IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 2, 2019, accepted November 12, 2019, date of publication November 25, 2019, date of current version December 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2955468

# Node Embedding With a *CN*-Based Random Walk for Community Search

**WEIJI ZHAO[1,2] AND FENGBIN ZHANG[1]**
[1]School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, China
[2]School of Information Engineering, Suihua University, Suihua 152061, China

Corresponding author: Fengbin Zhang (zhangfb@hrbust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61172168 and in part by the Fundamental Research Funds for the Universities of Heilongjiang Provincial Department of Education of China under Grant KYYWF10236180104.

**ABSTRACT** Community search is a query request-oriented community detection problem. Given a query node $v$ in network $G$, the goal of community search is to discover a community in $G$ that contains node $v$. Traditional algorithms rely on carefully engineered features to measure local neighborhood structures. Designing these features is a time-consuming process that limits their practical application. Motivated by node embedding using deep learning method to learn distributed representations for nodes in networks, we propose a two-stage community search algorithm based on node embedding. To address the drawbacks of existing node embedding methods, we propose a node embedding model with a $CN$-based random walk (NECNW) based on a skip-gram model in the first stage. Via NECNW, we learn a low-dimensional representation of nodes in networks. In the second stage, we propose a community quality metric *closeness-isolation* ($CI$) based on the learned vectors. Then, we expand the target community by greedy addition of a shell node that has maximum similarity with the current community. We evaluate the proposed algorithm on both real-world and synthetic networks with related community search and node embedding algorithms. The experimental results show that the proposed algorithm is more effective and efficient for community search than other algorithms.

**INDEX TERMS** Community search, node embedding, local community detection, community structure, random walk.

## I. INTRODUCTION

Community structure is a common property of complex networks. Essentially, a community is a group of nodes that are densely connected internally [1]. Identifying communities hidden in networks provides insight into the inner connections of networks and can be applied to many tasks such as friend recommendation, advertisement in e-commerce and hot spread node selection [2]–[4].

Traditional community detection algorithms [5]–[7] aim to identify all communities in a network based on the entire network structure. However, their computational complexities are proportional to the size of the networks [8] and are therefore too demanding to be applied to large networks. To solve this problem, community search [9]–[11] was proposed and has become a hot issue in network analysis research.

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

Community search is a query request-oriented community detection problem. Given a query node $v$ of network $G$, the goal of community search is to discover a high-quality community $D \subset G$, called the target community, which contains node $v$. The community search problem has also been studied as the local community detection problem in the literature [12], [13].

Community search has attracted much attention in recent years, and many algorithms have been proposed. Some algorithms find a community that satisfies a particular structure [14], such as $k$-core [10], [15], $k$-truss [16] and $k$-clique [17]. Other algorithms expand the target community from a seed node. Maximizing a goodness metric is the most useful and widely used strategy adopted by algorithms of this kind. This kind of algorithm usually takes the query node as a seed and expands the target community from the seed according to a particular goodness metric, such as $R$ [18], $M$ [19], *tightness* [20], or *compactness-isolation* [21].

Determining how to represent the network is a key problem in data mining of network data [22], [23]. Traditional community search algorithms rely on carefully engineered features to measure local neighborhood structures, and they face the limitation that designing these features is a time-consuming process [22]. In recent years, node embedding has adopted deep learning to learn distributed representations for nodes in networks, offering a new approach to map nodes into the points in a low-dimensional vector space. Meanwhile, the relationships among nodes in the origin networks are captured by the similarities between nodes in the vector space [23]. However, most of the existing node embedding methods treat adjacent nodes equally and neglect the fact that tie strengths differ between entities in complex networks.

As we know, the tie strengths are different in complex systems. However, in practice, for efficiency or because it is hard to quantify the closeness between entities in complex networks, this information is lost in the process of modeling complex networks as unweighted graphs. Therefore, we lose valuable information that could enhance the accuracy of detecting community structure.

To address this issue, we adopt an edge weighting strategy to recover the information lost in the modeling process. In detail, we adopt the common neighbors ($CN$) metric [24] to measure the similarity between nodes and develop a two-stage community search algorithm based on node embedding with a $CN$-based random walk approach. In the first stage, we adopt the $CN$ metric [24] to measure the similarity of nodes, and we present a $CN$-based random walk method. Moreover, we propose a node embedding model with a $CN$-based random walk (NECNW), via which we obtain a low-dimensional vector representation of nodes. In the second stage, based on the vector representation of nodes produced by NECNW, we propose a new goodness metric *closeness-isolation* ($CI$) and design a community search algorithm by maximizing this metric.

To summarize, our main contributions in this paper are summarized as follows:

- We design a $CN$-based random walk method via which we construct a corpus of node paths, and then, we propose a node embedding model, NECNW, based on the skip-gram model.
- We propose a metric, $CI$, for measuring the quality of a community based on node vectors produced by NECNW. Based on this, we propose a new community search algorithm by maximizing the $CI$ metric.
- We test the proposed algorithm on both real-world and synthetic benchmark networks. The experimental results show that the proposed algorithm is more effective for community search than baselines.

The rest of the paper is organized as follows. We first introduce some related works on community search and node embedding in Section II. Then, we present the formal problem definition of community search and evaluation metrics in Section III and describe the algorithm details in Section IV.

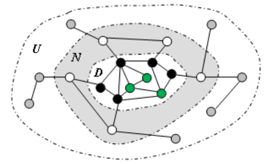



**FIGURE 1.** An illustration of the division of a network: target community $D$, $D$'s shell node set $N$ and unknown node set $U$.

We report the experimental results in Section V, followed by conclusions in Section VI.

## II. RELATED WORK

### A. COMMUNITY SEARCH

Community detection, also known as graph clustering [25], focuses on clustering nodes in a single network. Another related graph-based clustering problem is subspace clustering [26], [27], which aims to cluster data points drawn from a union of low-dimensional subspace. Although the problem of subspace clustering is different from community detection, many of the techniques involved are closely related [28].

Community search is a query-oriented variant of the community detection problem [10], [29]. Community search aims to discover the community of a query node $v$, while community detection aims to discover all communities in a network. The most related work to ours is that of seed node expansion-based community search algorithms. Algorithms of this kind usually take node $v$ as a seed and expand the seed node into a community according to a particular goodness metric. The metrics mainly depend on the local network structure around the target community, without requiring knowledge of the entire network structure.

As shown in Fig. 1, a network can be divided into three parts: community $D$, $D$'s shell node set $N$ and unknown node set $U$. We further divide nodes in community $D$ into two parts: the core nodes $C$ and the boundary nodes $B$. The nodes in $C$ are only connected with nodes in $D$, while the nodes in $B$ have at least one neighbor node in $N$. We refer to the edges within community $D$ as internal edges and the edges connecting nodes in $D$ with nodes in $N$ as external edges.

Goodness metrics mainly depend on the internal and external edges connected with target community $D$. Clauset [18] defines a community quality metric $R$ by considering the ratio of internal boundary edges to all edges associated with nodes in $B$. Luo *et al.* [19] define a local community modularity $M$, which is the ratio of the number of internal edges to the number of external edges connected with community $D$.

Both $R$ and $M$ count only the number of internal and external edges and give equal weight to them. However, this is not consistent with the fact that nodes of the same community are more similar with each other than with nodes outside of the community. To address this problem, Huang *et al.* [20] adopt structural similarity to measure similarity between two adjacent nodes, and they introduce a similarity-based community quality metric, *tightness*. Determining how to measure the similarity of nodes becomes a challenge for algorithms following this idea. Ma *et al.* [21] take into account nonadjacent nodes within $d$-steps and propose a $d$-neighbors similarity measurement. Zhao *et al.* [30] take into account the weight of neighbor nodes and propose a common neighbor-based similarity measurement with weighted neighbor nodes, CNWNN.

FlowPro [31] is another approach for community search based on flow propagation. The query node propagates a flow to its neighbors. Each node is able to store and propagate the received flow to its neighbors. The nodes that belong to the target community store higher flow than nodes outside the community.

In this paper, motivated by the advance of deep learning on networks, we propose a new similarity-based community search algorithm by combining node embedding technique.

### B. NODE EMBEDDING

Inspired by the success of distributed representations of words in natural language processing [32], Perozzi *et al.* [33] proposed a node embedding model in 2014, DeepWalk, which generalized the skip-gram model to process a sequence of randomly generated node paths. Since then, node embedding has become a hot issue in the field of complex network analysis.

DeepWalk [33] adopts an unbiased random walk on networks to generate node paths. Grover and Leskovec [34] propose node2vec, another skip-gram-based node embedding model for learning continuous feature representations for nodes in networks. By introducing the return parameter $p$ and in-out parameter $q$, node2vec adopts a flexible biased random walk that explores neighborhoods in BFS as well as DFS fashion. Liu *et al.* [35] take into account the similarity between nodes and propose a node embedding model, NEMCNB, based on closest-neighbor biased random walk.

Tang *et al.* [36] propose another highly successful node embedding model, LINE, which is not based on a random walk approach. LINE designs objective functions that optimize both the first-order and second-order proximities and proposes an edge-sampling algorithm for optimizing the objective, which tackles the limitation of the traditional stochastic gradient descent.

In contrast to the aforementioned algorithms, we adopt *CN* [24] to measure the closeness of nodes and propose a node embedding model, NECNW, with a *CN*-based random walk.

## III. PROBLEM DEFINITION AND EVALUATION METRICS

Complex networks are usually modeled as graphs. We first define the network and then present the problem definition of community search and evaluation metrics for measuring the effectiveness of a community search algorithm.

*Definition 1 Network [21]: We use graph $G = (V, E)$ to represent a network, where $V$ is the set of nodes and $E$ is the set of edges. $|V|$ denotes the number of nodes in $V$, and $|E|$ denotes the number of edges in $E$. For any node $v \in V$, $\Gamma(v)$ denotes the neighbor node set of $v$.*

A community is a group of nodes that are more similar to each other than to nodes of any other communities [37]. The problem of community search is defined as follows.

*Problem 1 Community Search: Given a network $G = (V, E)$ and a query node $s \in V$, the goal of community search is to find a particular community $C \subset G$ that contains query node $s$.*

The community search algorithms usually take query node $s$ as a seed and expand the seed into a community according to a particular goodness metric. We use three evaluation metrics *precision*, *recall* and *F-score* to measure the effectiveness of different community search algorithms, which are also adopted by other community search algorithms [21], [35], [38].

As mentioned above, $C$ denotes the ground-truth community that contains node $s$. Let $D$ denote the algorithmic community of node $s$; then, the definition of evaluation metrics is described as follows.

*Precision* is the fraction of correct nodes in the algorithmic community $D$, and *recall* is the fraction of correct nodes in the ground-truth community $C$ [38]. The formulas for *precision* and *recall* are defined as follows.

$$precision = \frac{|C \cap D|}{|D|} \tag{1}$$

$$recall = \frac{|C \cap D|}{|C|} \tag{2}$$

The algorithms usually return node sets with different sizes since there is no size constraint of target community $D$. For a given query node, an algorithm with more nodes returned would produce a higher *recall* value and a lower *precision* value, and an algorithm with fewer nodes returned would obtain a lower *recall* value and a higher *precision* value. Therefore, *precision* and *recall* can be thought of as two sides of the same coin. *F-score* is the harmonic mean of *precision* and *recall*. We use *F-score* to measure the effectiveness of different community search algorithms. A higher *F-score* value indicates better algorithmic performance. Its formula is defined as follows.

$$F\text{-}score = 2 \times \frac{precision \times recall}{precision + recall} \tag{3}$$

## IV. OUR ALGORITHM

For solving Problem 1, we propose a two-stage community search approach. The framework of our approach is shown in Fig. 2. In the first stage, we map nodes in network $G$ into points in a low-dimensional vector space using node embedding, and the relationships among nodes in origin network $G$ are captured by the similarities between nodes in the vector space [23]. In the second stage, based on the vector representation of nodes, we propose a new community search algorithm. In detail, we first design a new community quality metric, *closeness-isolation*, and then present our community search algorithm.

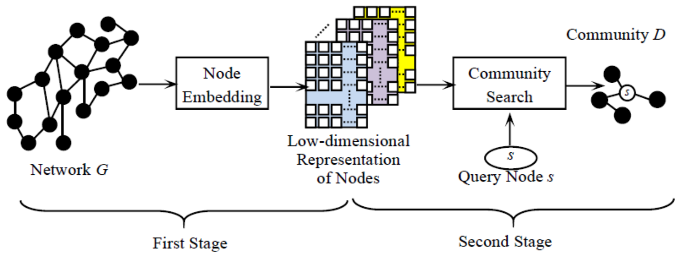



**FIGURE 2.** The framework of our community search approach.

In this section, we first propose a new node embedding model, NECNW, with a *CN*-based random walk, and then present our community search algorithm.

### A. NODE EMBEDDING MODEL NECNW

By simulating random walk on networks, we generate a corpus of node paths. The main difference among the word2vec-based node embedding algorithms is that they adopt different random walk approaches.

#### 1) *CN*-BASED RANDOM WALK

Formally, we represent a random walk path of fixed length $l$ as $[v_1, v_2, \ldots, v_l]$, where $v_i$ is the $i$th node in the path. In the process of simulating a random walk from node $v_1$, suppose the current node $v_i$ is $x$; then, the probability of node $y$ ($y \in \Gamma(x)$) being the next node $v_{i+1}$ is defined as follows.

$$P(v_{i+1} = y | v_i = x) = \frac{w_{xy}}{\sum_{u \in \Gamma(x)} w_{xu}} \tag{4}$$

$w_{xy}$ is the unnormalized transition probability between nodes $x$ and $y$. Algorithms have different definitions of $w_{xy}$. For example, DeepWalk sets $w_{xy} = 1$, while node2vec takes into account the shortest path length $spl$ between $v_{i-1}$ and $v_{i+1}$, and $w_{xy}$ is set as follows.

$$w_{xy} = \begin{cases} \frac{1}{p} & spl = 0 \\ 1 & spl = 1 \\ \frac{1}{q} & spl = 2 \end{cases} \tag{5}$$

where $p$ is the return parameter, and $q$ is the in-out parameter.

Liu *et al.* [35] propose a new opinion that the tie strengths are different among friends in real-world social networks and that closest friends are preferred. However, in practice, for efficiency or because it is hard to quantify the closeness between entities in complex networks, this information is lost in the process of modeling complex networks as unweighted graphs. The accuracy of detecting the community structure would be enhanced if we could recover the lost information [39]. Inspired by this work, we adopt *CN* [24] to measure the closeness of nodes, and we propose a *CN*-based random walk.

$$w_{xy} = |\Gamma(x) \cap \Gamma(y)| \tag{6}$$

Fig. 3 shows a simple network. Supposing that the current node $v_i$ is 5, $\Gamma(5) = \{2, 3, 4, 6\}$, and the $w$ and $p$ of node $y$ ($y \in \Gamma(5)$) are shown in Table 1.

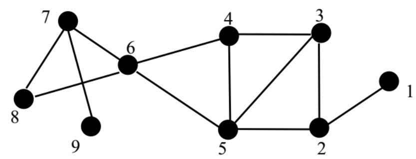



**FIGURE 3.** A simple network ***G***.

**TABLE 1.** The $w$ and $p$ of node $y$ ($y \in \Gamma(5)$).

| $y$ | $\Gamma(y)$ | $w_{5y}$ | $p(y\|5)$ |
|---|---|---|---|
| 2 | {1,3,5} | 1 | 1/6 |
| 3 | {2,4,5} | 2 | 2/6 |
| 4 | {3,5,6} | 2 | 2/6 |
| 6 | {4,5,7,8} | 1 | 1/6 |

The probability of node $y$ ($y \in \Gamma(v_i)$) being $v_{i+1}$ is proportional to its closeness with node $v_i$. Next, we adopt the alias method [36] to randomly sample a node from $\Gamma(v_i)$ according to the $p$ values.

We define a function to implement the operation of simulating a *CN*-based random walk from node $v$ with length $l$, which will be referred to as Algorithm 1 introduced in the next subsection. The pseudocode of the *CN*-based random walk is shown in Function *CNWalk*.

#### 2) NODE EMBEDDING MODEL NECNW WITH A *CN*-BASED RANDOM WALK

Based on the *CN*-based random walk, we propose a new node embedding model, NECNW. There are two steps in the NECNW model. In the first step, we construct a corpus of node paths by performing $r$ times *CN*-based random walks of fixed length $l$ from every node in network $G$. The corpus consists of $r * |V|$ node paths. In the second step, we consider nodes as words and node paths as sentences and learn vector representation of nodes via skip-gram [32], which has been proven to be successful in natural language processing. Given a node path $np = [v_1, v_2, \ldots, v_l]$, the context of node $v_i$, denoted as $C(v_i)$, is the nodes in a window of size $ws$ centered at $v_i$, i.e., $C(v_i) = [v_{i-ws}, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{i+ws}]$. We learn a node embedding function $f$ by maximizing the objective function

$$\max_f \sum_{v_i \in V} logp(C(v_i)|f(v_i)) \tag{7}$$

**Function** *CNWalk(G,v,l)*

```
Input: network G; start node v; walk length l
Output: node path np
begin
    initialize np = [v];
    i = 0;
    while i < l − 1 do
        cur = np[i];
        nbrs_p = {};
        foreach u ∈ Γ(cur) do
            //see Formula(6)
            nbrs_p[u] = |Γ(u) ∩ Γ(cur)|;
        end
        s = Σ_{x∈Γ(cur)} nbrs_p[x];
        foreach u ∈ Γ(cur) do
            nbrs_p[u] = nbrs_p[u]/s;
        end
        select a node in Γ(cur) according to the
        probabilities in nbrs_p, denoted as y;
        np.append(y);
        i++;
    end
    return np;
end
```

where we assume that the nodes in $C(v_i)$ are independent of each other; thus, Eq.(7) can be represented as

$$\max_{f} \sum_{v_i \in V} \prod_{u \in C(v_i)} logp(u|f(v_i)) \tag{8}$$

The learned vectors are expected to preserve as many properties of the original network as possible and thus can be used as feature vectors of nodes [40]. The pseudocode of NECNW is shown in Algorithm 1.

Via the NECNW model, we can learn low-dimensional vector representations of nodes. For description, we denote the feature vector associated with node $v$ as $f[v]$.

### B. COMMUNITY SEARCH ALGORITHM BASED ON NECNW

Based on the learned vectors of nodes produced by the NECNW model, we define a node similarity measurement as follows.

$$sim(u, v) = f[u] \cdot f[v] \tag{9}$$

Metric $sim(u, v)$ is the dot product of vectors associated with nodes $u$ and $v$. Next, we give the definition of *closeness-isolation* for measuring the quality of a community.

*Definition 2 (Closeness-Isolation Metric): Given a network $G = (V, E)$ and a node embedding function $f$, for a community $D$ with shell node set $N$, $N = \{x|v \in D, x \in \Gamma(v), x \notin D\}$, the closeness-isolation metric of community $D$, denoted by $CI(D)$, is defined as*

$$CI(D) = \frac{\sum_{u\in D, v\in D, (u,v)\in E} sim(u, v)}{1 + \sum_{a\in D, b\in N, (a,b)\in E} sim(a, b)} \tag{10}$$

**Algorithm 1** NECNW

```
Input: network G = (V, E); walks per node r; walk
       length l; windows size ws; dimension dn
Output: vector representations f of nodes in G
begin
    initialize nps = [];
    loop = 0;
    while loop < r do
        foreach u ∈ V do
            np = CNWalk(G, u, l);
            nps.append(np);
        end
        loop+ = 1;
    end
    f = Skip-gram(nps, ws, dn);
    return f;
end
```

The numerator of $CI$ is the closeness of community $D$, and the denominator is the isolation of $D$. The higher the $CI$ value is, the higher the quality of a community. Thus, we use $CI$ to measure the quality of a community.

For discovering the community of query node $s$, we initialize $D = \{s\}$, $N = \Gamma(s)$, and expand community $D$ by adding the nodes in $N$ one node at a time.

Nodes in $N$ have at least one neighbor node in $D$. The probability of a node $x$ ($x \in N$) belonging to community $D$ is directly proportional to the sum of similarities between node $x$ and nodes in $D$. Thus, we design $sim_{in}$ to measure the similarity of a node $x$ ($x \in N$) with community $D$.

$$sim_{in}(x, D) = \sum_{z\in\Gamma(x)\cap D} sim(x, z) \tag{11}$$

We choose the node with maximum $sim_{in}$ value in $N$ as candidate node $y$. If the $CI$ gain of adding node $y$ to community $D$ is positive, it is added to $D$ and $N$ is updated. Otherwise, it is removed from $N$. We repeat this operation until the shell node set $N$ is null. The pseudocode of our algorithm is shown in Algorithm 2.

## V. EXPERIMENTS

In this section, we evaluate the perform of the proposed algorithm by comparing it with related community search and node embedding algorithms on both real-world and synthetic networks.

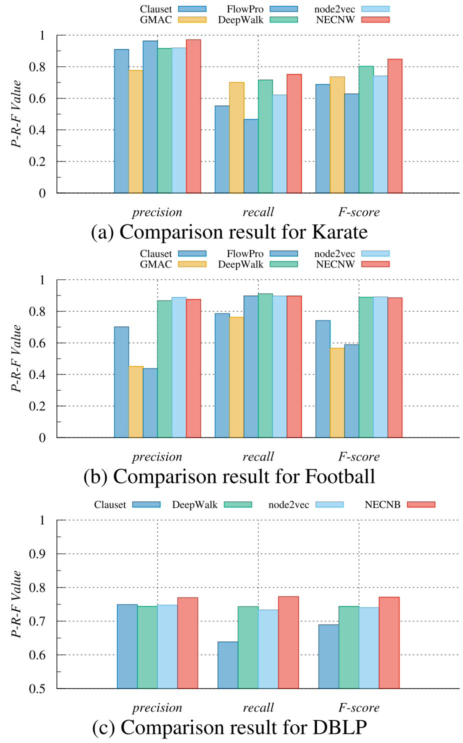
### A. EXPERIMENTAL SETUP

To validate the performance of our algorithm, we compare it with three community search algorithms, Clauset [18], GMAC [21] and FlowPro [31], together with two node embedding algorithms, DeepWalk [33] and node2vec [34].

We perform the experiments on three real-world networks and a group of LFR benchmark networks. Based on the LFR network generating model introduced by Lancichinetti *et al.* [41], we generate a group of 10 LFR

**Algorithm 2** Community Search Based on NECNW

```
Input: network G and its node embedding function f; a
       query node s
Output: target community D of node s
begin
    initialize D = [s], N = Γ(s);
    create a variable dis of map type to store the
    similarities of nodes in N with community D;
    while N ≠ null do
        foreach node v ∈ N do
            //see Formula(11)
            dis[v] = sim_in(v, D);
        end
        find node y such that dis[y] is maximum;
        N.remove(y);
        if CI(D ∪ y) > CI(D) then
            D.append(y);
            foreach node x ∈ Γ(y) do
                if x ∉ D then
                    N.append(x);
                end
            end
        end
    end
    return D;
end
```



(a) Comparison result for Karate
(b) Comparison result for Football
(c) Comparison result for DBLP

**FIGURE 4. Comparison results for real-world networks.**

networks by varying the degree of difficulty for community search.

For the four common parameters used in NECNW, DeepWalk and node2vec: walks per node $r$, walk length $l$, dimension $dn$, and window size $ws$, we set their values as follows. In experiments on real-world networks, we set walks per node $r = 400$, walk length $l = 6$, dimension $dn = 10$, and window size $ws = 2$. In experiments on LFR networks, we set walks per node $r = 10$, walk length $l = 80$, dimension $dn = 100$, and window size $ws = 10$.

For parameter $d$ used in the GMAC algorithm, we set $d = 3$ as suggested by authors [21]. For parameters $p$ and $q$ used in the node2vec algorithm, we set $p = 1$ and $q = 2$, which are adopted by authors in their experiments [34].

### B. EXPERIMENTS ON REAL-WORLD NETWORKS

Zachary's network of karate club members (Karate for short) [42], American college football network (Football for short) [1] and DBLP [43] are well-known benchmark networks for testing the performance of community detection algorithms [37]. Karate describes the friendships among members of a karate club at a US university, in which there are 34 nodes and 78 edges. Football describes American football games between Division IA colleges during the Fall 2000 regular season, in which there are 115 nodes and 613 edges. DBLP is a coauthorship network where two authors are connected if they publish at least one paper together, in which there are 317080 nodes and 1049866 edges.

We repeat the community search experiments on each network $|V|$ times, starting from each node in $V$ once, and report algorithmic average *precision*, *recall*, and *F-score*. The comparison results are reported in Fig. 4.

For the Karate network, NECNW achieves the greatest *precision*, *recall* and *F-score*. The node embedding-based algorithms, DeepWalk and node2vec, also perform better than other community search algorithms.

For the Football network, though node2vec achieves the greatest *F-score*, the differences among node2vec, NECNW and DeepWalk are minimal. Similar to the conclusion for the Karate network, the node embedding-based algorithms perform much better than the other algorithms.

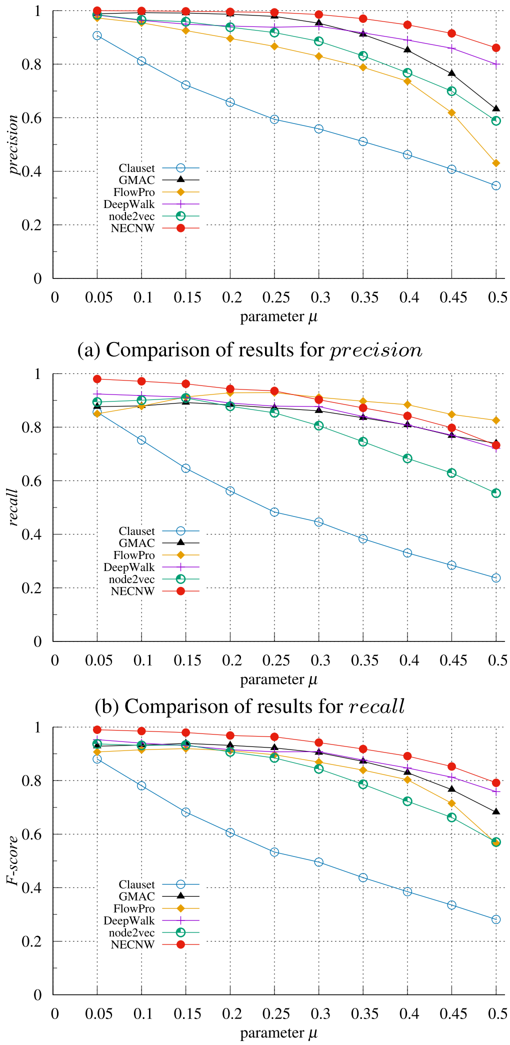
Because the DBLP network is too large for GMAC and FlowPro to handle, we only compare NECNW with Clauset, DeepWalk and node2vec. NECNW achieves the greatest *precision*, *recall* and *F-score*.

In summary, compared with the related algorithms, NECNW achieves the best performance on real-world networks.

### C. EXPERIMENTS ON SYNTHETIC NETWORKS

We first introduce the configuration of the LFR network generating model and then report the experimental results on the synthetic networks.

The LFR network generating model [41] includes four important parameters: the number of nodes $n$, the average degree of nodes $k$, the maximum degree of nodes $k_{max}$ and mixing parameter $\mu$. The mixing parameter $\mu$ of a node is the proportion of the edges outside its community to the total edges associated with it, which is used to control the difficulty




(a) Comparison of results for *precision*

(b) Comparison of results for *recall*

(c) Comparison of results for *F-score*

**FIGURE 5. Comparison of results on LFR benchmark networks.**

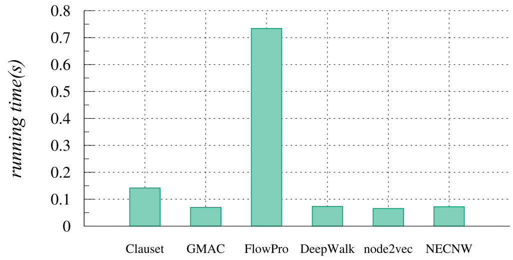



**FIGURE 6. The average running time of a node in the LFR networks.**

of community detection [37]. Therefore, greater $\mu$ values indicate higher difficulty for community search. We set $n = 5000$, $k = 10$, and $k_{max} = 50$ and generate synthetic networks by varying the mixing parameter $\mu$ from 0.05 to 0.5, with a span of 0.05. Thus, we obtain a total of 10 network datasets with different degrees of difficulty for community search.

We repeat the community search experiments on each network 5000 times, starting once from each node, and report algorithmic average *precision*, *recall*, and *F-score*. Fig. 5 shows the experimental results of *precision*, *recall*, and *F-score* on the LFR networks. We obtain the following conclusions from the experimental results.

Along with the growth of mix parameter $\mu$, all six algorithms suffer performance degradation. This observation is consistent with greater $\mu$ values indicating higher difficulty of community search. Among these algorithms, the performance of Clauset declines rapidly; meanwhile, the performances of the other algorithms decline slowly. This is because Clauset only counts the number of edges associated with boundary nodes and neglects the similarity between nodes.

For each of these ten LFR networks, NECNW has the largest *precision* value and achieves the largest *F-score* value, though its *recall* value is less than that of FlowPro when $\mu > 0.25$. In this experiment, the second best algorithm is DeepWalk, the third best algorithm is GMAC, and the fourth best algorithm is FlowPro. The average *F-score* of NECNW is 4.33% larger than that of DeepWalk, 5.75% larger than that of GMAC, and 9.40% larger than that of FlowPro.

In summary, compared with the related algorithms, NECNW achieves the best performance on the synthetic networks.

### *D. DISCUSSION OF ALGORITHMIC EFFICIENCY*

In this subsection, we discuss the efficiency of the proposed algorithm. We run community search experiments 50000 times, starting from each node in the LFR networks, and report the average running time of a node. The comparison results are shown in Fig. 6.

Among these algorithms, FlowPro is the slowest algorithm, and its average running time is 0.73 s per node. NECNW, node2vec, DeepWalk and GMAC are the most efficient algorithms, and their average running times are approximately 0.07 s per node. However, the effectiveness of DeepWalk, node2vec and GMAC is not as good as that of NECNW.

## VI. CONCLUSION AND FUTURE WORK

In this paper, we study the problem of community search. Community search is an important research problem in the era of big network data. Node embedding-based network representation learning provides a new viewpoint from which to study the community search problem. Inspired by the finding that making use of the closeness of nodes could enhance the accuracy of detecting community structure, we propose a node embedding model, NECNW, with a *CN*-based random walk based on the skip-gram model. Based on NECNW, we map nodes into points in vector space and propose a two-stage community search algorithm. Our algorithm achieves good performance for both real-world and synthetic networks.

For future work, we will apply the proposed algorithm to social networks and study the node embedding problem in heterogeneous networks.

## REFERENCES


[1] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Apr. 2002.

[2] J. Shan, D. Shen, Y. Kou, T. Nie, and G. Yu, "Approach for hot spread node selection based on overlapping community search," *Ruan Jian Xue Bao/J. Softw.*, vol. 28, no. 2, pp. 326–340, 2017.

[3] Y. Zhang, B. Wu, N. Ning, C. Song, and J. Lv, "Dynamic topical community detection in social network: A generative model approach," *IEEE Access*, vol. 7, pp. 74528–74541, 2019.

[4] Y. Fang, X. Huang, L. Qin, Y. Zhang, W. Zhang, R. Cheng, and X. Lin, "A survey of community search over big graphs," *CoRR*, vol. abs/1904.12539, pp. 1–41, Apr. 2019.

[5] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci. USA*, vol. 103, no. 23, pp. 8577–8582, 2006.

[6] J. Shao, Z. Han, Q. Yang, and T. Zhou, "Community detection based on distance dynamics," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1075–1084.

[7] X. Zhou, K. Yang, Y. Xie, C. Yang, and T. Huang, "A novel modularity-based discrete state transition algorithm for community detection in networks," *Neurocomputing*, vol. 334, pp. 89–99, Mar. 2019.

[8] Y. Fang, R. Cheng, X. Li, S. Luo, and J. Hu, "Effective community search over large spatial graphs," *Proc. VLDB Endowment*, vol. 10, no. 6, pp. 709–720, 2017.

[9] R. Andersen and K. J. Lang, "Communities from seed sets," in *Proc. Int. World Wide Web Conf.*, 2006, pp. 223–232.

[10] M. Sozio and A. Gionis, "The community-search problem and how to plan a successful cocktail party," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Washington, DC, USA, Jul. 2010, pp. 939–948.

[11] I. M. Kloumann and J. M. Kleinberg, "Community membership identification from small seed sets," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 1366–1375.

[12] Y. Wu, R. Jin, J. Li, and X. Zhang, "Robust local community detection: On free rider effect and its elimination," *Proc. VLDB Endowment*, vol. 8, no. 7, pp. 798–809, Feb. 2015.

[13] J. Zhu and C. Wang, "Approaches to community search under complex conditions," *Ruan Jian Xue Bao/J. Softw.*, vol. 30, no. 3, pp. 552–572, 2019.

[14] X. Huang, L. V. S. Lakshmanan, and J. Xu, "Community search over big graphs: Models, algorithms, and opportunities," in *Proc. 33rd IEEE Int. Conf. Data Eng.*, Apr. 2017, pp. 1451–1454.

[15] W. Cui, Y. Xiao, H. Wang, and W. Wang, "Local search of communities in large graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2014, pp. 991–1002.

[16] E. Akbas and P. Zhao, "Truss-based community search: A truss-equivalence based indexing approach," *PVLDB*, vol. 10, no. 11, pp. 1298–1309, 2017.

[17] W. Cui, Y. Xiao, H. Wang, Y. Lu, and W. Wang, "Online search of overlapping communities," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2013, pp. 277–288.

[18] A. Clauset, "Finding local community structure in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 72, no. 2, 2005, Art. no. 026132.

[19] F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," *Web Intell. Agent Syst., Int. J.*, vol. 6, no. 4, pp. 387–400, 2008.

[20] J. Huang, H. Sun, Y. Liu, Q. Song, and T. Weninger, "Towards online multiresolution community detection in large-scale networks," *PLoS One*, vol. 6, no. 8, 2011, Art. no. e23829.

[21] L. Ma, H. Huang, Q. He, K. Chiew, J. Wu, and Y. Che, "GMAC: A seed-insensitive approach to local community detection," in *Proc. DaWaK*, 2013, pp. 297–308.

[22] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *IEEE Data Eng. Bull.*, vol. 40, no. 3, pp. 52–74, Sep. 2017.

[23] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 833–852, May 2018.

[24] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *Eur. Phys. J. B*, vol. 71, no. 4, pp. 623–630, Oct. 2009.

[25] S. E. Schaeffer, "Graph clustering," *Comp. Sci. Rev.*, vol. 1, no. 1, pp. 27–64, 2007.

[26] X. Peng, Z. Yu, Z. Yi, and H. Tang, "Constructing the L2-graph for robust subspace learning and subspace clustering," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1053–1066, Apr. 2016.

[27] X. Peng, J. Feng, J. Lu, W.-Y. Yau, and Z. Yi, "Cascade subspace clustering," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2478–2484.

[28] X. Lan, M. Ye, R. Shao, B. Zhong, P. C. Yuen, and H. Zhou, "Learning modality-consistency feature templates: A robust RGB-infrared tracking system," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9887–9897, Dec. 2019.

[29] R.-H. Li, L. Qin, J. X. Yu, and R. Mao, "Influential community search in large networks," *Proc. VLDB Endowment*, vol. 8, no. 5, pp. 509–520, 2015.

[30] W. Zhao, F. Zhang, and J. Liu, "A novel local community detection algorithm based on common neighbors similarity measurement with weighted neighbor nodes," *J. Nanjing Univ. (Natural Sci.)*, vol. 54, no. 4, pp. 751–757, 2018.

[31] C. Panagiotakis, H. Papadakis, and P. Fragopoulou, "Local community detection via flow propagation," in *Proc. 2015 IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, 2015, pp. 81–88.

[32] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, pp. 1–12, Sep. 2013.

[33] 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*, 2014, pp. 701–710.

[34] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 855–864.

[35] J. Liu, D. Wang, S. Feng, Y. Zhang, and W. Zhao, "Learning distributed representations for community search using node embedding," *Frontiers Comput. Sci.*, vol. 13, no. 2, pp. 437–439, 2019.

[36] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.

[37] S. Fortunato and D. Hric, "Community detection in networks: A user guide," *Phys. Rep.*, vol. 659, pp. 1–44, Nov. 2016.

[38] Y.-J. Wu, H. Huang, Z.-F. Hao, and F. Chen, "Local community detection using link similarity," *J. Comput. Sci. Technol.*, vol. 27, no. 6, pp. 1261–1268, 2012.

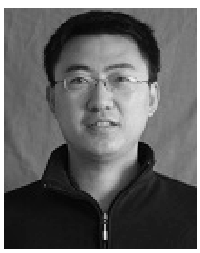
[39] W. Zhao, F. Zhang, and J. Liu, "Local community detection via edge weighting," in *Proc. 12th Asia Inf. Retr. Soc. Conf.*, 2016, pp. 68–80.

[40] D. Nguyen and F. D. Malliaros, "BiasedWalk: Biased sampling for representation learning on graphs," in *Proc. BigData*, Dec. 2018, pp. 4045–4053.

[41] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, no. 4, 2008, Art. no. 046110.

[42] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropol. Res.*, vol. 33, no. 4, pp. 452–473, 1977.

[43] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 181–213, 2015.

**WEIJI ZHAO** received the B.S. degree in computer science and technology from Qufu Normal University, China, in 2004, and the M.S. degree in computer application technology from the Liaoning University of Technology, China, in 2007. He is currently pursuing the Ph.D. degree in computer system structure with the Harbin University of Science and Technology. Since 2007, he has been a Teacher with Suihua University. His research interests include social computing, social network analysis, and data mining.

**FENGBIN ZHANG** received the Ph.D. degree in computer application from Harbin Engineering University, China, in 2005. He is currently a Supervisor and a Professor with the Harbin University of Science and Technology. He has presided over the conclusion of two National Natural Science Foundation of China projects. He is the author of more than 70 articles. His current research focuses on network and information security, as well as intrusion detection technology.