environment is stochastic, decision-making based on different responses from the environment will produce fewer errors than decisions based on one response. In this article, we will first introduce MWCACLA and then analyze its convergence.

In the second part of this article, we have used MWCACLA for predicting links in the future based on a multi-relation (heterogeneous) weighted social network. We can consider a multi-relation graph as a multi-layer graph in which each layer consist of the same type of nodes and different type of links. First, we partition our data set into two groups: the training data set and the test data set. In the proposed model, a module of CALAs is assigned to each test link. Each CALA in a module can generate a wave in the social network, which tries to learn the weight of the corresponding link according to the weight of the one type of link in the specific layer of social networks. Therefore, each of the waves generated by each of the CALA in a module tries to calculate the weight of the relevant test link based on the weight of one layer of the network's connections. The learning experiment of different CALAs in a module using the local information is merged to estimate the weight of a test link. Since in this algorithm, each of the CALAs in a module generates a wave on a specific layer, so each automaton in a module predicts the weight of the test link based on a specific layer in a parallel manner. For each layer, we do the following procedure in a parallel way. At each step of the introduced algorithm, each of the CALA in the module for each test link selects an action whose selected action represents the score of the corresponding test link. If the selected action is different

from the previous action, it activates the neighbors until a predefined depth. After that, we add the test links with their weights to the training graph on the corresponding layer and make a new graph. Based on the new graph, the weight of the training set in each layer is calculated by using some similarity criteria. Then, a reinforcement signal will generate based on the estimated weight of the new graph for each layer. The generated reinforcement signals for layers are merged and used to update the common normal distribution of a module. After some steps, the CALAs related to a test link converge to action because they use a common normal distribution. After convergence, we sort the weight of test links, and links with higher scores will appear in our result.

The significant contributions of this paper are summarized as follows:

1. A new CLA method is proposed, which can propagate multiple waves into the environment from a cell. The proposed method can be used in any problems in a continuous environment that the learning process must propagate in multiple ways and successive order.
2. The convergence analysis of the MWCALA is considered.
3. The proposed method is used as a framework for improving LP problem accuracy (MWCALA-WLP) in weighted multi-layer social networks.
4. The results of experiments for the MWCALA-WLP method are compared with other LA and CLA-based algorithms for LP.

The paper's upcoming sections have been sorted as follows: Sect. 2 provides a brief overview of the LP problem and the studies that have been proposed so far. In Sect. 3, a short description is given about the LAs and CLAs. Section 4 gives full explanations about the MWCALA and its convergence analysis. A new model based on MWCALA for weighted multi-layer LP problem (MWCALA-WLP) in the multi-layer social network has been introduced in Sect. 5. In Sect. 6, an example of the MWCALA-WLP is explained. In Sect. 7, the experimental study of MWCALA-WLP and their comparison with other methods have been described. In Sect. 8, the research's main findings have been briefly discussed.

2 LP methods

A social network is shown as a graph in which each vertex demonstrates an entity, and a connection shows a relation between two entities. A heterogeneous social network is illustrated formally as $G = \left(V_1, V_2, \ldots, V_N, E_1, E_2, \ldots, E_K\right)$ with different types of vertices and connections. In this social network $V_i (i = 1, 2, \ldots, N)$ shows the set of vertices with the equal type i, and $E_j (j = 1, 2, \ldots, K)$ shows the set of links with the equal type j. As we consider weighted social networks, a value is assigned to each link as the weight of the relevant link. The weight of each link shows the number of co-occurrences between two vertices, for instance, in a bibliographic network; it can be the number of papers co-authored between author pairs. In this study for experiments, we use a network with a single type of vertex ($N = 1$) and different

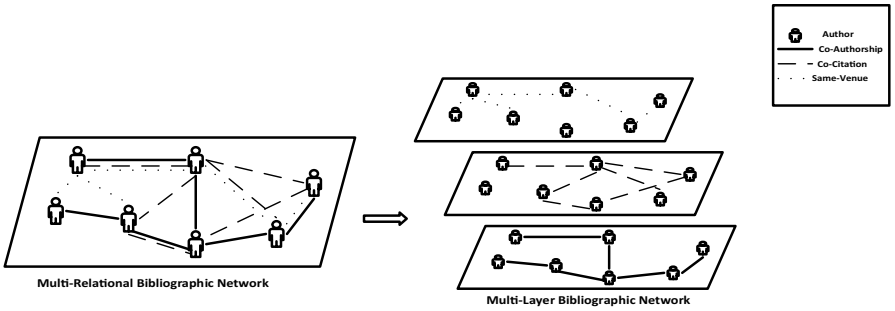


Fig. 1 A multi-relational bibliographic network as a multi-layer network

types of links. In other words, we consider a multi-relational network, and as it has shown in Fig. 1, we represent a multi-relational network as a multi-layer network. Given the graph in time T, our ultimate goal in the LP problem is to predict relations of different types ($j = 1, 2, \ldots, k$) in time $T+1$. In Fig. 1, we represent a heterogeneous graph for a bibliographic network. It consists of author vertices (one type vertex) and co-authorship, co-citation, and same venue links (three types of relations). There is a solid link between two vertices if two authors write at least an article together, there is a dashed link if two authors cite a common article, and a dotted line illustrates same-venue relation between two vertices. Also, we consider a multi-relational bibliographic network as a multi-layer network, so each type of relation between authors has been located in a specific layer. In the proposed method in this paper, we consider the weight of different types of links in different layers to predict future relations.

There are numerous LP methods for homogeneous and heterogeneous social networks that have been introduced in previous studies. Most LP methods for homogeneous social networks in the past are based more on the structural and topological features of the network. In this type of method, pairs of unconnected nodes are ranked according to a local or global topological similarity criterion. So, unconnected ranked nodes with the highest scores will be connected in the near future. A number of similarity criteria have been introduced in previous studies, such as the common neighborhood (CN), the Jaccard criterion (JC), the Adamic-Adar metric (AA) as the local similarity criteria, and the Katz metric as the global similarity criterion. The basic idea of local similarity criteria is that the probability of establishing a link between two nodes in the future depends on the mechanism of organization of their level one and two neighboring nodes. In Reference [23], the authors introduced weighted models for local similarity criteria. They have shown that using link weight has an effect on LP problems and strong links have a greater impact on LP. By contrast, Lu and Zhou in [24] have illustrated that weak links are much more important in LP. The intuition behind the use of global similarity criteria is that two vertices are more likely to be connected in the future if there are shorter paths between them. Local similarity criteria use only neighboring node information compared to global similarity criteria, while global methods use paths between nodes. Global similarity criteria are much more accurate than local criteria, but they

are not suitable for large-scale networks because these criteria are time-consuming. In Reference [25], the authors examine the similarities between the two nodes based on semantics and content and introduce a node-based similarity approach. In this method, some techniques such as mutual information, Dice coefficient, and Cosine coefficient are used. There is a feature vector for each node in the network, and the more similar the feature vector of the two nodes is, the more likely it is that the two nodes will be connected in the future.

Some LP methods based on LAs [5, 26–29] and ICLAs [30, 31] have been introduced. In [28], a new LP method based on CALA and temporal similarity metrics (CALA-LP) has been proposed. The LP problem has been modeled as an optimization problem, and then a series of CALAs has been used as an optimization tool for solving the problem. Authors define a weight for each similarity criteria and time periods to determine the importance of them. And each CALA in the proposed method tries to learn the optimal value for the relevant weight. A new time-series LP (TSLP) method based on LAs has been proposed in [27]. The score of each test link has been estimated by using an LA, which is assigned to the corresponding test link. Each LA tries to estimate the optimal score for the relevant link by using local similarity metrics in each time period. Also, in [26], a new method (FLP-DLA) based on fuzzy social networks and distributed LAs (DLAs) has been introduced. FLP-DLA estimates the fuzzy strength of each link based on the event time, then, by using the DLA and fuzzy strength of links, calculates the score of the test links. Another LA-based algorithm for weighted social networks (CALA-WLP) has been proposed in [5]. CALA-WLP uses the weight of connections in the network for estimating the score of test links. Likewise, authors in [30] combine irregular CLA and evolutionary computing and propose ICLA-EC. ICLA-EC as a TSLP algorithm uses local information of each node in different time periods to predict future connections. Another ICLA-based algorithm for the LP problem has been proposed in [31] (ICLA-LP). ICLA-LP considers local triangular communities in consecutive times for weighing the real links in the social networks and then by using the weighted social network estimates the score of test links.

It should be noted that mostly supervised methods have used unweighted homogeneous social networks to predict links. In the following sections, we will study previous supervised methods in weighted and heterogeneous networks.

In [23], it has been shown that LP based on graph proximity criteria is suitable for dynamic online social networks. So, the authors have introduced a new method for LP based on link weight and graph proximity criteria. Their experiments on the data sets of co-authors have shown greater efficiency than previous methods based on proximity criteria. In [32], an unsupervised method has been developed for estimating the strength of connections from social activities and entity similarity. Facebook and LinkedIn datasets have been used to evaluate the proposed method, and the achieved results have shown that the estimated weights for the corresponding networks improved the accuracy. In Reference [33], the authors used several weighted and unweighted local similarity criteria to predict links, but what was surprising was that the unweighted criteria performed better, this reminiscent of the Weak Tie Theory. Further studies in this article have shown that weak ties have a significant impact on better predicting links in the future. A new LP method based

on robust principal component analysis has been proposed in [34], and it has used the low rank (LR) and sparsity feature of the matrix. The results of different experiments show that the proposed method in [34] has a good performance for dense matrix compared to traditional methods.

All the methods mentioned so far in this paper are related to LP in homogeneous social networks. Most real social networks, however, have a heterogeneous structure. A new method based on tensor factorization for LP in heterogeneous social networks has been proposed in [35]. A new probabilistic LP method for a sparse heterogeneous network has been proposed in [36], and it is called Multi-Relational Influence Propagation (MRIP). MRIP employs temporal features for predicting links in the future. In [37], a Multi-Task Learning Metric (MTLM) model has been proposed. MTLM model uses the features of the network and learns a distance measure for each type of link. An unsupervised Multi-Relational LP (MRLP) model has been introduced in [38], and it is an extension of the Adamic-Adar measure for weighted social networks. In [39], authors have proposed new meta path-based topological features for heterogeneous networks. In this model first, path-based properties have been extracted, and then a regression-based model learns weight for each link in the networks. A random walk-based method has been proposed in [40] for the bibliographic network. This method has been compared with some new supervised models, and the achieved results illustrate that the introduced method has better accuracy in comparison with them. In [41], a Multi-Relational ProbFlow (MRPF) has been proposed, which is a combination of two methods to LP, Katz [42] and ProbFlow [40] measures.

As some similarity criteria are used in this paper, we have briefly studied the weighted and unweighted similarity criteria. As it has been described in [43], In the proposed metrics, the collection of neighbors of vertex x and the degree of vertex x in the social network is illustrated by $\Gamma(x), |\Gamma(x)|$ respectively. The weight of link *(x, y)* is shown by *w(x, y)*, and it should be noted that in our proposed method, an undirected social network is used, so $w(x, y) = w(y, x)$.

Common Neighborhood (CN)[44]: in this index, two nodes are more likely to be related in the future if they have many common neighbors compared to other neighboring nodes.

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)| \tag{1}$$

We can define the CN measure for the weighted social networks as the Eq. (2):

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

Jaccard Index (JC) [45]: In this case, the probability that a neighbor of v_i or v_j is a neighbor of both has been determined.

$$\mathrm{Jaccard}(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \tag{3}$$

For weighted social networks, JC measure is illustrated as:

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

Preferential Attachment (PA)[46]: if a pair of nodes have a higher degree, the corresponding connection will give a higher score.

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

The following Equation can calculate PA measure for weighted social networks:

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

Adamic-Adar Index (AA)[47]: it is counting common neighbors between two nodes by using the following Equation:

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

Also, for weighted social networks, it can be calculated as follows:

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

3 Background

In this Section, we briefly go over LAs and CLAs.

3.1 LAs

LAs [8] provide a technique for computational intelligence for solving complicated, uncertain problems or where plenty of information within the environment is lacking. LAs improve their performance by learning optimal actions from a limited number of actions in interaction with a random environment. As it has been shown in Fig. 2, the chosen action is applied as input to the environment according to LA's action probability distribution vector. Then the environment

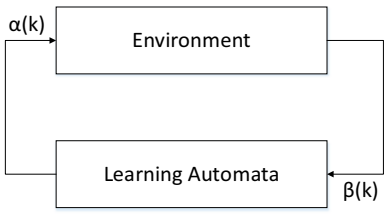


Fig. 2 Schematic of the relationship between an LA and its environment

sends the automaton a reinforcement signal as an input action response. The reinforcement signal can be used for updating the automata actions' probability distribution vector. As this process continues, the LA learns to pick out the best action from its team of actions, thus receiving the desired response from the environment. The LA can be divided into two main categories: continuous action-set learning automata (CALAs), and finite action-set learning automata (FALAs), which are explained in the following.

For FALA, there is a finite set of actions $\overline{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$, and r is the set of actions. Hence penalty function, f, and action probability functions, ζ are discrete probability distributions shown by $\overline{S} = \{s_1, s_2, \ldots, s_r\}$ and $\overline{p}(n) = \left[p_1(n), p_2(n), \ldots, p_r(n)\right]^T$, respectively. $\overline{S}$ and $\overline{p}$ called penalty strengths and action probability vector, respectively. As we have shown in Fig. 2 at the time k, the action chosen based on p's action probability vector by LA such as $\alpha(k)$ is given as input to the environment, and random response $\beta(k)$, i.e., reinforcement signal, is received from the environment. Then, the $p(k)$, actions' probability vector is updated by a learning algorithm.

By contrast, in CALA, automata choose their actions from the real line $\mathfrak{R}$. In these automata, the probability distribution for selecting an action is represented by a probability distribution function. The CALA updates this probability distribution function. In CALA, ζ at instant $n \geq 0$ is a normal distribution with mean μ_n and standard deviation σ_n [48]. This automaton interacts with the environment using two actions α_n and μ_n. Then it gets reinforcement signals from the environment for two applied actions. Let this reinforcement signals be β_{μ_n} and β_{α_n}. Finally, automaton updates it's normal distribution function using:

$$\mu_{n+1} = \mu_n + aF_1\left[\mu_n, \sigma_n, \alpha_n, \beta_{\alpha_n}, \beta_{\mu_n}\right] \tag{9}$$

$$\sigma_{n+1} = \sigma_n + aF_2\left[\mu_n, \sigma_n, \alpha_n, \beta_{\alpha_n}, \beta_{\mu_n}\right] - Ca[\sigma_n - \sigma_L] \tag{10}$$

where F_1, F_2, and $\emptyset$ are defined as below:

$$F_1\left(\mu_n, \sigma_n, \alpha_n, \beta_{\alpha_n}, \beta_{\mu_n}\right) = \left[\frac{\beta_{\alpha_n} - \beta_{\mu_n}}{\emptyset(\sigma_n)}\right]\left[\frac{\alpha_n - \mu_n}{\emptyset(\sigma_n)}\right] \tag{11}$$

$$F_2\left(\mu_n, \sigma_n, \alpha_n, \beta_{\alpha_n}, \beta_{\mu_n}\right) = \left[\frac{\beta_{\alpha_n} - \beta_{\mu_n}}{\emptyset(\sigma_n)}\right]\left[\left(\frac{\alpha_n - \mu_n}{\emptyset(\sigma_n)}\right)^2 - 1\right] \tag{12}$$

$$\emptyset(\sigma) = \left(\sigma - \sigma_L\right) I\left\{\sigma > \sigma_L\right\} + \sigma_L \tag{13}$$

The parameters of the CALA are σ_L which is the lower bound on σ , $K > 0$ is the large positive constant, $0 < \lambda < 1$ is the learning parameter [48].

3.2 CLAs

A CLA [11] is a CA which has been assigned an LA. The configuration of cells is defined as the state of all cells. A CLA works through its local rule. Based on the local rule and the cell's configuration, a reinforcement signal is produced, which penalizes or awards the LA in the cell. The operations of a CLA can be described according to the following stages. In the first stage, each LA in a cell picks out one of its actions through the automata probability distribution vector, thus defining the state of the cell. Then, a reinforcement signal is obtained using the CLA's local rule and configuration, which penalizes or awards the LA in each cell, therefore updating the LA's probability distribution vector according to a learning algorithm. This process is repeated until the desired result is achieved. Officially, a CLA is defined as:

Definition 1 A d-dimensional CLA is shown with the quintet $(z^d, N, \overline{\emptyset}, \overline{A}, F)$ so that: z^d is a lattice, d-tuple is an integer number, $N = \{\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_m\}$ is a limited subset of z^d which is called a neighborhood vector so that $\overline{x}_m \in z^d$، $\overline{\emptyset}$ shows a limited team of states. The state of the cell c_i is shown as $\overline{\emptyset}_i$. $\overline{A}$ demonstrates a set of LAs, each of which has been allocated to a cell of CLA. $F_i : \overline{\emptyset}_i \rightarrow \overline{\beta}$ shows the local rule of CLA in cell c_i, so that $\overline{\beta}$ indicates the set of probable values for the reinforcement signal.

CLAs can be categorized in different categories, as the following:

Closed vs. open CLAs [49]: In closed CLAs, the LA's action selection in each cell for the next stage is conducted according to its local environment. In open CLAs, by contrast, each LA's action selection for the next step depends not only on its local environment but on its exclusive and global environments as well.

Regular vs. irregular CLAs [50]: Based on their structure, CLAs can be categorized as irregular or regular. In the irregular CLAs, the assumption for having a regular structure has been dropped.

Synchronous vs. asynchronous CLAs [51]: In Synchronous CLAs, the LAs residing in different cells are operated by a synchronous global clock, while in asynchronous CLAs, only a limited number of the cells are active at a given time, while the state of others stays unchanged. Wavefront CLAs [21] are a class of asynchronous CLAs which are different from CLAs in two main ways. First, the neighbors of an LA are defined as successor LAs. Second, each LA in the network can forward a wave into its successors if its selected action is opposed to the previous action.

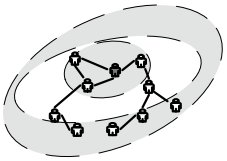
Static vs. dynamic CLAs [52]: The former demonstrates that the cellular structure of CLAs remains the same through time, whereas the latter illustrates that one of CLAs' sights, such as a local rule, structure, or adjacency radius, changes over time. In the Reference mentioned above, the models have been introduced, and it has been illustrated that for commutative rules (a class of rules), the mentioned models converge based on a stable general state.

4 Multi-wave CALA (MWCALA)

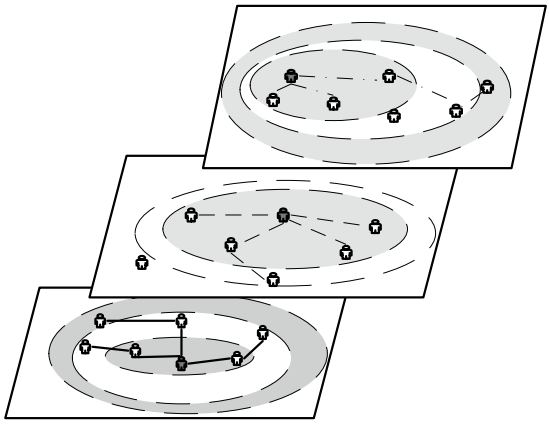
A Multi-Wave CALA (MWCACLA) is a type of Wavefront CLA (WCLA). As shown in Fig. 3, the main difference between the MWCACLA and WCLA is that in the WCLA, only one wave moves along the network in every instance, while in the MWCACLA, several waves can move along the network in every instance. Waves generated by several CALAs operate in parallel.

The initial idea by parallel operation of waves is that since the response from the environment is quite random, a decision made on multiple responses will have less chance of a mistake than a decision made on one response. Also, updating the action probability distribution will be much more precise and facilitating more quick convergence.

In some applications such as social networks, it is necessary that at each given time, more than one wave is active in the network. Each wave is considered for one decision-making variable, which can be matched by one CALA. The parallel operation of waves has to do with using a module of CALAs instead of one LA in cells. We have called this type of CLA the Multi-Wave CALA (MWCACLA). Similar to the WCLA, in the MWCACLA, its neighbors are defined as its successors, and in each cell, each CALA that is activated can send out a wave to its corresponding CALAs in its neighborhood cells, and they will be activated if the action chosen differs from the previous action (the propagation feature). In the k^{th} run, the CALAs in a cell are activated and choose one of their actions. If their chosen actions are different from their previous actions, they will send some waves over the network. Each cell receiving the wave will be activated, and the corresponding CALAs that have received the wave from the other cells should choose their actions. Then, the MWCACLA will create a reinforcement signal using the local rule as well as the actions chosen in the neighborhood so as to update the probability distribution for the active cell. The process is repeated until the desired outcome is achieved.



Wavefront Cellular Learning Automata



Multi-Wave Cellular Learning Automata in Multi-layer Network

Fig. 3 Multi-wave cellular learning automata in a simple view

The procedure of the MWCACLA is demonstrated in Fig. 4 from a straightforward viewpoint. In this figure, first, the root cell 1is activated and each CALA residing in the module of CALAs in cell 1 chooses one action based on a common normal distribution. As each CALA selects a different action in comparison with the previously selected action, each CALA activates all of its children CALAs in the next step (CALAs residing in cell 2 and cell 3, for example, $CALA_1^1$ generate wave one and forward the wave one to $CALA_2^1$ and $CALA_3^1$, likewise $CALA_1^2$ generate wave two and forward the wave two to $CALA_2^2$ and $CALA_3^2$ and so on). Then, each activated CALA selects a new action, and if its selected action is the same as its prior action, then the chain is stopped at that vertex (colored as red), and if the new action is not the same as the prior action, it activates its children, and this procedure continues. A dashed arrow indicates the direction of propagating each wave with a specific color (wave one with green color, wave two with blue color, wave three with green color). In the MWCACLA algorithm, since more than one wave is active in the environment at each given time, the algorithm is run in a parallel way. It is also close since it only uses the local environment for updates. It is also static since its structure does not change with time.

In the MWCACLA, the path of propagation for waves depends only on the neighborhood structure. Each wave propagated in the environment has two features: 1. Energy 2. Frequency. The energy of a wave is indicative of the wave's propagation intensity. The energy of waves diminishes as they move along the network until it vanishes, and then the wave disappears, and the movement stops. The frequency of waves is the length of time in which they are repeated. In the

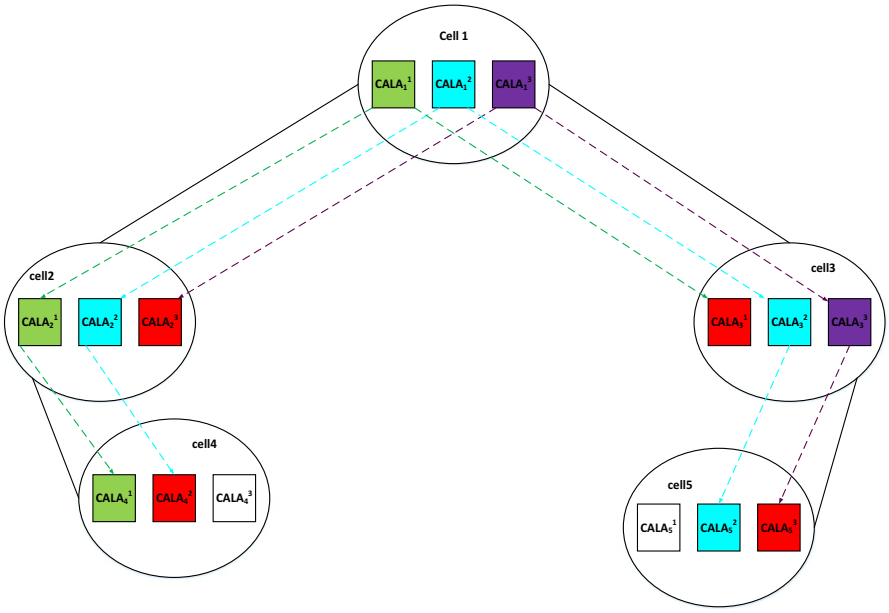


Fig. 4 MWCACLA example

MWCACLA, some waves are sent by a root cell over the network in each run. The waves are not repeated until the CALAs of the root cell choose new actions.

As mentioned before, the idea used in this paper is to employ a module of n CALAs in the MWCALA instead of a single LA in the WCLA. Provided that there are N LAs in the WCLA, so in MWCACLA, there will be N cells, each possessing a module with n CALAs. These CALAs now can create n waves working in parallel. The CALAs of the i^{th} cell in each such MWCACLA together create the i^{th} module and share a common actions probability vector x_i.

Definition (2): In stage k, the MWCACLA configuration is shown as $X(k)$, which is defined as the following as a probability vector for the entire LAs:

$$X(k) = (x_1^T(k), x_2^T(k), \ldots, x_N^T(k))^T \tag{14}$$

Consider a group of modules of CALAs.

Then,

$$x_i(k) = (\mu_i(k), \sigma_i(k))^T \tag{15}$$

where x_i is the state related to i^{th} CALA in each wave with mean μ_i and standard deviation σ_i, and x_i indicates the Gaussian distribution for selecting different values.

The n parallel waves interact with the environment based on the actions selected by their CALAs and receive a common reinforcement signal for updating the common Gaussian distribution of their CALAs. Supposed that $\beta_{x^\tau(k)}$ is the reinforcement signal by the τ th wave at time k, the objective of the MWCACLA will then be defined as:

$$\text{maximize} f(P) = \sum_{\tau=1}^{n} E[\beta_{x^\tau(k)}|X] \tag{16}$$

Since the n waves share the same Gaussian distribution$X_i, (i = 1, \ldots, N)$,

$$E[\beta_{x^1(k)}|X] = E[\beta_{x^2(k)}|X] = \cdots = E[\beta_{x^n(k)}|X] \tag{17}$$

Therefore:

$$f(P) = nE[\beta_{x^\tau(k)}|X] for any \tau \in \{1, 2, \ldots, n\} \tag{18}$$

Considering that $\beta_{x^\tau(k)}$ can vary with τ. Therefore, (16) equals to:

$$\text{maximize} f(P) = E[\beta_{x^\tau(k)}|X] \text{for any} \tau \in \{1, 2, \ldots, n\} \tag{19}$$

In the following the mechanism of n parallel CALA, which make n waves in a cell is described:

- The variance and mean of the Gaussian distribution for the ith module are initialized as $\sigma_i(0)$ and $\mu_i(0)$ at time step $k = 0$.
- Based on the Gaussian distribution n random numbers $x_i^\tau(k)(\tau = 1, \ldots, n)$ are generated, $N(\mu_i(k), \emptyset(\sigma_i(k)))$, which is the action probability distribution at k.

– The reinforcements $\beta_{x_i^\tau(k)}(\tau = 1, \ldots, n)$ and $\beta_{\mu_i(k)}$ are calculated from the environment.
– The standard deviation $\sigma_i(k)$ and mean $\mu_i(k)$ are updated as follows:

$$\mu_i(k+1) = \mu_i(k) + \tilde{\lambda} \sum_{\tau=1}^{n} F_1\left[\mu_i(k), \sigma_i(k), x_i^\tau(k), \beta_{x_i^\tau(k)}, \beta_{\mu_i(k)}\right] \tag{20}$$

$$\sigma_i(k+1) = \sigma_i(k) + \tilde{\lambda} \sum_{\tau=1}^{n} F_2\left[\mu_i(k), \sigma_i(k), x_i^\tau(k), \beta_{x_i^\tau(k)}, \beta_{\mu_i(k)}\right] - \tilde{\lambda}\, Ka\left[\sigma_i(k) - \sigma_L\right] \tag{21}$$

where, F_1 and F_2 are shown in Eqs. (11) and (12) and $\tilde{\lambda} = \frac{\lambda}{n}$ is a normalized learning parameter.

4.1 Convergence analysis for MWCACLA

In this Section, the asymptotic behavior of the MWCACLA algorithm is analyzed. For this purpose, an approach based on deriving an associated *Ordinary Differential Equation (ODE)* is used. For analyzing the convergence, we assume an MWCACLA with the following features. First, the structure of MWCACLA, S, is a directed graph without any directed cycle (directed acyclic graph) with E as the set of edges and V as the set of vertices. Second, we assume a finite set of successor cells at the predefined depth, N, as the neighborhood set for each cell.

The optimization problem (19) is considered as the goal of the MWCACLA. The target of the MWCACLA is to maximize the reinforcement signals that waves obtain from the environment. So, we want to display that MWCACLA maximizes the reinforcement signals obtained from the environment.

Note that the analysis in this paper is independent of the MWCACLA structure. Since in MWCACLA, more than one CALA is active at any time, and each has chosen an action, in this algorithm, only the normal distribution of active CALAs are updated, and the rest stay unchanged.

In essence, in this analysis, we need to understand how, over a long time, the CALAs action probabilities evolve in the introduced Eq. (20) and Eq. (21). This is achieved by learning algorithm estimation using an *ODE*. Since the objective for the learning algorithm is to solve the optimization problem Eq. (19), a connection should be established between the maximum of $f(P) = \sum_{\tau=1}^{n} E[\beta_{x^\tau(k)}|X]$ and the asymptotic behavior of the ODE.

In the MWCACLA algorithm, a multiplicity of CALAs works in a parallel fashion in a random environment and is in some shared situation at each given moment. Each CALA reacts to the situation by choosing an action and is then penalized or rewarded. The CALAs belonging to a cell cooperate with each other in some cases, such that they have a shared internal state representation. Actions chosen by each CALA depend on the shared normal distribution, but the chosen actions are independent of each other. Here, considering the above definitions, we analyze the learning algorithm used with one CALA in a wave and then develop the discussion for

multiple waves that work in parallel. Each CALA in the sequential case in a wave for updating its probability vector can use the learning algorithm which is given below:

$$X(k+1) = X(k) + \lambda G(X(k), \xi(k)) \tag{22}$$

where G represents the updating of the two components of $X(k)$ as given by (9) and (10) and $X(k) = [\mu(k), \sigma(k)]^T$ is the state at k and. Here we take $\xi(k) = [x(k), \beta_{x(k)}, \beta_{\mu(k)}]^T$ and the updating depends on both $X(k)$ and $\xi(k)$. $\xi(k)$ is random, and it takes different probabilities values based on $X(k)$. Thus the general learning algorithm as represented by (23) is a stochastic difference equation, and we are interested in the asymptotic properties of $X(k)$. In this Section, we derive an ODE whose solution provides a good approximation to the behavior of the algorithm when λ is sufficiently small. The main approach to analyzing such an algorithm is explained in the following.

Recall that $x(k)$ is the action selected at k and $\beta_{x(k)}$ and $\beta_{\mu(k)}$ are the reinforcements for the two actions $x(k)$ and $\mu(k)$, respectively. Let $f(x)$ shows the reward probability function. That is, $f(x) = E[\beta_{x(k)} | x(k) = x]$. Define:

$$J(\mu, \sigma) = \int f(x) dN(\mu, \sigma) = \int f(x) N(\mu, \sigma) dx \tag{23}$$

where $N(\mu, \sigma)$ shows the normal density, and it is defined as follow:

$$N(\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{24}$$

Considering Eq. (11) and (12), we have:

$$\begin{aligned}
&E\left[F_1\left(\mu(k), \sigma(k), x(k), \beta_{x(k)}, \beta_{\mu(k)}\right) | \mu(k) = \mu, \sigma(k) = \sigma\right] \\
&= \int E\left[\frac{\beta_x - \beta_\mu}{\emptyset(\sigma(k))} | x(k) = x, \mu(k) = \mu, \sigma(k) = \sigma\right] \left(\frac{x-\mu}{\emptyset(\sigma)}\right) dN\left(\mu, \emptyset(\sigma)\right) \\
&= \int \left(\frac{f(x) - f(\mu)}{\emptyset(\sigma)}\right) \left(\frac{x-\mu}{\emptyset(\sigma)}\right) \frac{1}{\emptyset(\sigma)\sqrt{2\pi}} exp\left(-\frac{(x-\mu)^2}{2\emptyset(\sigma)^2}\right) \mathrm{dx} \\
&= \int f(x) \left(\frac{x-\mu}{(\emptyset(\sigma))^2}\right) \frac{1}{\emptyset(\sigma)\sqrt{2\pi}} exp\left(-\frac{(x-\mu)^2}{2\emptyset(\sigma)^2}\right) \mathrm{dx} \\
&= \int f(x) \frac{\partial}{\partial\mu} \left(\frac{1}{\emptyset(\sigma)\sqrt{2\pi}} exp\left(-\frac{(x-\mu)^2}{2\emptyset(\sigma)^2}\right)\right) dx \\
&= \frac{\partial J}{\partial \mu}\left(\mu, \emptyset(\sigma)\right)
\end{aligned} \tag{25}$$

The last step above follows because $f(\mu)$ is independent of x and so comes out of the integral and the rest of the integral in that term is zero.

Similarly, we can show that

$$E\left[F_2\left(\mu(k),\sigma(k),x(k),\beta_{x(k)},\beta_{\mu(k)}\right)-K\left(\sigma(k)-\sigma_L\right)|\mu(k)=\mu,\sigma(k)=\sigma\right] = \frac{\partial J}{\partial \mu}\left(\mu,\emptyset(\sigma)\right)-K\left[\sigma-\sigma_L\right] \tag{26}$$

Equations (26) and (27) show that the expectation of the change in $\mu(k),\sigma(k)$ conditioned on their current values is proportional to the gradient of the average reinforcement, J. We call it the stochastic gradient following the property of the algorithm.

Theorem 1 *Since the state here has two components, namely, μ and σ, using* (26) *and* (27)*, the approximating ODE for the CALA algorithm is*:

$$\frac{d\mu}{dt}=\frac{\partial J}{\partial \mu}\left(\mu,\emptyset(\sigma)\right) \tag{27}$$

$$\frac{d\sigma}{dt}=\frac{\partial J}{\partial \sigma}\left(\mu,\emptyset(\sigma)\right)-K\left[\sigma-\sigma_L\right] \tag{28}$$

This completes the calculation of the approximating ODE for the CALA algorithm. To prove the approximating ODE, authors in the [9] verify assumptions A1 to A4 as follows.

Assumptions **A1**. $\{(X(k),\xi(k)),k\geq 1\}$ is a Markov Process.

A2. For any appropriate Borel set, B,

$$Prob[\xi(k)\in B|X(k),\xi(k-1)]=Prob[\xi(k)\in B|X(k)]$$

This is condition on $X(k)$, $\xi(k)$ is independent of $\xi(k-1)$.
A3. Define $g:\mathfrak{R}^N\rightarrow\mathfrak{R}^N$ by

$$g(x)=E[G(X(k),\xi(k))|X(k)=x]$$

We assume that $g(0)$ is independent of k and that is globally Lipschitz.
A4. Define

$$\theta(k)=G(X(k),\xi(k))-g(X(k))$$

We assume that $E||\theta(k)||^2<M<\infty$, for some M and $\forall k$.

As mentioned earlier, in the MWCACLA, a number of waves work in parallel. Therefore, the algorithm for updating the probability for MWCACLA is rewritten as (30). The updating algorithm vector in MWCACLA is achieved by adding up the update vectors in n waves, which uses the algorithm in each wave individually.

Remark 1 So, learning algorithm (23) for n waves in MWCACLA can be rewritten as:

$$X(k+1) = X(k) + \tilde{\lambda} \sum_{\tau=1}^{n} G(X(k), \xi^{\tau}(k)) \tag{29}$$

Here the step size for learning is $\tilde{\lambda} > 0$.

By the summations in Eq. (20) and (21) the strength of parallelization in the proposed MWCACLA is indicated. It obtains from an application of the ODE method of analysis that an n-fold increase in speed is also obtained. A formal statement of this fact is illustrated in Theorem 2 as follows.

Theorem 2 *For the algorithm given by* (20) *and* (21), *the approximating ODE is given by*.

$$\frac{d\mu}{dt} = n \frac{\partial J}{\partial \mu}\left(\mu, \emptyset\ (\sigma)\right) \tag{30}$$

$$\frac{d\sigma}{dt} = n \frac{\partial J}{\partial \sigma}\left(\mu, \emptyset\ (\sigma)\right) - K\left[\sigma - \sigma_L\right] \tag{31}$$

where J is the same function as in the case of a single CALA.

Proof: The proof is a straightforward extension of the derivation of the approximating ODE for a single wave CALA which is described for one wave earlier.

ODE (27), (28), and (30), (31) include a set of stationary points with identical characteristics with no difference except that ODE (30) and (31) have been compressed n-fold on the time axis. If λ enjoys the identical values for both ODEs, and n is thus that λ falls in the allowable range, for given time T ODE (30) and (31) will be faster than ODE (27) and (28) based on factor n.

In this step, we want to study the ODE of MWCALA, so first, we consider the ODE of one wave CALA.

Considering the stochastic gradient following property of the algorithm (23) as explained sooner, the ODE associated with the corresponding algorithm is as expected. Given Eqs. (28) and (29), the right-hand side (RHS) of the ODE is the gradient of J is evaluated at the point$(\mu, \emptyset(\sigma))$. The additional part in Eq. (28) is because of the penalty term used in the updating of$\sigma(k)$. As a result, we can expect μ and σ to converge to some maxima of$J$. But our goal is to find a maximum of f and not of$J$. Observe that J is the expected value of the reinforcement, where the expectation is taken with respect to both 1) the normal distribution with parameter μ and σ using which actions are chosen and 2) the unknown distribution which the reinforcement β_x is obtained. Except this, we have to consider the penalizing term for σ in (28). From (23), it is intuitively clear that $J(\mu, \sigma)$ would be close to $f(\mu)$ if σ is sufficiently small. So, we can expect that for sufficiently smallσ, $\frac{\partial J}{\partial \mu}(\mu, \sigma)$ would be close to $f\prime(\mu) \triangleq \frac{\partial f}{\partial x}|_{x=\mu}$ and as a result, maxima of J would be close to maxima of$f$. Relation between maxima of f and constrained maxima of J (to which the solution of the approximating ODE and therefore the algorithm converge) can be established. We can prove that the algorithm converges to a close approximation of an isolated local maximum of$f$.

From (19), any equilibrium point of the ODE should satisfy$\frac{\partial J}{\partial \mu}\left(\mu, \emptyset(\sigma)\right) = 0$. Suppose we can establish that in any equilibrium point of the ODE(μ, σ), we would have $\emptyset(\sigma)$ close toσ_L. Then the above results imply that if we choose σ_L sufficiently small, then, the ODE and hence the algorithm would converge to a maximum of J, which is close to a maximum of$f$. Suppose, $> \sigma_L$. Since (μ, σ) is an equilibrium point, we must have

$$\frac{\partial J}{\partial \sigma}\left(\mu, \emptyset\ (\sigma)\right) - K\left[\sigma - \sigma_L\right] = 0 \tag{32}$$

Lemma 1 *If we can choose* $K \gg \frac{\partial J}{\partial \sigma}\left(\mu, \emptyset(\sigma)\right)$ *uniformly for all* μ, σ, *then the above holds only for some* σ *close to* σ_L. *By assumptions as illustrated earlier and some assumption on the function* $f(x)$ *as explained in* [53], *such a choice of K exists. So, we can conclude that in any equilibrium point of the ODE* (μ, σ), *we would have* $\emptyset(\sigma)$ *as close to* σ_L *as desired by taking K sufficiently large*.

Putting all this together, we can conclude that by choosing a small value of $\sigma_L > 0$, a small step size $\lambda > 0$ and sufficiently high $K > 0$, we can ensure that the iterates $\mu(k)$ of the CALA algorithm will be close to a maximum of the function f with a high probability after a long enough time.

As for approximate precisions, there will be identical arguments for both sequential and parallel algorithms in the case of a unique stable equilibrium point. Please note that although the given algorithm works in a parallel fashion, it does not change the asymptotic properties of the sequential algorithm. Please also note that as the value of n increases, the convergence speed does not necessarily increase because there are limits in the state space.

In Reference [21], lemma 1 has been proven for ODE (22), and since ODE (30) is an n-factor in comparison to ODE (22), this lemma can be easily proven through the given considerations.

We think that MWCACLA is an appropriate model for the LP problem. Because MWCACLA has propagation probability, the learning process can be propagated in the network. Also, more than one wave can work in the environment each time, so MWCACLA is suitable for multi-layer social networks. Each wave can consider the information of one layer, and then the information of different layers is merged to predict future connections. So, in the next Section, we use MWCALA for the weighted multi-layer LP problem.

5 LP in weighted multi-layer social networks

In this Section, we have selected the problem of predicting links in social networks to examine the behavior of MWCACLA. A new algorithm for LP in weighted multi-layer social network using MWCACLA has been proposed, and it is called

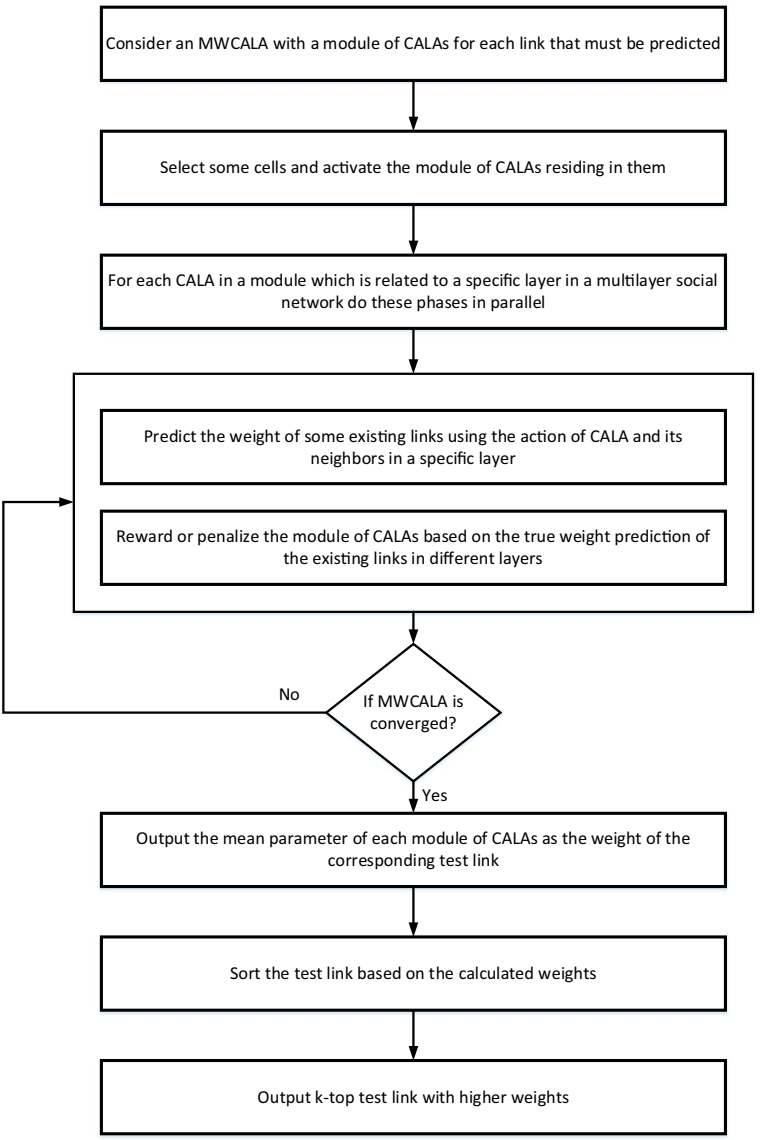


Fig. 5 Flowchart of the proposed weighted multi-layer link prediction based on MWCALA

MWCALA-WLP. The MWCACLA-WLP consists of three phases: modeling phase, training phase, and prediction phase.

The first phase- Modeling phase: In the first phase of the proposed method, we model a social network based on MWCACLA, as shown in Fig. 6. Our goal in the LP problem is to predict relations of different types of links. For example, in the multi-relational bibliographic dataset, we have three types of links, as we have considered earlier. We try to predict the probability of writing a paper in the future with

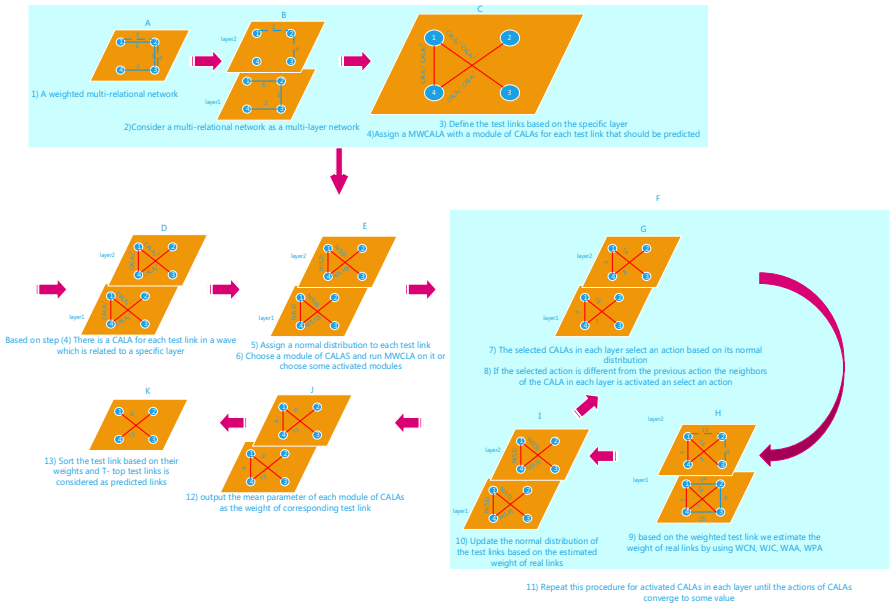


Fig. 6 Example of the proposed weighted multi-layer LP based on MWCACLA

two authors using the information of the connections in other layers. As we have shown in Fig. 5, the main phase of the proposed algorithm is as follow:

The first phase- step one: The co-authorship links have been divided into a training set and test set. The weight information of the co-authorship training set, as well as two other types of links, are used for predicting the near future co-authorship links.

The first phase- step two: In this step, we assign a module of CALAs to each test link that should be predicted (Fig. 6, Sub-Fig. 6(C)). The CALAs, which are assigned to a test link, cooperate with each other to learn the accurate score of the corresponding test link by using the connection weights in different layers of the social network. On the other hand, each CALA in a module tries to learn the weight of the corresponding test link based on a specific layer and generate a wave which is moving on the corresponding layer.

The second phase- Training phase: Due to the fact that in this method, we consider a multi-layer social network and there are some waves which are working parallel, and each wave is related to a specific layer, so in the following, we have explained the mechanism of CALAs in one layer.

The second phase- step one: in instant k, we randomly select some cells and activate the module of CALAs residing in the activated cell.

The second phase- step two: In this step each $CALA_i^{[\tau]}$ in an activated cell's module selects an action based on the normal distribution $N_i(\mu, \sigma)$ of the i^{th} cell. Each $CALA_i^{[\tau]}$ in the i^{th} the active module is related to τ^{th} layer. The selected action is assumed as the weight of the corresponding test link. If the selected action is distinct

from the prior action, the neighbors' CALAs are activated. In this way, a wave is generated and moves in a layer until a predetermined depth.

The second phase-step three: after all neighbors of $CALA_i^{[\tau]}$ in the τ^{th} layer choose their actions, we evaluate the actions, and then based on the reinforcement signals which are generated from different layers, the probability distribution of the i^{th} cell is updated. At instant k for each $CALA_i^{[\tau]}$ in a layer, two reinforcement signal will be calculated, $\beta_{\alpha_i^{\tau}(k)}(\tau = 1, 2, \ldots, n)$ for selected action and $\beta_{\mu_i(k)}$ for the mean parameter. To do so, for each layer, we make a weighted network $GI_{\text{Weighted}}^{[\tau]}$ that is made by using the selected action of $CALA_i^{[\tau]}$ and its neighbors. Also, we have a training set for each layer which is a real link with some weight values$G_{\text{weighted}}^{[\tau]}$. Now, by using the weight of links in the graph $G_{\text{weighted}}^{[\tau]}$, we calculate the weight of γ percentage of real links in$G_{\text{weighted}}^{[\tau]}$. For this purpose, we use one of the similarity metrics such as WCN, WPA, WAA, WJC. The percentage of the links in the graph $G_{\text{weighted}}^{[\tau]}$ is considered as a parameter, and we study it in the experiments. The reinforcement signal for $CALA_i^{[\tau]}$ in the τ^{th} layer can be calculated as follows:

$$\beta_{\alpha_i^{[\tau]}}(k) = \sum_{l=0}^{q} (w'^{[\tau]}_l(\alpha_i^{[\tau]}(k), \alpha^{[\tau]}_{\text{neighbors}(i)}(k))) - w_l^{[\tau]} \tag{33}$$

$$\beta_{\mu_i}(k) = \sum_{l=0}^{q} (w'^{[\tau]}_l(\mu_i(k), \alpha^{[\tau]}_{\text{neighbors}(i)}(k))) - w_l^{[\tau]} \tag{34}$$

where q is the number of real links in the τ^{th} layer of the graph G(graph $G^{[\tau]}$) that we calculate its weights, $w'^{[\tau]}_l(\alpha_i^{[\tau]}(k), \alpha^{[\tau]}_{\text{neighbors}(i)}(k)))$ can be the calculated weight of link l based on the selected action of $CALA_i^{[\tau]}$ and selected actions of its neighbors $\alpha^{[\tau]}_{neighbors(i)}(k)$ in the τ^{th} layer, $w'^{[\tau]}_l(\mu_i(k), \alpha^{[\tau]}_{neighbors(i)}(k)))$ can be the calculated weight of link l based on the mean of $CALA_i^{[\tau]}$ as well as selected actions of neighbors CALAs in the τ^{th} layer, $\alpha^{[\tau]}_{neighbors(i)}(k)$. Finally, $w_l^{[\tau]}$ is the real weight of link l in the τ^{th} layer. After calculating the $\beta_{\alpha_i^{[\tau]}}(k)$, and $\beta_{\mu_i}(k)$ of the different layers, the reinforcement signal will be generated based on Eq. (20) and (21). This procedure is repeated in each layer for each CALA that is activated. The general operation of the proposed method is repeated until the action of CALA converges to some value.

The third phase: in this phase, the final results of predicted links will be generated. In this way, we sort the calculated weights for test links, and the top T links with the most weights are considered as existence links.

6 Example

An example of the MWCALA-WLP method on a weighted multi-layer network has been given: consider Fig. 6., Sub-Fig. 6A illustrates a multi-relational weighted network in which there is more than one connection between two nodes. In Sub-Fig. 6B, we show the multi-relational weighted network as a weighted multi-layer

network with each connection in a layer has a weight value. Sub-Fig. 6C illustrates possible test connections in the multi-layer network and MWCLA structure with its module of CALAs. As we described earlier, we assign a module of CALAs to each test connection, with a normal distribution as its parameter. Each CALA in a module is related to the test connection in a specific layer, as it has been shown in Sub-Fig. 6D. Then one module of CALAs is activated, and one $CALA_i^{[\tau]}$ related to a specific layer chooses an action based on the module's normal distribution, and if the selected action is distinct from the prior one, it activates all neighbors. The chosen action of $CALA_i^{[\tau]}$ and its neighbors in the different layers are shown in Sub-Fig. 6G. Now we create a new weighted network for each layer $G_w^{[\tau]}$ and use it to calculate the weight of original links in each layer based on some weight similarity criteria such as WJC, WCN, WPA, WAA. Sub-Fig. 6H shows the weight of the real connections of each layer that are estimated based on a similarity metric. Likewise Sub-Fig. 6I shows that each module of CALAs updates its normal distribution based on its effect on the true weight calculation of the original network. Then this procedure repeats for each activated module of CALA again, as it demonstrated in Sub-Fig. 6F until the action of each CALA in a specific layer converges to some value as demonstrated in Sub-Fig. 6J. Then the test connections are sorted based on their scores, and the existence or absence of each test connection is predicted based on its weights, as illustrated in Sub-Fig. 6K.

7 Results and experiments

In this Section, the experiments performed to assess the MWCALA-WLP approach are explained. First, the social network data utilized and the procedure of experiments are explained (Sect. 7.1). In Sect. 7.2, we propose the evaluation metric that we utilize as a part of our analysis. In Sect. 7.3., three experiments for evaluating the MWCALA-WLP are presented, and the performance of MWCALA-WLP is compared with a number of LP methods.

7.1 Datasets

In this Section, we have proposed two groups of datasets which are used for experiments. The analysis of these datasets is given below:

1. **The Higgs dataset:** it has been collected by monitoring the Twitter social network. There are four networks which are extracted from user activities as retweeting (RT) network consists of 256,491 nodes and 328,132 edges, replying (RP) network consists of 38,918 nodes and 32,523 edges, mentioning (MN) network consists of 116,408 nodes and 150,818 edges, and friends' relationships (FR) consists of 456,626 nodes and 14,855,842 edges. This network makes four layers, and we have used the information of these layers for the LP problem.

2. **The DBLP dataset:** it has been used as the second dataset, which is collected from ArnetMiner, the nodes represent the authors, and the links represent the relations between authors. There are three types of relationships between two authors: co-author relation, if two authors have written at least one paper together, co-citation, if they cite the same paper and same-venue if they publish in the same venue. The complete dataset consists of 2,092,356 research papers, and 1,712,433 authors. We have selected 100,000 papers and extracted data from 1984 to 1995 and, based on the three different relations between authors, make three graphs: co-author, co-citation, and same-venue network. These three graphs are used as three layers for LP problems in multi-layer networks.

For making the weighted version of the network, the weight of each link is estimated by the total number of occurred events between the two connected nodes.

7.2 Metrics for evaluating the proposed LP algorithm

We use the AUC metric to help us compare MWCACLA-LP with other LP approaches as the following [44]:

AUC: Provided that all non-existence connections are ranked through their scores. For calculating the AUC, a missing connection and a non-existent one has been selected randomly to compare their scores at each time. We assess the AUC as the probability that a randomly chosen missing connection has been assigned with a higher score than a randomly selected non-existent connection. So, If in n comparisons, there are n' times missing connections have a higher score and n'' times have the same score, the AUC equals Eq. (35):

$$\text{AUC} = \frac{n^{'} + 0.5n\varepsilon}{n} \tag{35}$$

Precision: Provided that we predict that L connections to be connected and L_r connections out of L connections are right; thus, the precision equals Eq. (36).

$$\text{Precision} = \frac{L_r}{L} \tag{36}$$

7.3 Experiments

For evaluating the accuracy of the proposed MWCALA-WLP, we conduct some experiments. To do so, for the Higgs dataset, we use data between 1 to 6^{th} July 2012 as the training data and 7th July as the test data. Likewise, for the DBLP dataset we use the data from 1984 to 1994 as the training data and the year 1995 as the test data. For all following experiments, the initial parameter of each module of CALAs in the i^{th} cell $(\mu_0(i), \sigma_0(i))$ are chosen randomly,λ= 0.005, $\sigma_L = 10^{-3}$, and the wave energy or wave depth is equal to 4. We run the proposed

MWCALA-WLP with four similarity metrics (WCN, WJC, WAA, WPA) which were described in Sect. 2. We call the proposed algorithm MWCALA-WLP-XX if the proposed algorithm uses similarity metric XX in its procedure.

For evaluating the performance of the proposed MWCACLA-WLP, some experiments have been conducted, and it has been compared with some algorithms. We have selected some unweighted and weighted local similarity-based algorithms such as PA [46], AA [47], JC [45], CN [44], WCN, WJC, WAA,WPA,WRA [43] quasi-local similarity-based algorithms such as local path [33], some global similarity-based algorithm such as Katz [42], and WProbflow [40], some multi-relational LP method such as MRLP [38], MRIP [36], MRPF [41], and some recent algorithms such as LA-TSLP [27], ICLA-EC-TSLP [30], ICLA-LP [31], CALA-LP [28], FLP-DLA [26], CALA-WLP [5] to compare with MWCALA-WLP.

7.3.1 Experiment one

The results gained for the MWCALA-WLP method, as well as their results for other methods for Higgs, and DBLP datasets using AUC, are illustrated in

Table 1 Comparison of the MWCACLA-LP with similarity-based methods on the Higgs dataset

Method/dataset	Layer (FR)	Layer (MN)	Layer (RP)	Layer (RE)
CN [44]	0.6011	0.6128	0.5435	0.5924
JC [45]	0.6059	0.6095	0.5324	0.5872
PA [46]	0.6312	0.7025	0.6821	0.6697
AA [47]	0.6108	0.6228	0.5839	0.6057
Local Path [33]	0.6529	0.6685	0.5962	0.6328
Katz [42]	0.6712	0.6822	0.6932	0.7706
LA-TSLP [27]	0.7935	–	–	–
ICLA-EC-TSLP [30]	**0.8124**	–	–	–
ICLA-LP [31]	0.8065	–	–	–
CALA-LP [28]	**0.8227**	–	–	–
FLP-DLA [26]	0.8009	–	–	–
CALA-WLP [5]	**0.8105**	–	–	–
WCN [43]	0.6483	0.6589	0.5979	0.6215
WJC [43]	0.6449	0.6562	0.5943	0.6250
WPA [43]	0.7290	0.7671	0.7343	0.7644
WAA [43]	0.6485	0.6594	0.5949	0.6261
WRA [43]	0.6404	0.6172	0.5534	0.5841
Probflow [40]	0.6569	0.7128	0.6107	0.6440
MRLP [38]	0.6638	0.7085	0.6201	0.6495
MRIP [36]	0.6812	0.7122	0.6331	0.6534
MRPF [41]	0.7042	0.7321	0.6982	0.7196
MWCACLA-WLP	**0.8219**	**0.7914**	**0.7647**	**0.7824**

Table 2 Comparison of the MWCACLA-LP with some methods on the DBLP dataset

Method/Dataset	Layer (co-author)	Layer (co-citation)	Layer (same-venue)
CN [44]	0.5634	0.5309	0.5691
JC [45]	0.5512	0.5237	0.5541
PA [46]	0.5723	0.5439	0.5691
AA [47]	0.5843	0.5721	0.5789
Local Path [33]	0.6115	0.6002	0.6028
Katz [42]	0.6348	0.6227	0.6331
LA-TSLP [27]	0.7423	–	–
ICLA-EC-TSLP [30]	0.7895	–	–
ICLA-LP [31]	0.7632	–	–
CALA-LP [28]	**0.8013**	–	–
FLP-DLA [26]	0.7925	–	–
CALA-WLP [5]	0.7991	–	–
WCN [43]	0.5957	0.5872	0.5883
WJC [43]	0.5921	0.5763	0.5810
WPA [43]	0.6212	0.6032	0.6197
WAA [43]	0.6197	0.5991	0.6043
WRA [43]	0.6032	0.6023	0.6112
Probflow [40]	0.6843	0.6912	0.6834
MRLP [38]	0.6924	0.7029	0.6886
MRIP [36]	0.7001	0.7093	0.6927
MRPF [41]	0.7095	0.7116	0.7031
MWCACLA-WLP	**0.8114**	**0.7823**	**0.7797**

Tables 1 and 2. The parameters of the used methods are borrowed from their references. The best results are shown in bold. The results reported here demonstrate that the proposed MWCACLA-WLP is able to receive an AUC measure that is much better than the similarity-based methods, which calculate the score of test links just based on the local information of neighbors in depth one. Also, the achieved results are better than Katz, Local Path. According to Table 1, the suggested MWCALA-LP can reach AUC (0.8219) measure, which far surpasses all considered similarity-based methods, which consider or do not consider weight. Likewise, in comparison to LA and CLA-based methods, we see that the proposed algorithm achieves a better result in AUC measure. Therefore, we also conclude that MWCALA-WLP is superior to recent weighted LP methods for multilayer social networks and no worse than other recent ones.

7.3.2 Experiment two

In this experiment, we choose γ percentage of training links in each layer in the learning phase and study the impact of using the different number of layers as

Table 3 Comparison of the selection methods and number of waves on the performance of the proposed method.

Method	Data set	
	Higgs	DBLP
One Wave-CACLA-WLP Random	0.7624	0.6485
One Wave-CACLA-WLP Ascending	0.6539	0.5832
One Wave-CACLA-WLP Descending	0.6324	0.5729
Two Wave-CACLA-WLP Random	0.7909	0.8022
Two Wave-CACLA-WLP Ascending	0.6723	0.6051
Two Wave-CACLA-WLP Descending	0.6594	0.6008
Three Wave-CACLA-WLP Random	0.8201	**0.8114**
Three Wave-CACLA-WLP Ascending	0.6612	0.6423
Three Wave-CACLA-WLP Descending	0.6651	0.6739
Four wave-CACLA-WLP Random	**0.8219**	–
Four wave-CACLA-WLP Ascending	0.6789	–
Four wave-CACLA-WLP Descending	0.6812	–

well as various percentages of training links with different weight order. There are three suggested strategies:

- Using γ percentage training links in each layer using random order and considering a different number of layers
- Using γ percentage training links in each layer with the lowest weights and considering a different number of layers
- Using γ percentage training links in each layer with the highest weights and considering a different number of layers.

Table 3 compares the MWCALA-WLP method using three selection strategies, where the reported value related to the best value of the XXX-Wave-CALA-WLP-XX for all four used similarity metrics and XXX number of layers. From the results reported in Table 3, it can be concluded that random selection of training links besides using the information of each wave propagated in each layer is the best method where we do not have any special information whether higher or lower weights are important for the LP task. As it has been shown for the Higgs datasetFour-Wave-CACLA-WLP Random and for DBLP Three-Wave-CACLA-WLP, Random provides the best result. Also, in this experiment, we compare the impact of parameter γ which shows the percentage of training links that are selected for the proposed method evaluation. In this way, parameter γ can be varied from the set {30%, 65%, 100%} and we find that the best value of parameter γ is about 65%; because by choosing a small number of training set, the MWCALA-WLP underfits data and results in poor accuracy. Likewise, when we consider all training sets, the MWCALA-WLP overfits data and now again results in low accuracy. As a result, the best choice for parameter γ is about 65%

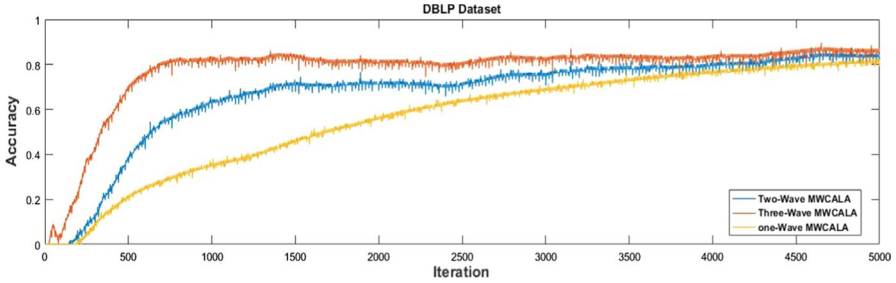


Fig. 7 The convergence diagram of the proposed MWCACLA-LP using different configuration

for all layers. So the illustrated results in all experiments are based on $\gamma = 65\%$. The best results are shown in bold in Tables 1, 2, and 3.

7.3.3 Experiment three

This experiment compares the impact of using a different number of waves; each of them uses the information of one layer in the social network. Figure 7 illustrates the result of this experiment for the DBLP dataset. This figure shows that the accuracy of the MWCALA-WLP is a little better when we use three waves, one wave for considering the information of each layer. In this case, we have a faster convergence compared to when we use one or two waves for considering the information of the one or two layers.

8 Conclusion

In this paper, we propose MWCACLA, which is an extension of continuous action set CLA. The proposed method supports the idea of using a larger number of waves, and these waves operate in a parallel manner. The parallel operation requires a module of CALAs instead of a single LA. The CALAs in the module share a common normal distribution for all, which is updated once each time. Each time, every CALA in the module chooses an action independently and extracts reinforcement from the environment, which forms the parallelism in the algorithm. The learning algorithm for MWCACLA has the ability to improve the speed while it maintains the required accuracy. This method is suitable for those problems where several actions can be simultaneously activated. The second feature of this approach is that the neighbors of each node are its successors' nodes, and each CALA in a module can generate a wave and forward the wave to the successors' nodes if its selected action is opposed to its previous action (the propagation property). These properties described here are useful in many applications such as web mining, social network analysis, and classification. Different types of CLAs can be extended to parallelization and propagation paradigm. We also evaluated the MWCACLA by weighted multi-layer LP problem in social networks and analyzed the behavior of the MWCACLA. The proposed

MWCALA-WLP uses continuous LAs to predict the existence or non-existence of each test link using the weight information of the different layers of the multi-layer network. The proposed method uses a module of CALA for each test link, and each CALA is responsible for learning the true score of the corresponding link based on the weight information in the specific layer of the multi-layer social network. All CALAs in a cell select the weight of their related link as their action based on the common normal distribution. Each CALA in a cell receives a reinforcement signal based on its influence on the true weight estimating of the training set in a specific layer. Based on the received reinforcement signals of CALAs in a cell, the normal distribution of the cell is updated. The final prediction is performed based on the estimated scores. The experimental results reported in this paper illustrate that the MWCALA-WLP is superior to other LP methods for weighted multi-layer social networks.

Data availability The data that support the findings of this study are available in Higgs Twitter Dataset at https://snap.stanford.edu/data/higgs-twitter.html [54], and DBLP at https://www.aminer.org/data/ [55] which are used in our experiments.

References

1. Jalili M, Orouskhani Y, Asgari M, Alipourfard N, Perc M (2017) Link prediction in multiplex online social networks. Royal Soc Open Sci 4(2):160863
2. Martínez V, Berzal F, Cubero J-C (2016) A survey of link prediction in complex networks. ACM Comput Surv (CSUR) 49(4):1–33
3. Li J-c, Zhao D-l, Ge B-F, Yang K-W, Chen Y-W (2018) A link prediction method for heterogeneous networks based on BP neural network. Physica A 495:1–17
4. De Bacco C, Power EA, Larremore DB, Moore C (2017) Community detection, link prediction, and layer interdependence in multilayer networks. Phys Rev E 95(4):042317
5. Moradabadi B, Meybodi MR (2018) Link prediction in weighted social networks using learning automata. Eng Appl Artif Intell 70:16–24
6. Wolfram S (1983) Statistical mechanics of cellular automata. Rev Mod Phys 55(3):601
7. Das D A survey on cellular automata and its applications. In, 2011. Springer, pp 753–762
8. Thathachar MAL, Sastry PS (2002) Varieties of learning automata: an overview. IEEE Trans Syst, Man, Cybernetics, Part B (Cybernetics) 32 (6):711–722
9. Thathachar MAL, Sastry PS (2011) Networks of learning automata: Techniques for online stochastic optimization. Springer Science & Business Media
10. Beigy H, Meybodi MR (2004) A mathematical framework for cellular learning automata. Adv Complex Syst 7:295–319
11. Navid AHF, Aghababa AB (2013) Cellular learning automata and its applications. Emerg Appl Cellular Automata 1:85–111
12. Esnaashari M, Meybodi MR (2008) A cellular learning automata based clustering algorithm for wireless sensor networks. Sens Lett 6(5):723–735
13. Ghavipour M, Meybodi MR (2017) Irregular cellular learning automata-based algorithm for sampling social networks. Eng Appl Artif Intell 59:244–259
14. Zhao Y, Jiang W, Li S, Ma Y, Su G, Lin X (2015) A cellular learning automata based algorithm for detecting community structure in complex networks. Neurocomputing 151:1216–1226
15. Manshad MK, Manshad AK, Meybodi MR (2012) Memory/search RCLA-EC: A CLA-EC for moving parabola problem. In: 2012. IEEE, pp 732–737
16. Manshad MK, Meybodi MR, Salajegheh A A new irregular cellular learning automata-based evolutionary computation for time series link prediction in social networks.

17. Mirsaleh MR, Meybodi MR (2016) A Michigan memetic algorithm for solving the community detection problem in complex network. Neurocomputing 214:535–545
18. Manshad MK, Manshad AK, Meybodi MR (2022) Memory/search RCLA-EC: A CLA-EC for moving parabola problem. In: 2011 6th International Conference on Computer Sciences and Convergence Information Technology (ICCIT), 2011. IEEE, pp 732–737
19. Vafashoar R, Meybodi MR, Azandaryani AHM (2012) CLA-DE: a hybrid model based on cellular learning automata for numerical optimization. Appl Intell 36(3):735–748
20. Patel DK, More SA (2013) Edge detection technique by fuzzy logic and Cellular Learning Automata using fuzzy image processing. In, 2013. IEEE, pp 1–6
21. Moradabadi B, Meybodi MR (2018) Wavefront cellular learning automata. Chaos: An Int J Nonlinear Sci 28(2):021101
22. Rezvanian A, Moradabadi B, Ghavipour M, Khomami MMD, Meybodi MR (2019) Wavefront Cellular Learning Automata: A New Learning Paradigm. In: Learning Automata Approach for Social Networks. Springer, pp 51–74
23. Murata T, Moriyasu S (2007) Link prediction of social networks based on weighted proximity measures. In, 2007. IEEE, pp 85–88
24. Lü L, Zhou T (2011) Link prediction in complex networks: A survey. Physica A 390(6):1150–1170
25. Al Hasan M, Chaoji V, Salem S, Zaki M (2006) Link prediction using supervised learning. In, 2006. pp 798–805
26. Moradabadi B, Meybodi MR (2017) Link prediction in fuzzy social networks using distributed learning automata. Appl Intell 47(3):837–849
27. Moradabadi B, Meybodi MR (2017) A novel time series link prediction method: Learning automata approach. Physica A 482:422–432
28. Moradabadi B, Meybodi MR (2016) Link prediction based on temporal similarity metrics using continuous action set learning automata. Physica A 460:361–373
29. Moradabadi B, Meybodi MR (2018) Link prediction in stochastic social networks: learning automata approach. Journal of computational science 24:313–328
30. Manshad MK, Meybodi MR, Salajegheh A (2020) A new irregular cellular learning automata-based evolutionary computation for time series link prediction in social networks. Appl Int 51:1–14
31. Manshad MK, Meybodi MR, Salajegheh A (2021) A variable action set cellular learning automata-based algorithm for link prediction in online social networks. J Supercomput 77:1–29
32. Xiang R, Neville J, Rogati M (2010) Modeling relationship strength in online social networks. In: 2010. pp 981–990
33. Zhou T, Lü L, Zhang Y-C (2009) Predicting missing links via local information. Eur Phys J B 71(4):623–630
34. Pech R, Hao D, Pan L, Cheng H, Zhou T (2017) Link prediction via matrix completion. EPL (Europhysics Letters) 117(3):38002
35. Ermiş B, Acar E, Cemgil AT (2015) Link prediction in heterogeneous data via generalized coupled tensor factorization. Data Min Knowl Disc 29(1):203–236
36. Yang Y, Chawla NV, Sun Y, Han J (2012) Link prediction in heterogeneous networks: Influence and time matters. In: Proceedings of The 12th IEEE International Conference on Data Mining, Brussels, Belgium
37. Negi S, Chaudhury S (2016) Link prediction in heterogeneous social networks. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp 609–617
38. Davis D, Lichtenwalter R, Chawla NV (2011) Multi-relational link prediction in heterogeneous information networks. In: 2011 International Conference on Advances in Social Networks Analysis and Mining. IEEE, pp 281–288
39. Sun Y, Barber R, Gupta M, Aggarwal CC, Han J (2011) Co-author relationship prediction in heterogeneous bibliographic networks. In: 2011 International Conference on Advances in Social Networks Analysis and Mining. IEEE, pp 121–128
40. Lichtenwalter RN, Lussier JT, Chawla NV (2010) New perspectives and methods in link prediction. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 243–252
41. Johnson RA, Yang Y, Aguiar E, Rider A, Chawla NV (2012) Alive: A multi-relational link prediction environment for the healthcare domain. Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, pp 36–46
42. Katz L (1953) A new status index derived from sociometric analysis. Psychometrika 18(1):39–43

43. Murata T, Moriyasu S (2007) Link prediction of social networks based on weighted proximity measures. In: IEEE/WIC/ACM International Conference on Web Intelligence (WI'07), IEEE, pp 85–88
44. Al Hasan M, Zaki MJ (2011) A survey of link prediction in social networks. In: Social network data analytics. Springer, pp 243–275
45. Jaccard P (1901) Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bull Soc Vaudoise Sci Nat 37:547–579
46. Newman ME (2001) Clustering and preferential attachment in growing networks. Phys Rev E 64(2):025102
47. Adamic LA, Adar E (2003) Friends and neighbors on the web. Social networks 25(3):211–230
48. Thathachar MA, Sastry PS (2002) Varieties of learning automata: an overview. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 32 (6):711–722
49. Beigy H, Meybodi MR (2007) Open synchronous cellular learning automata. Adv Complex Syst 10(04):527–556
50. Esnaashari M, Meybodi MR (2014) Irregular cellular learning automata. IEEE Trans Cyber 45(8):1622–1632
51. Beigy H, Meybodi MR (2008) Asynchronous cellular learning automata. Automatica 44(5):1350–1357
52. Saghiri AM, Meybodi MR (2017) A closed asynchronous dynamic model of cellular learning automata and its application to peer-to-peer networks. Genet Prog Evol Mach 18(3):313–349
53. Thathachar MA, Sastry PS (2011) Networks of learning automata: Techniques for online stochastic optimization. Springer Science & Business Media
54. De Domenico M, Lima A, Mougel P, Musolesi M (2013) The anatomy of a scientific rumor. Sci Rep 3(1):1–9
55. Bose AJ, Jain A, Molino P, Hamilton WL (2019) Meta-graph: Few shot link prediction via meta learning. arXiv preprint arXiv:191209867

Authors and Affiliations

Mozhdeh Khaksar Manshad[1] · Mohammad Reza Meybodi[2] · Afshin Salajegheh[3]

Mohammad Reza Meybodi
mmeybodi@aut.ac.ir

Afshin Salajegheh
a_salajegheh@azad.ac.ir

[1] Department of Computer Engineering, Omidiyeh Branch, Islamic Azad University, Omidiyeh, Iran

[2] Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran

[3] Department of Computer Engineering, South Tehran Branch, Islamic Azad University, Tehran, Iran