

Received 23 October 2023, accepted 8 November 2023, date of publication 13 November 2023, date of current version 20 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3332167

RESEARCH ARTICLE

Identification of Key Nodes in Complex Networks Based on Network Representation Learning

HEPING ZHANG¹, SICONG ZHANG¹, XIAOYAO XIE¹, (Member, IEEE),
TAIHUA ZHANG², AND GUOJUN YU¹

¹Key Laboratory of Information and Computing Science Guizhou Province, Guizhou Normal University, Guiyang 550001, China

²School of Mechanical and Electrical Engineering, Guizhou Normal University, Guiyang 550025, China

Corresponding author: Guojun Yu (254579993@qq.com)

This work was supported in part by the National Natural Science Foundation, China, under Grant 72061006; in part by the Science and Technology Planned Project of Guizhou Province, China, under Grant [2023]YB449; and in part by the Academic New Seedling Foundation Project of Guizhou Normal University, China, under Grant Qianshixinmiao-[2021]A30.

ABSTRACT Recently, some research has utilized machine learning methods to identify critical nodes in complex networks. However, existing approaches often lack a comprehensive consideration of network structural features during node feature extraction. Benefiting from the powerful feature extraction capability of network representation learning methods, a simple and effective algorithm for identifying key nodes in complex networks, termed Network Representation Learning and Key Node Identification (NRL_KNI), is proposed. The NRL_KNI algorithm utilizes network embedding techniques for learning node feature representations, followed by clustering and the utilization of quota-based limited sampling to obtain sampled nodes. Subsequently, these sampled nodes are employed to train a regression model for predicting the diffusion capability of unsampled nodes. To rank node influences, a Local Structure Influence Score (LSIS) based on the local structure is introduced to evaluate nodes' final impact. Experimental results on eight real-world datasets demonstrate that the NRL_KNI algorithm generally outperforms traditional centrality methods and network representation learning-based methods in terms of the Jaccard similarity coefficient and Kendall's Tau correlation coefficient evaluation metrics.

INDEX TERMS Complex networks, key node identification, network representation learning, regression model.

I. INTRODUCTION

With the rapid growth of social media platforms, and the widespread dissemination of information [1] has spawned many applications based on social networks. For example, the influence maximization problem aims to identify the most influential key nodes in the social network [2], so that under a specific propagation model (such as the Susceptible-Infectious-Recovered model), the key nodes can influence as many other nodes as possible. Critical node identification in complex networks is also crucial in other scenarios, such as viral marketing, personalized recommendation, information dissemination, etc. Therefore, how to effectively identify key

nodes in a complex network has become a research hotspot in different fields.

Traditional methods for identifying key nodes typically rely on node centrality indices for selection. For instance, degree centrality [3] primarily assesses a node's significance based on the count of its first-order neighbors; betweenness centrality [4] quantifies importance by measuring how frequently a node serves as an intermediary in the shortest paths; PageRank centrality [5] gauges a node's significance by considering both its connections to other nodes and their individual importance scores; K-shell coefficient [6], on the other hand, is a network-centric method for globally classifying the importance of nodes based on structural characteristics; Berahmand et al. [7] introduced a novel semi-local and parameter-free centrality measure for identifying the most influential key nodes. While these methods are

The associate editor coordinating the review of this manuscript and approving it for publication was Dominik Strzalka¹.

straightforward and extensively employed, their drawback lies in their partial utilization of the intricate structural traits of the network.

In the pursuit of more effectively identifying the most influential key nodes, machine learning-based algorithms for key node identification are emerging as promising tools. These approaches can be categorized as either supervised or unsupervised methods. The supervised method generally involves transforming the challenge of identifying key nodes into a regression or classification problem. It leverages the SIR epidemic model to simulate the actual spreading capability of each node, effectively using it as a node label. For instance, Asgharian Rezaei et al. [8] initially constructed an eigenvector for each node, with its dimension equal to the total number of nodes in the network. They employed 0.5 % of the network's nodes to train a regression model. However, a drawback of this method arises when the network scale increases, as the dimension of the eigenvector becomes excessively large, resulting in considerable time and space consumption during runtime. Similarly, Yu et al. [9] extracted feature representations from the nodes in the network and utilized Convolutional Neural Networks (CNNs) to train a regression model. Subsequently, the trained regression model was employed to predict the influence of nodes. Conversely, Zhao et al. [10] utilized the entire network to train a classification model, subsequently applying the trained classification model to predict the importance categories of nodes in a separate test network.

The crux of the supervised method lies in obtaining propagation ability labels for nodes, a process whose cost escalates significantly with the expansion of the network scale. Hence, unsupervised learning methods have gained popularity among researchers in the task of key node identification. Unsupervised learning approaches first employ network representation learning, also known as network embedding, to extract feature vectors of nodes. Subsequently, in conjunction with other methods, which identify key nodes within the network. For instance, the DeepIM method [11] utilizes network representation learning to tackle the key node identification challenge. In this method, the CARE algorithm [12] is first employed to acquire representation vectors for each node. These vectors are then used to compute the similarity between pairs of nodes, constructing a matrix of correlations among nodes. The selection of key nodes is based on the magnitude of node counts within the correlation matrix. Similarly, some studies [13] have also employed node embeddings as a foundation, coupled with clustering algorithms to partition network nodes into multiple clusters. Subsequently, within these node clusters, employ additional methodologies to select key nodes.

The above methods do not well combine the feature extraction ability of network representation learning and the generalization ability of machine learning models, resulting in poor adaptability and generalization performance of the above methods. Simultaneously, some algorithms constructed node feature dimensions that were excessively large,

leading to significant time and space consumption during runtime. Therefore, this paper proposes a simple and effective key node identification algorithm *NRL_KNI* based on the network embedding method. The network embedding algorithm can greatly reduce the representation dimension of nodes and preserve the node structure information as much as possible. In addition, to account for the structural diversity of training samples, a quota-based approach is employed for sampling the partitioned node clusters, and the regression model is trained by sampling 5% of the nodes in the network as the training set, which can not only save model training time but also help to enhance the model's generalization capacity.

Overall, the main contributions of this paper are as follows:

- A simple and effective key node identification algorithm *NRL_KNI* is proposed, which combines network embedding and regression models, which can make more effective use of node structure information and improve the prediction effect of the model. In this paper, only 5% of the nodes in the network are used to train the model, which improves the training efficiency and generalization of the model.
- In assessing the ultimate influence of nodes, this paper introduces a localized structure-based influence score (LSIS). LSIS integrates the propagation capability of first-order neighboring nodes with the respective node's local features, providing an effective evaluation of node influence.
- The proposed *NRL_KNI* method was evaluated for its performance on eight real datasets. Comparative analysis with the results from nine benchmark methods indicated that, in the majority of cases, the *NRL_KNI* method consistently outperformed in terms of the Jaccard similarity coefficient metric. Simultaneously, under the Kendall's Tau correlation coefficient metric, the *NRL_KNI* algorithm demonstrated performance improvements of up to 10% compared to the second-best performing benchmark method. Additionally, parameter sensitivity analysis experiments indicated that *NRL_KNI* exhibits strong robustness.

II. RELATED WORKS

In this section, we primarily focus on the relevant techniques associated with the *NRL_KNI* model, namely the SIR propagation model for obtaining node labels and the network embedding methods for learning node representation features.

A. SIR PROPAGATION MODEL

In the context of influence maximization, the identification of key nodes in complex networks aims to select a certain number of nodes that maximize the spread range under a specific propagation model. In graph G , the expected number of nodes that node v can influence under a specific propagation model is defined as its propagation capability (influence spread range).

In this study, the SIR epidemic model is employed to assess the propagation capability of nodes [8]- [10], [14], and [15]. The propagation process of this model can be succinctly described as follows: Initially, if node v is in an infected (activated) state (I), while the rest of the nodes remain susceptible (inactive) (S), nodes in an infected state (I) activate their susceptible (S) neighbors with a probability β , simultaneously, nodes in an infected state (I) transition to a recovered state (R) with a probability α , the propagation process terminates when no new nodes are activated (infected).

B. NETWORK REPRESENTATION LEARNING METHODS

Network representation learning, also referred to as network embedding, aims to learn a low-dimensional vector representation, denoted as $v_i \in R^{|V| \times d}$, for each node in the graph $G(V, E)$, where $d \ll |V|$ represents the dimensionality of the embedding vector.

In recent years, the application of deep learning for learning feature representations of network nodes has gained significant attention. The DeepWalk algorithm [16] was the first to introduce the Skip-gram natural language model and random walk strategy for learning node feature representations. It utilized a Depth-First Search (DFS) strategy to generate random walk sequences and employed these sequences as contextual information input into the skip-gram model to learn node representations. Building upon DeepWalk, Node2Vec [17] utilized a biased random walk approach to sample neighborhoods of nodes. This method incorporated two hyperparameters to control the sampling strategy using both Depth-First Search (DFS) and Breadth-First Search (BFS) to generate walk sequences. Subsequently, the Skip-gram natural language model [18] was employed to learn node embeddings.

However, this node's first-order neighborhood-based random walk strategy fails to capture richer contextual information. To capture higher-order contextual information, researchers integrated community information and role information into network representation learning.

The CARE algorithm [12] integrates community information into network representation learning. It accomplishes this by employing a community detection algorithm to partition network nodes into multiple communities. Then, by setting a hyperparameter, it samples either the community members of the current node or its immediate neighbors. The obtained walk sequences are subsequently fed into a Skip-gram model to obtain node representations. Similarly, algorithms such as MNMF [19], GEMSEC [20], CNRL [21], NRL_RWCE [22], CRARE [23], among others, also incorporate community information into network representation learning. On the other hand, the role2Vec algorithm [24] incorporates role information into single-layer network representation learning. Similarly, Zhang et al. proposed the RMNE algorithm [25] which integrates role information into multi-layer network representation learning.

TABLE 1. The symbols used in this article and their meanings.

Notations	Meaning
G	Undirected graph
V	Set of nodes
E	Set of edges
n	$n = V $
d	Embedding vector dimension
k	Number of clusters
r	Sampling ratio of nodes in the network
N	Network's node count
$d(u)$	Degree of node u
$\tau(v)$	The set of first-order neighboring nodes of node v
$\langle k \rangle$	Network's average degree
$\langle k^2 \rangle$	Network's second-order average degree
β_{th}	Network's theoretical diffusion threshold
β	The infection threshold

In this paper, the NRL_KNI model employs the SEGK algorithm [26] for learning node representations. In the following sections, we will elucidate the intricate details associated with learning node representations using the SEGK algorithm.

III. METHODOGY

According to convention, in this paper, an undirected network is defined as $G(V, E)$, where V represents the set of nodes, $E \subseteq V \times V$ represents the set of edges, and $(v_i, v_j) \in E$ indicates the presence of an edge between node v_i and node v_j .

The NRL_KNI model proposed in this paper is illustrated in Fig 1. It can be divided into three main components. The uppermost section in Fig 1 represents the first part of the model, which is primarily responsible for acquiring feature representations of nodes along with their corresponding propagation capability labels to form the training dataset. The middle section represents the second part, involving the model training using the training dataset and making predictions for non-sample nodes. The lowermost section corresponds to the third part, focusing on assessing the final influence of nodes based on their local structural features and combining them with their propagation capabilities. Ultimately, an influence ranking list for nodes is generated. The following sections will provide a detailed explanation of the theoretical concepts involved in each of these components.

A. NETWORK EMBEDDING AS FEATURE LEARNING IN NRL_KNI

The structural features of nodes in networks are widely utilized for identifying key nodes in complex networks. Consequently, recognizing structurally equivalent nodes holds paramount significance for downstream network analysis tasks. In this paper, we employ a network embedding algorithm called SEGK [26] to learn low-dimensional vector representations of nodes. This method integrates the structural information of nodes into the node representation learning process, by comparing different structural information of nodes across multiple scales to learn node embeddings.

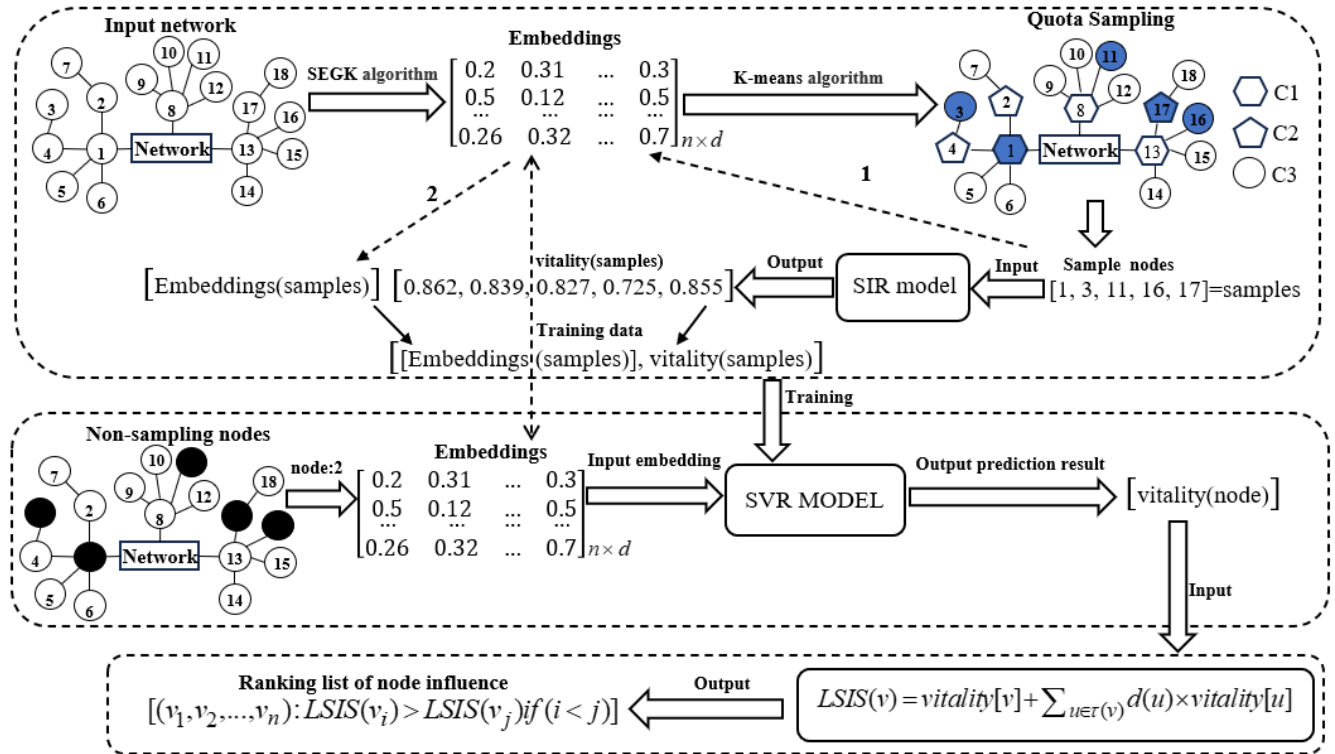


FIGURE 1. The overall framework of the NRL_KNI model.

Specifically, this algorithm defines an R-hop neighborhood for each node, where the R-hop neighborhood of node v_i is defined as $\{G_i^1, G_i^2, \dots, G_i^R\}$, and simultaneously establishes the computation method for the kernel between two nodes:

$$k(v_i, v_j) = \sum_{r=1}^R \hat{k}_G(G_i^r, G_j^r) \hat{k}_G(G_i^{r-1}, G_j^{r-1}). \quad (1)$$

The algorithm initializes matrix $K \in R^{n \times n}$ as a symmetric positive semidefinite matrix. The values of matrix K can be computed using formula (1), with the calculation equation being $K_{i,j} = k(v_i, v_j)$. Subsequently, the kernel matrix K is factorized to obtain $K = SS^T$, where $S \in R^{n \times d}$, and each row of S can be interpreted as the structural representation of nodes in a latent dimensional space. However, when the network scale becomes significantly large, the total number of network nodes denoted as n becomes substantial. This leads to a considerable computational cost for factorizing matrix K. Therefore, the algorithm employs an approximation technique (Nyström method [27]) to yield an approximation of K as $K \approx SS^T$, where $S \in R^{n \times d}$, and $d \ll n$ signifies the dimension of the embedding vectors.

B. OBTAINING SAMPLE NODES AND NODE LABELS AS THE TRAINING SET

For machine learning models, constructing training samples with diverse structures contributes to enhancing the model's generalization ability. Therefore, in this section, building upon node feature representations, we extract sample

nodes with varying structural characteristics through clustering algorithms and quota sampling to promote the model's generalization capacity.

The NRL_KNI model obtains the feature matrix embedding of nodes through the SEGK algorithm. In the context of the SEGK algorithm, it can map nodes with structurally similar attributes into the embedding space while preserving their proximity. Consequently, the K-means algorithm can assign nodes with structural similarity to the same cluster. In essence, this algorithm aims to partition n data points into k ($k \ll n$) clusters, so that the sum of squared distances between each sample point within a cluster and the centroid sample point of that cluster is minimized.

However, the clusters of nodes obtained through the k-means algorithm do not guarantee an even distribution of nodes within each cluster. Therefore, this algorithm employs a quota sampling approach to sample nodes within each cluster. Assuming the total number of nodes in node cluster c_i is m_i and the total number of sampled nodes is h, then the number of sampled nodes from node cluster c_i is $\frac{m_i}{n} \times h$. The advantage of this approach lies in avoiding excessive or insufficient sampling across clusters of different sizes, thus accommodating the structural diversity of sample nodes.

For the sampled data points, the SIR model is utilized to simulate their propagation capabilities as node labels. Subsequently, the embedding vectors of the sample nodes are combined with their corresponding propagation capabilities to construct the training set.

C. MODEL TRAINING AND PREDICTION FOR NON-SAMPLE NODES

In this paper, we set the proportion of sampled nodes to be 5% of the total number of network nodes. Considering that the smallest dataset used in this study contains approximately 1000 nodes, this implies that the number of sampled nodes is around 50. Therefore, the selection of a machine learning model that can efficiently train on small-scale datasets is crucial. In other words, we aim to train a model on a small-scale dataset and subsequently apply that model to a large-scale dataset.

In this study, we opt for utilizing Support Vector Regression (SVR) [8] to train the regression model. The SVR model has been extensively employed in the literature [8], [28] for modeling small-scale datasets. The central objective of SVR is to identify an optimal hyperplane within a high-dimensional feature space, intending to minimize the error between predicted and actual values while maintaining this error within a permissible tolerance range. Diverging from conventional regression techniques, SVR's distinctiveness lies in its robustness against outliers and nonlinear relationships. Through the incorporation of kernel functions, SVR can map data into higher-dimensional spaces, thereby accommodating intricate nonlinearity patterns.

Given the obtained training dataset, a regression model is trained using SVR on the sampled dataset. For non-sampled nodes, their feature representation vectors are obtained, and these vectors are then utilized as inputs to the SVR model to obtain their predicted values.

D. COMPUTE THE FINAL INFLUENCE OF NODES

However, the final ranking of network node influence is not directly sorted by the values predicted by the SVR model or simulated by the SIR model. Therefore, this paper introduces the Local Structural Influence Score (LSIS) based on node-local structures. It combines the propagation capability of first-order neighboring nodes with their local structural characteristics. The computation method for the LSIS score of node v is as follows:

$$LSIS(v) = vitality[v] + \sum_{u \in \tau(v)} d(u) \times vitality[u], \quad (2)$$

In equation (2), $\tau(v)$ represents the set of first-order neighboring nodes of node v , $d(u)$ represents the degree of node u , and $vitality[u]$ represents the propagation capability of node u , which is either the value predicted using the SVR model or simulated using the SIR model.

In the above equation, the importance of a node is measured by incorporating the propagation capabilities of its first-order neighboring nodes. If a node's neighboring nodes exhibit strong propagation capabilities, it implies that the node has a greater capacity for outward diffusion through these neighbors. Simultaneously, weights are allocated to the nodes in the first-order neighborhood based on their degrees. Building on this assumption, the larger the degree of a node, the more significant it is within the network. Therefore, by enhancing the contribution of neighboring nodes in LSIS

based on their degrees, this approach efficiently evaluates the significance of nodes within the network.

Algorithm 1 NRL_KNI Model

```

1: Input: Undirected graph  $G(V, E)$ , node embedding
   dimensions  $d$ , number of clusters  $k$ , sampling ratio  $r$ ,
   diffusion threshold  $\beta$ .
2: Output: Generate a ranked list of nodes based on the
   LSIS.
3: Embeddings = SEGK ( $G, d$ )
4:  $S\_num = \text{len}(G.nodes()) \times r$ 
5: Sample_nodes = []
6:  $C\_list = \mathbf{K-means}$  (Embeddings,  $k$ )
7: for  $c$  in  $C\_list$  do
8:   Sample_nodes.append(quota_sample( $c, S\_num$ ))
9: end for
10: Vitality = SIR (Sample_nodes,  $\beta$ )
11: SVR_model = SVR(Embeddings[Sample_nodes],
   Vitality)
12: for node in  $G.nodes() - \text{Sample\_nodes}$  do
13:   Vitality[node] = SVR_model(Embeddings[node])
14: end for
15: for  $v$  in  $V$  do
16:   Calculate the LSIS( $v$ ) based on Equation (2)
17: end for
18: return [ $(v_1, v_2, \dots, v_n) : LSIS(v_i) > LSIS(v_j)$  if ( $i < j$ )]

```

In Algorithm 1, the process begins with learning the node embedding matrix through the SEGK algorithm in the third line, and k node clusters are in the sixth line obtained using a clustering algorithm. Subsequently, lines 7 to 9 utilize a quota sampling method within the node clusters to acquire sample nodes. The propagation capabilities of the sample nodes are simulated using the SIR model in the 10th line. In the 11th line, a regression model is trained using the sample set. The subsequent lines from 12 to 14 are employed for predicting the propagation capabilities of non-sample nodes. The computation of the Local Structural Influence Score (LSIS) for nodes within the network occurs in lines 15 to 17. Ultimately, the algorithm concludes in the 18th line, outputting a ranked list of nodes based on LSIS.

IV. EXPERIMENTS

To evaluate the performance of different methods, this section first employs various approaches to generate node rankings. These rankings are then compared with the actual node rankings, and two different metrics are employed to assess the methods' performance. In recent years, researchers have proposed some novel node influence ranking algorithms [29], [30], to obtain the true ranking of node influence, this study runs the SIR model 1000 times for each node, and the average propagation capability is taken as the node's propagation ability label. Subsequently, nodes are ranked based on their propagation capabilities in descending order to establish a baseline ranking.

A. DATASETS

The study validates the performance of the proposed methods using networks from various domains, including citation networks (cora, citeseer), collaboration networks (CA-GrQc), social networks (Socfb-Reed98), interaction networks (ia-fb-message), biological networks (bio-CE-GT, bio-CE-LC), and language networks (wiki) [31]. The datasets mentioned above can be obtained from the NetworkRepository [32]. Detailed statistics for these datasets are provided in Table 2. In Table 2, the symbol β_{th} is employed to denote the theoretical diffusion threshold of the network, which can be calculated using the following formula: $\beta_{th} \approx \frac{\langle k \rangle}{\langle k^2 \rangle}$. In this equation, $\langle k \rangle = \frac{1}{N} \sum_i d_i$ is used to represent the network's average degree, $\langle k^2 \rangle = \frac{1}{N} \sum_i d_i^2$ indicates the second-order average degree of the network [33], and d_i denotes the degree of node i . Regarding the infection probability within the SIR model, this study consistently sets the value of β slightly greater than β_{th} , ensuring that large-scale propagation can be triggered at the corresponding β value [34].

TABLE 2. Detailed statistics for real datasets.

Dataset	#nodes	#edges	#max_D	#Avg_D	β_{th}	β
cora	2708	5429	168	3.89	0.091	0.101
citeseer	3312	4732	99	2.78	0.143	0.152
wiki	2405	12761	262	9.81	0.032	0.042
CA-GrQc	5241	14484	81	5.52	0.059	0.061
Socfb-Reed98	962	18812	313	39.11	0.014	0.016
ia-fb-message	1266	6451	112	10.19	0.036	0.040
bio-CE-GT	942	3239	151	7.01	0.034	0.037
bio-CE-LC	1387	1468	131	2.37	0.079	0.082

#max_D stands for max degree; #Avg_D stands for average degree.

B. EVALUATION METRICS

In this paper, the Jaccard similarity coefficient and Kendall's Tau correlation coefficient are utilized to quantify the similarity and correlation between the ordered lists generated by the algorithms and the ground truth lists.

1) JACCARD SIMILARITY COEFFICIENT [8]

This metric is used to compare the similarity or disparity between two samples, with values ranging from 0 to 1, where higher values indicate greater similarity between the two samples. According to the definition, for two ordered lists PR and TR, the Jaccard similarity coefficient $JS(k)$ for the top- k elements can be mathematically defined as:

$$JS(k) = \frac{|PR(k) \cap TR(k)|}{|PR(k) \cup TR(k)|}, \quad (3)$$

where PR is the node ranking list obtained by a specific algorithm, and TR is the true node ranking list.

2) KENDALL'S TAU CORRELATION COEFFICIENT [9]

This metric is used to quantify the degree of concordance between two lists. Its mathematical definition is as follows:

$$\tau(X, Y) = \frac{2(N_+ - N_-)}{n(n-1)}, \quad (4)$$

where X and Y respectively denote two distinct lists of length n , while N_+ and N_- represent the counts of concordant and discordant pairs between lists X and Y. For instance, the set of joint ranks obtained from X and Y is represented by $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$. If $x_i > x_j$ and $y_i > y_j$ or $x_i < x_j$ and $y_i < y_j$, then (x_i, y_i) and (x_j, y_j) form a concordant pair. If $x_i > x_j$ and $y_i < y_j$ or $x_i < x_j$ and $y_i > y_j$, then this pair is considered a discordant. Furthermore, the value of $\tau(X, Y)$ ranges from -1 to 1 . A larger value of $\tau(X, Y)$ indicates a stronger correlation between the two ranking lists.

C. BASELINE METHODS

The baseline algorithms in this paper include four centrality-based heuristic algorithms and five network embedding-based counting methods, which are detailed as follows:

1) DEGREE CENTRALITY (DC) [3]

Degree centrality measures the importance of a node by calculating the number of its first-order neighbors. Intuitively, if a node has more neighboring nodes, it is considered more significant in the network. Let d_v represent the degree of node v in network G . The normalized degree centrality of node v can be expressed as follows:

$$DC_v = \frac{d_v}{N-1}. \quad (5)$$

2) BETWEENNESS CENTRALITY (BC) [4]

Betweenness centrality measures the importance of a node by calculating the frequency at which the node appears as an intermediary in the shortest paths. The betweenness centrality of node v in the network can be defined as:

$$BC_v = \sum_{f \neq v \neq h} \frac{n_{fh}^v}{g_{fh}}, \quad (6)$$

where g_{fh} represents the total number of shortest paths between nodes f and h , and n_{fh}^v represents the count of paths passing through node v among the aforementioned total number of shortest paths. A higher betweenness centrality of node v indicates stronger control and information transmission capacity within the network.

3) PageRank CENTRALITY (PG) [5]

The core idea of this algorithm is that the importance of a node is not solely determined by the number of nodes it is connected to, but also by the importance of those nodes. In other words, a node's PageRank centrality is influenced by both its degree and the quality of its connections, emphasizing the collaborative impact of these two factors.

4) KATZ CENTRALITY(KC) [35]

This algorithm assesses the centrality of nodes based on the length of relationship paths between them. Unlike other centrality metrics, Katz centrality not only takes into account a node's direct connections but also considers multi-hop

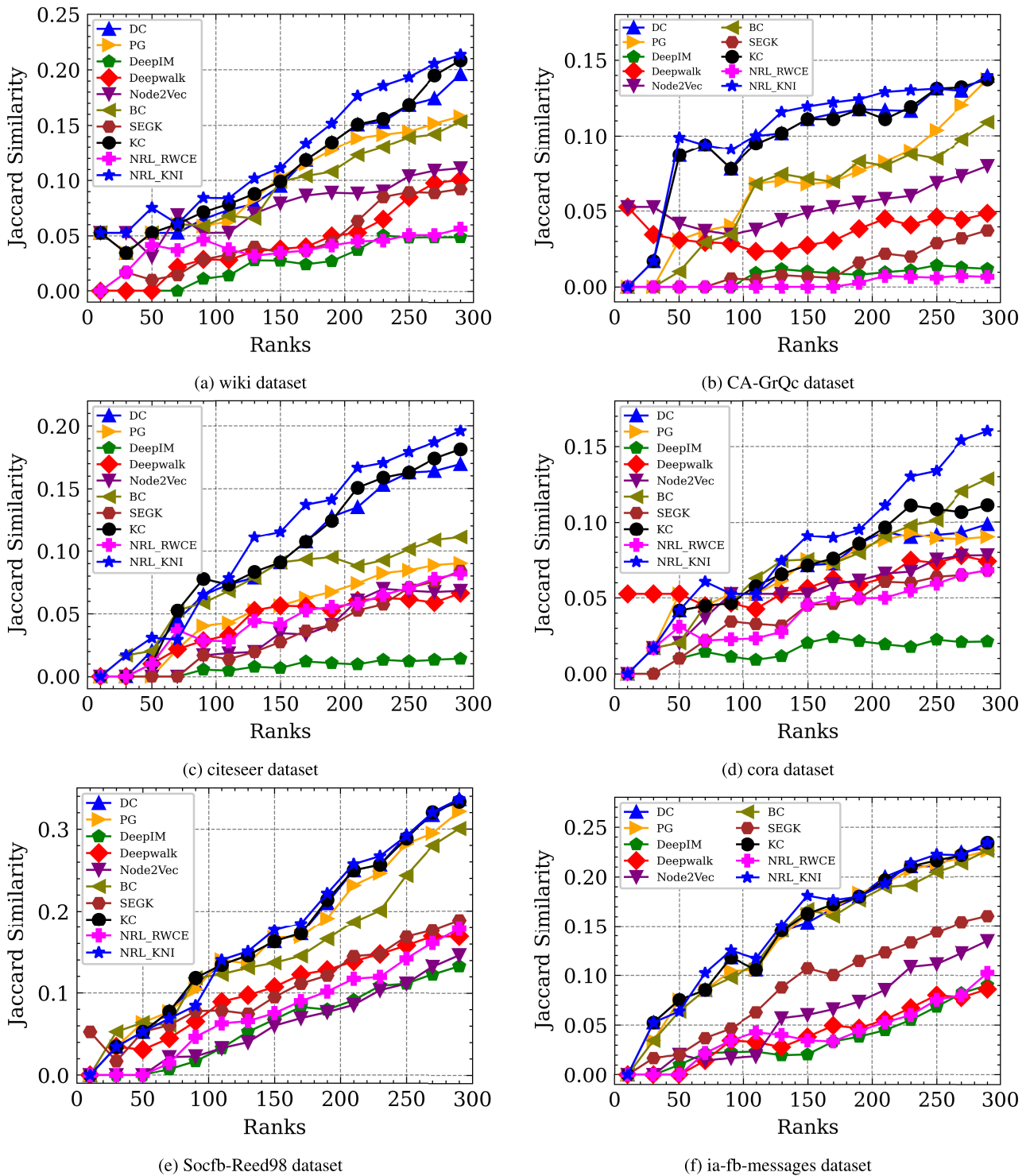


FIGURE 2. Jaccard similarity results in different datasets.

connections between nodes. The Katz centrality of node v_i can be defined as follows:

$$c_k(v_i) = \alpha \sum_{j=1}^N A_{i,j} c_k(v_j) + \xi, \quad (7)$$

where α is the attenuation factor, and α is strictly less than the reciprocal of λ_{max} . λ_{max} is the largest eigenvalue of the

adjacency matrix A , and parameter ξ provides an initial value for all nodes, typically set to 1.

5) DeepIM ALGORITHM [11]

This method introduces graph representation learning techniques for the identification of critical nodes in complex networks. Initially, the algorithm employs the CARE

TABLE 3. The Kendall's Tau correlation coefficient between the rankings obtained from different algorithms and the true rankings.

Algorithm	CA-GrQc	cora	citeseer	SocfbR	wiki	iafbm	bio-CE-GT	bio-CE-LC
DC	0.3788	0.2755	0.4233	0.3702	<u>0.3616</u>	<u>0.3496</u>	0.2423	0.1750
PG	0.0798	0.0889	-0.0395	0.3349	0.3062	0.3353	0.1198	-0.0598
BC	0.2368	0.2619	0.4180	0.2856	0.2421	0.3235	0.2125	<u>0.1910</u>
KC	<u>0.4423</u>	<u>0.3277</u>	<u>0.4808</u>	<u>0.3723</u>	0.3811	0.3543	<u>0.2714</u>	0.1789
DeepIM	-0.1846	-0.1242	-0.2254	-0.0838	-0.1169	-0.1409	-0.0834	-0.1183
Deepwalk	0.0712	0.0934	0.0262	-0.0362	0.0895	-0.1798	0.0562	0.0308
Node2vec	0.0896	0.1066	0.0286	-0.1344	0.1658	-0.0115	0.0301	0.0179
SEGK	-0.0009	0.0147	-0.0120	0.0430	0.0373	0.0186	0.0494	-0.0234
NRL_RWCE	0.0000	0.0098	0.0580	-0.0170	-0.0130	0.0003	-0.0036	0.1149
NRL_KNI	0.5400	0.3988	0.5805	0.3864	0.3563	0.3399	0.2846	0.2086

SocfbR stands for Socfb-Reed98; iafbm stands for ia-fb-message.

TABLE 4. The influence of different cluster numbers k on Kendall's Tau correlation coefficient.

k	CAG	cora	citeseer	SocfbR	wiki	iafam
k=5	0.5254	0.4037	0.5611	0.3868	0.3542	0.3400
k=10	0.5400	0.3988	0.5805	0.3864	0.3563	0.3399
k=15	0.5303	0.3926	0.5565	0.3864	0.3564	0.3402
k=20	0.5400	0.3988	0.5555	0.3864	0.3555	0.3407

CAG stands for CA-GrQc; SocfbR stands for Socfb-Reed98; iafam stands for ia-fa-message.

algorithm [12] to learn node embedding vectors. Subsequently, it calculates the cosine similarity between every pair of nodes, constructs a matrix of node correlations, and eventually selects the top-K nodes as critical nodes using a counting-based approach. In this paper, nodes are arranged in descending order based on their frequency of appearance, resulting in an ordered list reflecting the nodes' influence.

Building upon the approach of the DeepIM method, this paper also extends several other methods.

6) SEGK ALGORITHM [26]

This algorithm learns node embedding vectors by approximating and decomposing the kernel matrix containing the structural similarity between nodes. For the graph kernel used in this algorithm, the Weisfeiler-Lehman (WL) subtree kernel is selected in this paper. Similarly, after obtaining node embeddings, the selection of critical nodes follows the same approach as the DeepIM algorithm.

7) DeepWalk ALGORITHM [16]

This method starts by sampling nodes using a random walk approach. Subsequently, node sequences are used as inputs to train node embedding vectors using the Skip-Gram model [18]. The selection of the critical node set aligns with the approach of the DeepIM algorithm.

8) Node2Vec ALGORITHM [17]

This method introduces two hyperparameters during the sampling process to control the balance between depth-first search and breadth-first search for generating random walk sequences of nodes. Subsequently, these node sequences are used as inputs for training node embedding vectors using the Skip-Gram model [18]. Similarly, the selection of critical nodes follows the approach of the DeepIM algorithm.

TABLE 5. The influence of different sampling ratios r on Kendall's Tau correlation coefficient.

r	CAG	cora	citeseer	SocfbR	wiki	iafam
r=0.01	0.5266	0.3973	0.5461	0.3864	0.3547	0.3410
r=0.03	0.5310	0.4023	0.5751	0.3864	0.3544	0.3408
r=0.05	0.5254	0.4037	0.5611	0.3868	0.3542	0.3400
r=0.07	0.5304	0.4087	0.5687	0.3864	0.3563	0.3397

CAG stands for CA-GrQc; SocfbR stands for Socfb-Reed98; iafam stands for ia-fa-message.

TABLE 6. The influence of different embedding dimensions d on Kendall's Tau correlation coefficient.

d	CAG	cora	citeseer	SocfbR	wiki	iafam
d=16	0.5299	0.4037	0.5525	0.3864	0.3555	0.3406
d=32	0.5322	0.3953	0.5478	0.3864	0.3552	0.3404
d=64	0.5363	0.3956	0.5806	0.3862	0.3539	0.3409
d=128	0.5254	0.4037	0.5611	0.3868	0.3542	0.3400
d=200	0.5351	0.4038	0.5740	0.3868	0.3564	0.3410

CAG stands for CA-GrQc; SocfbR stands for Socfb-Reed98; iafam stands for ia-fa-message.

9) NRL_RWCE ALGORITHM [22]

This algorithm is a node embedding method based on community-aware random walks and incorporates an automatic parameter determination strategy for the random walks to ensure that the learned node representations preserve community information. The selection of the critical node set aligns with the approach of the DeepIM algorithm.

D. EXPERIMENTAL TOOLS AND PARAMETER SETTINGS

In this paper, all experiments were conducted on a computer equipped with the Windows operating system and executed using the CPU. The software tools utilized for experimental data processing included Python (version: 3.8), Networkx (version: 2.6.3), Matplotlib (version: 3.7.1), NumPy (version: 1.16.6), and Pandas (version: 1.4.3). Each experiment on the datasets involved the training of a support vector regression model for predicting the diffusion capability of non-sampled nodes. The model was configured with the following parameters: the regularization parameter C was set to 100, and the kernel function selected was the Radial Basis Function. Furthermore, for network embedding-based methods, the node embedding dimension d is uniformly set to 128, while the remaining parameters are used with the default values from the original literature. When constructing the correlation matrix, only the top 10 most relevant nodes for each node

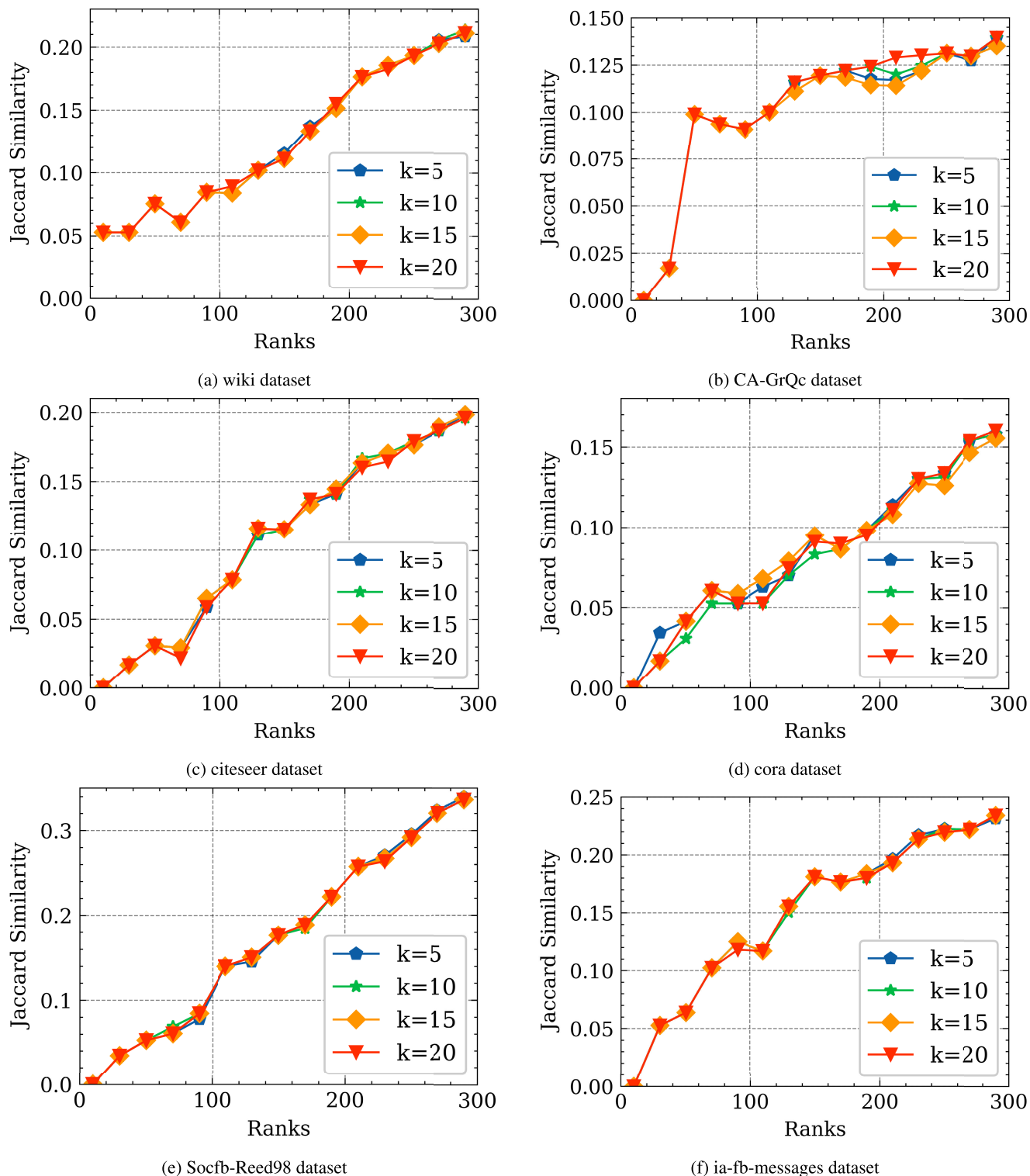


FIGURE 3. The influence of different cluster numbers k on Jaccard similarity coefficient.

are considered. In the SIR model, the infection probability β for each dataset is set to be greater than the threshold β_{th} , and the recovery probability is consistently set to 0.01. The clustering parameter k is chosen from the set $\{5, 10, 15, 20\}$. For all datasets, the sampling ratio r is uniformly set to 0.05.

E. EXPERIMENTAL RESULTS ANALYSIS

Across different datasets, the performance of various methods was compared based on the ranking of the top- k nodes, where k ranged from 10 to 290, and the corresponding ‘‘Ranks’’ values were shown on the graph. Fig 2 illustrates the variation

curves of Jaccard similarity coefficient values across six datasets. As depicted in Fig 2, our proposed NRL_KNI method attained the highest Jaccard similarity scores for the top-30 nodes in the wiki, ia-fb-message, and citeseer datasets. Similarly, it achieved the highest Jaccard similarity scores for the top-50 nodes in the CA-GrQc, wiki, and citeseer datasets. Furthermore, as “Ranks” values increased, the performance of NRL_KNI also exhibited improvement. In most scenarios, the NRL_KNI method consistently demonstrated the best outcomes. These results suggest that the NRL_KNI algorithm holds a certain advantage over baseline algorithms in identifying critical nodes within complex networks, thus affirming the effectiveness of the proposed approach presented in this study.

Table 3 presents the results of Kendall’s Tau correlation coefficient metric, where bold type indicates the best-performing result, and results marked with horizontal lines denote the second-best outcomes. In the CA-GrQc dataset, the performance of NRL_KNI improved by approximately 10% compared to the second-best Katz method. In the cora dataset, the performance of NRL_KNI improved by around 7% compared to the second-best algorithm. In the citeseer dataset, NRL_KNI’s performance improved by approximately 10% compared to the second-best method. Moreover, it is worth noting that NRL_KNI exhibited a performance improvement of approximately 1% compared to the Katz algorithm in the Socfb-Reed98 and bio-CE-GT datasets. Similarly, on the bio-CE-LC dataset, the performance of NRL_KNI also improved by around 1% compared to the second-best performing Betweenness Centrality algorithm. Additionally, Table 3 reveals that, in the majority of datasets, the performance of network embedding-based counting methods is subpar, whereas certain centrality-based heuristic algorithms exhibit favorable performance.

F. PARAMETER SENSITIVITY ANALYSIS

In the process of clustering, the number of clusters k can have a significant impact on the clustering outcomes. Therefore, this section performs parameter sensitivity analysis on six datasets. When evaluating the influence of the number of clusters k , while keeping other parameters constant, it is sufficient to vary only the value of k . The results are depicted in Fig 3. For the citeseer, wiki, Socfb-Reed98, and ia-fb-message datasets, the Jaccard similarity coefficient curve shows no noticeable fluctuations as the parameter k changes. The results indicate that the clustering parameter k is not highly sensitive on these datasets, and the algorithm presented in this paper demonstrates relatively stable outcomes. On the other hand, for the cora and CA-CrQc datasets, the Jaccard similarity coefficient curve exhibits slight fluctuations with changes in the parameter k , but these variations do not significantly alter the algorithm’s results, showing an overall trend towards stability. The results indicate that, in most cases, the parameter k has a minor effect on the Jaccard similarity coefficient values.

Table 4 presents the impact of different clustering numbers, denoted as k , on Kendall’s Tau correlation coefficient. Across the cora, Socfb-Reed98, wiki, and ia-fa-message datasets, as the clustering number k varies, the values of Kendall’s Tau correlation coefficient exhibit minimal fluctuations, indicating that parameter k is not highly sensitive to these datasets. However, in the CA-GrQc dataset, despite slight fluctuations in the correlation coefficient values due to changes in parameter k , the overall performance of the NRL_KNI algorithm remains relatively stable. Moreover, for the citeseer dataset, when $k = 10$, there is an approximately 3% performance improvement compared to $k = 15$ or $k = 20$. Nevertheless, considering the overall trend, the amplitude of this improvement is not substantial, and the proposed algorithm continues to outperform the baseline. The experimental results from Fig 3 and Table 4 collectively demonstrate that the proposed method is not highly sensitive to the clustering parameter k , further highlighting the robustness of the NRL_KNI algorithm.

The optimal values of k for different datasets are shown in Figure 3 and Table 4. Dataset CA-GrQc has an optimal k value of 20, while dataset citeseer has an optimal k value of 10. This indicates that different datasets have different optimal values for k . Tables 5 and 6 respectively present the impact of different parameter values r and d on the Kendall’s Tau correlation coefficient. The optimal values for parameters r and d vary across different datasets. For instance, the optimal r value for the citeseer dataset is 0.03, while for the cora dataset, it is 0.07. The optimal d value for the citeseer dataset is 64, whereas for the wiki dataset, it is 200. To maintain consistency with the benchmark methods, d is uniformly set to 128 in this study.

V. CONCLUSION

This paper presents a simple and effective key node identification algorithm named NRL_KNI. This algorithm leverages network representation learning techniques to acquire node embeddings, enabling efficient capture of the structural information of nodes within the network while significantly reducing the dimensionality of node representations. Moreover, this method introduces the Local Structural Influence Score (LSIS) to evaluate the final impact of nodes, taking into full consideration the influence factors of their local structures. The NRL_KNI approach employs a quota-based sampling technique on node clusters, using only 5% of the nodes within the network for model training, thus effectively reducing computational time and enhancing the algorithm’s generalization performance. Experimental results conducted on multiple real-world datasets indicate that the NRL_KNI method significantly outperforms the majority of baseline methods in terms of metrics such as the Jaccard similarity coefficient and Kendall Tau correlation coefficient.

However, there is still significant room for improvement for NRL_KNI. The shortcomings of this work include the reliance on node propagation capability labels on multiple

simulations of the SIR epidemic model. Although this approach is widely used in many literature works and this paper attempts to mitigate its impact on the overall algorithm by reducing the size of the training set (using only 5% of the network nodes), the influence of simulating node propagation capability on NRL_KNI is inevitable. Therefore, exploring a more efficient method to approximate node propagation capability becomes a direction for the improvement of this algorithm. Furthermore, the algorithm proposed in this paper can be extended for application in multi-layer networks, temporal networks, attribute networks, etc. It is also possible to explore alternative methods for constructing node feature vectors or employing diverse machine-learning models for training the sample set.

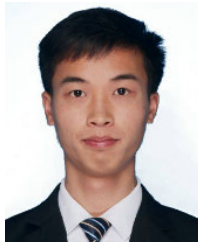
ACKNOWLEDGMENT

The authors express their sincere gratitude to the collaborative partners, who actively engaged in ideation, providing invaluable guidance, support, and in-depth discussions throughout the entirety of the research process. Furthermore, recognition is given to the laboratory for its contributions to this endeavor. Lastly, they are thankful to the numerous individuals who have contributed to their work, even if their names are not explicitly mentioned here; their contributions are equally significant to them.

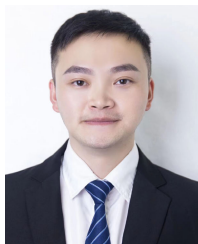
REFERENCES

- [1] H. Zhang, X. Chen, Y. Peng, G. Kou, and R. Wang, "The interaction of multiple information on multiplex social networks," *Inf. Sci.*, vol. 605, pp. 366–380, Aug. 2022.
- [2] G. Pallis, D. Zeinalipour-Yazti, and M. D. Dikaiakos, "Online social networks: Status and trends," in *New Directions in Web Data Management I* (Studies in Computational Intelligence). Berlin, Germany: Springer, 2011, pp. 213–234.
- [3] L. C. Freeman, "Centrality in social networks conceptual clarification," *Social Netw.*, vol. 1, no. 3, pp. 215–239, Jan. 1978.
- [4] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, vol. 40, no. 1, pp. 35–41, Mar. 1977.
- [5] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Comput. Netw. ISDN Syst.*, vol. 30, nos. 1–7, pp. 107–117, Apr. 1998.
- [6] Z. Zhao, D. Li, Y. Sun, R. Zhang, and J. Liu, "Ranking influential spreaders based on both node k-shell and structural hole," *Knowl.-Based Syst.*, vol. 260, Jan. 2023, Art. no. 110163.
- [7] K. Berahmand, A. Bouyer, and N. Samadi, "A new centrality measure based on the negative and positive effects of clustering coefficient for identifying influential spreaders in complex networks," *Chaos, Solitons Fractals*, vol. 110, pp. 41–54, May 2018.
- [8] A. Asgharian Rezaei, J. Munoz, M. Jalili, and H. Khayyam, "A machine learning-based approach for vital node identification in complex networks," *Expert Syst. Appl.*, vol. 214, Mar. 2023, Art. no. 119086.
- [9] E.-Y. Yu, Y.-P. Wang, Y. Fu, D.-B. Chen, and M. Xie, "Identifying critical nodes in complex networks via graph convolutional networks," *Knowl.-Based Syst.*, vol. 198, Jun. 2020, Art. no. 105893.
- [10] G. Zhao, P. Jia, C. Huang, A. Zhou, and Y. Fang, "A machine learning based framework for identifying influential nodes in complex networks," *IEEE Access*, vol. 8, pp. 65462–65471, 2020, doi: [10.1109/ACCESS.2020.2984286](https://doi.org/10.1109/ACCESS.2020.2984286).
- [11] M. M. Keikha, M. Rahgozar, M. Asadpour, and M. F. Abdollahi, "Influence maximization across heterogeneous interconnected networks based on deep learning," *Expert Syst. Appl.*, vol. 140, Feb. 2020, Art. no. 112905.
- [12] M. M. Keikha, M. Rahgozar, and M. Asadpour, "Community aware random walk for network embedding," *Knowl.-Based Syst.*, vol. 148, pp. 47–54, May 2018.
- [13] D. Chen, P. Du, B. Fang, D. Wang, and X. Huang, "A node embedding-based influential spreaders identification approach," *Mathematics*, vol. 8, no. 9, p. 1554, Sep. 2020.
- [14] A. Zareie, A. Sheikahmadi, M. Jalili, and M. S. K. Fasaee, "Finding influential nodes in social networks based on neighborhood correlation coefficient," *Knowl.-Based Syst.*, vol. 194, Apr. 2020, Art. no. 105580.
- [15] K. Berahmand, N. Samadi, and S. M. Sheikholeslami, "Effect of rich-club on diffusion in complex networks," *Int. J. Mod. Phys. B*, vol. 32, no. 12, May 2018, Art. no. 1850142.
- [16] 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*, New York, NY, USA, Aug. 2014, pp. 701–710.
- [17] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2016, pp. 855–864.
- [18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*.
- [19] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, Feb. 2017.
- [20] B. Rozenberczki, R. Davies, R. Sarkar, and C. Sutton, "GEMSEC: Graph embedding with self clustering," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*, New York, NY, USA, Aug. 2019, pp. 65–72.
- [21] C. Tu, X. Zeng, H. Wang, Z. Zhang, Z. Liu, M. Sun, B. Zhang, and L. Lin, "A unified framework for community detection and network representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1051–1065, Jun. 2019, doi: [10.1109/TKDE.2018.2852958](https://doi.org/10.1109/TKDE.2018.2852958).
- [22] K. Guo, Q. Wang, J. Lin, L. Wu, W. Guo, and K.-M. Chao, "Network representation learning based on community-aware and adaptive random walk for overlapping community detection," *Int. J. Speech Technol.*, vol. 52, no. 9, pp. 9919–9937, Jul. 2022.
- [23] H. Zhang, G. Kou, Y. Peng, and B. Zhang, "Role-aware random walk for network embedding," *Inf. Sci.*, vol. 652, Jan. 2024, Art. no. 119765.
- [24] N. K. Ahmed, R. A. Rossi, J. B. Lee, T. L. Willke, R. Zhou, X. Kong, and H. Eldardiry, "Role2vec: Role-based network embeddings," in *Proc. DLG KDD*, 2019, pp. 1–7.
- [25] H. Zhang and G. Kou, "Role-based multiplex network embedding," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 26265–26280.
- [26] G. Nikolentzos and M. Vazirgiannis, "Learning structural node representations using graph kernels," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 2045–2056, May 2021, doi: [10.1109/TKDE.2019.2947478](https://doi.org/10.1109/TKDE.2019.2947478).
- [27] C. K. Williams and M. Seeger, "Using the nystrom method to speed up kernel machines," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 13. Cambridge, MA, USA: MIT Press, 2001, pp. 682–688.
- [28] H. Khayyam, A. Jamali, A. Bab-Hadiashar, T. Esch, S. Ramakrishna, M. Jalili, and M. Naebe, "A novel hybrid machine learning algorithm for limited and big data modeling with application in industry 4.0," *IEEE Access*, vol. 8, pp. 111381–111393, 2020, doi: [10.1109/ACCESS.2020.2999898](https://doi.org/10.1109/ACCESS.2020.2999898).
- [29] R. Wang, X. Qiu, S. Wang, X. Zhang, and L. Huang, "Ranking nodes in complex networks based on TsRank," *Phys. A, Stat. Mech. Appl.*, vol. 624, Aug. 2023, Art. no. 128942.
- [30] C. Salavati, A. Abdollahpouri, and Z. Manbari, "Ranking nodes in complex networks based on local structure and improving closeness centrality," *Neurocomputing*, vol. 336, pp. 36–45, Apr. 2019.
- [31] J. Chen, Z. Gong, J. Mo, W. Wang, W. Wang, C. Wang, X. Dong, W. Liu, and K. Wu, "Self-training enhanced: Network embedding and overlapping community detection with adversarial learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 11, pp. 6737–6748, Nov. 2022.
- [32] R. Rossi and N. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. AAAI Conf. Artif. Intell.*, vol. 29, 2015.
- [33] J.-G. Liu, Z.-M. Ren, and Q. Guo, "Ranking the spreading influence in complex networks," *Phys. A, Stat. Mech. Appl.*, vol. 392, no. 18, pp. 4154–4159, Sep. 2013.

- [34] J. Bae and S. Kim, "Identifying and ranking influential spreaders in complex networks by neighborhood coreness," *Phys. A, Stat. Mech. Appl.*, vol. 395, pp. 549–559, Feb. 2014.
- [35] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, Mar. 1953.



HEPING ZHANG was born in Guizhou, China. He is currently pursuing the master's degree with the Key Laboratory of Information and Computing Science Guizhou Province, Guizhou Normal University, Guiyang, China. His research interests include network representation learning and influence maximization.



SICONG ZHANG was born in Chongqing, China. He received the Ph.D. degree in software engineering from Guizhou University. He is currently a Lecturer with the Key Laboratory of Information and Computing Science Guizhou Province, Guizhou Normal University, Guiyang, China. His research interests include artificial intelligence security and intrusion detection.



XIAOYAO XIE (Member, IEEE) was born in Guizhou, China. He is currently a Professor and the Ph.D. Supervisor with the Key Laboratory of Information and Computing Science Guizhou Province, Guizhou Normal University, Guiyang, China. He is also the Director of the Key Laboratory. His research interests include artificial intelligence, 5G, and IPV6.



TAIHUA ZHANG was born in Guizhou, China. He received the Ph.D. degree in mechanical engineering from Zhejiang University. He is currently a Professor and the Master's Supervisor with the School of Mechanical and Electrical Engineering, Guizhou Normal University, Guiyang, China. He is also the Deputy Dean of the School of Mechanical and Electrical Engineering. His research interests include knowledge engineering and knowledge management, intelligent manufacturing, industrial internet, and industrial big data.



GUOJUN YU was born in Guizhou, China. He received the Ph.D. degree in software engineering from Guizhou University. He is currently a Lecturer with the Key Laboratory of Information and Computing Science Guizhou Province, Guizhou Normal University, Guiyang, China. His research interests include artificial intelligence and data visualization.

...