Efficiently detecting overlapping communities using seeding and semi-supervised learning

Changxing Shang · Shengzhong Feng · Zhongying Zhao · Jianping Fan

Received: 28 January 2015 / Accepted: 6 February 2015 / Published online: 22 February 2015 © Springer-Verlag Berlin Heidelberg 2015

Abstract A common scheme for discovering overlapping communities in a network is to use a seeding process followed by an expansion process. Most seeding methods are either too complex to scale to large networks or too simple to select high-quality seeds. Additionally, the non-principled functions used by most expansion methods lead to poor performances when applied to diverse networks. This paper proposes a new method that transforms a network into a corpus. Each edge is treated as a document, and all the network nodes are treated as terms of the corpus. We propose an effective seeding method that selects seeds as a training set, and a principled expansion method based on semi-supervised learning that classifies the edges. We compared our new algorithm with four other community detection algorithms on a wide range of synthetic and empirical networks. Our experimental results show that the new algorithm significantly improved the clustering performance in most cases. Furthermore, the time complexity of the new algorithm is linear with respect to the number of edges, which means that the technique can be scaled to large networks.

Keywords Graph clustering · Social networks · Overlapping community detection

1 Introduction

Many complex systems can be abstracted using networks or graphs, where the elementary parts of a system are represented as nodes, and their mutual interactions are edges (or links). The community structure is a key property of many networks. Nodes with similar properties or functions have more edges than random pairs of nodes and tend to be gathered into distinct subgraphs, which are called communities (modules or clusters). Examples can be found in many complex systems in fields such as sociology, biology [1], and computer science [2]. In reality, a node may belong to multiple communities. For example, a researcher may be active in several areas, and a person typically has connections to several social groups (family, friends, colleagues, and so on). The aim of an overlapping algorithms is to discover a cover, which is defined as a set of communities such that each node belongs to at least one community.

Local expansion and optimization is a common scheme used to find overlapping communities [3, 4]. The detection process consists of two steps: selecting seeds and expanding the seeds to form communities.

Seed quality has an important impact on the performances of these types of methods. For example, Lee et al. demonstrated that the performance gap between greedy clique expansion (GCE) [3] and local fitness maximization (LFM) [4] is largely due to the different seed selection methods, because they use the same expansion process. The quality of the detecting communities can be improved by replacing the selection method [5]. Lancichinetti et al.
[6] also gave an example to show how seeding methods affect the expectation and maximization (EM) method proposed by Newman et al [7]. Our experimental results agree with this conclusion. Three types of methods are typically used for seed selection: random, maximal cliques, and ranking. Random methods often result in an unstable performance [4–6]. Maximal cliques methods are common [3, 8], and result in better community structures at the cost of a loss of scalability. The Bron-Kerbosch algorithm [9] used can find all maximal cliques, but is exponentially complex \( O(3^{n/3}) \), where \( n \) is the number of nodes in the network. In practice, the Bron-Kerbosch algorithm can run fast on networks that have less than \( 10^5 \) nodes, because of techniques such as pruning. However, it is hard to scale to larger networks. Ranking methods give each node or edge a rank, and then a removing or appending strategy is used to select seeds. PageRank is often used to compute rank values. Rank Removal (RaRe) [10] assumes that nodes with high ranks are responsible for a significant amount of communication, so it sequentially removes high-rank nodes until some “cores” are left as seeds. The above assumption is not appropriate, because many high-ranked nodes are authorities of their communities and suitable as seeds. For example, if we select node 34 in Fig. 1 as a seed, the community detection process will be easy. On the other end of the spectrum, appending methods such as the link aggregate (LA) [11] technique select seeds in decreasing order of rank. The drawback of appending methods is that many network hub nodes have high ranks. For example, nodes 12 and 49 in Fig. 2 are hubs, and expanding from them will result in poor communities. Another drawback of appending methods is that they prefer to select seeds from major communities to minor ones, so seed diversity cannot be guaranteed. We believe that these problems with ranking methods are due to their global ranking behavior. This paper proposes an efficient seeding method that overcomes these drawbacks, by first ranking edges locally and then selecting seeds globally.

Most expansion methods [3, 4] use a local fitness function to decide whether a node should be included in a community. Yang et al. [12] summarized 13 functions based on the intuition that links in communities are dense, whereas links between communities are sparse. An advantage of these functions is that they only use local (or neighborhood) information to decide if a node belongs to a community, which leads to a fast expansion. However, these functions are all heuristic and lack principled support, so they cannot be applied to a wide range of synthetic and empirical networks. Lee noted [3] that “just as there is no universally correct concept of community that spans all domains, one cannot argue that any given fitness function will be appropriate for all types of network data”. Another drawback is that each community expands independently, without negotiation with others. This can lead to highly similar communities that share a large number of nodes. A post-merging process can merge these communities together, but it can be hard to determine appropriate merging criteria. In this paper, we propose a new expansion method that replaces the local fitness function with a global optimization function to infer to which community an edge belongs. Negotiations are introduced by the global function. By virtue of the wide applicability of Bayesian inference, this new method can be applied to a diverse range of networks.

This paper proposes an algorithm called ITEM, which uses information theory and an EM process to discover communities in a network. ITEM first transforms a network into a corpus, where edges and nodes are treated as documents and terms, respectively. Then it classifies each edge into a community, and assigns two endpoints of the edge to the community. The contributions of the paper are as follows.

---

**Fig. 1** The Karate network, which represents friendship between members of a university sports club. The club is divided into two communities as a result of an internal dispute.
Fig. 2 An illustration of the steps in the seeding process, using the LM network as an example. The LM network is a co-appearance network of characters in the novel Les Misérables by Victor Hugo. In the left subfigure, $e_{12-49}$ (the sinewave edge) is the seed selected using only the reputation scores. The colored edges are seeds selected using reputation×strength scores. In the middle subfigure, the colored edges are seeds selected using reputation×strength×specificity scores. In the right subfigure, the bold colored edges are the seeds selected using the maximizing global information gain (MIGI) method with $th = 0.3$. The thin color edges are added later to enlarge the training examples.

1. We propose the concept of the Jaccard matrix\(^1\) of a network. The topic (i.e., the community) of an edge can be easily extracted using the Jaccard matrix.
2. We propose an efficient and effective seeding method that overcomes the drawbacks of traditional methods.
3. We propose a principled expansion method. By treating the selected seeds as a training set, we use semi-supervised learning technology to classify edges into communities.

We conducted experiments on a wide range of synthetic and empirical networks. Our experimental results show that ITEM significantly improves the clustering performance. The total computational complexity of ITEM is linear with respect to the number of edges, which means that ITEM can be scaled to large networks.

The remainder of this paper is organized as follows. In Sect. 2, we introduce the outline of the ITEM algorithm and some terminologies. We describe the seeding and expansion methods in Sects. 3 and 4, respectively. In Sect. 5, we introduce the experimental setup, including the benchmark networks and related algorithms. We present our experimental results and analyses in Sect. 6, and our conclusions and suggestions for future research in Sect. 7. We have published ITEM’s code on the internet, so that interested readers can reproduce these results\(^2\).

\(^1\) Borgatti [13] proposed the Jaccard similarity matrix, but the Jaccard matrix in this paper is different from Borgatti’s Jaccard similarity matrix. Using the definitions given by Borgatti [13], the Jaccard matrix is a 2-mode matrix derived from a 1-mode adjacent matrix. Borgatti’s Jaccard similarity matrix is a 1-mode matrix derived from a 2-mode women-by-events matrix.

\(^2\) https://github.com/cxshang/ITEM.

2 Terminology and outline of the ITEM algorithm

Some of the terminology used in this paper is defined in Table 1. We use two different notations to denote an edge. We use the double subscript notation $e_{i-j}$ when we must explicitly define its two endpoints ($v_i$ and $v_j$); otherwise, we use a single subscript notation (e.g., $e_i$ denotes the $i$th edge). We used the Karate network in Fig. 1 as an example to clearly illustrate these terminologies. In this section, we also explain the outline of the ITEM algorithm.

ITEM exploits text-mining technologies [14] to discover the communities of a network. We propose the concept of the Jaccard matrix of a network by observing that the community of an edge can be largely determined by its two endpoints and their shared neighbors. This is similar to how the topic of a document can be identified by a few key terms or features [15]. The Jaccard matrix is motivated by the Jaccard index, which is a statistic commonly used to measure the similarity of two endpoints of an edge. For the similarity between two endpoints of $e_{i-j}$, the Jaccard index is defined as $J(v_i, v_j) = \frac{\text{Nb}(v_i) \cap \text{Nb}(v_j)}{\text{Nb}(v_i) \cup \text{Nb}(v_j)}$.

The Jaccard matrix of a network $G$ is denoted as $M$, and is an $m \times n$ matrix ($m$ and $n$ are defined in Table 1). For each item $M_{ij}$, where $i \in \{1, 2, \cdots m\}$ and $j \in \{1, 2, \cdots n\}$,

$$M_{ij} = \begin{cases} 0 & \text{if } v_j \notin \text{Nb}(e_i) \\ w_j & \text{if } v_j \in \text{Nb}(e_i) \end{cases} $$

(1)

$w_j$ can simply be assigned a value of 1 or can be determined using the $tf \cdot idf$ [16] method. Table 2 displays some of the entries of the Karate network’s Jaccard matrix.

The Jaccard matrix has some particular features. First, a document $e_{i-j}$ only includes terms in $\text{Nb}(e_{i-j})$; all the other neighbors of $v_i$ and $v_j$ are discarded. The discarding
operation resembles the preprocessing step in text mining that removes low-frequency words from a document. After the discarding operation, the topic of document \( e_{i-j} \) is easier to identify. The topic becomes clearer because the edges are more specific to a certain community or topic than the nodes. Second, because of the increased density of a community and the more common neighbors shared by its edges, this produces many similar documents in the Jaccard matrix. Based on the similarity, we can cluster edges from the same community together.

The ITEM algorithm uses a semi-supervised learning method to classify edges into different communities. Many machine learning researchers have found semi-supervised learning can considerably improve learning accuracy, because it exploits both labeled and unlabeled information [17, 18]. The NB (naïve Bayes), SVM (support vector machine) [19, 20] and DT (decision tree) [21] methods can be used to classify edges. In this paper, we used the NB classifier, because it is simple and can effectively classify documents [22]. ITEM first selects some seeds as a training set, then an EM process is used to expand the edges into communities. In each expectation step, some previously unlabeled edges are assigned new labels (i.e., communities), and some edges change their communities. Because the above unlabeled edges are then used in the maximization step to model or refine the NB classifier, in this sense, ITEM uses semi-supervised learning to expand the communities. The EM iterations stop when a predefined condition is satisfied.

### 3 The seeding process

ITEM uses two steps to select the seeds. In the first step, to avoid the high computational complexity and drawbacks of global ranking methods, we propose a local ranking method called RSS (reputation, strength, specificity). "Local" has two meanings: RSS gives each edge a score using only its local information (i.e., the neighbors’
information); and the edges locally compare their RSS scores. More specifically, $e_i$ only compares its score with its incident edges [i.e., $Ic(e_i)$]. If there exists any one incident edge whose score is higher than $e_i$, then $e_i$ is filtered out and cannot be selected as a seed. If two edges have equal scores, then we keep the edge with the smallest index and remove the other. In the second step, the candidate seeds selected by RSS are input into the maximizing global information gain (MGIG) method [23], which is used to select distinctive and representative seeds from the candidates from a global perspective.

3.1 Selecting candidate seeds using the RSS method

The LM network in Fig. 2 is slightly more complicated than the Karate network. In this subsection, we use it to explain the motivations behind the RSS method.

When selecting seed nodes, we should not completely use or ignore the reputation values because both hub nodes and specificity nodes may have high reputations. Reputation and specificity are also important for the edges, and edges have another property that the nodes do not. We define strength as a measure of the link intensity between two endpoints of an edge. Intuitively, we define strength as a measure of the link intensity between two endpoints of an edge. Intuitively, $e_{12-49}$ in Fig. 2 is not suitable as a seed because it has a high reputation but low specificity. $e_{63-65}$ is more suitable as a seed than $e_{74-75}$, because $e_{63-65}$ is more reputable and stronger than $e_{74-75}$. But what is the reasoning behind this intuition? The number of common neighbors between two endpoints of an edge is a good indicator of an edge's suitability as a seed. We have now given formal definitions of the reputation, strength, and specificity, which all exploit the common neighbors concept either directly or indirectly. The seeds selected using reputation, strength, and specificity scores are listed in Fig. 2.

We must first calculate the similarity of two incident edges so that we can determine the RSS score of each edge. SimHash [24] is a popular method for evaluating the similarity of the two documents. This paper uses SimHash to convert document $e_i$ into a 64-bit binary number, which is denoted as $fp_i$, and called the fingerprint of $e_i$. We have not included any details of SimHash because of space constraints. For more information, please see [24, 25].

For each $e_i$, where $i \in \{1, 2, \ldots, m\}$, the reputation score is defined as

$$\text{reputation}(e_i) = \frac{1}{64} \sum_{(j | f_j \in Ic(e_i))} 64 - hd(fp_i, fp_j),$$

(2)

where $fp_i$ and $fp_j$ are the fingerprints of $e_i$ and $e_j$, and $hd(fp_i, fp_j)$ is the Hamming distance between $fp_i$ and $fp_j$. Equation (2) gives a high reputation score to $e_i$ if $Nb(e_i)$ and $Nb(e_j)$ share a large number of nodes [which makes $hd(fp_i, fp_j)$ small] or if $|Ic(e_i)|$ is large.

The strength of $e_{i-j}$ is defined as

$$\text{strength}(e_{i-j}) = \frac{|Nb(e_{i-j})| - 1}{\max(\deg(v_i), \deg(v_j))}.$$  

(3)

strength$(e_{i-j})$ measures the intensity of the connection between $v_i$ and $v_j$.

The specificity of each node is defined as

$$\text{specificity}(v_i) = \sum_{i \leq k} \frac{\text{strength}(e_j)}{|Ic(v_i)|}.$$  

(4)

specificity$(v_i)$ measures the average similarity between $v_i$ and its neighbors. The specificity of an edge is

$$\text{specificity}(e_{i-j}) = \min(\text{specificity}(v_i), \text{specificity}(v_j)).$$

For $\forall e_i \in E$, the reputation, strength, and specificity scores are all in $[0, 1]$. Now, we can calculate the RSS score of $e_i$ using

$$\text{RSS}(e_i) = \text{reputation}(e_i) \times \text{strength}(e_i) \times \text{specificity}(e_i).$$

The computational complexity of RSS is $O(dm)$, where $d$ is the average degree of the nodes. The low computational complexity is mainly due to the locality of RSS. However, only comparing with its incident edges makes an edge easy to select as a seed, which results in some similar seeds. For example, $e_{60-62}$, $e_{63-65}$, and $e_{64-66}$ are similar, so using them as seeds may split a community apart. As another side effect, an edge may be thrashed among its adjacent communities, which slows the convergence of the subsequent EM process. To overcome this drawback, similar edges should be removed. In Sect. 3.2, we explain how to select the final edges. In the following discussion, we call the edges selected by RSS candidate seeds, because the final seeds are selected from them.

3.2 Selecting the final seeds using the MGIG method

MGIG is an efficient feature selection method for text classification that effectively selects distinctive and representative terms [23]. Because of the duality between terms and documents [26], we use MGIG to select dissimilar edges (documents). The idea behind MGIG is very simple. In Table 2, the document representations of $e_{2-3}$ and $e_{2-4}$ are identical, so we lose no information when merging them into a virtual document (i.e., the document that includes all terms in $e_{2-3}$ and $e_{2-4}$). Additionally, there is no information loss by splitting the virtual document apart. However, information is released when we split $e_{2-3}$ and $e_{24-30}$ apart because they are dissimilar. MGIG tries to select edges that release the maximum information.
MGIG selects seeds one by one. Suppose there are $l$ candidate seeds. If $k - 1$ ($1 \leq k < l$) edges have been selected, then MGIG selects the $k$th edge from the unselected candidates, which results in the maximum

$$p(e_k)H(p(V|e_k)) = p(e_k)H(p(V|e_k^*)).$$  

(4)

Note that $e_k^*$ represents the $k$th edge selected from the $l$ candidates. $H(p(V|e_k^*))$ is the entropy of $p(V|e_k^*)$, and $p(V|e_k^*) = (p(v_1|e_k^*), p(v_2|e_k^*), \ldots, p(v_n|e_k^*))$ is the node distribution for a given edge. $e_k^* \approx S_k = \{e_1^*, e_2^*, \ldots, e_k^*\}$ is the set of already selected candidates, and $e_k^*$ is a virtual edge (if we view all the $k$ selected edges in $S_k$ as a whole). Hence,

$$p(e_k^*) = \sum_{i=1}^{k} p(e_i^*),$$

and

$$p(v_j|e_k^*) = \frac{\sum_{i=1}^{k} p(e_i^*)p(v_j|e_i^*)}{p(e_k^*)}.$$

For $p(v_j|e_i^*)$,

$$p(v_j|e_i^*) = \begin{cases} 0 & \text{if } v_j \notin Nb(e_i^*) \\ 1/|Nb(e_i^*)| & \text{if } v_j \in Nb(e_i^*) \end{cases}$$

Because Criterion (4) does not apply to $k = 1$, we select the first edge that maximizes $H(p(V|e_1^*))$.

After selecting $e_k^*$, we check each unselected candidate edge, $e^*$. $Nb(e^*) - Nb(e_k^*)$ is the set of nodes that are included in $Nb(e^*)$ but not included in $Nb(e_k^*)$. Clearly, a larger value of $|Nb(e^*) - Nb(e_k^*)|/|Nb(e^*)|$ means that $e^*$ is more likely to qualify as a seed. We introduce a scaling parameter $th \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ to control whether $e^*$ should be filtered out. If $|Nb(e^*) - Nb(e_k^*)|/|Nb(e^*)| < th$, $e^*$ is filtered out. There is no universal value for $th$ because of the diversity of network and community structure scales. However, for a specific network, the effective range of $th$ is typically narrow.

In the right subfigure of Fig. 2, the bold colored edges are the final seeds selected using the MGIG method with $th = 0.3$. Now, we have a training set that contains five seeds, which is not enough because it contains only one training example for each community. To improve the reliability of the subsequent semi-supervised learning process, we increase the training set as follows. Suppose that $e^*$ is a final seeding edge, for $\forall v_i, v_j \in Nb(e^*)$. If $\exists e_{i-j} \in E$, then $e_{i-j}$ is added to the training set and labeled with community $e^*$. In the same subfigure, these added edges are represented by thin colored lines. We call edges with the same color a committee. In the expansion process, committees are expanded to form communities, where $K$ is the final number of communities. For the LM network in Fig. 2, $K = 5$.

Suppose that $p = \frac{1}{m} \sum_{i=1}^{m} |Nb(e_i)|$ is the average number of neighbors for an edge. The computational complexity of using MGIG to select $K$ seeds is $O(pK)$. If we consider the RSS, the total computational complexity of the seeding process is $O(dm + pK)$, where $p \leq d + 1$ and $l \ll m$. Clearly, our seeding method is more scalable than the method of finding all maximal cliques.

4 The expansion process

The expansion process uses an EM algorithm to classify edges into communities.

4.1 The expectation step

The expectation step exploits both topological and topic information to decide the most suitable community for $e_i$. We first use the topological information to judge whether $e_i$ is a potential edge of $C_k$. The potential edges of a community $C_k$ are the edges that are not currently included in $C_k$, but which may be added into $C_k$ in the expectation step. If $e_i$ is a potential edge of $C_k$, then the topic information is used to evaluate the posterior probability, $p(C_k|e_i)$. Finally, $e_i$ is assigned to the community that maximizes $p(C_k|e_i)$. The community is denoted as $C^*(e_i)$.

Suppose that $Nd(C_k) = \bigcup_{e_{i-j} \in C_k} \{v_i, v_j\}$ is the set of nodes that are currently included in $C_k$. Clearly, if neither $v_i$ nor $v_j$ is included in $Nd(C_k)$, then $e_{i-j}$ is not adjacent to $C_k$ and cannot be a potential edge. However, if both $v_i$ and $v_j$ are included in $Nd(C_k)$ but $e_{i-j}$ is not included in $C_k$, $e_{i-j}$ is a potential edge of $C_k$. As the third option, $e_{i-j}$ may sit between the two extremes. That is to say, $Nd(C_k)$ only includes one endpoint of $e_{i-j}$. Without loss of generality, we can assume that $v_i$ is included in $Nd(C_k)$ but $v_j$ is not. In this case, $e_{i-j}$ is treated as a potential edge of $C_k$, if it satisfies any condition below

1. $od(e_{i-j}) > 0$ and $|Nb(e_{i-j}) \cap Nd(C_k)| \geq 2$,
2. $od(e_{i-j}) = 0$ and $od(v_j) = 0$ and $|Nb(e_{i-j}) \cap Nd(C_k)| \geq \frac{1}{2}d_{deg}(v_j)$

where $od(e_{i-j})$ is the order of $e_{i-j}$ and is defined as $od(e_{i-j}) = |Nb(e_{i-j})| - 2$, which is the number of triangles that have $e_{i-j}$ as an edge. For example, in Fig. 1, $od(e_{20-34}) = 0$, $od(e_{1-18}) = 1$, and $od(e_{2-3}) = 4$. $od(v_j) = \frac{1}{2} \sum_{e_{i-j} \in E(v_j)} od(e_{ij})$ is the order of $v_j$, which is the number of triangles that have $v_j$ as a vertex, so zero order nodes are not included in any triangles.

Conditions 1 and 2 give the constraints for non-zero order edges and zero order edges, respectively. Condition 1
indicates that at least two sponsors wish to pull \( v_j \) into \( C_k \). Additionally, these two sponsors and \( v_j \) are the three vertices of a triangle. Condition 2 imposes a rigorous requirement on a zero-order edge. If \( od(e_{i,j}) = 0 \) but \( od(v_j) > 0 \), \( v_j \) has other more intimate neighbors than \( v_j \). Therefore, it is more likely that \( v_j \) will be pulled into a community (not necessarily \( C_k \)) by other links but not by \( e_{i,j} \). If \( od(e_{i,j}) = 0 \) and \( od(v_j) = 0 \), \( v_j \) treats its neighbors equally. Then, \( |Nb(e_{i,j}) \cap Nd(C_k)| \geq \frac{1}{2} \deg(v_j) \) ensures that \( e_{i,j} \) can only be viewed as a potential edge of \( C_k \) for at least half the neighbors of \( v_j \) included in \( C_k \).

After ensuring that \( e_i \) is a potential edge of \( C_k \), the posterior probability \( p(C_k|e_i) \) is evaluated using an NB classifier. The final community of \( e_i \) is

\[
C^*(e_i) = \arg\max_{C_k} p(C_k|e_i) = \arg\max_{C_k} p(C_k) \prod_{v_j \in Nb(e_i)} p(v_j|C_k)^{w_j},
\]

(5)

where \( w_j \) is the weight of \( v_j \), set using the tf·idf method. \( p(C_k) \) and \( p(v_j|C_k) \) are evaluated in the maximization step.

Suppose that \( W_e = \sum_{j \in \{v_j \in Nb(e_i)\}} w_j \), then (5) can be written in the following form by taking logarithms, dividing by \( W_e \), and adding \( H(p(V|e_i)) \). Then,

\[
C^*(e_i) = \arg\min_{C_k} KL(p(V|e_i), p(V|C_k)) - \frac{1}{W_e} \log(p(C_k)),
\]

(6)

where \( KL(p(V|e_i), p(V|C_k)) \) is the Kullback-Leibler divergence between \( p(V|e_i) \) and \( p(V|C_k) \). Hence, our expansion method resembles the k-means algorithm because it uses (6) to measure distances and tries to minimize a global objective function. That is,

\[
\sum_{e_i \in PS} KL(p(V|e_i), p(V|C^*(e_i))) - \frac{1}{W_e} \log(p(C^*(e_i))),
\]

(7)

where \( PS \) denotes the set of potential edges.

4.2 The maximization step

In the maximization step, the unknown parameters are evaluated based on all the currently labeled edges. \( p(C_k) \) is the proportion of edges in \( C_k \) versus the edges in all current communities. \( p(v_j|C_k) \) is evaluated using

\[
p(v_j|C_k) = \frac{\sum_{\{e_i \in Nb(C_k)\}} w_{ik} \cdot \sum_{\{v_j \in Nb(e_i)\}} w_{ik}}{\sum_{\{e_i \in Nb(C_k)\}} w_{ik}}.
\]

(8)

In the above equation, \( Nd^*(C_k) \) is the set of terms occurring in the documents of \( C_k \), which is defined as \( Nd^*(C_k) = \bigcup_{v_j \in C_k} Nb(e_i) \). Clearly, \( Nd(C_k) \subseteq Nd^*(C_k) \).
Fig. 3 NMI comparisons for increasing om
results in Fig. 3 are the averages over 10 LFR networks. We used normalized mutual information (NMI) to measure the similarities between the ground truth and calculated communities [4]. The NMI value is between 0 and 1, with 1 corresponding to a perfect match between the true and calculated communities.

For empirical networks, we used the Facebook100 dataset [30, 31], which includes 100 friendship networks. These networks have metadata such as the year and dorm. It is difficult to evaluate these networks, because the ground truth communities are unknown. Therefore, NMI cannot be used on the Facebook100 dataset. Recently, Lee and Cunningham proposed a machine learning framework [32] for measuring how well communities were detected based on the assumption that “if a community detection algorithm is functioning well, then a classifier should be able to use the set of detected communities to infer missing values of a node attribute that is closely related to community structure.” So when a classifier is more accurate, the calculated communities are more appropriate.

5.2 Comparative algorithms

Many algorithms for discovering overlapping communities have been proposed. Some algorithms [33–35] use domain or prior knowledge. For example, [33] uses the view counts of videos to find the most popular videos as seeds. But this information is not always available, so we have compared algorithms that only use the information contained in the networks. We also tried to compare algorithms proposed by [36–38], but these codes were difficult to implement on our platform.

Xie et al. [39] conducted a good survey of overlapping community algorithms. They concluded that the GCE method, speaker-listener label propagation algorithm (SLPA) [40], community overlap propagation algorithm (COPRA) [41], order statistics local optimization method (OSLOM) [42], and LFM method perform best on larger networks. We have only compared ITEM with GCE and SLPA, because the other three algorithms are computationally complex. We also compared the clique percolation method (CPM) [43]. Last, we compared ITEM with the Poisson model (PM) algorithm [44], which is considered a state-of-the-art method.

CPM first finds all k-cliques, then combines one clique with another if they share k − 1 nodes. This process only stops when there are no adjacent cliques that share k − 1 nodes. The nodes in this k-cliques combination form a community. GCE uses maximal cliques as seeds, then expands the nodes by maximizing the fitness function, 

\[ f(C_i) = 2 \times m_{\text{in}}^C / (2 \times m_{\text{in}}^C + m_{\text{out}}^C), \]

where \( m_{\text{in}}^C \) is the number of edges on the boundary of \( C_i \). SLPA uses label propagation to discover overlapping communities, and nodes send and receive labels according to certain rules. SLPA uses a threshold variable \( r \) to filter out labels with probabilities less than \( r \). PM evaluates a set of parameters \( \theta_R \), which measure the extent that node \( i \) belongs to community \( C_k \).

We can compare the results of the ITEM, GCE, SLPA, CPM, and PM techniques to see which is superior. However, this will not tell us the reason for this superiority. We designed another algorithm to see the effects of our seeding and expansion methods. The expansion process of GCE is the same as minimizing the conductance between \( C_i \) and the rest of the network, so we designed a mixing algorithm called seeding and minimizing conductance (SMC), which uses ITEM’s seeding process with the GCE expansion process. It is clear that SMC and GCE are identical except for their seeding methods, and SMC and ITEM are identical except for their expansion methods. So we can determine the effects of our seeding and expansion methods using the performance differences between GCE and SMC, or SMC and ITEM.

CPM, GCE, ITEM, and SMC require at least one parameter. For CPM and GCE, we let the clique level \( k \) vary between 3 and 6, and set the other GCE parameters to their default values. For ITEM and SMC, \( h_1 = \{0.2, 0.3, 0.4\} \) for the LFR networks, and \( h_1 = 1 \) for the Facebook100 networks. The \( r \) value for SLPA was in \{0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5\}. To run PM, we must set the number of communities. Hence, we only evaluated PM on LFR synthetic networks by providing the true number of communities. A threshold value \( \theta_R \in \{0.6, 0.8, 1.0, 1.2, 1.4\} \) was also used to filter out communities \( C_k \) of \( v_i \) if \( \theta_R < \theta_R \). For each network, we ran all the algorithms multiple times by iterating through their parameters and picking the best result.

6 Experiments and analyses

We ran a wide range of experiments on the LFR networks. Figure 3 shows the performances of the algorithms for different numbers of node memberships. Each subfigure in Fig. 3 was produced using a different set of parameters.

We first compared ITEM with GCE, SLPA, CPM, and PM. Figure 3 shows that no algorithm was clearly better over all the 32 subfigures. However, ITEM performed the best in most experiments. To determine if there were statistical differences between the different results, we used the Friedman test on an NMI result set with five treatments (ITEM, GCE, SLPA, CPM, and PM) and 224 blocks (32 parameter settings and seven options for \( omi \)). The p-value (2.2e−16) of the Friedman test shows that there were statistical differences. To further identify the differences, we ran post hoc analyses using the paired Wilcoxon test. These
results are shown in the left subfigure of Fig. 4. The results using ITEM and PM were significantly different than those using GCE, SLPA, and CPM. More specifically, ITEM improved the NMI by more than 0.1 in approximately half of these experiments. PM improved the results by 0.05. The 32 subfigures in Fig. 3 also demonstrate the wide applicability of ITEM. Compared with other algorithms, ITEM always performs adequately in more subfigures.

We then compared ITEM with GCE and SMC. We ran the same statistical analyses, except that we replaced the original treatments with ITEM, GCE, and SMC. The results are shown in the right subfigure of Fig. 4. We evaluated the effectiveness of our seeding method by comparing SMC with GCE, which is considered the best seeding method [5]. The right subfigure indicates that the qualities of the seeds selected by SMC and GCE are comparable because there were no statistical differences between SMC and GCE. However, it is less computationally complex to find committees than to find all cliques. Committees are often dense sub-graphs that are similar to cliques, so finding committees to be used as seeds means that we relax the seeding criteria to increase the efficiency. We believe that the effectiveness of this method can be attributed to the following. First, RSS’s local ranking method overcomes the problems of the global ranking method. Second, MGIG selects seeds from a global perspective. Third, it is more appropriate to use committees as seeds than cliques, because the clique criteria are too strict.

There are significant differences between ITEM and SMC. This indicates that the expansion method using semi-supervised learning and Bayesian inference is superior to the heuristic method used by GCE. The following differences between ITEM’s and GCE’s expansion methods may explain this superiority. First, because it is based on Bayesian inference, ITEM promotes the weights of authoritative nodes and suppresses the weights of hub nodes. The heuristic functions used by GCE and other methods [12] treat all nodes equally. The universality of Bayesian inference means that it can be extensively applied in many areas. Second, semi-supervised learning uses both labeled and unlabeled edges to model the NB classifier. Unlabeled edges may theoretically decrease the performance of an NB classifier [45]. However, this is an ideal scenario for semi-supervised learning, because the document presentations of edges in the same communities are very similar and we use a conservative strategy of only adding potential edges in each iteration. Third, competition is introduced by the ITEM because we try to minimize the global objective function (7). In other words, each belonging is determined by the consultations of multiple communities, and not only by a local fitness function.

We also ran experiments using the Facebook100 dataset. We used 40 networks with the fewest nodes to evaluate the communities found by ITEM, GCE, and SLPA. CPM is excluded because it cannot finish running in 48 hours. For each combination of network and algorithm, Figs. 5 and 6 display the accuracy given by the training classifier when using “year” and “dorm” attributes as labels, respectively. Each value represents the average of three runs. Figure 5 indicates that ITEM performed consistently better than GCE and SLPA for all 40 networks. In Fig. 6, we can see that GCE performed better than ITEM and SLPA on most networks. We believe that there are two reasons why ITEM performed better than GCE when using “year” as a label, but worse when using “dorm” as a label. First, ITEM tends...
to generate larger communities than GCE. Second, the true communities with “year” attributes are larger than the true communities with “dorm” attributes. The second reason also explains why ITEM and GCE were both more accurate in Fig. 5 than in Fig. 6, for most networks. That is, the benchmark classifier has more training examples when using “year”. The different behaviors of GCE and ITEM also suggest that it may be advantageous to combine multiple orthogonal algorithms to discover more true communities in a network.

It is worth comparing ITEM with Linkcomm [46], because they both cluster edges into communities. But there is an important difference between these methods. We refer to ITEM a partitioning method because our expansion method resembles the k-means technique. Linkcomm uses the hierarchical agglomerative algorithm to cluster edges. It consequently inherits two shortcomings of the hierarchical agglomerative algorithm. First, the greedy nature of these algorithms yield sub-optimal clusters when compared with partitioning algorithms, which use collective information to generate edge clusters. Agglomerating algorithms only consider information from two clusters at each agglomeration step [47]. Second, an edge cannot change to another community once it has been assigned (most expansion methods also have this drawback). We believe these reasons can partly explain the poor performance of Linkcomm [39].

The computational complexity of ITEM’s expansion method is $O(tpm)$, where $t$ is the number of iterations of the EM algorithm. $t$ may be associated with $\log(n)$ because the average path length is proportional to $\log(n)$ for “small-world” networks [48]. In our experiments, the expansion process stops when only a few edges change labels. The overall computational complexity of ITEM is approximately $O(tpm)$, because the seeding complexity can be ignored in most cases (when compared with the expansion complexity). Hence, ITEM is appropriate for detecting communities in large networks. Table 3 lists ITEM’s running times for some empirical networks.\(^4\) The

\(^4\) http://snap.stanford.edu/data/index.html.
configuration was identical to the previous configuration (please see footnote 3). ITEM processed the Dblp network in 12 min. We believe that ITEM could efficiently process even larger networks using more machines, because of the ideal parallelism of the k-means method [49].

7 Conclusions and future work

In this work, we proposed an overlapping community detection algorithm called ITEM. ITEM was designed to efficiently select high-quality seeds and to use an expansion method that is applicable to a wide range of networks. The high-quality seeds are selected using local and global methods, and the expansion method uses semi-supervised learning and an NB classifier. Our experimental results demonstrate that our seeding and expansion methods perform well, and our statistical analyses show that ITEM performs significantly better than most existing algorithms.

We must urgently develop methods for automatically generating ground truth communities based on a network’s metadata. As more high-quality ground truth communities become available, we can further verify ITEM on social networks like Twitter and LinkedIn. Additionally, high-quality benchmarks will make the comparisons among different algorithms more convincing.

At present, ITEM can only run on unweighted and undirected networks, but we will extend it in the future. ITEM may be treated as a preparatory step for other tasks such as link prediction and key point detection, because we
can use it to determine parameters such as $p(v_j|C_k)$ and $p(C_k|v_j)$. Last, it is also worth implementing ITEM for incremental or dynamic networks.

**Acknowledgments** The authors would like to thank the anonymous referees for their comments. This research is partially supported by grants from the National Natural Science Foundation of China (No. 61303167 and No. 61363047) and by funds from the Basic Research Program of Shenzhen (Grant No. JCYJ20130401170306838).

**References**

1. Girvan M, Newman ME (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821
2. Borgs C, Chayes J, Madhian M, Saberi A (2004) Exploring the community structure of newsgroups. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 783–787, 2004
3. Lee C, Reid F, McDaid A, Hurley N (2010) Detecting highly overlapping community structure by greedy clique expansion. In: SNA-KDD’10: Proceedings of the 4th Workshop on Social Network Mining and Analysis, 2010
4. Lancichinetti A, Fortunato S, Kertész J (2009) Detecting the overlapping and hierarchical community structure in complex networks. New J Phys 11(3):033015
5. Lee C, Reid F, McDaid A, Hurley N (2011) Seeding for pervasively overlapping communities. Phys Rev E 83(6):066107
6. Lancichinetti A, Fortunato S (2009) Community detection algorithms: a comparative analysis. Phys Rev E 80(5):056117
7. Newman ME, Leicht EA (2007) Mixture models and exploratory analysis in networks. Proc Natl Acad Sci 104(23):9564
8. Shen H, Cheng X, Cai K, Hu MB (2009) Detect overlapping and hierarchical community structure in networks. Phys A Stat Mech Appl 388(8):1706
9. Bron C, Kerbosch J (1973) Algorithm 457: finding all cliques of an undirected graph. Commun ACM 16(9):575
10. Baumes J, Goldberg MK, Krishnamoorthy MS, Magdon-Ismail M, Preston N (2005) Finding communities by clustering a graph into overlapping subgraphs. IADIS AC 5:97
11. Baumes J, Goldberg M, Magdon-Ismail M (2005) Intelligence and security informatics. Springer, New York, pp 27–36
12. Yang J, Leskovec J (2012) Proceedings of the ACM SIGKDD Workshop on mining data semantics, ACM, p 3, 2012
13. Borgatti SP (2012) Computational complexity—theory, techniques, and applications. In: Meyers RA (ed). Springer, New York, pp 2912–2924
14. Berry MW, Castellanos M (2004) Survey of text mining. Springer, New York
15. Koller D, Sahami M (1997) Proceedings of ICMC–97, 14th International Conference on machine learning. Morgan Kaufmann Publishers, Burlington, pp 170–178
16. Robertson S (2004) Understanding inverse document frequency: on theoretical arguments for IDF. J Doc 60(5):503
17. Zhu X (2006) Semi-supervised learning literature survey. Comput Sci Univ Wis Madison 2:3
18. Jiang J, Yan X, Yu Z, Guo J, Tian W (2014) A Chinese expert disambiguation method based on semi-supervised graph clustering. Int J Mac Learn Cybern 3(3):247
19. Maulik U, Chakraborty D (2012) A novel semisupervised SVM for pixel classification of remote sensing imagery. Int J Mac Learn Cybern 3(3):247
20. Chen WJ, Shao YH, Hong N (2014) Laplacian smooth twin support vector machine for semi-supervised classification. Int J Mac Learn Cybern 5(3):459
21. Tinha J, van Someren M, Afsarmanesh H (2015) Semi-supervised self-training for decision tree classifiers. Int J Mac Learn Cybern :1–16
22. Domingos P, Pazzani M (1997) On the optimality of the simple Bayesian classifier under zero-one loss. Mach Learn 29(2–3):103
23. Shang C, Li M, Feng S, Jiang Q, Fan J (2013) Feature selection via maximizing global information gain for text classification. Knowl Based Syst 54:298
24. Charikar MS (2002) Similarity estimation techniques from rounding algorithms. In: Proceedings of the thirteenth annual ACM symposium on Theory of computing, ACM, pp 380–388, 2002
25. Manku GS, Jain A, Das Sarma A (2007) Detecting near-duplicates for web crawling. In: Proceedings of the 16th international conference on World Wide Web, ACM, pp 141–150, 2007
26. Dhillon IS, Mallela S, Modha DS (2003) Information-theoretic co-clustering. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 89–98, 2003
27. Miladic D, Grobelnik M (1999) Feature selection for unbalanced class distribution and naive bayes. ICML 99:258–267
28. Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110
29. Lancichinetti A, Fortunato S (2009) Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys Rev E 80(1):016118
30. Traud AL, Kelsic ED, Mucha PJ, Porter MA (2011) Comparing community structure to characteristics in online collegiate social networks. SIAM Rev 53(3):526
31. Traud AL, Mucha PJ, Porter MA (2012) Social structure of Facebook networks. Phys A Stat Mech Appl 391(16):4165
32. Lee C, Cunningham P (2014) Community detection: effective evaluation on large social networks. J Comp Netw 2(1):19
33. Gargi U, Lu W, Mirrokni VS, Yoon S (2011) Large-Scale Community Detection on YouTube for Topic Discovery and Exploration. ICWSM
34. Subbian K, Aggarwal CC, Srivastava J, Yu PS (2013) Community Detection with Prior Knowledge. In: Proceedings of the 2013 SIAM International Conference on data mining, SIAM, pp 405–413, 2013
35. Yang T, Jin R, Chi Y, Zhu S (2009) Combining link and content for community detection: a discriminative approach. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, pp 927–936, 2009
36. Gopalan PK, Blei DM (2013) Efficient discovery of overlapping communities in massive networks. Proc Natl Acad Sci 110(36):14534
37. Andersen R, Bleich DF, Mirrokni V (2012) Overlapping clusters for distributed computation. In: Proceedings of the fifth ACM international conference on Web search and data mining, ACM, pp 273–282, 2012
38. Bleich DF, Seshadhri C (2012) Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, pp 597–605, 2012
39. Xie J, Kelley S, Szymanski BK (2013) Overlapping Community Detection in Networks: The State-of-the-art and Comparative Study. ACM Comput Surv 45(4):43. doi:10.1145/2501654.2501657
40. Xie J, Szymanski BK, Liu X (2011) Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In: Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on, IEEE, pp 344–349, 2011
41. Gregory S (2010) Finding overlapping communities in networks by label propagation. New J Phys 12(10):103018
42. Lancichinetti A, Radicchi F, Ramasco JJ, Fortunato S (2011) Finding statistically significant communities in networks. PLoS One 6(4):e18961
43. Palla G, Derényi I, Farkas I, Vicsek T (2005) Uncovering the overlapping community structure of complex networks in nature and society. Nature 435(7043):814
44. Ball B, Karrer B, Newman MEJ (2011) Efficient and principled method for detecting communities in networks. Phys Rev E 84(3):036103
45. Chapelle O, Schölkopf B, Zien A (2006) Risks of semi-supervised learning: how unlabeled data can degrade performance of generative classifiers, in semi-supervised learning. MIT Press, Massachusetts, pp 57–72
46. Ahn YY, Bagrow JP, Lehmann S (2010) Link communities reveal multiscale complexity in networks. Nature 466(7307):761
47. Ding C, He X (2002) Cluster merging and splitting in hierarchical clustering algorithms. Data mining, 2002, ICDM 2003. Proceedings. 2002 IEEE International Conference on, IEEE, pp 139–146, 2002
48. Newman ME (2003) The structure and function of complex networks. SIAM Rev 45(2):167
49. Stoffel K, Belkoniene A (1999) Parallel K/h-Means Clustering for Large Data Sets. In: Proceedings of the 5th International Euro-Par Conference on parallel processing. Springer, New York, pp 1451–1454, (Euro-Par ’99)