Journal of Nonlinear and Convex Analysis
Volume 26, Number 6, 2025, 1811–1821

Yokohama Publishers
ISSN 1880-5221 ONLINE JOURNAL

A REPRESENTATION LEARNING-BASED LINK PREDICTION ALGORITHM FOR EVOLVING BIPARTITE NETWORKS

DONGQI WANG, JIE WANG, QIANQIAN GAN, MINGSHUO NIE, AND DONGMING CHEN*

Abstract. Bipartite networks are prevalent in various real-life scenarios. The structures and link relationships evolve over time in evolving bipartite networks (EB networks), making it challenging for traditional meta-path-based representation learning methods to capture the internal knowledge of the two disjoint sets effectively. This study aimed to propose a method to address this issue by constructing network projections for training based on the historical interactions of the EB network. The goal was to learn both the internal information of the two disjoint sets and the information between them in a meta-path-based learning process. The experimental results on real-life datasets showed that the proposed approach significantly outperformed baseline methods in link prediction tasks. The proposed algorithm mostly achieved optimal or suboptimal results, indicating its effectiveness in capturing the underlying knowledge of evolving bipartite networks.

1. Introduction

Networks are powerful data structures representing complex systems that are the foundation of our world. These complex systems are a common thread that runs through the fabric of everyday life, connecting people, places, and things in intricate ways, ranging from wireless sensor networks [4] to road networks [14] and social networks [3,7]. As high-dimensional, non-Euclidean entities, they elude straightforward analysis, necessitating the development of sophisticated techniques for their comprehension. Network representation learning is a transformative solution that has emerged to address this challenge [2]. Various techniques have been developed to harness this learning process, such as the DeepWalk algorithm [10] and its progeny, which have used local context to distill meaningful node representations. The LINE method has approached the problem probabilistically, aiming to preserve network proximity within the embedding space [13]. Other innovative approaches, such as BUDDY [1] and RecNS [16], have also been developed to capture the essence of network structures and enhance the learning process.

Representation learning for evolving bipartite networks (EB networks) extends this concept to dynamic systems where nodes are categorized into two separate

2020 *Mathematics Subject Classification.* 05C82, 06F30.

Key words and phrases. Evolving bipartite network, link prediction, meta-path, network representation learning.

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 2024GFZD03, and in part by the Applied Basic Research Project of Liaoning Province under Grant 2023JH2/101300185, in part by the Key Technologies Research and Development Program of Liaoning Province in China under Grant 021JH1/10400079, in part by the Key Technologies Research and Development Program of Liaoning Province (Grant No.2024JH2/102400072).

*Corresponding author.

sets. This learning process must encapsulate not only the topological and attribute information but also the temporal dynamics, including the chronological evolution of network topologies and the time-varying attributes of nodes. Further, network representation learning helps researchers decipher the structural and behavioral nuances of complex networks, enabling the projection of the original network and facilitating critical inferences such as the prediction of previously unseen or lost connections [5, 8, 11], the inference of node labels [15], and the determination of node community affiliations [12].

The existing meta-path-based representation learning algorithms usually extract information from the network across both temporal and spatial dimensions for learning. These algorithms, however, often disregard the relational information between network nodes and do not effectively capture the associations within the two distinct node modes. In this study, we proposed an evolving bipartite network representation learning algorithm based on a network projection to address these issues, named LP4EBN. The proposed algorithm solution could simultaneously construct the association between nodes of the same mode and different modes, resulting in more abundant node representations. We conducted link prediction experiments on five real-life datasets to evaluate the effectiveness of the proposed algorithm, realizing that our proposed algorithm achieved better performance on most datasets.

2. Methodology

2.1. ***Algorithm process.*** The training process of the LP4EBN algorithm, depicted in Figure 1, consists of several steps. First, a batch of interaction data, comprising EB network data within a specific time interval, is input into the algorithm. The algorithm then integrates the historical interaction records of each node to extract the behavior attribute information of the node, which is used for node attribute modeling. Simultaneously, the algorithm generates two projection networks based on the node modes of the original EB network and their corresponding mode internal interaction records. The algorithm aggregates information from both projection networks to obtain the neighborhood information of the target node. Additionally, it collects relevant evolving node behavior attributes from the original EB network for the training. Finally, a recurrent neural network (RNN) is used to learn the association of the node within the time interval. The final node representation is learned by combining the historical information of the node, the predicted association information, and the neighborhood information of two-mode projection networks. The algorithm performance is tested on downstream link prediction tasks.

The RNN plays a central role in learning the temporal associations between nodes. It takes the initial embedding vectors of each node as input, encapsulating the historical behavioral attributes of the nodes. Additionally, the embedding vectors of neighbor nodes, weighted by their interaction intensity with the target node over time, are included to reflect the strength of temporal connections. Specifically, the timestamp information is incorporated as part of the sequence data input into the RNN, ensuring that the model captures the temporal dependencies in the evolution of the network. The output of the RNN is the updated node representation, which not only includes the static features of the nodes but also integrates their dynamic behavioral patterns over time.

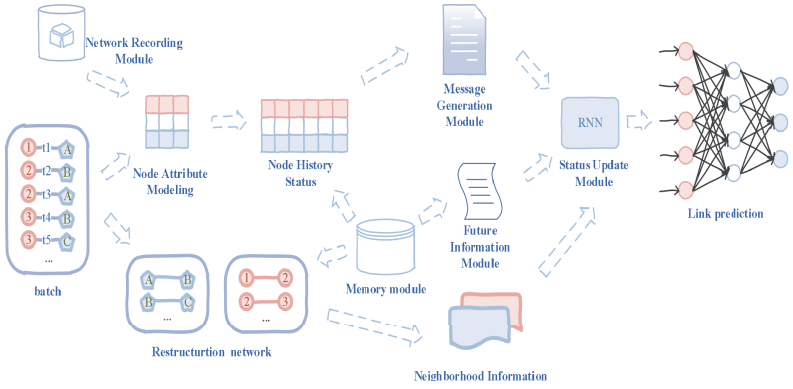

FIGURE 1. Overall framework of LP4EBN

In this study, we specifically considered the impact of the time set T on the network dynamics. Each point in time t was associated with the edges in the network, mapped to the edge attributes in the form of timestamps. This implied that we recorded not only the nodes it connected but also the specific time when the edge was created or the interaction occurred for every edge in the network. This mapping allowed us to capture and analyze the evolution of the network over time. The temporal information was used to weigh the contributions of neighbor nodes and influence the update of node representations in every algorithm step.

2.2. ***Construction of the projection network.*** Let us assume the existence of a bipartite evolving network $G = (X, Y, E, T)$, where the sets X and Y represent two disjoint node sets. Nodes within set X do not form edges among themselves and only interact with nodes in set Y. Similarly, nodes within set Y do not form edges among themselves and only interact with nodes in set X. E denotes the set of edges, recording the interaction information between nodes, and T represents the timestamp information.

An evolving bipartite network is a special kind of evolving network. Existing studies mostly use meta-path-based learning node representation for this type of network. However, meta-paths generally need to be manually defined, and the edges of the meta-path are linked by two different modes. Nodes and the information associated with nodes of the same mode are ignored. This study mapped the evolving bipartite network to obtain two projection networks with a single node mode. The evolving bipartite network $G = (X, Y, E, T)$ was mapped on the node set X and Y, respectively, to obtain the projection network$G_X = (X, E_X, T_X), G_Y = (Y, E_Y, T_Y)$ so as to learn the associated information between nodes of the same mode.

The construction process of the projection network in this study was inspired by the mapping process from a bipartite graph to a monopartite graph. Taking the evolving bipartite network shown in Figure 2 as an example, the mutual information I of the evolving bipartite network is expressed as follows:$I = \{(1, A, t1), (1, B, t2), \ldots, (4, B, t9), (4, C, t10)\}$, according to the bipartite graph mapping to obtain a single point. The idea of the graph is that interaction information $\{(1, A, t1), (1, B, t2)\}(t1<t2)$ exists in the bipartite graph, that is, node A and node

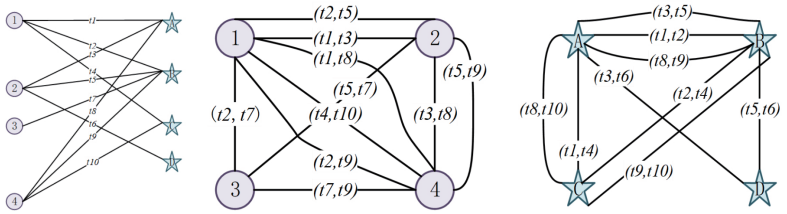

FIGURE 2. Construction of the projection network.

B have interacted with node 1 successively. An edge is present between node A and node B in a monopartite graph. Based on the aforementioned mapping process, an edge exists between node 1 and node 2 due to an interaction$\{(1, A, t1), (2, A, t3)\}$ ($t1$<$t3$) in the bipartite graph. According to the aforementioned rules, the projection networks derived from mapping the evolving bipartite network onto sets X and Y are shown in Figure 2, respectively.

Considering timestamp information on the edges of the evolving bipartite network, this study preserved the timestamp information of the two interactions involved in the network projection from the evolving bipartite network mapping to ensure the integrity of the timing information. For example, if a pair of interaction information $\{(xi, Y, ti), (xj, Y, tj)\}(ti<tj)$ is present in the evolving bipartite network, then the interaction information saved on the projection network is recorded as $\{[xi, xj, (ti, tj)]\}$. This shows that the timestamp information on the edges of the projection network is stored in the form of tuples $(ti, tj)(ti<tj)$. When learning the node representation on the projection network later, the timestamp information contained in this tuple will be used to calculate the corresponding weight of each neighbor node in the projection network.

2.3. ***Calculation of weight.*** Our LP4EBN algorithm used two projection networks to capture the historical interaction records of the evolving bipartite network for a given batch of interaction information. The interaction information before and after projection is presented in Figure 3. The timestamp information of the two interactions related to the mapping process was maintained in the projection networks to preserve the timing information in the original EB network. The algorithm stored the two interaction timestamps in tuple form and used this information to calculate the influence of each neighbor of the target node in the corresponding projection network, assigning specific weight values accordingly.

Two projection networks can be obtained by mapping in the aforementioned manner for an evolving bipartite network at a given time t. Let us assume that the interaction information on the projection network is $[x_i, x_k, (t_i, t_k)](t_i \leq t_k \leq t)$, where x_i represents the target node, x_k represents the neighbors of the target node on the projection network nodes, and $t_i$and t_k represent the timestamp information of the two interactions involved. The LP4EBN algorithm needs to first calculate the time interval $\triangle t_k$ between two interactions, and the specific calculation formula is shown in Equation (2.1).

$$\triangle t_k = t_k - t_i. \tag{2.1}$$

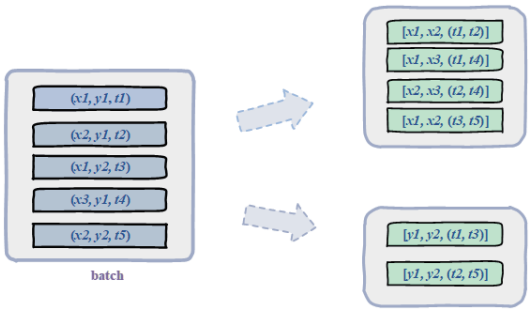

FIGURE 3. Original evolving bipartite network and projection network.

If two nodes interact with the same node at a relatively close time, then the two nodes are more similar, and hence the weight value corresponding to the neighbor node in the projection network is larger. On the contrary, if two nodes interact with the same node and the time interval is large, the similarity between the two nodes is not high and the weight value corresponding to the neighbor node in the projection network is small. Therefore, the corresponding weight is larger for the neighbor node x_k with a smaller time difference $\triangle t_k$. Otherwise, its corresponding weight is smaller. For the neighbor node x_k of the target node in the projection network, the corresponding weight calculation formula is shown in Equation (2.2), where K is the number of neighbors of the target node in the projection network.

$$Weight_1^k = \frac{e^{\frac{1}{\triangle t_k}}}{\sum_{k=1}^{K} e^{\frac{1}{\triangle t_k}}}. \tag{2.2}$$

For the interaction $[x_i, x_k, (t_i, t_k)](t_i \leq t_k \leq t)$ in the projection network, the time information t_k on the connection edge is far from the current time t, that is, the neighbor node is in a long time. If no relevant interaction exists within the target node, then the influence of the neighbor node on the target node and the corresponding weight is small. Conversely, if the time on the edge in the projection network is extremely close to the current time t, it indicates that the interaction has just occurred. Therefore, the target node should learn node representation, and its corresponding weight is larger. Therefore, we should consider the time interval $\triangle t'_k$ between the time information (t_i, t_k) on the projection network and the current moment t. The specific calculation formula is shown in Equation (2.3).

$$\triangle t'_k = t - t_k. \tag{2.3}$$

The closer the interaction time information in the projection network is to the current moment, the smaller the $\triangle t'_k$. Otherwise, the larger the $\triangle t'_k$. According to $\triangle t'_k$, the weight corresponding to the neighbor node x_k in the projection network can be obtained. The weight calculation formula is similar to the previous weight calculation, as shown in Equation (2.4). Among these, K still takes the value of 10

when the algorithm is implemented.

$$Weight_2^k = \frac{e^{\frac{1}{\triangle t_k'}}}{\sum_{k=1}^{K} e^{\frac{1}{\triangle t_k'}}}. \tag{2.4}$$

The weight $weight_1^k$ and $weight_2^k$ of the neighbor node x_k can be obtained based on the time information on the projection network. In this study, the two weights were simply averaged to obtain the neighbor node x_k in the projection network. The final weight value $weight^k$ is calculated using the specific formula shown in Equation (2.5).

$$Weight^k = \frac{\left(weight_1^k + weight_2^k\right)}{2}. \tag{2.5}$$

After obtaining the weight value corresponding to the neighbor node x_k in the projection network, we should read the historical state information of the neighboring node in the storage module and then multiply it by the weight to obtain the influence of the node on the target node. The assigned weights gauge connection strengths and are vital for capturing the dynamic evolution of the network. They reflect the interaction degree between the focal node and its neighbors within a given time frame. These weights allow the network to assign varying influence levels to neighbor nodes based on timestamp data by adjusting the RNN input. Recent interactions contribute more information due to larger weights, whereas earlier ones contribute less. Additionally, weights help balance the impact of different neighbor nodes, preventing model bias from extreme values.

2.4. ***Negative sampling.*** In this study, we presented a novel negative sampling strategy specifically designed for the evolving nature of bipartite networks. Our strategy involved choosing nodes that had historical interactions with node x but were not present in the interaction data of the batch being predicted, for example y, as negative sampling. This selection process necessitated both the predicted interaction data and the existing knowledge of past interactions, thereby converting the training process into a supervised learning task. As the algorithm was tailored for bipartite networks featuring two distinct node types, the interactions between nodes of the same kind were inherently absent. Therefore, when selecting negative samples for node x in set X, only nodes in set Y could be selected. If we chose the nodes of the set X, they must not interact, serving as weak negative samples that were easily judged as negative examples. In this study, we designed a strong negative sample selection strategy. The specific selection method is shown in Equation (2.6).

$$Y_x^{negative} = Y_x^{batch_{i-1}} - Y_x^{batch_i} = Y_x^{batch_{i-1}} - (Y_x^{batch_i} \cap Y_x^{batch_{i-1}}), \tag{2.6}$$

$$Y_x^{negative} = Y^{all} - Y_x^{batch_{i-1}} - Y_x^{batch_i} \tag{2.7}$$

where $Y_x^{batch_{i-1}}$ represents the set of nodes that have interacted with node x obtained according to known historical interaction information. $Y_x^{batch_i}$ represents the interaction information currently input to the model for link prediction. The node

set that will interact with node x. $Y_x^{negative}$ means the negative sample node set formed by the selected node y constitutes a negative sample with node x. The subtraction in the formula means the subtraction between sets, and $\cap$ means the intersection of sets. When the number of selectable negative sample nodes is less than the size of the negative sample node set $Y_x^{negative}$, that is, when the number of nodes obtained in Equation (2.6) is insufficient, negative samples are randomly selected from the set shown in Equation (2.7). Y^{all} represents nodes in set Y that are currently known to have occurred in all interactions.

3. Experiments

3.1. ***Datasets.*** As shown in Table 1, experiments were carried out on five real-life representative datasets to evaluate the proposed algorithm. The LP4EBN algorithm was implemented using the same dataset division method and ratio as the baseline algorithm, dividing the data into training, validation, and test sets in a 70:15:15 ratio. The parameter settings for the training process were as follows: the batch size was set to 200, the learning rate was 0.0002, the dropout rate was 0.1, the embedding dimension of the nodes was 100, and the number of neighbors aggregated in the reconstructed network (K) was 10.

Table 1. Dataset-related information

Dataset	Node	Edge	Duration
Wikipedia [6]	9227	157,474	1 month
Reddit [6]	10,984	672,447	1 month
MOOC [6]	7144	411,749	17 months
LastFM [6]	1980	1,293,103	1 month
UCI [9]	1899	59,835	196 days

3.2. ***Experimental results and analysis.*** Table 2 shows that our algorithm could achieve good results in link prediction tasks in most cases. The algorithm did not perform well on the Reddit dataset, which was because the Reddit dataset represented the situation of users posting in forums. Active users posting in the same forum section may have polarized opinions. In summary, the LP4EBN algorithm achieved good link prediction results on most evolving bipartite network datasets, proving that the introduction of the projection network to learn the association relationship between nodes of the same mode was beneficial to the node representation on the evolving bipartite network.

Some baseline algorithms can only perform the transductive link prediction task due to the limitation of the application range of the baseline algorithm. Therefore, the aforementioned comparative experiments were all performed on the transductive link prediction task. However, the LP4EBN algorithm proposed in this study could handle both the transductive and inductive link prediction tasks. The specific experimental results of the algorithm are shown in Table 3.

We conducted ablation experiments using MOOC and Reddit datasets as examples to examine whether introducing the reconfigured network positively affected node representation learning. The experimental results of the MOOC dataset are

TABLE 2. Comparison of AUC and AP values obtained using each algorithm on the link prediction task

Dataset	JODIE		DyRep		TGAT		CAWN		EdgeBank		LP4EBN	
	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP	AUC	AP
Wikipedia	0.79	0.77	0.79	0.81	0.74	0.76	0.84	**0.89**	0.77	0.71	**0.88**	**0.89**
Reddit	0.79	0.77	0.80	0.79	0.78	0.77	**0.85**	**0.89**	0.77	0.70	0.77	0.79
MOOC	0.77	0.70	**0.80**	0.74	0.61	0.59	0.60	0.66	0.60	0.57	0.79	**0.76**
LastFM	0.69	0.68	0.70	0.71	0.50	0.50	0.40	0.56	0.76	0.69	**0.80**	**0.79**
UCI	0.71	0.62	0.44	0.45	0.57	0.61	0.73	0.79	0.69	0.65	**0.76**	**0.80**

TABLE 3. Performance of LP4EBN in transductive and inductive link prediction tasks

Dataset	AP(Transductive)	AUC(Transductive)	AP(Inductive)	AUC(Inductive)
Wikipedia	0.89	0.88	0.88	0.86
Reddit	0.79	0.77	0.61	0.58
MOOC	0.76	0.79	0.82	0.82
LastFM	0.79	0.80	0.88	0.89
UCI	0.80	0.76	0.77	0.74

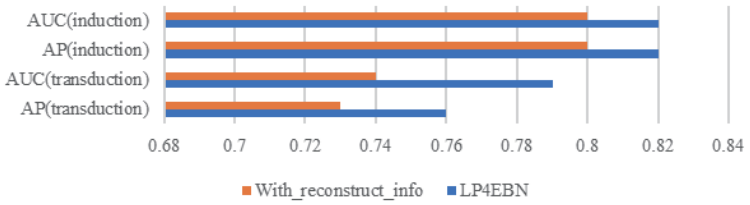

FIGURE 4. Ablation experiments on the MOOC dataset.

shown in Figure 4. In the figure, the method of not introducing the projection network is recorded as Without_reconstruct_info, and the control group is the complete LP4EBN algorithm.

The results of the ablation experiments showed that the introduction of projection network–related information in the evolving bipartite network helped the algorithm to learn node representation. Whether it was in the transductive link prediction task or the inductive link prediction task, the introduction of the projection network in this study could improve the AP and AUC indicators on the link prediction task to a certain extent.

The projection network is built on historical interactions. Therefore, long-term history is not useful for current representation learning and may even harm it. Thus, when aggregating neighbor node information on the projection network, only K recently interacted neighbors are selected for information aggregation. A parameter sensitivity experiment was conducted using the Wikipedia dataset to examine the impact of the number of aggregated neighbors on node representation learning. The results are presented in Figure 5.

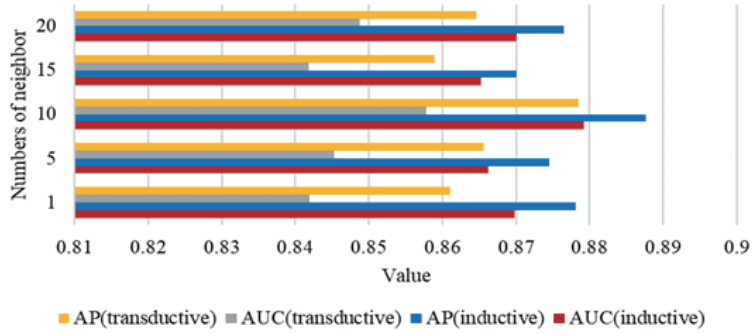

FIGURE 5. Aggregating the influence of different numbers of neighbor node representations on link prediction.

Figure 5 illustrates the impact of aggregating varying numbers of neighbor nodes within the projection network on the outcomes of link prediction. Specifically, incorporating a larger set of neighbor nodes in inductive link prediction scenarios is advantageous for modeling the representations of nodes lacking a historical interaction record. Nonetheless, this aggregation does not inherently improve with an indefinite increase in the number of neighbors. The experiments indicated that the aggregation of representation data from 10 neighbor nodes optimized the link prediction performance.

4. Conclusions

This study developed a novel link prediction algorithm tailored for evolving bipartite networks. The algorithm introduced a unified projection network framework integrating nodes from both modes, coupled with a meta-path learning approach enabling dual feature extraction across the projected and original networks. By capturing node representations that comprehensively reflect network complexity, the model consistently achieved or surpassed state-of-the-art performance in experiments, advancing link prediction research in evolving bipartite networks.

References

[1] B. P. Chamberlain, S. Shirobokov, E. Rossi, F. Frasca, T. Markovich, N. Hammerla, M. M. Bronstein and M. Hansmire, *Graph Neural Networks for Link Prediction with Subgraph Sketching*, in: Proc. 39th International Conference on Machine Learning (ICLR), OpenReview.net, 2023.

[2] P. Cui, X. Wang, J. Pei and W. Zhu, *A survey on network embedding*, IEEE Transactions on Knowledge and Data Engineering **31** (2018), 833–852.

[3] Y. Du, Q. Zhou, J. Luo, X. Li and J. Hu, *Detection of key figures in social networks by combining harmonic modularity with community structure-regulated network embedding*, Information Sciences **570** (2021), 722–743.

[4] S. Huang, A. Liu, S. Zhang, T. Wang and N. N. Xiong, *BD-VTE: A novel baseline data based verifiable trust evaluation scheme for smart network systems*, IEEE Transactions on Network Science and Engineering **8** (2021), 2087–2105.

[5] A. Kumar, S. S. Singh, K. Singh and B. Biswas, *Link prediction techniques, applications, and performance: A survey*, Physica A: Statistical Mechanics and its Applications **553** (2020): 124289.

[6] S. Kumar, X. Zhang, and Leskovec, *Predicting dynamic embedding trajectory in temporal interaction networks*, in: Proc. 25th ACM SIGKDD international conference on Knowledge discovery and data mining, Association for Computing Machinery, 2019, pp. 1269–1278.

[7] J. Li and G. Yang, *Network embedding enhanced intelligent recommendation for online social networks, Future Gener*, Future Generation Computer Systems **119** (2021), 68–76.

[8] L. Lü and T. Zhou, *Link prediction in complex networks: A survey*, Physica A: Statistical Mechanics and Its Applications **390** (2011), 1150–1170.

[9] P. Panzarasa, T. Opsahl and K. Carley, *Patterns and dynamics of users' behavior and interaction: Network analysis of an online community*, Journal of the American Society for Information Science and Technology **60** (2009), 911-932.

[10] B. Perozzi, R. Al-Rfou and S. Skiena, *DeepWalk: Online learning of social representations*, in: Proc. 20th ACM SIGKDD international conference on Knowledge discovery and data mining, Association for Computing Machinery, 2014, pp. 701–710.

[11] Shabaz, Mohammad and U. Garg, *Predicting future diseases based on existing health status using link prediction*, World Journal of Engineering **19** (2022), 29–32.

[12] X. Su, S. Xue, F. Liu, J. Wu, J. Yang, C. Zhou, W. Hu, C. Paris, S. Nepal, D, Jin, Q. Sheng, Yu, S. Philip, *A comprehensive survey on community detection with deep learning*, IEEE Transactions on Neural Networks and Learning Systems **35** (2022), 4682–4702.

[13] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan and Q. Mei, *LINE: Largescale information network embedding*, in: Proc. 24th International Conference on World Wide Web , International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2015, pp. 1067–1077.

[14] C. Wu, M. Hu, R. Jiang and Q. Hao, *Effects of road network structure on the performance of urban traffic systems*, Physica A: Statistical Mechanics and its Applications, **563** (2021): 125361.

[15] S. Xiao, S. Wang, Y. Dai and W. Guo, *Graph neural networks in node classification: survey and evaluation*, Machine Vision and Applications **33** (2022),1–19.

[16] Z. Yang, M. Ding, X. Zou, J. Tang, B. Xu, C. Zhou and H. Yang, *Region or Global a Principle for Negative Sampling in Graph-based Recommendation*, IEEE Transactions on Knowledge and Data Engineering **35** (2023), 6264–6277.

Manuscript received May 19, 2024
revised October 10, 2024

Dongqi Wang
Software College, Northeastern University, China
E-mail address: wangdq@swc.neu.edu.cn

Jie Wang
Software College, Northeastern University, China
E-mail address: 2271453@stu.neu.edu.cn

Qianqian Gan
Software College, Northeastern University, China
E-mail address: 2071273@stu.neu.edu.cn

Mingshuo Nie
Software College, Northeastern University, China
E-mail address: niemingshuo@stumail.neu.edu.cn

D. Chen
Software College, Northeastern University, China
E-mail address: chendm@mail.neu.edu.cn