IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 10, 2019, accepted August 23, 2019, date of publication August 29, 2019, date of current version September 12, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2938202

Link Prediction Algorithms for Social Networks Based on Machine Learning and HARP

HAO SHAO, LUNWEN WANG, AND YUFAN JI
College of Electronic Engineering, National University of Defense Technology, Hefei 230037, China

Corresponding author: Hao Shao (shaohao64@outlook.com)

This work was supported by the China National Defense Science and Technology Innovation Special Zone Project under Grant 17-H863-01-ZT-003-204-03.

ABSTRACT In Node2Vec, the global structure of the network is neglected and the stochastic gradient descent (SGD) method is easy to fall into local optimum. Based on this algorithm, an improved link prediction algorithm combining machine learning and hierarchical representation learning for network (HARP) is proposed. This method first uses adaptive learning optimizer Adam instead of SGD to improve Node2Vec, then divides the nodes and edges of the original network graph into a series of smaller layered graphs by merging them according to HARP, and then uses the improved Node2Vec algorithm to extract features continuously, so as to realize network embedding. Finally, a social network link prediction model based on machine learning and HARP is established. A series of social network link prediction experiments are carried out. The results show that the new method has excellent performance and stability.

INDEX TERMS Adam, HARP, link prediction, machine learning, Node2Vec.

I. INTRODUCTION

With the appearance of social media sites, the analysis of social networking services has received widespread attention in the past few years. The social network which consists of the relationships (edges) between persons, teams or communities (nodes) and nodes is ubiquitous and becomes the basic part of modern information infrastructure. For the network G = (V, E) of which time t_i is given, it needs to find out the edges that will appear in the network when $t_2 > t_1$ for the problem of link prediction [1]. Prediction of possible new links, as a basic element of graph mining, is applied in multiple fields. In social media, recommendation of friends and contents is often defined as an issue of link prediction [2]. In addition, link prediction is applied to identify credit card fraud in the field of network security inspection [1], [3], predict interaction between protein in biological information science [4], give shopping and film recommendation in e-commerce [5], and even identify hidden terrorist groups according to the activities of criminals and terrorists [6], [7].

Since a great number of social networks may contain millions of nodes and edges, it is extremely difficult to execute complicated reasoning processes in the whole network. A technology that has been used to deal with such matter is dimensionality reduction whose core idea is to search a mapping function and convert nodes in the graph into different latent representations with a lower dimension. Afterwards, these representations may be used as the characteristics of common tasks in the graph, such as classification of multiple tags, clustering and link prediction.

In the research on DeepWalk, a network representation learning algorithm, some scholars proposed to use neural network to cope with such online learning approach with restriction on extensibility [8]. Since then, more and more scholars have conducted plenty of work [9]–[11]; Node2vec which is similar to DeepWalk [10], [12] maintains high order proximity by increasing the probability of having subsequent nodes amid random walk within a fixed length to the uttermost. What is different from DeepWalk is that node2vec seeks balance between breadth first (Bfs) and depth first (Dfs) graph search through a developed flexible biased random walk, so as to embed more information with higher quality than that of DeepWalk. By choosing the appropriate balance may enable node2vec to maintain the community structure and structural equivalence between nodes. Ozcan A proposed a new multi-variable algorithm for link prediction of evolved heterogeneous networks based on non-linear autoregressive neural networks by outer inputs, and experiments on various

The associate editor coordinating the review of this article and approving it for publication was Eyuphan Bulut.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

data sets show that the accuracy of this algorithm is greater than that of previous algorithms which adopt a single variable time series in distinct system for link prediction [13], [14]. These methods which use neural network have been proven to be highly extensible with high performance, producing desirable results in classification of large networks and link prediction tasks.

Despite success, Node2vec has inherent weaknesses. First of all, it is a local approach which is confined to the surrounding structure of nodes. Node2vec [10] employs short random walk to explore local neighborhood of nodes. Such attention to local structure implicitly ignores the global relationship of long distance, and the learned representation may not be able to reveal the important global structure model. Second, it relies on non-convex optimization objectives which are solved by means of stochastic gradient descent method [15], [16]. Under the circumstance when network prior information falls short, implicit representation is always initialized with a stochastic number or zero. Using such initialization plan plus the application of stochastic gradient descent (SGD) in Node2vec algorithm to optimize the objective function, the algorithm is updated frequently, which is likely to cause vibration at the saddle point or local minimum point, thus making it possible to fall into local minimum of the range. In the meantime, SGD adopts the constant emulating rate for all parameter updates. If the network data is sparse, we would rather hope that the method can change learning rate by self-adaptive behavior, so as to conduct greater updates on features with a lower frequency of appearance [17]. Much as some scholars have put forward a new representation learning paradigm HARP algorithm to address the problems of Node2vec including ignorance of the network global structure and initialization [6], they fail to improve the objective function optimizer SGD or use adaptive learning algorithm which demonstrates better optimization effects. As a result, it is still possible to fall into local minimum.

Based on the aforementioned problems, in this paper, we will first divide the nodes and links of original network graph into a set of smaller layered graphs based on merging them according to HARP, a hierarchical representation learning algorithm proposed by Chen H et al. Then, we adopt the adaptive learning algorithm Adaptive Moment Estimation (Adam) to replace the optimizer SGD in Node2vec, utilize the improved Node2vec to extract features continuously, so as to realize final link prediction. Finally, we conduct link prediction in Twitter and Facebook datasets, and respectively compare the differences of similar link prediction algorithms of nodes and the methods of feature extraction with machine learning from the algorithm in this paper.

II. RELEVANT STUDIES

A. VARIATIONAL GRAPH AUTO-ENCODER (VGAE)

N. Kipf evaluated the performance of variational auto-encoder in graph embedding tasks [18], [19]. The method adopts a graph convolutional network (GCN) encoder and a scalar product decoder, relies GCN to emulate the high-order dependency between each node with the adjacency matrix as input. VGAE is used with the identity matrix for the feature space. Their work shows that compared to most existing models for unsupervised learning which are mostly applied to graph structure data and link prediction, using variational graph auto-encoder can naturally combine node features and has significantly improved the prediction performance of many benchmark datasets [18], [19].

B. ADAPTIVE MOMENT ESTIMATION (Adam)

Adam (Adaptive Moment Estimation) is a stochastic objective function optimization algorithm based on low-order moment adaptive estimation proposed by Kingma and Ba [20]. This algorithm, in essence, is an RMSprop with the item of Momentum. It automatically modifies the learning rate of parameters by utilizing the first-order and the second-order moment evaluations of the gradient. The optimization idea of Adam is that after offset correction, any iterative learning rate is within a certain scope. It can bring more stable parameters than before. Combining Adagrad's ability to deal with sparse gradients and RMSprop's ability to cope with non-stationary targets, Adam has a weaker demand for memory and calculates different adaptive learning rates of different parameters. It can be used for great majority non-convex optimization objective functions, big datasets and high-dimensional spaces, being similar to Momentum:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1)\, g_t \\ v_t &= \beta_2 v_{t-1} + (1-\beta_2)\, g_t^2 \end{aligned} \tag{1}$$

where, m_t, v_t are respectively the evaluations of the first-order moment (mean value) and the second-order moment (uncertain variance) of the gradient. When m_t and v_t are initialized to the vector of 0, P. Kingma and L. Ba observed that they were biased towards zero, especially in the initial time step, or when the decay rate is very low (that is, β_1 and β_2 are close to 1).

Therefore, they compute the first-order moment and second-order moment evaluation of the offset correction in order to offset these biases

$$\begin{aligned} \hat{m}_t &= \frac{m_t}{1-\beta_1^t} \\ \hat{v}_t &= \frac{v_t}{1-\beta_2^t} \end{aligned} \tag{2}$$

Then, the update rules of Adam are concluded:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon}\hat{m}_t \tag{3}$$

where, the default set value suggested by the hyper parameter is that $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. The experiments have shown that Adam works well in practice and has advantages over other adaptive learning method algorithms [17].

C. IMPROVED Node2vec ALGORITHM

Similar to DeepWalk [10], [12], Node2vec depends on direct encoding and uses an inner product-based decoder. However, Node2vec does not attempt to decipher a constant deterministic distance metrics; instead, it optimizes the embedding in order to encode statistics with random walk. The basic concept of Node2vec is to study embedding so that (roughly):

$$\mathrm{DEC}\left(\mathbf{z}_i, \mathbf{z}_j\right) \cong \frac{e^{\mathbf{z}_i^\top \mathbf{z}_j}}{\sum\limits_{v_k \in \mathcal{V}} e^{\mathbf{z}_i^\top \mathbf{z}_k}} \approx p_{\mathcal{G},T}\left(v_j|v_i\right) \tag{4}$$

where, $\mathrm{DEC}\left(\mathbf{z}_i, \mathbf{z}_j\right)$ is the decoded (i.e., estimated) proximity value, $p_{\mathcal{G},T}\left(v_j|v_i\right)$ is the probability of visit to v_i in a fixed-length random walk T beginning at v_j. T is usually defined within the range of $T \in \{2, \ldots, 10\}$. $p_{\mathcal{G},T}\left(v_j|v_i\right)$ is both random and asymmetrical.

More precisely, Node2vec tries to make the following cross entropy loss which is introduced below to a minimum:

$$\mathcal{L} = \sum_{(v_i, v_j) \in \mathcal{D}} -\log\left(\mathrm{DEC}\left(\mathbf{z}_i, \mathbf{z}_j\right)\right) \tag{5}$$

The training set under such circumstance is created by sampling random walks starting from one node (i.e., the N pairs of v_i at each node are sampled from the distribution of $\left(v_i, v_j\right) \sim p_{\mathcal{G},T}\left(v_j|v_j\right)$)). Yet, it is extremely expensive to directly assess the loss of such cross entropy given the computing costs, especially for $O(|\mathcal{D}||\mathcal{V}|)$ because evaluating the denominator of the equation (4) has time complexity $O(|\mathcal{V}|)$. Accordingly, Node2vec and DeepWalk use different optimization and approximation methods to calculate the cross entropy loss in equation (5). DeepWalk adopts ''hierarchical softmax technology'' to calculate normalization factors and uses a binary tree structure to speed up such calculation [8]. On the contrary, Node2vec employs the ''negative sampling'' approximation equation (5): it uses a random set of ''negative samples'' [10] to approximate the normalization factor instead of normalizing the entire set of vertices.

In addition to the differences mentioned above, the core distinguishes between Node2vec and DeepWalk is that Node2vec proposes an agile definition of random walk, and DeepWalk prefers simple unbiased random walk. Especially, Node2vec uses two random walk hyperparameters named p and q, which tend to random walk. The hyper parameter p controls the possibility of revisiting the node immediately upon walking, and g controls the possibility of walking to revisit the node in single neighborhood. Because of using these hyper parameters, Node2vec is capable of interpolating smoothly between walks that are more similar to BFs or DFs. Grover *et al.* proved that adjusting p and q allows the Node2vec to strike a balance between the emphasis on the learning embedding of community structures or emphasis on the role embedding of local structures [10]. Nonetheless, Node2vec also has inherent weaknesses. As it relies on non-convex optimization objectives which are solved by means of stochastic gradient descent method (SGD) [16], [21], under the circumstance when network prior information falls short, implicit representation is always initialized as a random number or zero. By using the initialization plan plus the application of SGD in Node2vec algorithm to optimize the objective function, the algorithm is updated frequently, which is likely to cause vibration at the saddle point or local minimum point, thus making it possible to fall into local minimum of the range. In the meantime, SGD adopts the constant learning rate for all parameter updates. If the network dataset is sparse, we would rather hope that the learning rate can be adaptively adjusted, so as to conduct greater updates on features with a lower frequency of appearance [17]. Meanwhile, the machine learning algorithm Adam can adopt different adaptive learning rates for different parameters, being unlikely to get into local minimum, and thus suitable for most non-convex optimization objective functions, big datasets and high-dimensional spaces. Concerning these merits, we replace the SGD algorithm in Node2vec which is used to solve non-convex optimization objectives with Adam algorithm, thus enhancing Node2vec in the hope of minimizing the influences caused by SGD data initialization and non-adaptive learning rate problems of Node2vec.

Algorithm 1 The New Node2Vec Algorithm

LearnFeatures (Graph $G = (V, E, W)$, Dimensions d, Walks per node r, Walk length l, Context size k, Return p, In-out q)
π = PreprocessModifiedWeights(G, p, q)
$G' = (V, E, \pi)$
Initialize *walks* to Empty
for *iter* = 1 **to** r **do**
  for all nodes $u \in V$ do
  walk = node2vecWalk $\left(G', u, l\right)$
  Append *walk to walks*
f = **Adaptive Moment Estimation** (k, d, *walks*)
return f

node2vecWalk (Graph $G' = (V, E, T)$, Start node u, Length l)
Inititalize *walk* to [u]
for *walk_iter* = 1 **to** l **do**
curr = *walk*[−1]
V_{urr} = GetNeighbors(*curr*, G')
s = AliasSample(V_{curr}, π)
Append s to *walk*
return *walk*

Algorithm1 introduces the steps of the improved algorithm Node2vec in the form of pseudo-random code. It is noteworthy that the implicit bias exists because of the selection of the starting node in arbitrary random walks. Therefore, the algorithm learns the representation of all nodes and counteracts such bias by simulating a random walk with a constant length of l staring from each node. At each step of walking, sampling is performed according to the transition probability π_{vx}. The transition probability π_{vx} of the second-order Markov chain

Algorithm 2 HARP(G, *Embed*())

Input:
graph $G = (V, E)$
arbitrary graph embedding algorithm EMBED()
Output: matrix of vertex representations $\Phi \in \mathrm{R}^{|V|\times d}$

1: $G_0, G_1, \cdots, G_L \leftarrow$ GraphCoarsening(G)
2: Initialize $\Phi' G_L$, by assigning zeros
3: $\Phi G_L \leftarrow \text{EMBED}\left(G_L, \Phi' G_L\right)$
4: **for** $i = \mathrm{L} - 1$ to 0 **do**
5: $\quad \Phi' G_i \leftarrow$ PROLONGATE $(\Phi G_i, G_{i+1}, G_i)$
6: $\quad \Phi G_i \leftarrow \text{EMBED}\left(G_i, \Phi' G_i\right)$
7: **end for**
8: **return** ΦG_0

Algorithm 3 GraphCoarsening(G)

Input: graph $G = (V, E)$
Output: Series of Coarsened Graphs $G_0, G_1, \cdots, G_L$

1: $L \leftarrow 0$
2: $G_0 \leftarrow G$
3: **while** $|V_L| \geq$ *threshold* **do**
4: $\quad L \leftarrow L + 1$
5: $\quad G_L \leftarrow$ EDGECOLLAPSE(STARCOLLAPSE(G))
6: **end while**
7: **return** $G_0, G_1, \cdots, G_L$

can be calculated in advance, so that the node sampling of the simulated random walk can be effectively completed by using the alias sampling in the $O(1)$ time. At the third phase of node2vec, the preprocessing of the transition probability is carried out and calculated according to the sequence, the random walk is simulated, and the optimization of Adam is utilized. Each phase is parallelizable and asynchronous, which is conducive to the overall scalability of node2vec.

D. HIERARCHICAL NETWORK REPRESENTATION LEARNING ALGORITHM (HARP)

Chen et al. found in their research that the literature on graph representation learning aims to optimize non-convex functions. Since there is no priori graph information, these methods are prone to lousy local minima due to unfavorable initialization. In addition, these methods are primarily intended to maintain local proximity in the graph but they ignore the global structure. To that end, Chen et al. put forward a ''meta-strategy'' called HARP, which is a new paradigm for learning low-dimensional embedding of graph nodes, being used to improve various random walk approaches via graph preprocessing steps. Algorithm 2 and Algorithm 3 shows the steps of the HARP by pseudo-random code [6].

1) Graph coarsening (row 1 in Algorithm 2): The graph G is given, and the hierarchical structure of continuous smaller graphs $G_0, G_1, \cdots, G_L$ is constructed with the graph coarsening algorithm, in which $G_0 = G$. The coarser graph

Algorithm 4 HARP(G, *Newnode2vec*())

Input:
graph $G = (V, E)$
New node2vec algorithm
NEWNODE2VEC()

Output: matrix of vertex representations $\Phi \in \mathrm{R}^{|V|\times d}$
1: $G_0, G_1, \cdots, G_L \leftarrow$ *GraphCoarsening*(G)
2: Initialize $\Phi' G_L$, by assigning zeros
3: $\Phi G_L \leftarrow NEWNODE2VEC\left(G_L, \Phi' G_L\right)$
4: **for** $i = \mathrm{L} - 1$ to 0 **do**
5: $\quad \Phi' G_i \leftarrow PROLONGATE\ (\Phi G_i, G_{i+1}, G_i)$
6: $\quad \Phi G_i \leftarrow NEWNODE2VEC\left(G_i, \Phi' G_i\right)$
7: **end for**
8: **return** ΦG_0

maintains the global structure of the original graph but the number of nodes and edges has obviously decreased.

2) Graph embedding in the coarsened graph (row 2 to 3 in Algorithm 2): The provided graph embedding algorithm is used to gain graph node embedding on the coarsened graph $\mathrm{G_L}$. Since the size of $\mathrm{G_L}$ is often small, it is more likely to obtain graphical representation with high quality.

3) Graph representation extension and refinement (row 4 to 7 in Algorithm 2): The graph representation is iteratively extended and refined from the coarsened graph to the refined graph. For different graph $\mathrm{G_i}$, we extend the graph representation of $\mathrm{G_{i+1}}$ as initial embedding Φ'_{G_i}. Then, the embedded learning algorithm *Embed* () is applied to $\left(G_i, \Phi'_{G_i}\right)$ to further refine Φ'_{G_i}, so as to achieve precise embedding of Φ'_{G_i}.

4) Node embedding of the original graph (row 8): Return Φ_{G_i} to complete node embedding of the original graph.

III. INTEGRATION OF IMPROVED Node2vec AND HARP LINK PREDICTION

Given that Node2vec is a local approach which is limited to the structure around the node, it uses short random walk to find the local neighborhood of nodes. Such attention to local structure implicitly overlooks the long-distance relationship in whole network, and the studied representation may be unable to reveal the important global structure model [10]. Meanwhile, the ''meta-strategy'' HARP proposed by Chen et al. is a new paradigm for learning low-dimensional embedding of graph nodes, being used to enhance varied random walk methods through graph preprocessing steps. The algorithm incorporates the global structure relationship of the network, which can compensate for the deficiency of Node2vec in this regard. Hence, this paper names the algorithm HARP that combines the improved Node2vec and HARP (NewN2V).

The description framework of such algorithm is displayed in Algorithm 4. Row 1 divides the original graph into multilayer small graphs in form of recursive coarse graining; row 2 and row 3 list the initialization parameters $\Phi' G_L$, and G_L

TABLE 1. An overview of the four social network datasets.

Dataset	Number of Nodes	Number of Edges	Average Degree	Density
Twitter	81306	1768149	43.493	0.0002
Facebook1	4039	88234	43.691	0.0108
Facebook2	1046	27794	53.143	0.0509
Facebook3	546	5360	19.562	0.0358

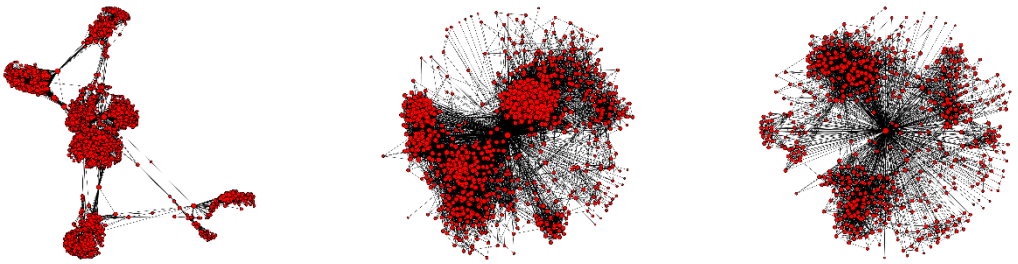


FIGURE 1. Network diagrams of three Facebook datasets.

represents the last smallest graph being divided, with the node embedding being obtained directly with the improved Node2vec; row 4 to row 7 are the combination and extension of embedding results from iterative comparison of the higher-level (larger) graphs and the lower-level (smaller) graphs. Then it's regarded as the embedding parameter input of larger graphs for embedding. In the end, row 8 returns the embedding results of the original graph.

IV. EXPERIMENT AND RESULT ANALYSIS

A. EXPERIMENTAL DATA

In this paper, the data of four real social media sites were chosen for testing, respectively:

1) 3 Facebook datasets; (https://snap.stanford.edu/data/egonets-Facebook.html)

2) Twitter dataset;
(https://snap.stanford.edu/data/egonets-Twitter.html)

Table 1 describes the basic parameters of four social networks, and it concludes the number of nodes, number of edges in the networks, average degree and density of each network.

Figure 1 shows the visualization of 3 Facebook datasets by NetworkX and MatplotLib which shows the link relationships for all nodes in each network. Since the number of nodes and the number of edges in the Twitter dataset are quite huge, they are not visually processed.

B. INDEXES OF MEASUREMENT

We choose two traditional measurement indexes that are extensively adopted in the issue of link prediction, that is, area under Roc curve (AUC) and precision. The definition of AUC is as follows [22]:

$$AUC = \frac{S_0 - n_0 (n_0 + 1)/2}{n_0 n_1} \tag{6}$$

where, n_0 and n_1 respectively represent the number of positive and negative class links. $S_0 = \sum r_i$. In this equation, r_i indicates the rank of the i positive class link reversely sorted by similarity index. $AUC \in [0, 1]$. The larger the AUC is, the higher the prediction accuracy of the algorithm is.

The index precision is defined as the proportion of links which are actually positive class among all links that are predicted as positive class and that are forecast as positive class, being expressed as follows [23]:

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

where, TP represents a link that is actually a positive class and is predicted correctly, and FP implies a link that is actually a negative class and is predicted wrongly. $Precision \in [0, 1]$. The higher the precision, the more accurate the algorithm used.

C. BENCHMARK METHODS

To obtain the effectiveness and prediction accuracy of the algorithm, this paper selects the following link prediction algorithms as the control methods.

1) Similarity link prediction algorithms, including Adamic-Adar (AA), Jaccard Coefficient (JC) and Preferential Attachment (PA);

2) Machine learning algorithms, including Variational Graph Auto-Encoders (VGAE) and Node2Vec.

Now, following are the physical meanings and equations of AA, JC and PA.

$$s_{AA} = \sum_{x \in \Gamma(i) \cap \Gamma(j)} \frac{1}{\log k_x} \tag{8}$$

$$s_{JC} = \frac{|\Gamma(i) \cap \Gamma(j)|}{|\Gamma(i) \cup \Gamma(j)|} \tag{9}$$

$$s_{PA} = k_x \cdot k_y \tag{10}$$

TABLE 2. Prediction results of each algorithm on 4 datasets.

Link Prediction algorithm	Twitter		Facebook1		Facebook2		Facebook3	
	AUC	P	AUC	P	AUC	P	AUC	P
HARP(NewN2V)	**0.955**	**0.963**	**0.993**	**0.988**	0.927	0.910	**0.903**	**0.886**
Node2Vec	0.896	0.907	0.969	0.951	0.871	0.881	0.853	0.862
VGAE	0.895	0.898	0.795	0.806	**0.929**	**0.938**	0.859	0.885
AA	0.897	0.897	0.898	0.942	0.878	0.913	0.734	0.829
JC	0.897	0.895	0.901	0.934	0.856	0.839	0.699	0.708
PA	0.891	0.894	0.831	0.846	0.809	0.837	0.760	0.788

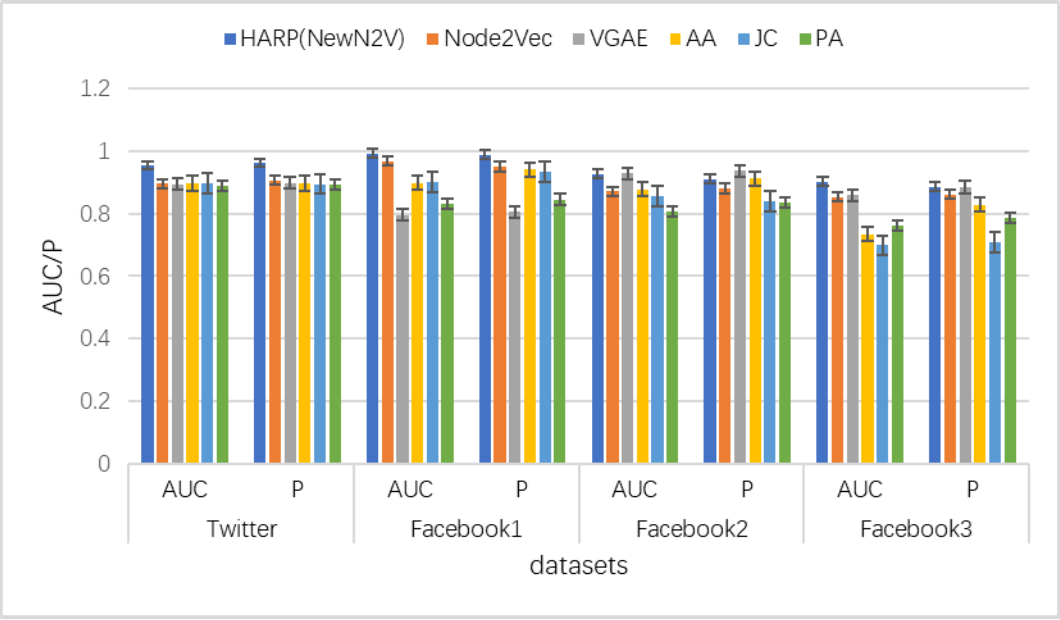


FIGURE 2. Diagram of link prediction results of each algorithm on 4 data sets.

AA believes that the links between two neighbor nodes of a node with small degree are easier to exist than those of a node with large degree. It's easy to understand, for example, two fans of a celebrity are less likely to know each other, but if two people follow a person with very few fans at the same time, they probably have the same interests or social circle. JC considers that the probability of link existence is proportional to the number of neighbors of two nodes. In Twitter, for example, if two users have exactly the same concerns, they are likely to have some kind of connection. Different from AA and JC, some studies show that the rate of an edge connected to a node is proportional to the degree of this node. Therefore, PA considers that the probability of a new edge connecting two nodes is related to the degree of these nodes at the same time. Generally speaking, two famous people probably know each other, but two ordinary people probably don't.

D. EXPERIMENTAL RESULTS

The algorithm HARP (NewN2V) proposed in this paper was tested on the four datasets shown in Table 1. In order to present the prediction results more accurately, the independent experiment was repeated on all datasets for 60 times, and the mean value of the AUC and precision (P) of these 60 experiments was calculated as the eventual outcome of link prediction. Table 2 provides the prediction results of each algorithm on 4 datasets; Figure 2 vividly displays the prediction results of each algorithm on 4 datasets; Table 3 lists the degree of improvement in AUC indicators and precision of the proposed algorithms regarding 4 datasets compared with the Node2Vec algorithm.

It is not hard to see in Table 2 and Fig.2 that in four datasets, the HARP (NewN2V) algorithm put forward in this paper demonstrates both greater AUC indicators and precision compared with Node2Vec. In addition, except Facebook 2 dataset, the AUC and P of HARP(NewN2V) are the highest in all datasets. And the performance of HARP(NewN2V) is almost the same as the optimal accuracy in Facebook 2 dataset. As shown in Table 3, for the HARP (NewN2V) algorithm which is tested in four datasets, the AUC index increased by 5.34 on average while the precision indicator P was raised by 3.89% compared with Node2Vec.

TABLE 3. AUC and P indicators improvement ratio %.

Datasets	HARP(NewN2V)/Node2Vec improvement ratio %	
	AUC	P
Twitter	↑6.58	↑6.17
Facebook1	↑2.47	↑3.70
Facebook2	↑6.43	↑2.90
Facebook3	↑5.86	↑2.78
Mean	↑5.34	↑3.89

It can be noted from the analysis of link prediction results mentioned above that the combination of the improved Node2Vec algorithm and the ''meta-strategy'' HARP plays a positive role in improving the AUC indicator and precision of link prediction, which proves the feasibility and superiority of the proposed algorithm.

V. CONCLUSION

This paper attempts to tackle the problems of Node2Vec algorithm in the process of optimizing non-convex functions, that is, the likeliness to run into the lousy local minimum value with unfavorable initialization due to lack of prior network information, and the inability of its optimizer SGD to conduct adaptive adjustment in the learning rate, which does not help with the treatment of sparse network. Besides, Node2Vec is aimed to retain local neighborhood of the graph, implicitly ignoring the global structure of the graph. Hence, this paper proposed to first use the machine learning optimizer Adam to replace SGD so as to optimize the Node2Vec algorithm. Meanwhile, HARP can merge input network graphs into smaller graphs yet with similar structures in a recursive manner to capture the global structure of input graphs, then study the graph representation of a set of smaller graphs and thus obtain a favorable initialization solution for input network graphs. Furthermore, such multi-level paradigm is combined with the improved Node2Vec algorithm to establish a link prediction model which is based on machine learning and HARP. The results of experiments conducted on real social network datasets show that the proposed algorithm is feasible and effective with certain superiority.

REFERENCES

[1] D. Liben-Nowell and J. Kleinberg, ''The link-prediction problem for social networks,'' *J. Amer. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, 2007.
[2] L. M. Aiello, A. Barrat, R. Schifanella, C. Cattuto, B. Markines, and F. Menczer, ''Friendship prediction and homophily in social media,'' *ACM Trans. Web*, vol. 6, no. 2, p. 9, May 2012.
[3] P. Wang, B. W. Xu, Y. R. Wu, and X. Y. Zhou, ''Link prediction in social networks: The state-of-the-art,'' *Sci. China Inf. Sci.*, vol. 58, no. 1, pp. 1–38, Jan. 2015.
[4] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, ''Mixed membership stochastic blockmodels,'' *J. Mach. Learn. Res.*, vol. 9, no. 5, pp. 1981–2014, Jun. 2008.
[5] M. Bressan, F. Chierichetti, R. Kumar, S. Leucci, and A. Panconesi, ''Counting graphlets: Space vs time,'' in *Proc. 10th Acm Int. Conf. Web Search Data Mining*, Feb. 2017, vol. 6, no. 2, pp. 557–566.
[6] H. Chen, B. Perozzi, Y. Hu, and S. Skiena, ''Harp: Hierarchical representation learning for networks,'' in *Proc. 32nd AAAI Conf. Artif. Intell.*, Apr. 2018, pp. 1–8.
[7] P. Pezeshkpour, Y. Tian, and S. Singh, ''Investigating robustness and interpretability of link prediction via adversarial modifications,'' May 2019, *arXiv:1905.00563*. [Online]. Available: https://arxiv.org/abs/1905.00563
[8] B. Perozzi, R. Al-Rfou, and S. Skiena, ''DeepWalk: Online learning of social representations,'' in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Dec. 2014, pp. 701–710.
[9] S. Cao, L. Wei, and Q. Xu, ''Grarep: Learning graph representations with global structural information,'' in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 891–900.
[10] A. Grover and J. Leskovec, ''node2vec: Scalable feature learning for networks,'' in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, vol. 1, no. 2, pp. 855–864.
[11] B. Perozzi, V. Kulkarni, H. Chen, and S. Skiena, ''Don't Walk, Skip!: Online learning of multi-scale network embeddings,'' *Comput. Sci.*, vol. 24, no. 1, pp. 1–4, Jun. 2017.
[12] W. L. Hamilton, R. Ying, and J. Leskovec, ''Representation learning on graphs: Methods and applications,'' *Bull. IEEE Comput. Soc. Tech. Committee Data Eng.*, vol. 3, no. 4, pp. 88–98, Apr. 2017.
[13] A. Ozcan and S. G. Oguducu, ''Link prediction in evolving heterogeneous networks using the NARX neural networks,'' *Knowl. Inf. Syst.*, vol. 55, no. 2, pp. 333–360, 2018.
[14] A. Ozcan and S. G. Oguducu, ''Multivariate time series link prediction for evolving heterogeneous network,'' *Int. J. Inf. Technol. Decis. Making*, vol. 18, no. 1, pp. 241–286, 2019. doi: 10.1142/S0219622018500530.
[15] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, ''Distributed representations of words and phrases and their compositionality,'' in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2013, vol. 26, no. 4, pp. 3111–3119.
[16] Y. Goldberg and O. Levy, ''word2vec Explained: Deriving Mikolov *et al.*'s negative-sampling word-embedding method,'' vol. 2, nos. 1–2, Feb. 2014, *arXiv:1402.3722*. [Online]. Available: https://arxiv.org/abs/1402.3722
[17] S. Ruder, ''An overview of gradient descent optimization algorithms,'' *Int. J. Numer. Methods Eng.*, vol. 58, no. 7, pp. 1019–1031, Feb. 2016.
[18] T. N. Kipf and M. Welling, ''Variational graph auto-encoders,'' Nov. 2016, *arXiv:1611.07308*. [Online]. Available: https://arxiv.org/abs/1611.07308
[19] T. N. Kipf and M. Welling, ''Semi-supervised classification with graph convolutional networks,'' in *Proc. ICLR*, no. 9, Sep. 2016, pp. 1–14.
[20] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' Dec. 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980
[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ''Efficient estimation of word representations in vector space,'' Jan. 2013, *arXiv:1301.3781*. [Online]. Available: https://arxiv.org/abs/1301.3781
[22] D. J. Hand and R. J. Till, ''A simple generalisation of the area under the ROC curve for multiple class classification problems,'' *Mach. Learn.*, vol. 45, no. 2, pp. 171–186, 2001.
[23] M. Sokolova and G. Lapalme, ''A systematic analysis of performance measures for classification tasks,'' *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009.



HAO SHAO received the B.S. degree in communication engineering from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2017. He is currently pursuing the M.S. degree with the National University of Defense Technology, China. He has published a few research articles in domestic and abroad journals. Some of his research articles have been accepted by international conferences. His current research interests include the areas of network topology, data mining, and machine learning.



LUNWEN WANG received the B.S. and M.S. degrees from the Electronic Engineering Institute of the PLA, Hefei, China, and the Ph.D. degree in communication and information systems from Anhui University, Hefei, China, in 2002. He is currently a Professor and a Ph.D. Supervisor with the National University of Defense Technology. His research interests include neural networks and data mining.



YUFAN JI received the B.S. degree in electronic information engineering from the National University of Defense Technology, China, in 2017, where she is currently pursuing the M.S. degree. Her research interests include machine learning, image recognition, and target tracking.