Hindawi
Complexity
Volume 2021, Article ID 6673444, 13 pages
https://doi.org/10.1155/2021/6673444

WILEY | Hindawi

Research Article

An Approach of Community Search with Minimum Spanning Tree Based on Node Embedding

Jinglian Liu,[1,2] **Daling Wang**,[1] **Shi Feng**,[1] **and Yifei Zhang**[1]

[1]*School of Computer Science and Engineering, Northeastern University, Shenyang 110169, China*
[2]*School of Information Engineering, Suihua University, Suihua 152061, China*

Correspondence should be addressed to Daling Wang; wangdaling@cse.neu.edu.cn

Received 5 December 2020; Revised 22 February 2021; Accepted 14 March 2021; Published 15 April 2021

Academic Editor: Hocine Cherifi

Community search is a query-oriented variant of community detection problem, and the goal is to retrieve a single community from a given set of nodes. Most of the existing community search methods adopt handcrafted features, so there are some limitations in applications. Our idea is motivated by the recent advances of node embedding. Node embedding uses deep learning method to obtain feature representation of nodes directly from graph structure automatically and offers a new method to measure the distance between two nodes. In this paper, we propose a two-stage community search algorithm with a minimum spanning tree strategy based on node embedding. At the first stage, we propose a node embedding model NEBRW and map nodes to the points in a low-dimensional vector space. At the second stage, we propose a new definition of community from the distance viewpoint, transform the problem of community search to a variant of minimum spanning tree problem, and uncover the target community with an improved Prim algorithm. We test our algorithm on both synthetic and real-world network datasets. The experimental results show that our algorithm is more effective for community search than baselines.

1. Introduction

Community detection is one of the most popular problems in social network analysis, and its goal is to identify all communities in a network [1–3]. Discovering communities in social networks may offer insight on how the networks are organised and have many applications [4, 5]. However, there are many application scenarios in which we are interested in a particular community instead of all communities in networks. For example, in a scholar network such as DBLP, we are interested in a group of data mining experts, but not all experts in the community are known [6]. As another example in recommendation systems, for offering a tourist the most relevant and personalized local venue recommendations, his local interesting community needs to be mined first [7]. Both of the above examples are query-oriented variant of community detection problem where only a single community shall be detected [8–10]. The community search problem has also been studied as local community detection [11, 12] or seed set expansion [6, 13, 14].

The traditional community detection methods aim to enumerate all the communities in a network, and the running time is proportional to the size of the entire graph; thus, their efficiency is inadequate for community search which aims to find a particular community [15]. To address this limitation, a lot of research studies have been devoted in the community search problem. Luo et al. [16], Huang et al. [17], Ma et al. [18], and Liu et al. [19] study the scenario in which only a query node requires to be pre-known in the target community C, but sometimes they perform poorly since they have no size limitation of the algorithmic returned results. Kloumann and Kleinberg [6] and Clauset [20] study the scenario in which a researcher need to pre-know the number of members in target community C, and Kloumann and Kleinberg [6] make a further assumption that a node set of size $|C|/10$ in C also requires to be pre-known which is hard to set in real application.

In this paper, we focus on a particular case of community search problem: for a graph G, given a node $s \in G$, the goal is to find k nodes which are in the same community with s.

Motivated by node embedding providing a new approach to learn node features from graph structure directly, we propose a two-stage community search algorithm. At the first stage, we propose a **N**ode **E**mbedding model with a **B**iased **R**andom **W**alk (NEBRW) based on the Skip-gram model and map nodes to the points in a low-dimensional vector space. Moreover, we transform the proximity between each pair of nodes into their distance. At the second stage, we define the community of a query node as the nodes which are connected via shortest distance. Therefore, the problem of community search is transformed to a variant of minimum spanning tree problem. For the purpose, we define a new measurement of the distance between two nodes and implement a new community search algorithm with a minimum spanning tree-based approach.

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

We propose a node embedding model NEBRW based on the Skip-gram model and map nodes to the points in a low-dimensional vector space. Moreover, we define a new measurement of the distance between two nodes.

We propose a new definition of community from distance viewpoint: each community is a group of nodes which are connected via the shortest distance and transform the problem of community search to a variant of minimum spanning tree problem.

Based on the above definition, we design a novel **C**ommunity **S**earch algorithm with a **M**inimum **S**panning **T**ree approach (CSMST) and test the algorithm on both synthetic and real-world network datasets. The experimental results show that our algorithm is more effective at community search than baselines.

The rest of the paper is organised as follows. Section 2 introduces some related work. We give a formal definition of the community search problem in section 3 and give the detailed algorithm in section 4. We report experimental results in section 5, followed by conclusions in section 6.

2. Related Work

Our work is partly inspired by the work on community search and partly by the work on node embedding. In this section, we review both lines of work below.

2.1. Community Detection and Community Search. Community detection is an interesting problem in social network analysis, and various types of algorithms have been proposed, including modularity maximization model [3], hierarchical clustering model [21], and distance dynamics model [22]. The goal of community detection is to enumerate all the communities in a network, and the recent work is reviewed in the literature [2, 4]. Community search is a query-oriented variant of community detection problem, and the goal is to obtain a single community from a given set of nodes [8, 10]. The traditional community detection methods aim to enumerate all the communities in a network; thus, their efficiency is inadequate for community search.

Community search has attracted a lot of attention, and lots of algorithms have been proposed. However, the problem definition is not in complete accord. Among them, a mainstream direction of efforts focuses on querying the community from a query node. Luo et al. [16] define a local modularity M and identify the subgraph with the maximum value of M starting from a query node with a locally optimized approach. Huang et al. [17] introduce a similarity-based community quality function tightness and design an algorithm LTE for revealing the natural community of a query node via local optimization of the tightness measure. Different from Huang's similarity measure which only focuses on the adjacent nodes, Ma et al. [18] introduce a d-NS measure which also takes into account nonadjacent vertices within a distance away and propose a d-NS based community search algorithm. In addition, Clauset [20] and Panagiotakis et al. [23] also assume that the approximate size of target community requires to be pre-known. Clauset [20] defines a local modularity measure R and proposes an algorithm to identity the community with a fixed number of nodes by maximizing R in a greedy fashion. Panagiotakis et al. [23] propose a flow propagation algorithm FlowPro to find the community surrounding a query node. Another direction of efforts focuses on finding the community from a set of query nodes. Kloumann and Kleinberg [6] study the scenario in which a researcher needs to pre-know an initial node set of size $|C|/10$ from C.

Furthermore, there is another kind of minimum spanning tree-based community detection algorithms. Saoud and Moussaoui [24] construct the minimum spanning tree of the network based on the dissimilarities of nodes for each edge and get groups of nodes by removing the highest edges dissimilarities, and then, they merge group pairs to identify the final community structure maximizing the modularity. In order to overcome the limitation of modularity maximization, Asmi et al. [25] propose a new algorithm to reveal the communities in social networks based on minimum spanning tree and the strength of similarity between two nodes.

2.2. Node Embedding. The key challenge in networked data mining is how to find a proper representation of network structure that can be exploited by downstream tasks [19, 26]. Most of the existing data mining models are designed to handle vectorized data, and the networked data cannot be directly input into these models. Node embedding enables the automatic discovery of vector representation of nodes directly from graph structure [27], and the relevant work is reviewed in the literature [28–30]. Besides homogeneous networks, there are also some heterogeneous networks [31] based on embedding approaches [27, 32, 33] proposed in recent years.

According to literature [30], network graph embedding output includes node embedding, edge embedding, hybrid embedding, and whole-graph embedding. Our work belongs to node embedding. The most related work to ours is the

word2vec-based node embedding algorithms, such as DeepWalk [34], node2vec [35], and NEMCNB [19]. By viewing nodes as words and random walk paths on networks as sentences, these methods generalize word embedding techniques in natural language processing from lists of sentences to graphs [14]. These algorithms usually include two steps. Firstly, node paths are generated by performing random walks on a network. Secondly, vector representation of nodes is learned by adopting word embedding technique.

Recently, research on incorporating node embedding into community detection has attracted great interest of scholars. A line of research is to learn low-dimensional vector representations of nodes from network topology and feed them as node features to clustering algorithms such as k-means. For improving the community detection accuracy, Jin et al. [36] define a new pairwise Markov Random Field framework which not only utilizes network embedding but also uses network topology to adjust of the improper division of nodes.

Motivated by the above work, we propose a new node embedding model to learn vector representation of nodes and design a minimum spanning tree-based community search algorithm.

3. Problem Definition and Solution Approach

A network can be represented by graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. A community in network G is a subgraph within which the nodes are in close proximity. The general community search problem is defined as follows.

Problem 1. (community search). For a given network G, we are interested in a potential community $C \subset G$, but pre-know only a member $s \in C$, and the goal is to find out the members in C.

The traditional way of quantifying the quality of a community focuses on the density of internal edges, e.g., local modularity M [16] is the ratio of the number of internal edges to external edges and R [20] is the fraction of boundary edges which are internal to the community. In this paper, we quantify the quality of a community with the shortest distance connecting all nodes in it from a viewpoint of distance. Based on this, we formally define the community search problem based on minimum spanning tree as follows.

Problem 2. (community search based on minimum spanning tree). For a network $G = (V, E)$ and distance function dis between nodes, given a query node $s \in V$ and a size constraint k, we seek to find an induced subgraph $H = (V_H, E_H)$ of G, such that

(1) H contains s

(2) $|V_H| = k$

(3) H is connected

(4) $\sum_{(u,v)\in E_H} dis(u, v)$ is minimized among all feasible choices for H

We now discuss the problem of finding subgraph H. Firstly, H is a connected subgraph, and there are at least $k - 1$ edges in it. For a connected subgraph H, there are at most $k - 1$ edges in it when $\sum_{(u,v)\in E_H} dis(u, v)$ is minimized. We put these two things together and get the conclusion that there are exactly $k - 1$ edges in H. Secondly, suppose a subgraph $H' = (V_{H'}, E_{H'})$ and $V_{H'} = V_H$, for any nodes $u, v \in V_{H'}$, if $(u, v) \in E$, then $(u, v) \in E_{H'}$. When $\sum_{(u,v)\in E_H} dis(u, v)$ is minimized, it means that connecting all nodes in H' with shortest total length; thus, H is the minimum spanning tree of graph H'.

Based on the above discussion, the community search problem is transformed into a variant of minimum spanning tree problem which starts from node s and contains only k nodes. However, our problem is different from the classic minimum spanning tree [37, 38]. The problem definition of classical minimum spanning tree is described as follows. Given a set of nodes, connect them by a network having the smallest sum of the edge lengths [38]. It differs from our problem in two aspects. The first aspect is that the classic minimum spanning tree aims to connect all nodes in network G with shortest edge lengths, while we aim to connect the nodes in H' which is a small part of G. The second is that the nodes in G are pre-known, but we do not know which nodes belong in H' (H) except that node s belongs in it and there are k nodes in it. Thus, we cannot adopt the classic minimum spanning tree algorithms directly. In our solution, we design an improved Prim algorithm to solve this problem.

To solve Problem 2, we propose a two-stage community search algorithm CSMST which includes node embedding representation and community search. The illustration is shown as Figure 1.

At the first stage, we focus on the representation of complex networks. How to represent networked data is an important aspect when we apply data mining techniques to analyze network datasets. Instead of traditional handcrafted feature extraction based on domain experts' knowledge, we learn vector representation of nodes automatically from the graph structure via node embedding technique.

At the second stage, we focus on community search problem. Based on the vector representation of nodes obtained at the first stage, we define a distance measurement between pairs of nodes. We treat the community of a query node s as the node set connected via shortest distance and implement a community search algorithm with a minimum spanning tree approach.

4. The Algorithm of CSMST

CSMST is an algorithm with two stages. At the first stage, we focus on the network representation problem and propose a Node Embedding model with a Biased Random Walk (NEBRW) to learn low-dimensional vector representations for nodes. Moreover, a distance measurement between nodes based on their vector representations is given. At the second stage, we identify the target community of a query node with a variant minimum spanning tree approach.

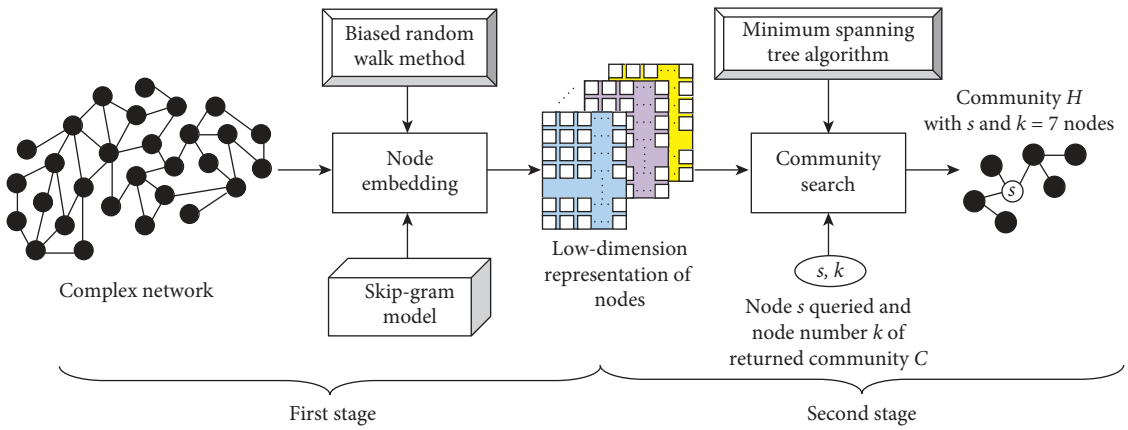

Figure 1: An illustration of CSMST.

4.1. Our Node Embedding Algorithm. This is the first stage of our CSMST process. We first introduce a Skip-gram model for network and then give our node embedding model NEBRW.

4.1.1. Skip-Gram Model for Network. Given a network $G = (V, E)$, the goal of node embedding is to learn a mapping from nodes to a low-dimensional space, $f: V \longrightarrow R^d$, where $d \ll |V|$ and each node u in V is associated with a real-valued d-dimensional vector $f[u]$. By viewing nodes as words and random walk paths on G as a corpus, we can learn the embedding of nodes via Skip-gram model [39].

Given a random walk $walk = [v_1, v_2, \ldots, v_l]$, the context of node v_i, denoted as $C(v_i)$, is the nodes in a window of size w centered at v_i, i.e., $C(v_i) = [v_{i-w}, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{i+w}]$. The embedding of nodes is learned by the maximizing objective function:

$$\max_f \sum_{v_i \in V} \log p\left(C\left(v_i\right) | f\left(v_i\right)\right) \tag{1}$$

where we assume that the nodes in $C(v_i)$ is independent of each other; thus, equation (1) can be expressed as

$$\max_f \sum_{v_i \in V} \prod_{u \in C\left(v_i\right)} \log p\left(u | f\left(v_i\right)\right) \tag{2}$$

The learned vectors are expected to be able to preserve as many properties of the network as possible; thus, they can be an alternative to traditional handcrafted features extracted from the graph [40].

4.1.2. Node Embedding Model NEBRW. In this section, we introduce our node embedding model NEBRW, a method for learning low-dimensional vector representations of nodes in a network based on the Skip-gram model. We learn node representations from a network in two steps, and the process is shown in Figure 2.

In line 2 to 10, by simulating random walk on G of fixed length l starting from each node r times, we get a list of node paths. In line 11 to 13, by viewing nodes as words and random walk paths on G as a corpus, we leverage the Skip-gram model to learn vector representations of nodes in G. The detailed node embedding algorithm is shown in Algorithm 1.

The main difference among NEBRW, DeepWalk, and node2vec is that they adopt different random walk strategies. DeepWalk [34] uses pure random walk over networks. node2vec [35] adopts a biased random walk method by capturing the first-order and second-order proximity between nodes. In detail, we adopt a closest-neighbor biased random walk method [19] in the NEBRW model. Formally, we use $[v_1, v_2, \ldots, v_l]$ to denote a random walk of fixed length l, and v_i is the ith node in the walk. In the process of a random walk, suppose the current node is v_i, and $\Gamma(v_i)$ is the neighbor node set of v_i. We use additional information w_{xv_i} for the neighbor node x of v_i in order to estimate the proximity between v_i and x:

$$w_{xv_i} = \frac{\left|\Gamma(x) \cap \Gamma\left(v_i\right)\right|}{\left|\Gamma(x) \cup \Gamma\left(v_i\right)\right|}. \tag{3}$$

The probability of a neighbor node x being the next node v_{i+1} is proportional to, i.e.,

$$p_x = \frac{w_{xv_i}}{\sum_{u \in \Gamma\left(v_i\right)} w_{uv_i}}. \tag{4}$$

The detailed algorithm of biased random walk is shown in Function rw (Algorithm 2).

After mapping nodes to points in a low-dimensional vector space, the proximity of nodes can be measured by their distance. The distance between two nodes grows in inverse proportion to their similarity. Therefore, the distance dis between nodes u and v is defined as follows:

$$dis(u, v) = \begin{cases} 1 - \$ f[u] \cdot f[v] \$, & (u, v) \in E, \\ \infty, & (u, v) \notin E, \end{cases} \tag{5}$$

where $f[u] \cdot f[v]$ is the dot product of $f[u]$ and $f[v]$, which is the proximity score between nodes u and v.

4.2. The Algorithm of Community Search with Minimum Spanning Tree. This is the second stage of our CSMST process. Based on the learned vector representations of

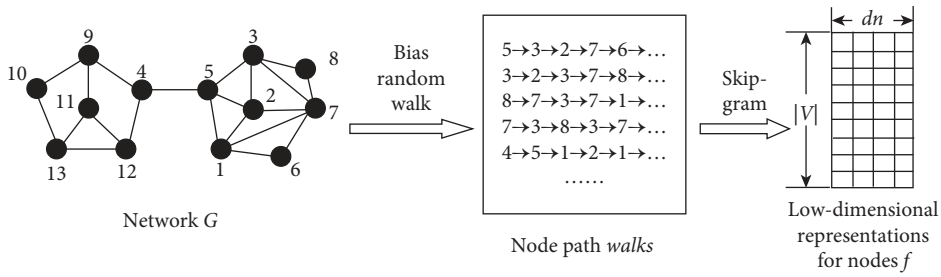

Figure 2: The illustration of NEBRW.

Input: a 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 *walks* =
  $loop = 0$
  while $loop < r$ **do**
    foreach $v \in V$ **do**
    //rw is a random walk function
      $walk = \mathrm{rw}(G, v, l)$
      append *walk* into *walks*
    end
    $loop+ = 1$
  end
  construct a corpus T consisting of $r * |V|$ sentences which are stored in *walks*
  use the Skip-gram to learn the mapping f by treating *walks* as a corpus
  return f
end

Algorithm 1: Node embedding model NEBRW.

Function $\mathrm{rw}(G, u, l)$
Input: network G; start node u; walk length l
Output: node path *walk*

begin
  Initialize $walk = [u]$;
  $i = 1$;
  while $i < l$ **do**
    $cur = walk[i]$;
    nbrs_weight =;
    foreach $v \in \Gamma(cur)$ **do**
      $nbrs\_weight[v] = p_v$;
    end
    randomly select a node in $\Gamma(cur)$ with the probabilities in *nbrs_weight*, denoted as x;
    add x to *walk*;
    $i++$;
  end
  return *walk*;
end

Algorithm 2: Biased random walk function rw.

nodes, we compute the distance between pairs of nodes by formula (5) and propose a novel community search algorithm CSMST.

In what follows, we give the description of CSMST. According to the analysis of Problem 2, we identify the target community of a query node s by constructing a minimum spanning tree H with k nodes. We initialize a target community $H = \{s\}$, a shell node set $N = \Gamma(s)$, and expand H by iteratively adding the node $b \in N$, which has shortest length with the current subgraph H, at a time until its node number reaches k. Figure 3 shows an example of our minimum spanning tree algorithm. Starting from a query node s in G, a compact subgraph H with 8 nodes is discovered by constructing a minimum spanning tree. The pseudocode of CSMST is shown in Algorithm 3.

5. Experiments

In this section, we evaluate the effectiveness of our community search algorithm CSMST on synthetic as well as real-world networks.

5.1. Experiment Setup. To validate the performance of CSMST, we give the experiment setup in this section.

5.1.1. Baselines. We compare CSMST against the following five community search algorithms for proving the advantage of our community search method:

(1) Clauset's algorithm [20]: this is a classical community search algorithm which discovers the target community by maximizing metric R.

(2) GMAC [18]: this is a classical similarity-based community search algorithm which uses S_{xy}^{d} as the node similarity measurement. We fix $d = 3$ in the following experiments as suggested by the authors.

(3) FlowPro [23]: this is a representative community search algorithm based on flow propagation. When the algorithm converges, the flow stored in the nodes that belong to the community of query node is higher than that stored in the nodes of other communities. The top k nodes with higher flow stored are chosen as the predicted community.

(4) NEMCNB [19]: it is a recent proposed community search algorithm which discovers the target community by adding a node iteratively from the shell node set to the target community that has the largest similarity with nodes in the current community. The purpose of choosing NEMCNB as a baseline is to evaluate the effectiveness of retrieving communities with a minimum spanning tree strategy.

(5) MSTW [25]: this is a community detection algorithm based on minimum spanning tree and the strength of similarity between two nodes W proposed by Asmi et al. The purpose of choosing MSTW as a baseline is to evaluate the effectiveness of node similarity measurement using node embedding.

For fair comparison with the other community search algorithms, a few modifications are required. GMAC, NEMCNB, and MSTW do not specify the number of nodes k to be added to the predicted community as the stopping condition. Thus, we naturally choose the top k members from algorithmic result as the predicted community. We also compare our node embedding model NEBRW in CSMST against the following two embedding baselines:

(1) DeepWalk [34]: this is the first node embedding algorithm which generalized the advancements of word embedding in natural language processing from sequences of words to graphs.

(2) node2vec [35]: this is another node embedding algorithm based on a biased random walk procedure that can explore neighborhoods in a BFS as well as DFS fashion. For learning representations where nodes that are close in the original network have similar embeddings, we set $p = 1$ and $q = 2$ in the following experiments which are also adopted by the authors in their experiments.

In the experiments, we use DeepWalk and node2vec to learn the vector representations of nodes and then retrieve communities with the minimum spanning tree strategy which is adopted by CSMST. The purpose of choosing DeepWalk and node2vec is to evaluate the effectiveness of our node embedding method NEBRW.

5.1.2. Datasets and Evaluation Metrics. We employ both synthetic and real-world networks for the evaluations. The widely used synthetic benchmark for community detection is a class of LFR benchmark networks introduced by Lancichinetti et al. [41]. We generate four groups of LFR benchmark networks, and in each group, there are ten networks.

In addition, we use four real-world network datasets to evaluate the performance of the community search algorithms. (1) Zachary Karate Club Network (Karate for short) [42], in which there are 34 nodes and 78 edges, describes the friendships among 34 members of a karate club at a US university. (2) NCAA football network (Football for short) [1], in which there are 115 nodes and 613 edges, describes American football games between Division IA colleges during regular season Fall 2000. (3) Books about US politics network (Polbooks for short) [43], in which there are 105 nodes and 441 edges, is a network of books about US politics published around the time of the 2004 presidential election and sold by Amazon.com. (4) YouTube social network (YouTube for short) [44], in which there are 1134890 nodes and 2987624 edges, is a video-sharing website that includes a social network.

Both the synthetic and real-world networks have ground-truth community structure. In the experiments, we set the same query node s and the same number of returned nodes k for different algorithms. If an algorithmic result contains more corrected nodes, it will obtain a higher value of recall and precision:

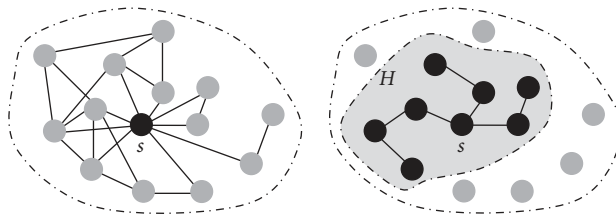

Figure 3: An example of the improved Prim algorithm. (a) Graph G and a query node s. (b) A minimum spanning tree H in G with 8 nodes starting from s.

Input: network G and its node embedding f; query node s; expected number of returned nodes k
Output: target community D
begin
  initialize $D = [s]$, $N = \Gamma(s)$
  define a variable *dis* of map type
  foreach *node* $v \in N$ **do**
    $dis[(s, v)] = 1 - f[s] \cdot f[v]$
  end
  while $len(dis) > 0$ and $|D| < k$ **do**
    find edge (a, b) such that $dis[(a, b)]$ is minimum
    add b to D
    foreach *node*
    $y \in \Gamma(b)$ **do**
      if $y \in D$ **then**
        del $dis[(y, b)]$
      end
      else
        $dis[(b, y)] = 1 - f[b] \cdot f[y]$
      end
    end
  end
  return D
end

Algorithm 3: CSMST.

$$\text{recall} = \frac{|C \cap D|}{|C|}, \qquad \text{precision} = \frac{|C \cap D|}{k}, \tag{6}$$

where C is the set of nodes in ground-truth community of the query node and D is the set of nodes obtained by community search algorithm. In the experiments, we set $k = |C|$, and the value of recall is equal to precision. So, we use the evaluation metric recall to compare algorithmic performance.

5.2. Evaluation on Synthetic Networks. The parameters of the LFR network generating model are introduced as follows: the number of nodes n, the average degree of nodes k, the maximum degree of nodes $k_{\max}$, and others except mixing parameter μ are set to their default values. Mixing parameter μ is the fraction of the number of edges of each node outside its community, which is used to control the difficulty of community detection [4], and larger μ would result in lower community detection accuracy.

We generate four groups of LFR networks by varying parameter n and mixing parameter μ. In each group of LFR networks, we vary mixing parameter μ from 0.05 to 0.5 with a span of 0.05 and get ten networks. The detailed parameter values are set as Table 1. There are total forty networks with ground-truth communities.

For experiments of each algorithm on each dataset, we repeat the community search experiments for n (n is the number of nodes in the network) times which start from each node at a time, and then report algorithmic average recall on this dataset. We evaluate our algorithm on these four groups of LFR network datasets, together with five community search baselines and two node embedding methods. We set the common parameter values as following for NEMCNB, CSMST, DeepWalk, and node2vec: walks per node $r = 10$, walk length $l = 80$, dimension dn = 100, and window size ws = 10. LFR30K

Table 1: LFR datasets.

Group name	n	k	$k_{\max}$	μ
LFR5K	5000	10	50	0.05, 0.1, . . ., 0.5
LFR10K	10000	10	50	0.05, 0.1, . . ., 0.5
LFR30K	30000	10	50	0.05, 0.1, . . ., 0.5
LFR50K	50000	10	50	0.05, 0.1, . . ., 0.5

and LFR50K networks are too big for FlowPro to handle because of its high time complexity. Thus, we only compare CSMST with the other baselines on these two groups of networks. The experimental results are shown in Figures 4 and 5, respectively, and we can get the following conclusions.

Firstly, combining Figures 4 and 5, we can discover that increasing mixing parameter μ leads to performance degradation due to increased difficulty of community detection. This is because the higher the mixing parameter μ of a network, the weaker community structure it has. Empirical studies of the community search algorithms on the four groups of LFR networks verify this.

Secondly, Figure 4 shows that, with the increase of μ, the performances of Clauset and MSTW drop rapidly; meanwhile, the other algorithms drop slowly. Compared with the other five community search baselines, CSMST algorithm achieves the best performance on the four groups of LFR networks, followed by NEMCNB, FlowPro, and GMAC. The main difference between CSMST and NEMCNB is that the community expansion strategy adopted is different, and the comparison results show that minimum spanning tree strategy is better than similarity-based strategy. The main difference between CSMST and MSTW is that the node similarity measurement adopted is different, and the comparison results show that the node similarity measurement based on node embedding is better than that based on network structure.

Thirdly, Figure 5 shows that our node embedding algorithm NEBRW is better than DeepWalk and node2vec in community search experiments on LFR networks.

5.3. Evaluation on Real-World Networks. We adopt the same experimental method on real-world networks as that, on synthetic networks and report algorithmic average recall on these datasets. Firstly, we perform the experiments on Karate, Football, and Polbooks. The common parameters are set as the following for NEMCNB, CSMST, DeepWalk, and node2vec: walks per node $r = 400$, walk length $l = 6$, dimension dn = 10, and window size ws = 2. The comparison results with both community search and node embedding baselines on these real-world network datasets are reported in Figures 6 and 7, respectively.

Then, we perform the experiment on YouTube. The common parameters are set as follows: $r = 10$, $l = 30$, dn = 100, and ws = 3. We compare with Clauset, DeepWalk, and MSTW because the network is too big for other algorithms to handle due to their high time complexity. The comparison results are reported in Figure 8.

Compared with the other five community search baselines, we can see that CSMST algorithm achieves the best performance on Karate, Football, and YouTube datasets. On Polbooks, MSTW achieves the best performance; however, the difference among MSTW, CSMST, Clauset, and FlowPro is small.

Compared with DeepWalk and node2vec, NEBRW achieves the best performance on Karate and Polbooks datasets. And, on Football dataset, DeepWalk algorithm achieves the best performance; however, the differences among DeepWalk, NEBRW, and node2vec are small. This further proves that NEBRW model and CSMST algorithm have greater advantage in community search tasks.

5.4. Discussion of Parameter k. Parameter k is important in the definition of community search problem, and it is interpreted as the number of nodes in the target community. However, there are some scenarios in which we do not pre-know it. In this section, we discuss the effect of parameter k in CSMST algorithm. We choose LFR5K($\mu = 0.2$), LFR5K($\mu = 0.35$), and LFR5K($\mu = 0.5$) as the test network datasets. We perform experiments by varying parameter k from $0.2n$ to $1.8n$ with a span of $0.4n$ (n is the number of nodes in the ground-truth community). The larger k will return more corrected nodes, which leads to a higher recall value but a lower precision value. Thus, in addition to the recall and precision metrics, we also adopt $F-$score to measure algorithmic performance of different values of k. The experimental results are shown in Figure 9:

$$F-\text{score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}. \tag{7}$$

We discuss the experimental results in detail. Firstly, with the increasing of mixing parameter μ, the difficulty of community detection on correspondent LFR networks is increasing. The experimental results verify this point. Secondly, on each test network dataset, the experimental results show a consistent pattern that, with the increase of parameter k, the values of precision metric decrease, the values of recall metric increase, and the values of $F-$score increase first and then decrease. The larger the parameter k is, the more nodes are returned; thus, the recall metric shows an upward tendency and the precision metric shows a downward tendency. The $F-$score achieves the maximal value when the value of parameter k is equal to the number of nodes in target community.

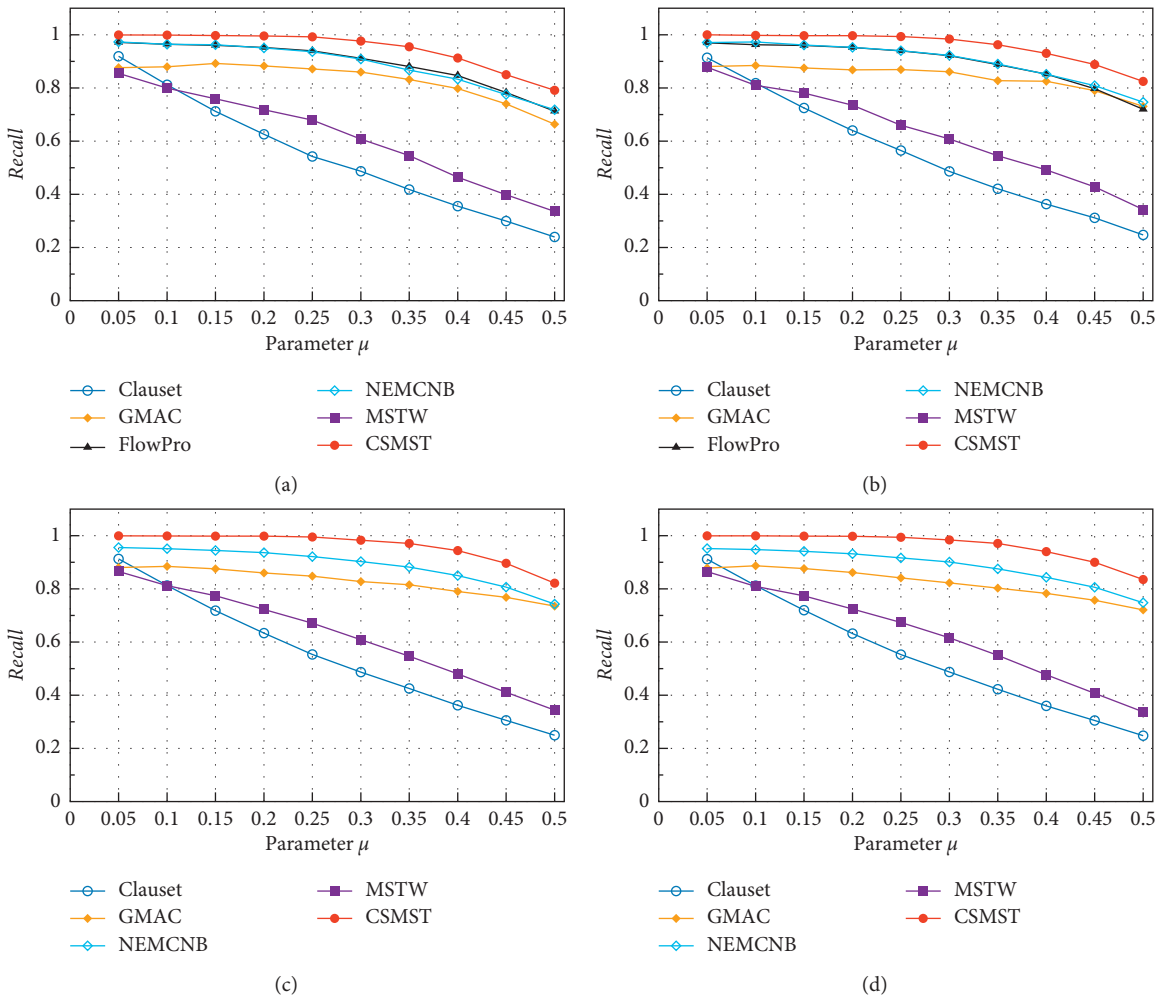

Figure 4: Comparison results on LFR networks with community search baselines. (a) Comparison results on LFR5K. (b) Comparison results on LFR10K. (c) Comparison results on LFR30K. (d) Comparison results on LFR50K.

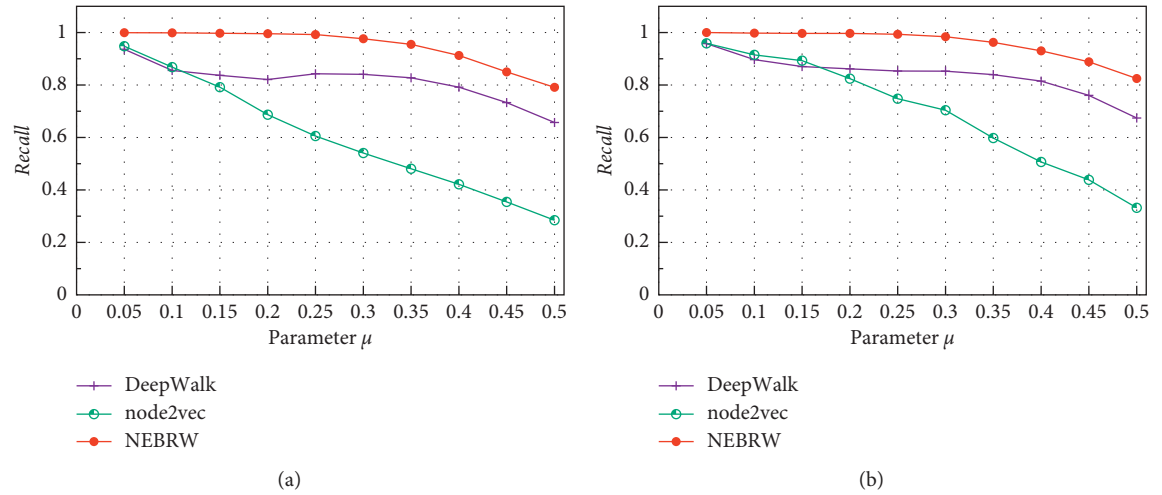

Figure 5: Continued.

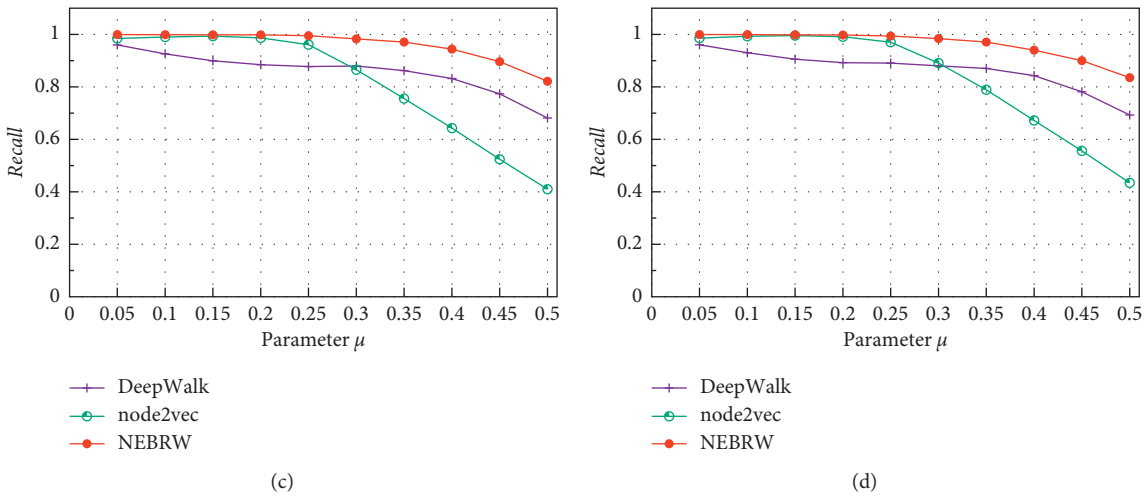

Figure 5: Comparison results on LFR networks with node embedding baselines. (a) Comparison results on LFR5K. (b) Comparison results on LFR10K. (c) Comparison results on LFR30K. (d) Comparison results on LFR50K.

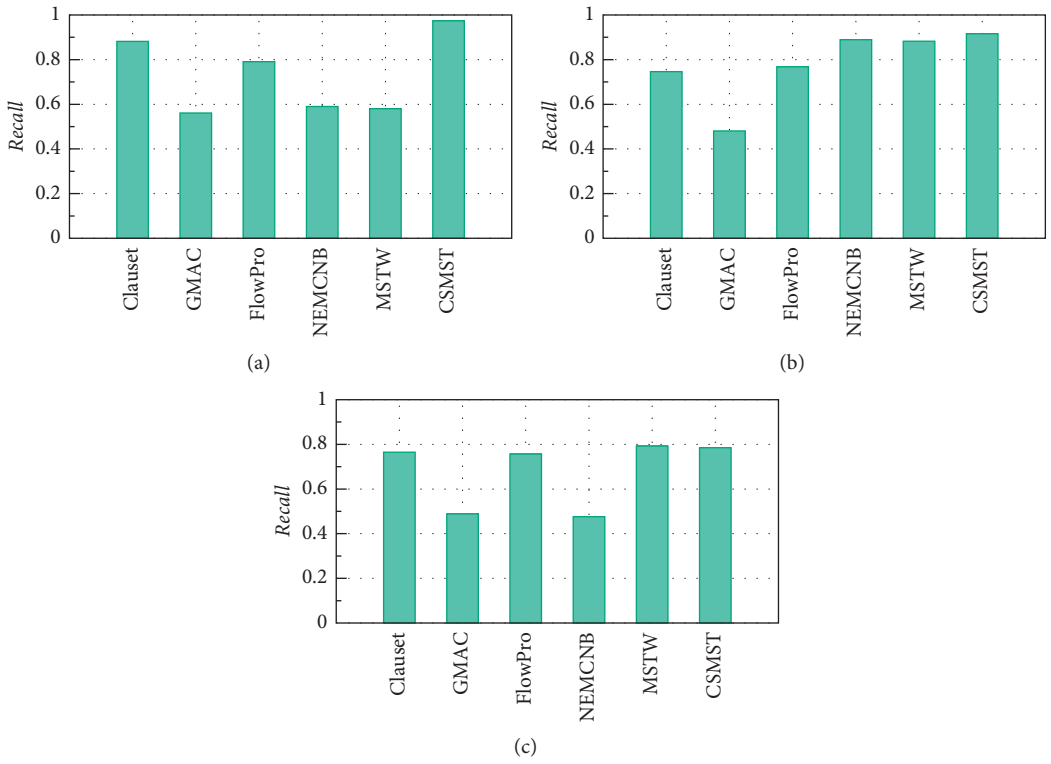

Figure 6: Comparison results on real-world networks with community search baselines. (a) Comparison result on Karate. (b) Comparison result on Football. (c) Comparison result on Polbooks.

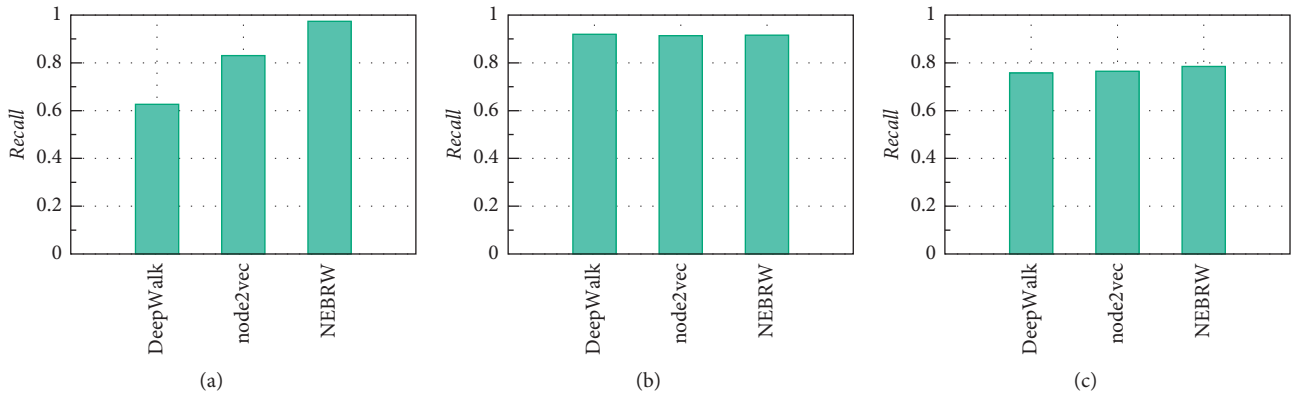

Figure 7: Comparison results on real-world networks with node embedding baselines. (a) Comparison result on Karate. (b) Comparison result on Football. (c) Comparison result on Polbooks.

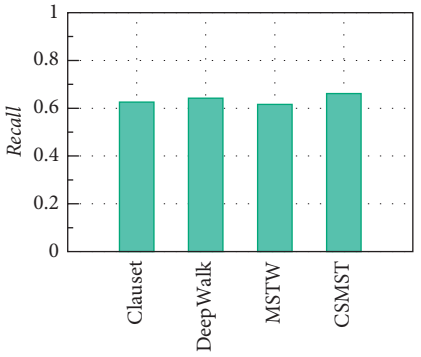

Figure 8: Comparison results on YouTube network.

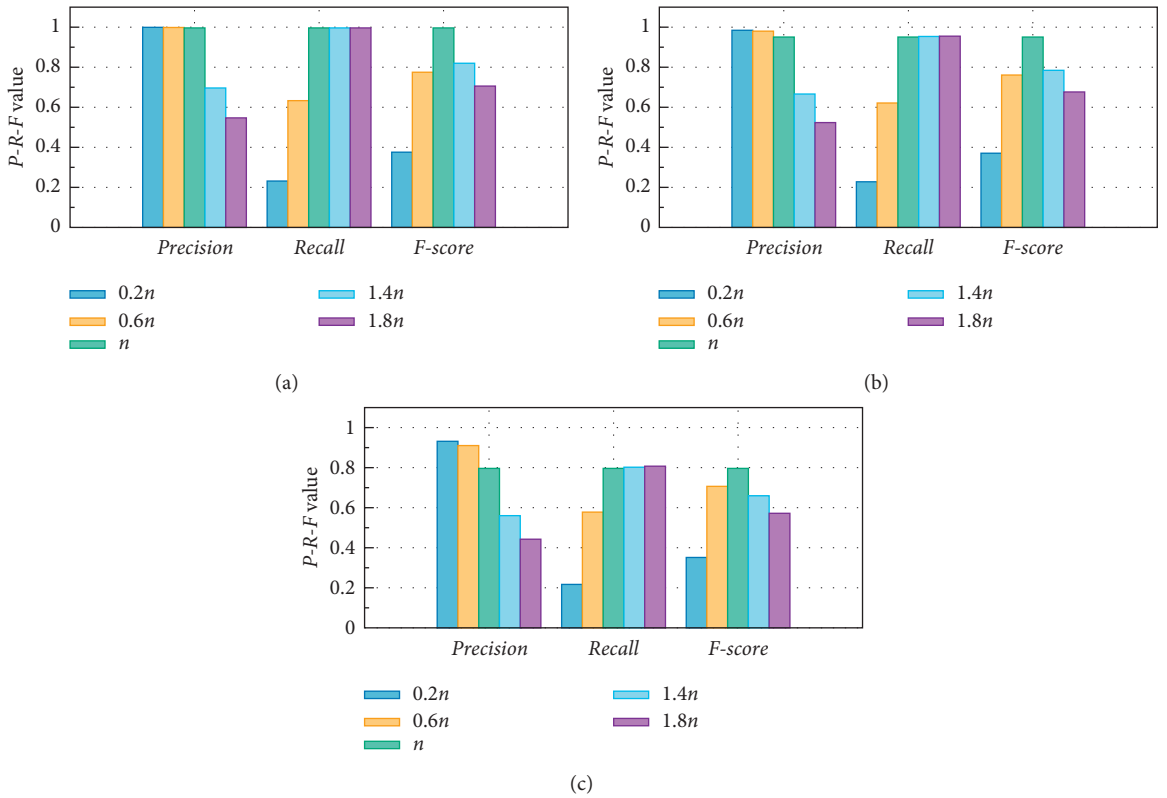

Figure 9: Experimental results of varying parameter k. (a) Experimental result with different k ($0.2n$~$1.8n$) on LFR5K ($\mu = 0.2$). (b) Experimental result with different k ($0.2n$~$1.8n$) on LFR5K ($\mu = 0.35$). (c) Experimental result with different k ($0.2n$~$1.8n$) on LFR5K ($\mu = 0.5$).

6. Conclusion and Future Work

In this paper, we study communities from the viewpoint of distance and transform community search problem into a variant of minimum spanning tree problem. Moreover, we propose a node embedding model NEBRW based on Skip-gram and design a new community search algorithm CSMST via an improved Prim-based approach. Communities detected by CSMST are the node sets connected with minimum total distance. CSMST algorithm achieves good performance on both synthetic and real-world networks.

In the future, we will study the node embedding technique in heterogeneous social media networks and study the community search problem in heterogeneous networks.

Data Availability

The data and code used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The project was supported by the National Key R&D Program of China under Grant 2018YFB1004700, National Natural Science Foundation of China (61772122 and 61872074), and Fundamental Research Funds for the Universities of Heilongjiang Province of China (YWK10236200141).

References

[1] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," in *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[2] H. Cherifi, G. Palla, B. K. Szymanski, and X. Lu, "On community structure in complex networks: challenges and opportunities," *Applied Network Science*, vol. 4, no. 1, p. 117, 2019.

[3] J. Cao, D. Jin, L. Yang, and J. Dang, "Incorporating network structure with node contents for community detection on large networks using deep learning," *Neurocomputing*, vol. 297, pp. 71–81, 2018.

[4] S. Fortunato and D. Hric, "Community detection in networks: a user guide," *Physics Reports*, vol. 659, pp. 1–44, 2016.

[5] Q. Ye, C. Zhu, G. Li, Z. Liu, and F. Wang, "Using node identifiers and community prior for graph-based classification," *Data Science and Engineering*, vol. 3, no. 1, pp. 68–83, 2018.

[6] I. M. Kloumann and J. Kleinberg, "Community membership identification from small seed sets," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1366–1375, New York, NY, USA, August 2014.

[7] Y.-L. Zhao, L. Nie, X. Wang, and T.-S. Chua, "Personalized recommendations of locally interesting venues to tourists via cross-region community matching," *Acm Transactions on Intelligent Systems and Technology*, vol. 5, no. 3, pp. 1–26, 2014.

[8] M. Sozio and A. Gionis, "The community-search problem and how to plan a successful cocktail party," in *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 939–948, Washington, DC, USA, July 2010.

[9] W. Cui, Y. Xiao, H. Wang, and W. Wang, "Local search of communities in large graphs," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 991–1002, New York, NY, USA, June 2014.

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

[11] Y. Wu, R. Jin, J. Li, and X. Zhang, "Robust local community detection," in *Proceedings of the VLDB Endowment*, vol. 8, no. 7, pp. 798–809, 2015.

[12] L. Yang, J. Xin-sheng, L. Caixia, and W. Ding, "Detecting local community structures in networks based on boundary identification," *Mathematical Problems in Engineering*, vol. 2014, no. 1, pp. 1–8, 2014.

[13] R. Andersen and K. J. Lang, "Communities from seed sets," in *Proceedings of the International World Wide Web Conferences*, pp. 223–232, Scotland, UK, May 2006.

[14] Y. Li, K. He, D. Bindel, and J. Hopcroft, "Uncovering the small community structure in large networks: a local spectral approach," *Computer Science*, vol. 36, no. 6, pp. 658–668, 2015.

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

[16] F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," *Web Intelligence and Agent Systems: An International Journal*, vol. 6, no. 4, pp. 387–400, 2008.

[17] 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, p. 492, 2011.

[18] L. Ma, H. Huang, Q. He, K. Chiew, J. Wu, and Y. Che, "GMAC: a seed-insensitive approach to local community detection," in *Proceedings of Data Warehousing and Knowledge Discovery, in: DaWaK*, pp. 297–308, Prague, Czech Republic, August 2013.

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

[20] A. Clauset, "Finding local community structure in networks," *Physical Reviews E*, vol. 72, no. 2, p. 026132, 2005.

[21] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Reviews E*, vol. 70, no. 6, p. 066111, 2004.

[22] J. Shao, Z. Han, Q. Yang, and T. Zhou, "Community detection based on distance dynamics," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1075–1084, Sydney, UK, August 2015.

[23] C. Panagiotakis, H. Papadakis, and P. Fragopoulou, "Local community detection via flow propagation," in *Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Paris, France, August 2015.

[24] B. Saoud and A. Moussaoui, "Community detection in networks based on minimum spanning tree and modularity," *Physica A: Statistical Mechanics and Its Applications*, vol. 460, pp. 230–234, 2016.

[25] K. Asmi, D. Lotfi, and M. E. Marraki, "A novel approach based on the minimum spanning tree to discover communities in social networks," in *Proceedings of the 2016 International*

Conference on Wireless Networks and Mobile Communications, pp. 286–290, Fez, Morocco, October 2016.

[26] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: large-scale information network embedding," in *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077, Florence, Italy, May 2015.

[27] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: scalable representation learning for heterogeneous networks," in *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 135–144, Halifax, Canada, August 2017.

[28] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: methods and applications," *IEEE Data Engineering Bulletin*, vol. 40, no. 3, pp. 52–74, 2017.

[29] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: a survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, 2018.

[30] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[31] Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, and T. Wu, "Rankclus: integrating clustering with ranking for heterogeneous information network analysis," in *Proceedings of 12th International Conference Extending Database Technology: Advances Database Technology*, pp. 565–576, Saint Petersburg Russia, March 2009.

[32] T. Fu, W. C. Lee, and Z. Lei, "Hin2vec: explore meta-paths in heterogeneous information networks for representation learning," in *Proceedings of the 2017 ACM Conference on Information and Knowledge Management: CIKM*, pp. 1797–1806, Singapore, Singapore, November 2017.

[33] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 357–370, 2019.

[34] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701–710, New York, NY, USA, August 2014.

[35] A. Grover and J. Leskovec, "node2vec: scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, San Francisco, CF, USA, August 2016.

[36] D. Jin, X. You, W. Li et al., "Incorporating network embedding into markov random field for better community detection," in *Proceedings of The Thirty-Third AAAI Conference on Artificial Intelligence*, pp. 160–167, Honolulu, HI, USA, January 2019.

[37] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," in *Proceedings of the American Mathematical Society*, vol. 7, no. 1, p. 48, 1956.

[38] R. C. Prim, "Shortest connection networks and some generalizations," *Bell System Technical Journal*, vol. 36, no. 6, pp. 1389–1401, 1957.

[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient Estimation of Word Representations in Vector Space*, Cornell University, Ithaca, NY, USA, 2013.

[40] D. Nguyen and F. Malliaros, "Biasedwalk: biased sampling for representation learning on graphs," in *Proceedings of 2018 IEEE International Conference on Big Data (Big Data)*, pp. 4045–4053, Seattle, WA, USA, December 2018.

[41] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Reviews E*, vol. 78, no. 4, pp. 046110–1–046110–5, 2008.

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

[43] M. E. J. Newman, "Modularity and community structure in networks," in *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.

[44] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 181–213, 2012.