Knowledge-Based Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Ensemble-based overlapping community detection using disjoint community structures

Tanmoy Chakraborty [a,*], Saptarshi Ghosh [b], Noseong Park [c]

[a] Department of CSE, IIIT Delhi, India
[b] Department of CSE, IIT Kharagpur, India
[c] Department of Information Sciences and Technology, George Mason University, USA

ARTICLE INFO

Article history:
Received 7 March 2018
Received in revised form 19 August 2018
Accepted 24 August 2018
Available online xxxx

Keywords:
Ensemble algorithm
Overlapping communities
Community detection

ABSTRACT

While there has been a plethora of approaches for detecting disjoint communities from real-world complex networks, some methods for detecting *overlapping* community structures have also been recently proposed. In this work, we argue that, instead of developing separate approaches for detecting overlapping communities, a promising alternative is to infer the overlapping communities from multiple disjoint community structures. We propose an ensemble-based approach, called EnCoD, that leverages the solutions produced by various disjoint community detection algorithms to discover the overlapping community structure. Specifically, EnCoD generates a feature vector for each vertex from the results of the base algorithms and learns which features lead to detect densely connected overlapping regions in an unsupervised way. It keeps on iterating until the likelihood of each vertex belonging to its own community maximizes. Experiments on both synthetic and several real-world networks (with known ground-truth community structures) reveal that EnCoD significantly outperforms nine state-of-the-art overlapping community detection algorithms Finally, we show that EnCoD is generic enough to be applied to networks where the vertices are associated with explicit semantic features. To the best of our knowledge, EnCoD is the second *ensemble-based overlapping community detection approach* after MEDOC Chakraborty (2016).

1. Introduction

Real-world networks are complex, high-dimensional and multi-faceted, thus can be interpreted in many different ways. A fundamental property of a real-world network is its "community structure", which is often assumed as organizational units in social networks [1], functional modules in biological networks [2], scientific communities in citation networks [3], and so on. Despite a huge amount of effort devoted in past decade or more [4,5], the problem of community detection (CD) has turned out to be more complicated because of two reasons — (i) CD is an *ill-defined* problem [4]; therefore one can obtain multiple solutions for a given network, each of which is important in its own way. Moreover a single objective function (even the "best", if exists) may not be able to effectively model such vast dimensions present in a network. (ii) The vertices in real-world networks often belong to multiple communities [6,7], leading to the community structure being overlapped. None of the existing disjoint CD algorithms are able to detect overlapping regions inside a community structure.

Although a growing body of literature is focusing on discovering the overlapping community structure after the introductory work of Palla et al. [6], the number is considerably less compared to the vast amount of literature on disjoint CD. Moreover, in recent years, Chakraborty et al. [8,9] showed that one can successfully detect overlapping regions inside a community structure by leveraging huge number of diverse and accurate disjoint community structures.

In this paper, we attempt to utilize multiple views of the disjoint community structure observed in a given network (obtained from different disjoint CD algorithms) by merging these views, distilling all their good qualities into an *ensemble solution* [10,11]. The motivation for our work comes from the fact that ensemble approaches have already been proved to be successful in clustering and classification tasks [12]. Hence, ensemble approaches, if designed properly, can perform well in overlapping community detection as well. However, the challenge in applying ensemble techniques to network-based analysis is due to the lack of sufficient information such as representative features of each vertex, extent of similarity between pair-wise vertices, and so on.

Empirical observations and Motivation. Our algorithm is primarily built on two interesting empirical observations on six large real networks (see Section 4 for dataset description). While analyzing the ground-truth community structures of these networks, we find many solutions by running each of the base disjoint CD algorithms on every single network. The solutions have significant

* Corresponding author.
E-mail addresses: tanmoy@iiitd.ac.in (T. Chakraborty), saptarshi@cse.iitkgp.ac.in (S. Ghosh), npark9@gmu.edu (N. Park).

https://doi.org/10.1016/j.knosys.2018.08.033
0950-7051/

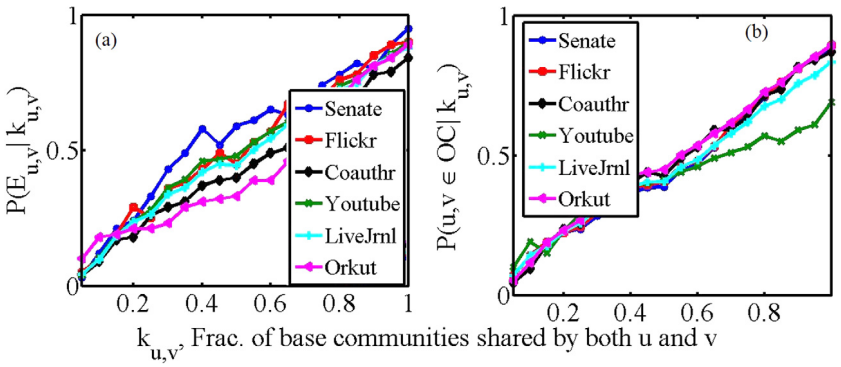

Fig. 1. Conditional probability P (in y-axis) of (a) existence of an edge between two vertices, and (b) common overlapping community (OC) membership of two vertices in the ground-truth community structure of six real networks, given the fraction of the base disjoint communities that both the vertices share. We conclude that the more the vertices share common base disjoint communities, the more the chance that they are connected in the graph and belong to the same overlapping community.

(dis)similarity with the ground-truth, corroborating the observation in [13]. We represent the memberships of each vertex in different base disjoint communities by a feature vector. Then we examine the probability of two vertices u and v being connected in the network, given that they have $k_{u,v}$ fraction of common base community memberships. Fig. 1(a) plots this conditional probability for the six networks. We see an increasing relationship, i.e., *the more base disjoint communities a pair of vertices has in common, the higher the probability of an edge connecting them*.

Further, we examine the conditional probability that two vertices belong to the same ground-truth overlapping community given that they have $k_{u,v}$ fraction of common base disjoint community memberships. Once again we observe a positive correlation in Fig. 1(b). Thus, *the probability of a pair of vertices belonging to the same overlapping community increases with the increase of their shared base community memberships*. The above observations motivated us to develop an overlapping CD algorithm, by combining disjoint community structures identified for a given network.

Ensemble-based overlapping community detection. Our method iteratively learns major dimensions of features for vertices that lead to densely-connected groups. In practice, since base disjoint CD algorithms produce many (and significantly different) community structures from a network, we leverage this information to extract the latent feature information associated with each vertex.[1] Our algorithm, named EnCoD is built on two hypotheses — first, an overlapping community is formed by a group of densely-connected vertices; second, vertices within a community have significantly high feature similarity. We represent the memberships of each vertex in different base disjoint communities by a feature vector, and then use the similarity between two vertices. We then learn important features that may lead to discover densely-connected overlapping communities [14,15]. To the best of our knowledge, EnCoD is the second algorithm after MEDOC [9] to propose *ensemble-based* overlapping CD algorithm by leveraging several disjoint community structures. Note that other two algorithms, namely PEACOCK [16] and PVOC [8] can detect overlapping community from a *single* disjoint community structure by applying an efficient post-processing technique. They do not leverage *multiple* disjoint community structures.

Summary of the evaluation. Experiments are conducted on both synthetic and six large real-world networks with known ground-truth overlapping community structure. We experiment with various disjoint CD algorithms as the base algorithms for EnCoD. For comparative evaluation, we choose seven non-ensemble based overlapping CD algorithms. Additionally, we compare EnCoD with PVOC [8] and MEDOC [9], two most recent overlapping CD algorithms that leverage disjoint community structures. Few notable observations are as follows.

(i) EnCoD is almost independent of how perfect the base CD algorithms are; however, stronger base algorithms have more impact in the aggregation stage than the weaker base algorithms. Also, the accuracy of EnCoD never decreases with the increase in base solutions.

(ii) EnCoD tends to get saturated after aggregating a certain number of base solutions, after which there is almost no effect on the final performance. However, the accuracy never decreases with the increase in base solutions.

(iii) On average, EnCoD outperforms the best baseline overlapping CD algorithm in terms of the overlapping NMI metric (*ONMI*) [17], Ω index and F-Score.

(iv) We further apply EnCoD to networks where, apart from the topological structure, every vertex is associated with additional feature information. In such cases, EnCoD systematically combines the features into the model and produces more accurate overlapping community structure.

Contributions of the paper. The major contributions of the paper are four-fold:

- Ensemble algorithms for community detection have rarely been studied. This paper will pave the way for enhancing the state-of-the-art on ensemble based community detection.
- The idea of using the membership of a vertex in different disjoint community structures as features is unique and, to our knowledge, has not been used earlier in community detection.
- Empirical results indicate that EnCoD is either as good as the best baseline CD algorithm, or sometimes performs even better than that. Therefore, one can rely on such ensemble framework instead of choosing and picking the best one from the jungle of CD algorithms available at present.
- The framework used in EnCoD is generalized enough to be used to combine any feature set of vertices to detect community structure (as shown in Section 6). Therefore, if the topology of a network is incomplete and sparse in nature and the properties of vertices are available, one can use the proposed framework to combine the features into the model for overlapping community detection.

2. Related work

Due to the abundance of literature on community detection (CD) algorithms [4,5,18], we restrict our discussion to some selected works that we deem as pertinent to our study. Two extensive survey articles on CD are [4] and [5].

[1] Note that the set of communities returned by a CD algorithm is termed as 'community structure'.

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.

Traditional community detection algorithms. Most of the early research in CD assumed the communities to be disjoint. Many different CD approaches have been proposed, including, modularity optimization [19–21], information theoretic approaches [22], vertex similarity-based approaches [23], significance-based approaches [24], label propagation [25], diffusion-based approach [26], and so on (see [4,5] for surveys). Gradually, researchers realized that in real-world networks, a vertex can belong to multiple communities, resulting overlapping community structure. Palla et al. proposed 'CFinder' [6], the first ever method to detect overlapping communities based on clique-percolation technique. Subsequently, several overlapping CD techniques have been proposed, including modified modularity [27], fitness function [28], local density function [29], affinity propagation function [30], integrated approach [31], node location analysis based approach [32], non-negative matrix factorization [7], and so on. Recently, Li et al. [33] proposed a local spectral clustering for overlapping community detection. Huang et al. [34] proposed an overlapping community detection algorithm in heterogeneous social networks via the user model. Sattari and Zamanifar [35] suggested a spreading activation-based label propagation algorithm for overlapping community detection in dynamic social networks. Notably, though there have been several recent studies on the topic, the depth of study on overlapping community detection is much lesser than that for the disjoint case.

Ensemble-approach for community detection. Ensemble approach has been well-studied in traditional data mining for clustering data points (see [12] for a detailed review). These approaches can be classified into two categories [12]: object co-occurrence based approaches and median partitioning based approaches. However, this number is significantly less when we talk about clustering vertices in the network. A pioneering attempt was made by Dahlin and Svenson [10] to propose ensemble clustering for network data. Raghavan et al. [36] showed the effectiveness of merging outputs of several community detection algorithms. Ovelgönne and Geyer-Schulz proposed CGGC [37], an ensemble-based modularity maximization approach. Kanawati proposed YASCA [38] that for each seed vertex considers its ego-centric network, partitions the network in different ways and combines the outputs. He further tuned two parameters — quality and diversity of the partitions for final combination [39]. Recently, Lancichinetti and Fortunato [11] proposed 'consensus clustering' which utilizes a consensus matrix for disjoint community detection. However, all the above works focused on *identifying disjoint communities*.

Recently, two algorithms, PEACOCK [16] and PVOC [8] argued that intelligently post-processing a disjoint solution can lead to the detection of overlapping community structure. PVOC was reported to outperform PEACOCK [8]. The difference between PVOC and EnCoD is that PVOC uses a novel post-processing technique that modifies a given disjoint community structure using a metric called 'permanence' [40,41], whereas EnCoD does not do any post-processing; rather it considers multiple disjoint community structures and uses them as features of a vertex for detecting overlapping regions. Another ensemble algorithm, called MEDOC [9] uses the idea of meta community (community of communities). It creates a multi-partite network using the base disjoint community structure. After this, it runs an existing CD algorithm to partition the multi-partite network. Finally, a membership function is to determine the membership of a vertex to a community. EnCoD does not create any meta-community using base community structures; rather it uses them to generate features for each vertex to be used further in the optimization algorithm. In Section 5.2, we will show that our algorithm outperforms both PVOC and MEDOC.

Table 1
Important notations used in this paper.

Notation	Description
$G(V, E)$	An undirected network with sets of vertices V and edges E
$\mathcal{AL}$	$\{Al_{m=1}^{M}\}$, set of M base disjoint CD algorithms
$\mathbb{C}_m^k$	$\{C_m^{1k}, \ldots, C_m^{ak}\}$, base community structure discovered by algorithm Al_m on kth vertex ordering
Γ_m	$\{\mathbb{C}_m^k\}_{k=1}^{K}$, set of base disjoint community structures discovered by base algorithm Al_m on K different vertex orderings
Γ	$\Gamma_{m=1}^{M}$, set of all MK base disjoint community structures
$\bar{a}$	Average size of a base community structure
ξ	$\bar{a}MK$, approx. total number of communities in Γ
$\mathbb{OC}$	$\{OC_1, OC_2, \ldots\}$, final overlapping community structure consisting of overlapping communities
$\mathbb{OC}_t$	Overlapping community structure at iteration t of EnCoD
ζ	$\{\tau_1, \tau_2, \ldots\}$, set of thresholds where τ_j corresponds to overlapping community OC_j
τ_L	Global community threshold used to control the community size
F_v	Feature vector of vertex v

3. Ensemble-based overlapping CD algorithm

We design an overlapping CD algorithm by leveraging the solutions obtained from several disjoint CD algorithms. This section details the proposed overlapping CD algorithm, named EnCoD. We start by formally defining the ensemble-based overlapping community detection problem.

Definition 1 (*Ensemble-based Overlapping Community Detection*)**.** Given a network $G = (V, E)$ and a set of M disjoint community detection algorithms $\mathcal{AL} = \{Al_m\}_{m=1}^{M}$, we aim at grouping the vertices into communities $\mathbb{OC} = \{OC_1, OC_2, \ldots\}$ in such a way that a vertex may belong to multiple communities and the following conditions are satisfied: $|\bigcup_i OC_i| = |V|$ and $\nexists i : OC_i = \phi$.

We now describe our algorithm, called EnCoD, an **En**semble based Overlapping **Co**mmunity **D**etection. The overall framework of the algorithm is presented in Algorithm 1 and important notations are summarized in Table 1.

Inputs to the algorithm: Two main inputs of the algorithm are the network $G(V, E)$ and the set of M base algorithms $\mathcal{AL} = \{Al_m\}_{m=1}^{M}$ which yield disjoint community structures. Apart from these, one needs to specify: K, number of vertex orderings of a particular network on which a base CD algorithm will run[2]; $\mathbb{INV}(v, C)$, an involvement function that will determine to what extent v is a part of a base community C; $\mathbb{SIM}'(C, v)$, a function that will measure the similarity of a vertex v with a community C; and τ_L, a threshold that confirms the minimum similarity value required by the vertices in a community to remain together, failing which the community is discarded. Possible definitions of these functions are discussed in Section 5.1.

Summary of the algorithm: Motivated by the ego-centric circle detection algorithms proposed in [43,44], we design EnCoD that aims at maximizing the likelihood of the membership of each node inside a community. EnCoD starts by running all the base disjoint CD algorithms on different vertex orderings[3] of a network, which produce dissimilar base community structures (Section 3.1). Following this, the base community information is used to generate a feature vector for each vertex (Section 3.2); this in turn transforms the vertices in the network into a feature space. The algorithm now tries to infer the overlapping community structure from the vector

[2] Previous research reported that most disjoint CD algorithms produce different community structures depending on the ordering of vertices in the input [42,40,11].

[3] We here use random vertex ordering strategy [42].

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.

representation of vertices, by iteratively optimizing an objective function (detailed in Section 3.3). The actual iteration starts by assigning each vertex to a singleton community, and a high similarity threshold value is assigned to each community to maintain high integrity among its constituent vertices. In each iteration, the algorithm manipulates the community structure by randomly removing vertices from their assigned communities and allocating vertices to some unassigned communities, such that the similarity condition is not violated (Section 3.4). After each iteration, the similarity threshold associated with each community is updated (Section 3.5). The iteration continues as long as the value of the objective function does not decrease.

In the rest of the section, we elaborate the subroutines mentioned in Algorithm 1.

Algorithm 1: EnCoD: An **En**semble based Overlapping **Co**mmunity **D**etection

Data: Network $G(V, E)$;
A set of base algorithms $\mathcal{AL} = \{Al_m\}_{m=1}^{M}$;
K: No of iterations;
$\mathbb{INV}(.,.)$: Involvement function;
$\mathbb{SIM}'(.,.)$: Vertex-to-community similarity function;
τ_L: Lower limit for community threshold
Result: Overlapping community structure $\mathbb{OC}$

1 $\Gamma = \phi$
2 **for** *each algorithm* $Al_m \in \mathcal{AL}$ **do**
3 Generate K different vertex orderings of G, run Al_m on each vertex ordering and obtain K disjoint community structures Γ_m. The size of each community structure $\mathbb{C}_m^k \in \Gamma_m$ is different. Each such set is denoted by $\mathbb{C}_m^k = \{C_m^{1k}, ..., C_m^{ak}\}$
4 $\Gamma = \Gamma \cup \Gamma_m$
5 **for** *each* $v \in V$ **do**
6 $F_v = ExtractFeatures(v, \Gamma, \mathbb{INV})$;
// Initialization
7 $\mathbb{OC} = \phi$ // Final overlapping community structure
8 $\mathbb{OC}_0 = \phi$ // Initial set of overlapping community
9 $\zeta_0 = \phi$ // Initial set of community threshold
10 **for** *each* $v \in V$ **do**
11 $OC_0^v = \{v\}$ // Each vertex is in each community
12 $\mathbb{OC}_0 = \mathbb{OC}_0 \cup OC_0^v$
13 $\tau_0^v = \infty$ // Arbitrary high threshold
14 $\zeta_0 = \zeta_0 \cup \tau_0^v$
15 $Plogl = -1$ // Previous log likelihood
16 $Clogl = -1$ // Current log likelihood
17 $t = 1$ // Current number of iterations
18 $\mathbb{OC}_t = \mathbb{OC}_0$
19 $\zeta_t = \zeta_0$
20 **while** $Clogl \geq Plogl$ **do**
21 $\mathbb{OC} = \mathbb{OC}_t$
22 $Plogl = Clogl$
23 $\mathbb{OC}_{t+1} = ManipulateComm(V, \mathbb{OC}_t)$
// Update thresholds and communities
24 **for** $OC_{t+1}^j \in \mathbb{OC}_{t+1}$ **do**
25 $\tau_{t+1}^j \leftarrow min\{\mathbb{SIM}'(OC_{t+1}^j, v) | v \in OC_{t+1}^j\}$
26 **if** $\tau_{t+1}^j < \tau_L$ **then**
// Threshold condition is violated
27 $\mathbb{OC}_{t+1} \leftarrow \mathbb{OC}_{t+1} \setminus \{OC_{t+1}^j\}$
28 **for** *each* $v \in OC_{t+1}^j$ **do**
29 $\mathbb{OC}_{t+1} \leftarrow \mathbb{OC}_{t+1} \cup \{v\}$
30 $\zeta_{t+1} \leftarrow \zeta_{t+1} \setminus \tau_{t+1}^j$
31 $Clogl = l_{\zeta_{t+1}}(G; \mathbb{OC}_{t+1})$ [Eq. 8]
32 $t = t + 1$
33 **return** $\mathbb{OC}$

3.1. Generate base communities

Assume that we are given a network $G = (V, E)$ and M different base CD algorithms $\mathcal{AL} = \{Al_m\}_{m=1}^{M}$. For each such base algorithm Al_m, EnCoD generates K different vertex orderings of G and runs Al_m on each vertex ordering. This results in K different community structures denoted as $\Gamma_m = \{\mathbb{C}_m^k\}_{k=1}^{K}$, where each community structure $\mathbb{C}_m^k = \{C_m^{1k}, \ldots, C_m^{ak}\}$ constitutes a specific partitioning of vertices in G. The size of each $\mathbb{C}_m^k$ might be different. We choose to use different vertex orderings of G since it has been shown that many of the disjoint CD algorithms produce different community structures depending on the ordering of vertices in the input [40, 11]. Here we use random vertex ordering strategy which ensures that the base algorithm starts from different seed vertices without any predefined bias and produces diverse community structures. At the end of this step, we obtain Γ, a set of MK base community structures (Step 3 of Algorithm 1).

Function 1: *ExtractFeatures*(Current vertex: v, Set of base disjoint communities: Γ, Involvement function: $\mathbb{INV}$)

1 $F_v = \phi$; $D_v = 0$; $\xi = 0$; $AF(v) = \phi$; // Auxiliary vector
2 **for** *each* $\Gamma_m \in \Gamma$ **do**
3 **for** *each* $\mathbb{C}_m^k \in \Gamma_m$ **do**
4 **for** *each* $C \in \mathbb{C}_m^k$ **do**
5 Measure $d_v^C = 1 - \mathbb{INV}(v, C)$;
6 $AF_v = AF_v \cup \{d_v^C\}$;
7 **if** $d_v^C \geq D_v$ **then**
8 $D_v = d_v^C$;
9 $\xi = \xi + 1$;
10 **for** *each* $AF_i(v) \in AF(v)$ **do**
11 Compute $P(C_i|v) = \frac{D_v - AF_i(v) + 1}{\xi . D_v + \xi - \sum_{k=1}^{\xi} AF_k(v)}$;
12 $F_v = F_v \cup \{P(C_i|v)\}$;
13 **return** F_v

3.2. Generate feature vectors for vertices

The ensemble step in Function 1 (which is invoked in step 5 of Algorithm 1) intends to leverage the base community structures Γ to construct the feature vector of vertices.

Definition 2 (*Feature Vector for a Vertex*)**.** Given a set of base community structures Γ, the feature vector of vertex v is denoted by F_v, whose ith entry $F_v(i) = P(C_i|v)$ indicates the probability of v being a member of the base community C_i, given vertex v is observed. The length of F_v is $|F_v| = \xi = \sum_{\Gamma_m \in \Gamma} \sum_{\mathbb{C}_m^k \in \Gamma_m} |\mathbb{C}_m^k|$. The value of $P(C_i|v)$ is derived by measuring the involvement of v into C_i.

We now describe how we compute F_v for a vertex v using Function 1. To start with, we measure $\mathbb{INV}(v, C)$, the involvement v into each base community $C \in \mathbb{C}_m^k$ (see Section 5.1 for the possible definitions of $\mathbb{INV}$). Note if v is not a part of a certain community C, we leave the corresponding field as zero. The value returned by $\mathbb{INV}$ is subtracted from 1 to get the distance of v from the community (Step 5). Following this, we construct an auxiliary feature vector $AF(v)$ for vertex v, which consists of elements indicating the distance of v from each base community (Step 6). The number of communities $\xi \in \Gamma$ can be approximated by $\xi = \bar{a}MK$, where the average size of a community is denoted by $\bar{a}$. Therefore, the size of $AF(v)$ is same as ξ. $D_v = \max_i AF_i(v)$ denotes the maximum distance of v from any of the base community present in Γ (Step 8).

To construct the *actual feature vector* F_v for vertex v, we measure the probability associated with each base community C_i given that vertex v is observed. For a given vertex v, the community label C_i is assumed to be a random variable from a distribution with probabilities $\{P(C_i|v)\}_{i=1}^{\xi}$. One can provide a non-parametric estimation of such probabilities based on the evidences of the data and the community structure. To transfer the distance into probability (the smaller the distance, the larger the probability),

we use the following conditional probability [9] for vertex v to be a part of C_i (Step 11):

$$P(C_i|v) = \frac{D_v - AF_i(v) + 1}{\xi.D_v + \xi - \sum_{k=1}^{\xi} AF_k(v)} \quad (1)$$

The denominator of the above equation ensures that $\sum_{i=1}^{\xi} P(C_i|v) = 1$. Adding 1 in the numerator ensures a non-zero probability for all vertex-community pairs. For smaller distance values, $P(C_i|v)$ increases proportionately to $D_v - AF_i(v)$, i.e., the larger the deviation of $AF_i(v)$ from D_v, the more the increase. As a consequence, the corresponding community C_i becomes more likely for v. Now we can construct the actual feature vector F_v for v as $F_v = \{P(C_k|v)\}_{k=1}^{\xi}$ (Step 12). This process essentially maps a vertex to a ξ-dimensional vector space.

3.3. Objective function

After the ensemble step, each vertex is represented by a feature vector. Our method of detecting the overlapping community structure (denoted by $\mathbb{OC} = \{OC_1, OC_2, \ldots\}$) aims at maximizing a certain objective function which represents the likelihood of each vertex being a part of its own community. The likelihood function assumes that if two vertices u and v are found to have very similar feature vectors, then they – (i) should be connected by an edge in the network, and (ii) should stay in the same community. These two assumptions are supported by the observations in Fig. 1. We use the knowledge of the edges in the network to validate the inferred community structure.

To start with, let us define few measures required to formulate the objective function. Note that all the communities mentioned in this subsection correspond to the overlapping communities that we aim to detect from a network.

Definition 3 (*Similarity between Two Vertices*)**.** For vertices u and v, let $F_u = \{F_u^i\}_{i=1}^{\xi}$ and $F_v = \{F_v^i\}_{i=1}^{\xi}$ represent their feature vectors respectively. Then the similarity between two vertices u and v is calculated using the cosine similarity among the feature vectors.

$$\mathbb{SIM}(u, v) = \frac{F_u.F_v}{|F_u|\,|F_v|} = \frac{\sum_{i=1}^{\xi} F_u^i F_v^i}{\sqrt{\sum_{i=1}^{\xi} {F_u^i}^2}\sqrt{\sum_{i=1}^{\xi} {F_v^i}^2}} \quad (2)$$

We also tried other similarity measures (see Section 5.1), and obtained best results with cosine similarity. With this similarity measure, we further define the similarity of a vertex with a community as follows.

Definition 4 (*Similarity between a Vertex and a Community*)**.** Given a community OC and a vertex v, the similarity of v and OC is calculated as the average similarity of v with all other vertices in OC.

$$\mathbb{SIM}'(OC, v) = \frac{\sum_{u \in OC} \mathbb{SIM}(u, v)}{|OC|} \quad (3)$$

Each community is associated with a minimum similarity threshold which every vertex needs to satisfy in order to be a part of the community.

Definition 5 (*Similarity Threshold per Community*)**.** Each community OC_j in our model is assigned a similarity threshold τ_j in such a way that if vertex $v \in V$ is in OC_j then it should satisfy $\mathbb{SIM}'(OC_j, v) \geq \tau_j$.

Given $\mathbb{OC} = \{OC_1, OC_2, \ldots\}$, an overlapping community structure and a set of thresholds $\zeta = \{\tau_1, \tau_2, \ldots\}$, we then define a *membership similarity* between two vertices based on their membership in different overlapping communities.

Definition 6 (*Membership Similarity of Pair-wise Vertices*)**.** The membership similarity of two vertices u and v is defined in terms of their memberships in different communities as follows:

$$\beta(u, v) = \exp\{[\beta_1(u, v)]^2 - [\beta_2(u, v)]^2\} \quad (4)$$

where,

$$\beta_1(u, v) = \sum_{OC_j : \{u,v\} \subseteq OC_j} (\mathbb{SIM}(u, v) - \tau_j + \lambda)^{-1}$$
$$\beta_2(u, v) = \sum_{OC_j : \{u,v\} \nsubseteq OC_j} (\mathbb{SIM}(u, v) - \tau_j + \lambda)^{-1} \quad (5)$$

We consider $\{u, v\} \subseteq OC_j$ if both the vertices u and v belong to the same community OC_j; whereas $\{u, v\} \nsubseteq OC_j$ if OC_j does not contain either u or v or both. One should consider λ as large as possible so that individual terms in the summation are always positive. In practice, we set λ as the maximum of all the threshold values corresponding to the communities mentioned in Definition 5 (i.e., $\lambda = max\{\tau_1, \tau_2, \ldots\}$). The value of $\beta_1(u, v)$ (*resp.* $\beta_2(u, v)$) tends to become high if both u and v share (*resp.* do not share) common communities with very high thresholds. Furthermore, the value of membership similarity $\beta(u, v)$ depends not only on the common community memberships, but also on the threshold specific to the communities. In particular, with the increase of the number of the common communities where both u and v belong to and the community threshold, the value of $\beta(u, v)$ also increases.

Given the membership similarity of all pair-wise vertices, we further define the probability that a pair of vertices u and v are connected by an edge $E_{u,v}$ as follows:

$$p(E_{u,v} \in E) = \frac{\beta(u, v)}{1 + \beta(u, v)} \quad (6)$$

Similarly, the probability of two vertices *not* being connected is $p(E_{u,v} \notin E) = 1 - p(E_{u,v} \in E)$. The value of $p(E_{u,v} \in E)$ increases with the increase in $\beta(u, v)$, and add-one smoothing is applied to $p(E_{u,v} \in E)$ in order to avoid "divided by zero" situation. One can thus obtain the predicted probability $p(E_{u,v} \in E)$ from the information of the overlapping community structure $\mathbb{OC}$ and the set of thresholds ζ. The idea is that, the more the similarity between the feature vectors of two vertices, the common communities both of them are part of and the similarity threshold of the common communities, the more the membership similarity between them. Given the probability estimation, our model should therefore assure that its resultant network corresponds to the real network.

Let us assume that each edge is generated independently. Then the joint probability of G and $\mathbb{OC}$ is estimated as:

$$P_\zeta(G; \mathbb{OC}) = \prod_{E_{u,v} \in E} p(E_{u,v} \in E) \prod_{E_{u,v} \notin E} p(E_{u,v} \notin E) \quad (7)$$

The log likelihood of the joint probability is:

$$\begin{aligned} l_\zeta(G; \mathbb{OC}) &= \log(P_\zeta(G; \mathbb{OC})) \\ &= \sum_{E_{u,v} \in E} \log(p(E_{u,v} \in E)) + \sum_{E_{u,v} \notin E} \log(p(E_{u,v} \notin E)) \\ &= \sum_{E_{u,v} \in E} \log(\beta(u, v)) - \sum_{E_{u,v} \in V \times V} \log(1 + \beta(u, v)) \\ &= \sum_{(u,v) \in E} \phi(u, v) - \sum_{E_{u,v} \in V \times V} \log(1 + \exp\{\phi(u, v)\}) \end{aligned} \quad (8)$$

where $\phi(u, v) = \log(\beta(u, v)) = ([\beta_1(u, v)]^2 - [\beta_2(u, v)]^2)$

EnCoD attempts to maximize the objective function mentioned in Eq. (8) (see line 31 in Algorithm 1) in order to obtain the final overlapping community structure $\mathbb{OC}$.

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.

Function 2: *ManipulateComm*(Set of vertices: V, Current overlapping community structure at iteration t: $\mathbb{OC}_t$)

1 $\mathbb{OC}_{t+1} = \mathbb{OC}_t$
2 **for** *each* v *in* V **do**
3 $S1_t^v = \{OC_t^j | OC_t^j \in \mathbb{OC}_t \wedge v \in OC_t^j\}$
4 $S2_t^v = \{OC_t^j | OC_t^j \in \mathbb{OC}_t \wedge v \notin OC_t^j\}$
// *Rand(x, y): a random number between x and y*
5 $p_1 \leftarrow Rand(1, |S2_t^v|)$
6 $AC_{t+1}^v \leftarrow \left\lceil \frac{p_1 + |S1_t^v|}{|S1_t^v|} \right\rceil$
7 $p_2 \leftarrow Rand(1, |S1_t^v|)$
8 $RC_{t+1}^v \leftarrow \left\lceil \frac{p_2 + |S1_t^v|}{|S1_t^v|} \right\rceil$
9 Choose randomly OC_{AC} such that $OC_{AC} \subseteq S2_t^v$, $|OC_{AC}| = AC_{t+1}^v$
10 **for** *each* $OC_t^j \in OC_{AC}$ **do**
11 $OC_{t+1}^j \leftarrow OC_{t+1}^j \cup \{v\}$
12 Choose randomly OC_{RC} such that $OC_{RC} \subseteq S1_t^v$, $|OC_{RC}| = RC_{t+1}^v$
13 **for** *each* $OC_t^j \in OC_{RC}$ **do**
14 $OC_{t+1}^j \leftarrow OC_{t+1}^j \setminus \{v\}$
15 **return** $\mathbb{OC}_{t+1}$

3.4. Manipulate community structure

The framework for manipulating the overlapping community structure in each iteration is shown in Function 2. The function originally starts with the initializations mentioned in lines 7–9 of Algorithm 1. Initially at iteration $t = 0$, each vertex v is assigned to a separate community OC_0^v with a very high threshold value τ_0^v. In every iteration, we change the memberships of v by removing it randomly from certain communities it belongs to, and assigning it to certain communities where it did not belong to. The threshold corresponding to each community is updated accordingly such that the constraint in Definition 5 is not violated. For a certain community C_j, if threshold τ_j falls below a certain lower limit τ_L, the community is discarded because such a community is too heterogeneous to represent a dense group in our model. The value of τ_L is empirically determined as described in Section 5.1. Since we intend to discover the communities of each vertex, in every iteration, we concentrate more on those vertices which are yet assigned to less number of communities, than those vertices which are already assigned to many communities. Therefore, with the increase in the number of communities a vertex is already part of after iteration t, it is less likely that the community membership of v is manipulated at iteration $t + 1$.

At each iteration t, the function *ManipulateComm* takes the set of overlapping communities $\mathbb{OC}_t = \{OC_t^1, OC_t^2, \ldots\}$. Then for each vertex v, $S1_t^v = \{OC_t^j | OC_t^j \in \mathbb{OC}_t \wedge v \in OC_t^j\}$ indicates the set of communities where v belongs to at iteration t. Similarly, $S2_t^v = \{OC_t^j | OC_t^j \in \mathbb{OC}_t \wedge v \notin OC_t^j\}$ indicates those set of communities where v is currently not a member of.

Now, let AC_{t+1}^v denote the number of communities to add v to, and RC_{t+1}^v denote the number of communities to remove v from. These variables are measured as follows:

$$AC_{t+1}^v \leftarrow \left\lceil \frac{p_1 + |S1_t^v|}{|S1_t^v|} \right\rceil, \quad RC_{t+1}^v \leftarrow \left\lceil \frac{p_2 + |S1_t^v|}{|S1_t^v|} \right\rceil \tag{9}$$

Here, p_1 is a random integer chosen from $[1, |S2_t^v|)$ such that the number of communities AC_{t+1}^v to add v to, never exceeds $|S2_t^v|$, indicating the number of v's current communities. Similarly, p_2 is an integer randomly chosen from $[1, |S1_t^v|)$ such that the value of RC_{t+1}^v is less than or equal to $|S1_t^v|$ (number of communities that v is currently part of). Note both the values AC_{t+1}^v and RC_{t+1}^v tend to decrease with the increase of $|S1_t^v|$, ensuring that, the more the number of communities v is currently part of, the less the manipulation of v in the current iteration, and vice versa.

After selecting the number of communities v is to be added to and to be removed from, we choose the set of communities OC_{AC} from $S2_t^v$ randomly such that $|OC_{AC}| = AC_{t+1}^v$ and assign v to OC_{AC} (Step 12). Similarly, we randomly choose RC_{t+1}^v many communities (indicated by the set OC_{RC}) from $S1_t^v$ and remove v from it (Step 14). The corresponding changed communities are updated accordingly.

3.5. Update thresholds and compute objective function

Once the manipulation of the current overlapping community structure is over (as described above), we obtain a new overlapping community structure $\mathbb{OC}_{t+1}$. The next task is to update the threshold corresponding to the new overlapping community structure (Steps 23–31 in Algorithm 1). For community $OC_{t+1}^j \in \mathbb{OC}_{t+1}$, we set the new threshold τ_{t+1}^j as the minimum value of all the similarities of the OC_{t+1}^j's constituent vertices with OC_{t+1}^j itself, i.e., $\tau_{t+1}^j = min\{\mathbb{SIM}'(OC_{t+1}^j, v) | v \in OC_{t+1}^j\}$ (Step 25). This assignment ensures that Definition 5 is never violated.

Apart from the minimum threshold assigned to each community, we also maintain another global threshold τ_L, which ensures the overall coherence among the members in each community. Once all the community-centric thresholds are updated, we further check whether all these thresholds are higher than τ_L (Step 26). If some threshold τ_{t+1}^j corresponding to community OC_{t+1}^j falls below τ_L, we discard OC_{t+1}^j and set each of its constituent members as a separate community (Steps 26–29). The value of τ_L controls the size of the final overlapping communities — a higher value of τ_L results in smaller community size. The selection of the value of τ_L is described later in Section 5.1.

Finally, we compute the log likelihood after the current iteration $l_{\zeta_{t+1}}(G; \mathbb{OC}_{t+1})$ using Eq. (8) (Step 31) and compare it with $l_{\zeta_t}(G; \mathbb{OC}_t)$ (Step 20). If the former is greater than the latter, we keep the current community structure $\mathbb{OC}_{t+1}$ and the current set of thresholds ζ_{t+1} and continue the iteration. Otherwise, we discard the updated sets and retain the sets obtained in the previous iteration (Step 21). We terminate the process when there is no sufficient increase of the log likelihood. Note that this step is not shown in Algorithm 1 (which rather shows that the iteration immediately stops once the condition is violated) as it depends on the users' confidence on the number of iterations she wants to continue after the log likelihood condition is violated. In practice, continuing the iterations ensures that the algorithm does *not* get stuck at any local optima of the objective function.

3.6. Complexity analysis

EnCoD terminates when it reaches a maxima of the objective function and the objective function does not change further. In the worst case, after any iteration the maximum number of overlapping communities is $|V|$ and the maximum number of vertices within each community is also $|V|$. Therefore, the runtime after each iteration is $\mathcal{O}(|V| + |\mathbb{OC}|)$. Importantly, a manipulation of the current community structure is only possible if the log likelihood increases. Therefore, EnCoD generally converges after a finite number of iterations. In practice, we assume that the local maxima is reached if the log likelihood does not change after $|V|$ iterations.

Note that, since EnCoD is an ensemble algorithm, it requires the running of all base algorithms. However, speedup might be achieved by parallelizing the base algorithms, or by using fewer base algorithms (potentially at the cost of performance, as analyzed later in Section 5).

4. Experimental setup

In this section, we describe the datasets and the experimental setup used to evaluate the performance of EnCoD.

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.

Table 2
Description of the real-world networks (τ: avg. edge-density per community, $\bar{c}$: avg. community size, V_c: avg. number of community memberships per vertex).

Network	Vertex type	Edge type	Community type	Reference
Senate	Senate	Similar voting pattern	Congress	[45]
Flickr	User	Friendship	Joined group	[46]
Coauthorship	Researcher	Collaborations	Publication venues	[2]
Youtube	User	Friendship	User-defined group	[7]
LiveJournal	User	Friendship	User-defined group	[7]
Orkut	User	Friendship	User-defined group	[7]

Network	# vertices	# edges	# communities	τ	$\bar{c}$	V_c
Senate	1,884	16,662	110	0.65	81.59	4.76
Flickr	80,513	5,899,882	171	0.046	470.83	18.96
Coauthorship	391,526	873,775	8,493	0.231	393.18	10.45
Youtube	1,134,890	2,987,624	8,385	0.732	43.88	2.27
LiveJournal	3,997,962	34,681,189	310,092	0.536	40.02	3.09
Orkut	3,072,441	117,185,083	6,288,363	0.245	34.86	95.93

4.1. Datasets

We use two types of network datasets to evaluate various CD algorithms: (1) **Synthetic networks:** We use the LFR benchmark model [47] to generate synthetic networks along with their ground-truth communities. We follow the parameters configuration suggested in [8] while generating networks: number of vertices $N = 10{,}000$, average degree $\bar{k} = 50$, maximum degree $k_{max} = 150$, maximum community size $c_{max} = 150$, minimum community size $c_{min} = 50$, percentage of overlapping vertices $O_n = 15\%$, number of communities to which a vertex belongs $O_m = 20$, and mixing parameter $\mu = 0.3$ (representing the ratio of inter- and intra-community edges; the lower the value of μ, the better the communities are). For each configuration, we create 100 such LFR networks and report the average result. (2) **Real-world networks:** We use six real-world networks of widely different scales, whose ground-truth communities are available *a priori* (see Table 2).

4.2. Evaluation metrics

To compare the detected overlapping community structure with the ground-truth, we use three standard metrics: (1) Overlapping Normalized Mutual Information (*ONMI*) [17], (2) Omega (Ω) Index [7], and (3) F-Score [7].

4.3. Base disjoint CD algorithms for aggregation

There exist numerous disjoint CD algorithms, which differ in the way they define the community structure. Here we select the following disjoint CD algorithms from different categories based on their working principles: (i) *Modularity-based approach*: FastGreedy [21], Louvain [19] and CNM [20]; (ii) *Node similarity-based approach*: WalkTrap [23]; (iii) *Compression-based approach*: InfoMap [22]; (iv) *Diffusion-based approach*: Label Propagation [25]. EnCoD ensembles the outputs of all these six algorithms and generates the feature vectors for the vertices.

4.4. Baseline algorithms

We compare EnCoD with the following seven non-ensemble based overlapping CD algorithms that cover different types of overlapping CD heuristics: OSLOM [24], EAGLE [48], COPRA [49], SLPA [50], MOSES [51], BIGCLAM [7] and IEDC [31]. We further compare EnCoD with PVOC [8] and MEDOC [9] which are the most recent algorithms that use disjoint community structure to detect the overlapping communities.

5. Experimental results

In this section, we discuss how the performance of EnCoD depends on various parameters, and compare its performance with state-of-the-art overlapping CD algorithms. We run EnCoD on both synthetic and real-world networks, and compare its performance with that of seven state-of-the-art overlapping CD algorithms.

5.1. Parameter selection

For parameter selection in EnCoD, we perform experiments over LFR networks of widely-varying quality of ground-truth community structures, obtained by varying the mixing parameter μ from 0.1 to 0.8 with the increase of 0.1.

- **Involvement Function** ($\mathbb{INV}$): The following two functions are used to quantify what extent a vertex is involved in a community [9]: (i) *Closeness Centrality* measures how close vertex v is from other vertices where v is a part of, i.e., $\frac{|C|}{\sum_{u \in C} dist(u,v)}$, where $dist(u, v)$ is the shortest distance from u to v; (ii) *Permanence* [40] measures how permanent is vertex is in its own community, i.e., $Perm(v, C) = \frac{I(v,G[C])}{D(v)E_{max}(v)} - (1 - c_{in}(v, G[C]))$, where $G[C]$ is the induced subgraph of C, $D(v)$ is the total degree of v, $I(v, G[C])$ is the internal degree of v w.r.t. $G[C]$, $E_{max}(v)$ is the maximum external connections to any one of the external communities of v, and $c_{in}(v, G[C])$ is the clustering coefficient of v w.r.t. $G[C]$ (see more in [40]).[4] Fig. 2(a) shows that the performance of EnCoD tends to be always superior with $\mathbb{INV}$ as permanence.
- **Similarity between two vertices** ($\mathbb{SIM}$): Although in Definition 3, we have mentioned to measure the similarity between the feature vectors of two vertices using cosine similarity, we also measure the similarity with *Pearson correlation coefficient* between two feature vectors. Cosine similarity turns out to be the better similarity measure than Pearson correlation coefficient (see Fig. 2(b)).
- **Number of iterations** (K): We set K as a function of the number of vertices N in the network. The analysis in Fig. 2(c) confirms that for most of the networks, especially the ones which possess distinct community structure (such as LFR network with $\mu = 0.1$, Senate), the performance of EnCoD converges at $K = 0.2|V|$. We therefore consider this value as default.
- **Global threshold** (τ_L): τ_L controls the size of the overlapping communities — the size changes inversely with the value of τ_L. We vary τ_L from 0.05 to 0.50 (in steps of 0.05) and observe that the maximum accuracy is attained at $\tau_L = 0.20$ for most of the networks (Fig. 2(d)).

[4] We also checked the performance of the algorithm by simply assigning 1 to the entry of v's feature vector if it belongs to the corresponding community, and 0 otherwise (without measuring its involvement). However, we did not obtain good results.

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.

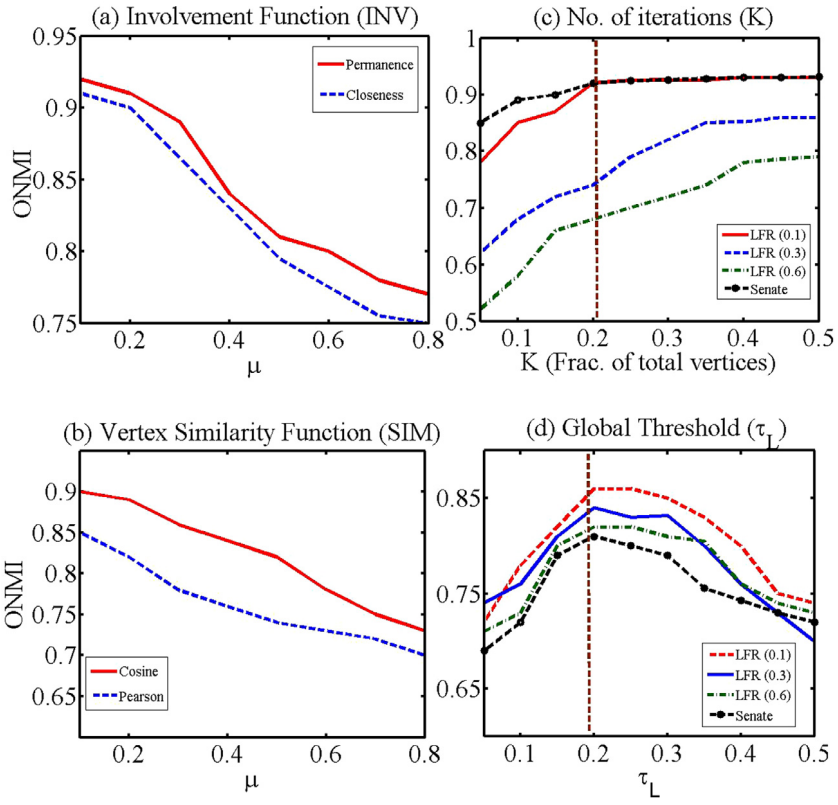

Fig. 2. Change in accuracy of EnCoD with different values of the algorithmic parameters. We vary μ (the mixing parameter of LFR) from 0.1 to 0.8. In (c) and (d), we vary K and τ_L respectively and report the accuracy for three different LFR and Senate networks.

Therefore, unless otherwise stated, we will report the results of EnCoD with the following parameter settings: $\mathbb{INV}$ = *permanence*, $\mathbb{SIM}$ = *cosine*, $K = 0.2|V|$ and $\tau_L = 0.20$.

5.2. Comparative evaluation

Here we perform a two-fold evaluation of EnCoD with the baseline algorithms.

Comparison with state-of-the-art overlapping CD algorithms. We run EnCoD and seven other state-of-the-art overlapping CD algorithms on default LFR network and six real-world networks, and compare the output with the ground-truth community structures in terms of three evaluation metrics. Tables 3, 4, and 5 show the performance of the competing methods in terms of ONMI, Ω Index and F-Score respectively.

Overall, EnCoD shows the best performance, followed by BIGCLAM. The absolute average ONMI of EnCoD for all the networks taken together is 0.82, which is 14.08% higher than BIGCLAM (0.71). The absolute average values of Ω index and F-Score for EnCoD are 0.80 and 0.80 respectively.

5.3. Effect of base algorithms

One can anticipate some noise in the ensemble process due to those base CD algorithms which are generally weak in detecting accurate communities. Therefore we test which base algorithm has the most effect on the final output of EnCoD. To this end, we first consider all base algorithms together and measure the performance of EnCoD. Subsequently, we remove each base algorithm in isolation (i.e., one at a time) and compare the change in accuracy due to the removal. Fig. 3(a) shows that for all the removals, the accuracy decreases significantly. InfoMap (which has been shown to be the best among all the base algorithms [22]) seems to be the one for which the accuracy of EnCoD deteriorates significantly. But Label Propagation and WalkTrap seem to have less effect on the performance of EnCoD.

In another experiment, we remove the base algorithms one after another (not in isolation), and report the performance of EnCoD in Fig. 3(b). Even with only the two 'weakest' base CD algorithms

Table 3
Performance (ONMI) of the competing algorithms w.r.t. the ground-truth community structure of LFR and six real-world networks.

Algorithm	ONMI						
	LFR	Senate	Flickr	Coauth	Youtube	LiveJ	Orkut
OSLOM	0.217	0.798	0.158	0.546	0.410	0.429	0.436
COPRA	0.870	0.688	0.197	0.624	0.393	0.413	0.504
SLPA	0.87	0.579	0.276	0.585	0.639	0.607	0.798
MOSES	0.783	0.512	0.395	0.391	0.369	0.388	0.537
EAGLE	0.174	0.714	0.158	0.195	0.492	0.712	0.714
BIGCLAM	0.791	0.728	0.678	0.654	0.705	0.701	0.741
IEDC	0.771	0.732	0.453	0.398	0.384	0.356	0.587
EnCoD	**0.87**	**0.84**	**0.79**	**0.78**	**0.82**	**0.81**	**0.84**

Table 4
Performance (Ω Index) of the competing algorithms w.r.t. the ground-truth community structure of LFR and six real-world networks.

Algorithm	Ω Index						
	LFR	Senate	Flickr	Coauth	Youtube	LiveJ	Orkut
OSLOM	0.735	0.656	0.10	0.592	0.415	0.395	0.563
COPRA	0.472	0.656	0.15	0.592	0.674	0.641	0.717
SLPA	0.577	**0.82**	0.15	0.592	**0.83**	**0.79**	**0.82**
MOSES	0.525	0.765	0.25	0.691	0.622	0.592	0.563
EAGLE	0.787	0.765	0.20	0.543	0.570	0.543	0.666
BIGCLAM	0.735	0.765	0.667	0.657	0.708	0.706	0.701
IEDC	0.765	0.743	0.565	0.435	0.421	0.452	0.613
EnCoD	**0.84**	**0.82**	**0.75**	**0.79**	**0.83**	**0.79**	**0.82**

No.	Base Algo.	F-Score	Ω-index	ONMI
(1)	All	0.90	0.84	0.87
(2)	(1) \ FastGreedy	0.87	0.83	0.85
(3)	(1) \ Louvain	0.86	0.81	0.84
(4)	(1) \ CNM	0.87	0.82	0.85
(5)	(1) \ InfoMap	0.82	0.80	0.81
(6)	(1) \ WalkTrap	0.83	0.83	0.86
(7)	(1) \ LabelProp	0.89	0.83	0.85

No.	Base Algo.	F-Score	Ω-index	ONMI
(1)	All	0.90	0.84	0.87
(2)	(1) \ FastGreedy	0.87	0.82	0.84
(3)	(2) \ Louvain	0.84	0.81	0.81
(4)	(3) \ CNM	0.83	0.80	0.79
(5)	(4) \ InfoMap	0.76	0.75	0.75
(6)	(5) \ WalkTrap	0.73	0.74	0.73
(7)	(6) \ LabelProp	N.A.	N.A.	N.A.

Fig. 3. Effect of base CD algorithms: performance of EnCoD after (a) removing base algorithms in isolation, i.e., one at a time, (b) removing base algorithms one after another. When all base algorithms are removed, EnCoD does not work (N.A.).

Table 5
Performance (F-Score) of the competing algorithms w.r.t. the ground-truth community structure of LFR and six real-world networks.

Algorithm	F-Score						
	LFR	Senate	Flickr	Coauth	Youtube	LiveJ	Orkut
OSLOM	0.112	0.579	0.387	0.48	0.25	0.259	0.421
COPRA	0.506	0.526	0.72	0.48	0.55	0.570	0.421
SLPA	0.562	0.316	**0.886**	0.64	0.35	0.311	0.632
MOSES	0.562	0.474	0.498	0.48	0.35	0.415	0.474
EAGLE	0.112	0.474	0.498	0.426	0.45	0.466	0.579
BIGCLAM	0.731	0.692	0.626	0.707	0.720	0.754	0.718
IEDC	0.821	0.703	0.598	0.667	0.489	0.549	0.705
EnCoD	**0.90**	**0.79**	0.72	**0.80**	**0.80**	**0.83**	**0.79**

(WalkTrap and Label Propagation), EnCoD achieves more than 80% of its performance when all six base algorithms are used.

In general, these results indicate that combining results from all sorts of (strong or weak) base CD algorithms is important to enhance the final performance of the ensemble method. Importantly, using more base algorithms never decreases the performance of the ensemble method.

Comparison with PVOC and MEDOC. As mentioned earlier, EnCoD is the fourth algorithm after PEACOCK [16], PVOC [8] and MEDOC [9] which use disjoint community structure to detect the overlapping communities.[5] PVOC has been proved to outperform PEACOCK [8]; hence we compare EnCoD with PVOC along with MEDOC and present the results in Table 6. For all the networks, EnCoD performs significantly better than both PVOC and MEDOC. The performance of EnCoD (PVOC, MEDOC) averaged over all the datasets is as follows — ONMI: 0.82 (0.74, 0.75), Ω Index: 0.81 (0.75, 0.70), F-Score: 0.80 (0.68, 0.76).

The superior performance of EnCoD over PVOC and MEDOC is potentially due to the following reason. In a sense, the final performance of both PVOC and MEDOC depends upon the performance of a single CD algorithm. PVOC brings in the overlapping property as a post-processing step after finding a single disjoint community structure, and hence the quality of the overlapping community structure is dependent upon that of the disjoint community structure found initially. MEDOC uses a CD algorithm to partition the multi-partite network that is created using the base disjoint community structures; hence the quality of the final overlapping community structure depends on the performance of the CD algorithm used over the multi-partite network. On the other hand, EnCoD obtains the features of vertices from the disjoint community structures given by several disjoint CD algorithms, and uses these features in an optimization setting. Thus the performance of EnCoD does not depend upon that of any one CD algorithm, and it can effectively learn from multiple disjoint community structures.

[5] Recall that MEDOC is an ensemble algorithm that uses multiple disjoint community structures, whereas PEACOCK and PVOC apply a post-processing on a single disjoint community structure to detect overlapping regions.

6. Community detection with vertex features

In Section 3, one could notice that the solutions of the base algorithms are used to derive only the feature vector of each vertex. This approach is useful for most real networks, where additional information about the vertices is unavailable or difficult to obtain [7]. However, we hypothesize that if vertex-centric features are available along with the network structure, our algorithm can equally be effective to adopt these features. In this case, one needs to replace the ensemble-based features with the real features associated with each vertex (i.e., Step 3 in Algorithm 1 is not required in this case).

To show the generic adaptation of EnCoD for overlapping community detection on networks where vertex-features are available, we set the following experimental framework. We acquire a *coauthorship network*[6] of Computer Science domains, used in [3], where vertices are authors and edges represent collaborations, with at least one paper written by the adjacent authors. There are 24 research areas (such as Algorithms, Data Mining, etc.), and each paper is annotated with one or more research areas. The (overlapping) ground-truth communities are marked by the publication venues (conferences/journals). The network contains 103,677 vertices, 352,183 edges and 1,705 communities.

To obtain vertex features, we assign each author a with a feature vector F_a of size 24 where $F_a(i)$ denotes the fraction of papers published by a in the ith research area. We run all seven competing algorithms (see Section 4) along with modified non-ensemble based EnCoD (we call it MEnCoD) and detect the underlying community structure. We further consider SPAEM [1], a joint probabilistic model to detect communities by combining node attributes and topological structure, as another baseline. Table 7 shows the performance of all the competing algorithms along with EnCoD and MEnCoD. We observe that MEnCoD is superior to all other competing algorithms (followed by EnCoD).

7. Conclusion

We proposed EnCoD, the second ensemble algorithm for overlapping community detection by leveraging the outputs of many disjoint CD algorithms. We examined the dependencies of our algorithm on different parameters. We showed that EnCoD significantly outperforms several standard overlapping CD algorithms, and can be used for detecting the overlapping community structure of networks whose vertices are associated with features.

There is still a lack of proper theoretical explanations to justify the superiority of network-based ensemble approaches. Also, since all base CD algorithms are not equally important in terms of community assignment for vertices (as observed in Section 5.3), it might be possible to retain only the most important base CD

[6] This network is different from the one mentioned in Table 2.

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.

Table 6
Comparison of EnCoD with PVOC and MEDOC on different networks.

Network	PVOC			MEDOC			EnCoD		
	F-score	Ω index	ONMI	F-score	Ω index	ONMI	F-score	Ω index	ONMI
LFR	0.75	0.78	0.79	0.80	0.78	0.82	0.90	0.84	0.87
Senate	0.69	0.73	0.71	0.78	0.80	0.76	0.79	0.82	0.84
Flickr	0.62	0.65	0.67	0.70	0.73	0.71	0.72	0.75	0.79
Coauthorship	0.64	0.68	0.69	0.74	0.71	0.70	0.80	0.79	0.78
Youtube	0.63	0.63	0.67	0.70	0.66	0.64	0.80	0.83	0.82
LiveJournal	0.74	0.72	0.71	0.80	0.76	0.78	0.83	0.79	0.81
Orkut	0.69	0.72	0.75	0.79	0.81	0.80	0.79	0.82	0.84

Table 7
Performance of the competing algorithms on coauthorship network with vertex feature (O = OSLOM, C = COPRA, S = SLPA, M = MOSES, E = EAGLE, B = BIGCLAM, I = IEDC, PV = PVOC, ME = MEDOC, SP = SPAEM).

	O	C	S	M	E	B	I	PV	ME	SP	EnCoD	MEnCoD
F-Sc	0.78	0.76	0.81	0.78	0.75	0.79	0.73	0.74	0.76	0.80	**0.83**	**0.84**
Ω	0.76	0.80	0.83	0.78	0.78	0.76	0.76	0.73	0.74	0.84	**0.86**	**0.87**
ONMI	0.80	0.81	0.83	0.81	0.75	0.79	0.80	0.76	0.79	0.81	**0.86**	**0.88**

algorithms through a correlation study or a machine learning-based approach. This could also potentially reduce the runtime cost of the algorithm. We plan to explore this possibility in future.

Acknowledgments

The authors thank the anonymous reviewers and the editor for their constructive suggestions that helped to improve the paper. The first author would like to acknowledge the support of Ramanujan Faculty Fellowship and the Infosys Center for Artificial Intelligence, IIIT Delhi, India.

References

[1] F. Zhang, Community detection based on links and node features in social networks, in: MMM, Sydney, NSW, Australia, 2015, pp. 418–429.
[2] G. Palla, I.J. Farkas, P. Pollner, I. Derényi, T. Vicsek, Fundamental statistical features and self-similar properties of tagged networks, New J. Phys. 10 (12) (2008) 123026.
[3] T. Chakraborty, S. Sikdar, V. Tammana, N. Ganguly, A. Mukherjee, Computer science fields as ground-truth communities: their impact, rise and fall, in: ASONAM, ACM, Niagara, Canada, 2013, pp. 426–433.
[4] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3–5) (2010) 75–174.
[5] J. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: the state-of-the-art and comparative study, ACM Comput. Surv. 45 (4) (2013) 43:1–43:35.
[6] G. Palla, I. Derenyi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (7043) (2005) 814–818.
[7] J. Yang, J. Leskovec, Overlapping community detection at scale: A nonnegative matrix factorization approach, in: WSDM, Rome, Italy, 2013, pp. 587–596.
[8] T. Chakraborty, Leveraging disjoint communities for detecting overlapping community structure, J. Stat. Mech 2015 (5) (2015) P05017.
[9] T. Chakraborty, N. Park, V.S. Subrahmanian, Ensemble-based algorithms to detect disjoint and overlapping communities in networks, in: ASONAM, San Fransisco, USA, 2016, pp. 73–80.
[10] J. Dahlin, P. Svenson, Ensemble approaches for improving community detection methods, CoRR abs/1309.0242 (2013).
[11] A. Lancichinetti, S. Fortunato, Consensus clustering in complex networks, Nat. Sci. Rep. (2) (2012).
[12] R. Xu, D. Wunsch, II, Survey of clustering algorithms, Trans. Neur. Netw. 16 (3) (2005) 645–678.
[13] T. Chakraborty, Z. Cui, N. Park, Metadata vs. ground-truth: A myth behind the evolution of community detection methods, in: Companion Proceedings of the The Web Conference, 2018, pp. 45–46.
[14] Z. Li, Y. Yang, J. Liu, X. Zhou, H. Lu, Unsupervised feature selection using nonnegative spectral analysis, in: AAAI, 2012, pp. 1026–1032, URL http://dl.acm.org/citation.cfm?id=2900728.2900874.
[15] Z. Li, J. Tang, Weakly supervised deep matrix factorization for social image understanding, Trans. Img. Proc. 26 (1) (2017) 276–288, http://dx.doi.org/10.1109/TIP.2016.2624140, URL https://doi.org/10.1109/TIP.2016.2624140.
[16] S. Gregory, Finding overlapping communities using disjoint community detection algorithms, in: Complex Networks: CompleNet, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 47–61.
[17] A.F. McDaid, D. Greene, N.J. Hurley, Normalized mutual information to evaluate overlapping community finding algorithms,, CoRR abs/1110.2515 (2011).
[18] T. Chakraborty, A. Dalmia, A. Mukherjee, N. Ganguly, Metrics for community analysis: A survey, ACM Comput. Surv. 50 (4) (2017) 54:1–54:37, http://dx.doi.org/10.1145/3091106, URL http://doi.acm.org/10.1145/3091106.
[19] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech 2008 (10) (2008) P10008.
[20] A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 066111.
[21] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.
[22] M. Rosvall, C.T. Bergstrom, Maps of random walks on complex networks reveal community structure, Proc. Natl. Acad. Sci. USA 105 (4) (2008) 1118–1123.
[23] P. Pons, M. Latapy, Computing communities in large networks using random walks, J. Graph Algorithms Appl. 10 (2) (2006) 191–218.
[24] A. Lancichinetti, F. Radicchi, J.J. Ramasco, S. Fortunato, Finding statistically significant communities in networks, PLoS ONE 6 (4) (2011) e18961.
[25] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (3) (2007) 036106.
[26] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (2007) 036106.
[27] D. Chen, M. Shang, Z. Lv, Y. Fu, Detecting overlapping communities of weighted networks via a local algorithm, Physica A 389 (19) (2010) 4177–4187.
[28] F. Havemann, M. Heinz, A. Struck, J. GlÃ₫ser, Identification of overlapping communities and their hierarchy by locally calculating community-changing resolution levels, JSTAT 2011 (01) (2011) P01023.
[29] J. Baumes, M.K. Goldberg, M.S. Krishnamoorthy, M. Magdon-Ismail, N. Preston, Finding communities by clustering a graph into overlapping subgraphs, in: IADIS AC, 2005, pp. 97–104.
[30] F. Ding, Z. Luo, J. Shi, X. Fang, Overlapping community detection by kernel-based fuzzy affinity propagation, in: ISA, 2010, pp. 1–4.
[31] M. Hajiabadi, H. Zare, H. Bobarshad, Iedc: an integrated approach for overlapping and non-overlapping community detection, Knowl.-Based Syst. 123 (2017) 188–199.
[32] W. Zhi-Xiao, L. Ze-chao, D. Xiao-fang, T. Jin-hui, Overlapping community detection based on node location analysis, Knowl.-Based Syst. 105 (2016) 225–235.
[33] Y. Li, K. He, K. Kloster, D. Bindel, J. Hopcroft, Local spectral clustering for overlapping community detection, ACM Trans. Knowl. Discov. Data 12 (2) (2018) 17:1–17:27.
[34] M. Huang, G. Zou, B. Zhang, Y. Liu, Y. Gu, K. Jiang, Overlapping community detection in heterogeneous social networks via the user model, Inform. Sci. 432 (2018) 164–184.
[35] M. Sattari, K. Zamanifar, A spreading activation-based label propagation algorithm for overlapping community detection in dynamic social networks, Data Knowl. Eng. 113 (2018) 155–170.
[36] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (2007) 036106.
[37] M. Ovelgönne, A. Geyer-Schulz, An ensemble learning strategy for graph clustering, in: Graph Partitioning and Graph Clustering, in: Contemporary Mathematics, vol. 588, AMS, 2012, pp. 187–206.
[38] R. Kanawati, Yasca: An ensemble-based approach for community detection in complex networks, in: COCOON, Springer, Cham, 2014, pp. 657–666.

[39] R. Kanawati, Ensemble selection for community detection in complex networks, in: SCSM, Springer, CA, USA, 2015, pp. 138–147.
[40] T. Chakraborty, S. Srinivasan, N. Ganguly, A. Mukherjee, S. Bhowmick, On the permanence of vertices in network communities, in: SIGKDD, New York, USA, pp. 1396–1405.
[41] P. Agarwal, R. Verma, A. Agarwal, T. Chakraborty, DyPerm: maximizing permanence for dynamic community detection, in: Advances in Knowledge Discovery and Data Mining - 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part I, 2018, pp. 437–449.
[42] T. Chakraborty, S. Srinivasan, N. Ganguly, S. Bhowmick, A. Mukherjee, Constant communities in complex networks, Nat. Sci. Rep. 3 (2013).
[43] J. Mcauley, J. Leskovec, Discovering social circles in ego networks, ACM TKDD 8 (1) (2014) 4:1–4:28.
[44] T. Chakraborty, S. Patranabis, P. Goyal, A. Mukherjee, On the formation of circles in co-authorship networks, in: SIGKDD, Sydney, NSW, Australia, 2015, pp. 109–118.
[45] L.G.S. Jeub, P. Balachandran, M.A. Porter, P.J. Mucha, M.W. Mahoney, Think locally, act locally: Detection of small, medium-sized, and large communities in large networks, Phys. Rev. E 91 (2015) 012821.
[46] X. Wang, L. Tang, H. Liu, L. Wang, Learning with multi-resolution overlapping communities, Knowl. Inf. Syst. 36 (2) (2013) 517–535.
[47] A. Lancichinetti, S. Fortunato, Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities, Phys. Rev. E 80 (2009) 016118.
[48] K.C.H. Shen, X. Cheng, M.B. Hu, Detect overlapping and hierarchical community structure in networks, Physica A 388 (8) (2009) 1706–1712.
[49] S. Gregory, Finding overlapping communities in networks by label propagation, New J. Phys. 12 (10) (2010) 103018.
[50] J. Xie, B.K. Szymanski, Towards linear time overlapping community detection in social networks, in: PAKDD, Malaysia, 2012, pp. 25–36.
[51] A. McDaid, N. Hurley, Detecting highly overlapping communities with model-based overlapping seed expansion, in: ASONAM, Odense, Denmark, 2010, pp. 112–119.

Please cite this article in press as: T. Chakraborty, et al., Ensemble-based overlapping community detection using disjoint community structures, Knowledge-Based Systems (2018), https://doi.org/10.1016/j.knosys.2018.08.033.