Ensemble Detection and Analysis of Communities in Complex Networks

TANMOY CHAKRABORTY, IIIT Delhi, India
NOSEONG PARK, George Mason University, USA
AYUSH AGARWAL, IIIT Delhi, India
V. S. SUBRAHMANIAN, Dartmouth College, USA

Though much work has been done on ensemble clustering in data mining, the application of ensemble methods to community detection in networks is in its infancy. In this article, we propose MeDOF, an ensemble method which performs disjoint, overlapping, and fuzzy community detection and represents one of the first ever ensemble methods for fuzzy and overlapping community detection. We run extensive experiments on both synthetic and several real-world datasets for which community structures are known. We show that MeDOF beats the best-known existing stand-alone community detection algorithms. We further show that MeDOF can help explore core-periphery structure of network communities, identify stable communities in dynamic networks, and help solve the "degeneracy of solutions" problem, generating robust results.

CCS Concepts: • **Mathematics of computing → Graph algorithms**; • **Information systems** → *Clustering*;

Additional Key Words and Phrases: Social networks, ensemble models, community detection, core-periphery organization, stable communities

ACM Reference format:
Tanmoy Chakraborty, Noseong Park, Ayush Agarwal, and V. S. Subrahmanian. 2020. Ensemble Detection and Analysis of Communities in Complex Networks. *ACM/IMS Trans. Data Sci.* 1, 1, Article 2 (January 2020), 34 pages.
https://doi.org/10.1145/3313374

1 INTRODUCTION

Community detection, a task of grouping similar nodes together, is an important problem in network science due to its diverse applications in various domains. A few such applications include clustering users together to increase the accuracy of prediction models (link prediction, churn

A preliminary version of the article was published in *The 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (ASONAM) [13].
T. Chakraborty would like to acknowledge the support of the Infosys Center for AI, IIIT Delhi, India, and the Ramanujan Faculty Fellowship (DST) and Early Career Research Award (SERB, DST), SERB, DST, India. V. S. Subrahmanian is supported in part by ONR grants N000141612739 and N00141612918.
Authors' addresses: T. Chakraborty (corresponding author) and A. Agarwal, Dept. of Computer Science & Engg., IIIT Delhi, Okhla Industrial Estate, Phase III, New Delhi, India - 110020; emails: {tanmoy, ayush14029}@iiitd.ac.in; N. Park, 424, Research Hall, George Mason University, 4400 University Dr, Fairfax, VA 22030; email: npark9@gmu.edu; V. S. Subrahmanian, 6211 Sudikoff Lab, Dartmouth College, Hanover, NH 03755; email: vs@dartmouth.edu.

2577-3224/2020/01-ART2 $15.00
https://doi.org/10.1145/3313374

prediction, etc.), estimating unknown features of entities present in a network, detecting networks of fraudulent websites (as such websites tend to link to each other and form communities), and designing recommendation systems in e-commerce applications by clustering customers via their shopping patterns. Though most human beings recognize a community of people when they see one, coming up with a formal mathematical definition of a community has proved challenging, leading to a plethora of diverse technical definitions which capture the intuitive human understanding of a community with different degrees of accuracy [34]. Because of this, different definitions of a community use different objective functions (such as modularity [71], significance [94], permanence [1, 8, 16, 17]), whose optimization leads to the detection of the underlying community structure.

We ask ourselves the question: *can we come up with a computational model of known communities in a network that accurately approximates real-world communities in the network by leveraging the best known existing network community detection algorithms and their associated objective functions?* We are not the first to examine this question; ensemble methods have already been pioneered in network community detection by [23, 58], building on past work on clustering (in non-network settings) in data mining [103].

Apart from these factors, many other factors suggest that an ensemble-based approach may lead to significantly improved accuracy:

- **Dependence on vertex ordering:** Existing community finding algorithms are highly dependent on vertex ordering [15, 38]. If we let an algorithm start from different seed vertices in different iterations for a particular network, it might produce completely different community structures. Each structure can be considered as a different view of what communities might exist.
- **Degeneracy of solution:** Existing community finding algorithms suffer from the "degeneracy of solutions" problem because they admit an exponential number of distinct high-scoring solutions and typically lack a clear global maximum. Each such solution represents a possible view of the underlying community structure and there is no reason to prefer one over another.
- **Lack of ground-truth communities:** Most real-world networks do not have (partial) ground-truth community structure to validate predicted communities. Therefore, it is extremely challenging to do any cross-validation to tune the input parameters of an algorithm.

In this article, we extend our previous study [13] where we suggested how to combine multiple solutions to generate ensemble community detection algorithms. However, it was unclear how one can select base solutions. Moreover, there was no single algorithm which can be able to detect three types of communities: disjoint, overlapping, and fuzzy. In this article, we present a comprehensive experiment to show the superiority of our proposed algorithm, `MeDOF`, along with our previous findings [13]. In particular, the summary of the contributions presented in this article is as follows:

(1) We propose `MeDOF`, a meta-clustering-based algorithm that, to the best of our knowledge, is the first ensemble-based generalized community detection algorithm to detect disjoint, overlapping, and fuzzy communities in a network. The idea behind this algorithm is to generate meta-communities from the "base communities" (i.e., communities generated by existing community detection algorithms) by grouping redundant solutions together. We propose a vertex-to-community association function that provides an accurate estimate of the community membership of different vertices in a network. This association function further allows us to detect overlapping and fuzzy community structures from the network (Section 3).

Table 1. Important Notations Used in This Article

Notation	Description
$G(V, E)$	An undirected network with V as a set of vertices and E as a set of edges
$\mathcal{AL}$	$\{Al_{m=1}^{M}\}$, set of M base disjoint community detection algorithms
K	Number of iterations (number of vertex orderings)
$\mathbb{C}_m^k$	$\{C_m^{1k}, \ldots, C_m^{ak}\}$, base community structure discovered by a base disjoint algorithm Al_m on k^{th} vertex ordering
Γ_m	$\{\mathbb{C}_m^k\}_{k=1}^{K}$, where $\mathbb{C}_m^k$ indicates the base disjoint community structures obtained from algorithm Al_m on K^{th} vertex ordering
Γ	$\Gamma_{m=1}^{M}$, set of all MK base disjoint community structures
$\mathbb{M}$	Ensemble matrix, where $\mathbb{M}(u, v)$ indicates the similarity of vertices u and v
$W(C_i, C_j)$	Matching function between two communities
$\mathcal{F}(v, C)$	A function measuring the association between vertex v and community C
$\mathbb{C}_{GP}$	$\{C_{GP}^l\}_{l=1}^{L}$, meta-communities obtained from P-partite graph GP
$\mathbb{A}$	Association matrix, where $\mathbb{A}(v, l)$ indicates the association of v in meta-community l
$\hat{\mathbb{A}}$	Normalized association matrix, where $\hat{\mathbb{A}}(v, l) = \frac{\mathbb{A}(v,l)}{\sum_{l' \in L} \mathbb{A}(v,l')}$
τ	A threshold needed to detect the overlapping community structure in MeDOF
$\mathbb{OC}$	Final disjoint community structure
$\mathbb{OC}$	Final overlapping community structure
$\mathbb{FC}$	Final fuzzy community structure

(2) We conduct an experimental analysis using both synthetic and real-world networks whose ground-truth community structure is known. Experimental results are shown separately for disjoint (Section 4), overlapping (Section 5), and fuzzy (Section 6) community detection. We show that MeDOF outperforms all state-of-the-art baseline algorithms by a significant margin including the best known and best performing "consensus clustering" algorithm for disjoint community detection [58]. We also present a detailed explanation of how to choose the parameters of MeDOF.

(3) We provide four strategies to choose a subset of "base" community detection algorithms in order to obtain highly accurate communities. These strategies depend on two entities: *quality* and *diversity*. We show that a tradeoff of these two quantities is required to select the best subset of base solutions (Section 7).

(4) MeDOF further allows us to explore the core-periphery structure of communities in a network. We observe that vertices with stronger association within a community form the core unit of the community, whereas peripheral vertices are loosely associated with the community. Furthermore, we show that one can use MeDOF to detect communities in a dynamic time-varying network that are stable, i.e., remain almost invariant over time (Section 8).

(5) Finally, we show that MeDOF significantly reduces the problem of "degeneracy of solutions" in community detection (Section 9) and is much faster than consensus clustering [58], a recently proposed ensemble-based disjoint community finding algorithm (Section 10). Additionally, we remind the readers that consensus clustering does not handle overlapping and fuzzy clustering.

In this article, the major additions of new contributions with our existing work [13] are as follows: (i) We present MeDOF, the first ensemble-based algorithm that can detect disjoint, overlapping,

and fuzzy community structures (Section 3). (ii) We present results of detecting fuzzy community structures from synthetic and real-world networks (Section 6). To the best of our knowledge, this is the first case where a fuzzy community detection algorithm is verified against real-world ground-truth. (iii) We present four strategies to select a subset of base solutions which produce near-optimal results (Section 7). (iv) We present two new implications of the MeDOF algorithm: (a) it can explore the core-periphery structure of communities in a network and (b) it can detect stable communities in dynamic networks (Section 9). (v) We present a detailed analysis to show how our proposed algorithms reduce the effect of "degeneracy of solutions" compared to other baseline algorithms (Section 9).

2 RELATED WORK

In this section, we present the literature pertaining to community detection in three subparts. First, we describe past efforts on disjoint community detection. Second, we discuss fuzzy and overlapping community detection. Finally, we present related work on ensemble-based community detection. As the literature on community detection is abundant, we restrict our discussion to best known and/or most recent works. Extensive survey articles on community detection are available in [34], [10], and [100].

2.1 Disjoint Community Detection

Past efforts devoted to community detection largely assume that nodes are densely connected within a community and sparsely connected across communities. Such efforts include modularity optimization [6, 21, 42, 71, 72], spectral graph-partitioning [73, 85], clique percolation [30, 78], local expansion [4, 59], random-walk-based approaches [26, 81], information-theoretic approaches [87, 88], diffusion-based approaches [83], significance-based approaches [60], and label propagation [83, 101, 102]. Most of these efforts detect communities in static networks. On the other hand, a large number of algorithms were proposed to detect communities in dynamically evolving networks (i.e., Internet, Online Social Networks), such as LabelRankT [99], Estrangement [54], and intrinsically dynamic community detection algorithm [68]. Several pre-processing techniques [3, 86] have been developed to improve the quality of the solutions generated by these algorithm. These methods generate preliminary community structure on a set of selected vertices and then modify the structure over successive steps to cover all the vertices. Recently, [15] presented a analysis of the causal effect of node orders on community detection algorithms.

2.2 Fuzzy and Overlapping Community Detection

Another set of community detection algorithms allow a vertex to be a part of multiple communities. CFinder [78] was the first method of this kind which was established upon the principles of the clique-percolation method. However, since many real-world networks are sparse in nature, CFinder generally produces low quality output [35]. The idea of partitioning links instead of vertices to discover community structure has also been explored [2]. Some algorithms are based on local expansion and optimization such as LFM [59], OSLOM [60], EAGLE [43], MOSES [65], and GCE [61]. [41, 109] proposed a fuzzy community detection technique. BIGCLAM [104] uses the Nonnegative Matrix Factorization (NMF) framework for fuzzy/overlapping community detection. Zhang et al. used NMF to find the overlapping communities given the feature vector of vertices and known number of communities [106].

There is another type of algorithm which exploits the idea of local expansion and optimization to find communities which overlap. For example, the "RankRemoval" algorithm [5] uses a local density function. LFM [59] and MONC [45] maximize a fitness function over successive iterations. OSLOM [60] measures the statistical significance of a cluster w.r.t. a global null model during

community expansion. [18] proposed a combination of "belongingness" and "modified modularity" for local expansion. EAGLE [43] and GCE [61] use an agglomerative pipeline for the detection of community instances that overlap. COCD [28] starts by recognizing the cores following which the rest of the vertices are blended with cores with which they are connected more prominently.

[70] presented community detection as a constrained optimization problem and solved it by using simulated annealing methods. [74, 75, 84, 105] used mixer models to solve this problem. [27] used affinity propagation clustering methods for overlapping community detection. [98] proposed a seed set expansion approach for community detection.

In recent work, the label propagation algorithm has been extended to accommodate the possibility of finding overlapping communities. COPRA [40] updates the "belonging coefficient" of a vertex by taking an average of the coefficients of the neighbors at each timestep. SLPA [101, 102] propagates labels across vertices based on the pairwise interaction rules. [19] proposed a game-theoretic pipeline in which a community is associated with a Nash local equilibrium.

Apart from the above, CONGA [39] uses the GN algorithm [37] to split a vertex into different copies. [108] brought about the idea that one can see this using an iterative method that strengthens the network topology and proximity which is seen as a probability of a pair of vertices that they lie in the same community. [49] introduced a technique which had its roots in the concept of centrality-based influence functions. [92] proposed a fuzzy clustering-based disjoint community detection technique.

2.3 Community Detection using Ensemble Approach

There has been a plethora of research in traditional data mining (not involving networks) to cluster data points using an ensemble approach (see [103] for a detailed review). These approaches can be classified into two categories [103]: object co-occurrence-based approaches and median partitioning-based approaches. But, if we look for instances of clustering vertices into networks, there are few such attempts. [23] proposed an instance-based ensemble clustering method for network data by fusing different community structures. [11, 83] addressed the advantages of combining multiple community structures. CGGC [77] presents a modularity maximization-based ensemble technique. YASCA is another ensemble approach that can detect ego-centered communities [51, 52], and identify the importance of the quality and diversity of base outputs [53].

Recently, two ensemble algorithms have been proposed: [58] proposed "consensus clustering" which leverages a *consensus matrix* for disjoint graph clustering, and [13] proposed `EnDisCo`, an ensemble-based disjoint community detection method that outperforms most state-of-the-art algorithms. However, they are unable to detect overlapping and fuzzy community structure.

Here, we consider consensus clustering and `EnDisCo` as baseline techniques for disjoint community detection and compare them with `MeDOF`. For overlapping and fuzzy community detection, we present the first ever "ensemble-based" algorithm in the literature.

3 MEDOF: META-CLUSTERING APPROACH

`MeDOF` (**Me**ta-Clustering-based **D**isjoint, **O**verlapping, and **F**uzzy Community Detection) begins by running all baseline community detection algorithms. All of these have different order of vertices. At this stage, we have a set of community structures. Next, it creates a multipartite network following which, another community detection algorithm partitions the resulting multipartite network. At the end, a function measuring the vertex-to-community association decides the strength of membership of a vertex in a community. `MeDOF` can yield disjoint, overlapping, and fuzzy community structures from the network.

3.1 Algorithmic Description

MeDOF consists of the following basic steps (Algorithm 1 contains the pseudocode; a toy example of the work-flow of MeDOF is presented in Figure 1, and important notations are shown in Table 1):

(i) Constructing multipartite network. MeDOF takes a set $\mathcal{AL} = \{Al_m\}_{m=1}^{M}$ of M base community detection algorithms as input and runs each Al_m multiple times for K varied orderings of vertices of G. For each ordering k, Al_m produces a community structure $\mathbb{C}_m^k = \{C_m^{1k}, \ldots, C_m^{ak}\}$ of distinct size (Step 1). Each such different run produces its own community structure, thus yielding K community structures $\Gamma_m = \{\mathbb{C}_m^k\}_{k=1}^{K}$. Thus, at the end of Step 1, there are K community structures from each of M algorithms (this implies, the availability of $P = M.K$ structures). We create a P-partite network GP as follows: vertices are constituents of $\bigcup_m \mathbb{C}_m^k$, i.e., each of the communities inside the structure which was procured as a base community in Step 1 ($\mathcal{AL}$ and any vertex ordering) is a vertex of GP. A connecting edge can be drawn from community C_m^{ik} to a community $C_n^{jk'}$ and on top of that a scalar weight can be associated given by $W(C_m^{ik}, C_n^{jk'})$ (Step 1). Section 3.2 will further provide the plausible definition of W. In any partition, the vertices are not connected and this is due to the fact that each $\mathbb{C}_m^k$ is disjoint.

一共M.K个划分结果

ALGORITHM 1: MeDOF: A **Me**ta Clustering based **D**isjoint, **O**verlapping and **F**uzzy Community Detection

Input: Graph $G(V, E)$; Multiple base algorithms $\mathcal{AL} = \{Al_m\}_{m=1}^{M}$; K: Iteration count; $W(.,.)$: check for a match between a pair of communities; *RAlgo*: re-clustering algorithm;
Predicate: $\mathcal{F}(.,.)$: vertex-to-community association;
Parameter: τ: value of threshold to determine overlap in communities
Output: Community candidates that are Disjoint ($\mathbb{DC}$), overlapping ($\mathbb{OC}$) and fuzzy ($\mathbb{FC}$)
// **Step 1:** Making a multipartite network
for Al_m *in* $\mathcal{AL}$ **do**
  Run Al_m on G and get K community structures, represented by the set $\Gamma_m = \{\mathbb{C}_m^k\}_{k=1}^{K}$ corresponding to the runs on K different vertex orderings; each community structure $\mathbb{C}_m^k \in \Gamma_m$ can be of a distinct possible size and can be written as $\mathbb{C}_m^k = \{C_m^{1k}, \ldots, C_i^{ak}\}$;
Make a P-partite graph GP (where $P = M.K$) that contains $M.K$ partitions, each related to the respective community obtained in the last step: partition m^k are communities in $\mathbb{C}_m^k$ and there are edges connecting paired vertices i.e. paired communities C_m^{ik} and C_n^{jk} with the edge weight $W(C_m^{ik}, C_n^{jk'})$;
// **Step 2:** The Multipartite Graph being re-clustered
Here, use the *RAlgo* to cluster the vertices belonging to GP again and obtain a meta-community structure $\mathbb{C}_{GP} = \{C_{GP}^l\}_{l=1}^{L}$;
// **Step 3:** Building such a matrix
Build an association matrix $\mathbb{A}_{|V|\times L}$, where $\mathbb{A}(v, l) = \mathcal{F}(v, C_{GP}^l)$, which provides a pointer to the association of a node v to a meta-community C_{GP}^l;
// **Step 4:** Extracting the structure of the community at the end
Every row in the association matrix $\mathbb{A}$ shows the memberships of the concerned node in L meta-communities;
Allocate a node v to obtained community $C^* = \text{argmax}_C\ \mathbb{A}(v, C)$ to further derive $\mathbb{DC}$;
For the next case, to obtain $\mathbb{OC}$, we allot a node v to some communities C_v^* such that $\forall C \in C_v^* : \mathbb{A}(v, C) \geq \tau$;
To get $\mathbb{FC}$, we first normalize each entry in $\mathbb{A}$ by the sum of entries in the corresponding row and obtain a normalized association matrix $\hat{\mathbb{A}}$, i.e., $\hat{\mathbb{A}}(v, l) = \frac{\mathbb{A}(v,l)}{\sum_{l' \in L} \mathbb{A}(v,l')}$; and assign a vertex v to a community C with the membership probability of $\hat{\mathbb{A}}(v, C)$;
return $\mathbb{DC}$, $\mathbb{OC}$, $\mathbb{FC}$

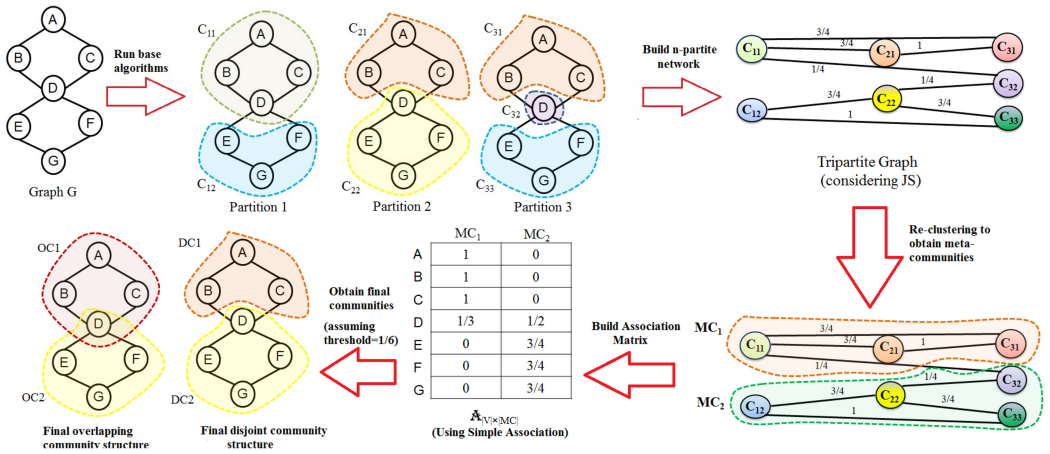


Fig. 1. A toy example depicting the work-flow of the MeDOF algorithm. The broken lines indicate the community boundaries. Assume that the base algorithms produce three different community structures in Step 2, and we use Jaccard Similarity (JC) and simple association ($\mathcal{F}$) as matching and association functions, respectively. The definitions of JC and $\mathcal{F}$ are described in Section 3.2. The threshold τ is chosen as $\frac{1}{6}$.

(ii) Re-clustering the multipartite network. In this step, we use a standard community detection program *RAlgo* on the multipartite network GP to get an output structure consisting of L communities $\mathbb{C}_{GP} = \{C^l_{GP}\}^L_{l=1}$. In this step of the process, one clusters the structures that were carried forward from Step 1; thus, any community C^l_{GP} we get in this step can be considered as a "meta-community" (or community of communities) (Step 2).

The motivation behind re-clustering the multipartite network is to alleviate the shortcomings of the base stand-alone algorithms; two such major shortcomings are "resolution limit" (tendency to combine small communities into a large community) and "degeneracy of solutions" (tendency to produce different results across different iterations for the same network). The multipartite network obtained from different base disjoint community structures allows the re-clustering algorithm not to move the nodes which tend to remain together to different communities, leading to more robust and deterministic results across different iterations (as shown in Section 9). The resolution limit is also tackled as it obtains evidence from multiple base results and merges the nodes accordingly. Therefore, it will allow those nodes which are supposed to to remain together, to form a community. Note that the idea of generating an intermediate network from the original network has already been proved to be effective in the past [15, 58] to address the shortcomings of the existing stand-alone methods.

(iii) Building an association matrix. Calculate the link between a node v and a meta-community C^l_{GP} by using a function $\mathcal{F}$: use this to build the association matrix $\mathbb{A}$ of size $|V| \times L$. Every entity in the matrix is given by $\mathbb{A}(v, l) = \mathcal{F}(v, C^l_{GP})$ (Step 3). The definitions of $\mathcal{F}$ will be provided below in Section 3.2.

(iv) Deriving the community structure at the end. The terminal vertex-to-community connections are made using $\mathbb{A}$. Contents in each row of the matrix A show the membership likelihood of the corresponding vertex in the L communities. Label each vertex in the graph by l, the community which it has the highest probability to lie in according to $\mathbb{A}$, i.e., $l^* = \text{argmax}_l\ \mathbb{A}(v, l)$. In cases of a tie break, the situation is handled by selecting the community which is connected to most of the vertex's direct neighbors. It is important to note that it isn't necessary that every meta-community

has at least one vertex assigned to it. This further implies that we cannot guarantee the existence of L communities in the final structure. In this system, to check for an overlapping community, vertex v is allocated to communities based on whether it satisfies the threshold of having a probability greater than τ. The method of determining this threshold is discussed in Section 3.2.

For fuzzy community detection, we first normalize each entry of $\mathbb{A}(v, l)$ by the sum of entries in the corresponding row so that the membership probability of each vertex in different communities sums up to 1. This in turn returns a new association matrix $\mathbb{A}(\hat{v}, l)$. Accordingly, we assign each vertex v to a community C with the membership probability of $\hat{\mathbb{A}}(v, C)$ (see Step 4).

3.2 Parameter Selection

We now describe the parameters used in the MeDOF algorithm:

— **Matching Function (W):** Suppose we have two communities C_i and C_j. We can determine the similarity/match using the Jaccard Coefficient ($\mathcal{JC}$) $= \frac{|C_i \cap C_j|}{|C_i \cup C_j|}$ and Average Precision (AP) $= \frac{1}{2}(\frac{|C_i \cap C_j|}{|C_i|} + \frac{|C_i \cap C_j|}{|C_j|})$.

Example 3.1. In Figure 1, the Jaccard Coefficient between C_{11} and C_{21} is $JC(C_{11}, C_{21}) = \frac{3}{4}$. The average precision between them is $AP(C_{11}, C_{21}) = \frac{1}{2}(\frac{3}{4} + \frac{3}{3}) = \frac{7}{8}$.

— **Association Function ($\mathcal{F}$):** Suppose we have C, a meta-community that has (say) γ communities; the association entry of v with C can be given by $\mathcal{F}(v, C) = \frac{\sum_{l=1}^{\gamma} \delta(v, C^l)}{\gamma}$. The function δ returns 1 if v is a part of C^l and 0 otherwise. Alternatively, a possible weighted association measure may work on a score which it gives to a vertex v w.r.t. C. This score is dependent on the co-occurrence of other members of the community where v resides, i.e., $\mathcal{F}_w(v, C) = \frac{|\bigcap_{C^l \in C} C^l \delta(v, C^l)|}{|\bigcup_{C^l \in C} C^l \delta(v, C^l)|}$.

Example 3.2. In Figure 1, the simple association between vertex A and meta-community MC_1 is $\mathcal{F}(A, MC_1) = \frac{3}{3} = 1$ because A is present in C_{11}, C_{21}, and C_{31} communities which are parts of MC_1. On the other hand, the weighted association between A and MC_1 is $\mathcal{F}_w(A, MC_1) = \frac{|\{A,B,C,D\} \cap \{A,B,C\} \cap \{A,B,C\}|}{|\{A,B,C,D\} \cup \{A,B,C\} \cup \{A,B,C\}|} = \frac{3}{4}$.

— **Threshold (τ):** The threshold τ was selected in an automatic fashion: In the first step, every vertex is attached to the community it has the greatest chance of being given to which results in a disjoint community structure. For any i, the i'th row in the association matrix $\mathbb{A}$ builds a feature vector $F(v_i)$ which is used to denote the vertex v_i. The average similarity is calculated using the following formula: $AS(C) = \frac{\sum_{(u,v)|u,v \in C \wedge E_{uv} \in E_C} COS(F(u), F(v))}{|E_C|}$, where E_C denotes the collection of edges that are fully inside the community C, E_{uv} is an edge between the vertices u and v, and COS is cosine similarity. The probability of assignment of two vertices to C can then be given by

$$P(C) = \frac{e^{[AS(C)]^2}}{1 + e^{[AS(C)]^2}}. \tag{1}$$

For a given vertex v, if $P(C \cup \{v\}) \geq P(C)$, we make a new connection of v to C apart from the attachment to its present community.

Example 3.3. In Figure 1, let us measure the threshold for community $DC1$ that we obtain in the final step after discovering the disjoint community structure. From the association matrix $\mathbb{A}$ in the figure, $F(A) = \{1, 0\}$, $F(B) = \{1, 0\}$, and $F(C) = \{1, 0\}$. So the average similarity of vertices in $DC1$ in terms of cosine similarity is $AS(DC1) = 1$. Then the probability

of two vertices in $DC1$ being connected is $P(DC1) = 0.88$. If we add D to $DC1$, the new probability $P(DC1 \cup D) = 0.80$ and $P(DC1 \cup D) < P(DC1)$. Therefore, D is not assigned to $DC1$.

It is worth noting that the selection of a threshold depends upon which community we start with. For instance, in Figure 1, if we start from community $DC2$ and calculate the increase in probability after assigning B to it, it would treat this assignment as a valid assignment. If we continue assigning the other vertices, the entire network would get assigned to a single community. To avoid this situation, we start from that community for which the membership probability $P(C)$ is highest and keep assigning other vertices to it. In Figure 1, we start from $DC1$. Once the members in a given community are finalized, we will not perturb their community membership further. This in turn also reduces runtime.

We make comparisons between our method where we select the threshold with the following method: Assignment to highly probable communities—each vertex is marked to the top $n\%$ communities based on probability (n is set to 5% or 10%). Our threshold selection algorithm works better and helps MeDOF provide superior results (see Figure 3(g) and (i)).

— ***RAlgo*: The Re-Clustering Algorithm:** In order to cluster the vertices which are part of the ensemble matrix again, the base community detection algorithms are considered. Our goal is to show that existing community detection algorithms can perform even better when instead of using the adjacency matrix of the graph G, they consider using the ensemble matrix of G. Still, there is scope for using any community detection method in this stage to help determine the structure. Section 4.4.2 contains an analysis of the system with the use of various detection algorithms in the above step.

— **Number of Iterations (K):** To avoid setting a fixed threshold on the iterations, we set K as a variable dependent on V, the number of nodes in the graph. The value of K is varied in the range 0.01–0.50 (with step 0.05) of $|V|$ and it can be said after experimentation that in most cases, the accuracy of the method is the best at $K = 0.2|V|$, which is shown below in Figure 2(c). This is the reason that we assign $K = 0.2|V|$ in our experiments.

3.3 Complexity Analysis

In the above MeDOF algorithm, the costliest step is the building of the multipartite network in Step 1. The worst-case scenario can be described as a situation where every node in one set is linked to every node in another partition when the number of base algorithms is M, the number of vertex orderings is K, and the mean size of the ground community structure is $\bar{a}$. In the above case, the total number of edges is $O(\bar{a}^2M^2K^2)$, which can be simply written as $O(\bar{a}MK)$ edges in practice due to the high sparsity of the network. This occurs due to the fact that in sparse networks $O(|V|) \sim O(|E|)$. The time complexity of building the association matrix is $O(NL)$ iterations (where $L \ll N$). Recall that L is the number of communities obtained from the multipartite network.

4 EXPERIMENTS: OUTCOME OF DISJOINT COMMUNITY DETECTION

In this section, we assess the efficacy of using MeDOF for disjoint community detection. We start by explaining the datasets used in this experiment, followed by the baseline algorithms taken to compare against our algorithms. We also describe in detail the evaluation metrics used to check our detected communities against the gold standard. Then, in the experimental results we will show how we select the parameters and the comparative analysis.

4.1 Datasets

We use both synthetic networks with community structures embedded, and real-world networks of different sizes with known ground-truth community structure.

Table 2. Properties of the Real-World Networks with Disjoint Community Structure

Network	N	E	C	ρ	S
University	81	817	3	0.54	27
Football	115	613	12	0.64	9.66
Railway	301	1,224	21	0.24	13.26
Coauthorship	103,677	352,183	24	0.14	3,762.58

N: # of vertices, E: # of links, C: # of community structures, ρ: average edge-density per community, S: average size of a community.

4.1.1 Synthetic Networks. The data was synthesized with variations in the number of vertices n, average degree $\bar{k}$, maximum degree k_{max}, the mixing parameter μ (a measure to depict the ratio between inter- and intra-community connections), minimum (maximum) community size c_{min} (c_{max}), average percentage O_n of overlapping vertices, and the average number O_m of communities to which a vertex belongs. The LFR benchmark [57] was used to generate the above graphs with the ground-truth community structure. The mixing parameter is used to regulate the quality of the structure. A larger value of μ results in more inter-community edges and decreases the quality of the community structure. We vary this parameter in order to generate different network and community structures. We also vary O_m and O_n to obtain communities in different extents of overlapping (see Section 5). Except for the cases which are specially identified, we create the networks with the same parameter configuration used in [16, 52] for disjoint community structure: $n = 10{,}000, \bar{k} = 50, k_{max} = 150, \mu = 0.3, O_n = 0, O_m = 1, c_{max} = 100$, and $c_{min} = 20$. For every collection of parameters, 50 LFR networks are generated for different experiments and the mean results of all these runs are reported.

4.1.2 Real-World Networks. We also use the following four real-world networks mentioned in Table 2 for experiments.

Football network: This network obtained from [21] consists of regular season American football games between Division IA colleges from Fall 2000. Nodes of the graph here denote the teams in the network (which are college names) while the edges denote games between the two teams which are joined. The analogy of communities here is incorporated through conferences in the division which typically formed with 8–12 teams.

Railway network: This network is proposed in [16]. The graph here is formed by considering a railway station as a node and an available train route which has v_i and v_j as marked stops on the line, as an edge. The states/provinces can be considered communities because the states contain a greater number of trains plying within them than the number of trains between two separate states.

University network: This network generated by [70] is a friendship network of a faculty of a UK university, consisting of 81 vertices (individuals) and 817 connections. The school affiliation of each individual is stored as a vertex attribute. Schools act as communities in this network.

Coauthorship network: Another example of a real-word network is this one which has been introduced by Chakraborty et al. [16]. This network was extracted from a citation dataset [14]. Here each vertex represents an author. There exists an undirected edge between two people if both of them have coauthored at least one paper. Since most collaborations occur when two authors from the same research field come together, the set of research fields are considered as different communities. It may be possible that an author has worked on multiple fields, which causes the communities to overlap. We assign an author to that research community in which he/she has

published the most papers. However, later in Section 6, for fuzzy community detection we will leverage this information to prepare the fuzzy ground-truth community structure.

Note that there are many other publicly available networks such as Dolphin [63], Karate,[1] Polbooks,[2] and Polblogs,[3] which are often used to evaluate a disjoint community detection algorithm [15]. In this article, we did not consider them because they do not have the ground-truth communities marked. When they are used for the purpose of evaluation, the detected communities are evaluated based on intrinsic community scoring metrics, such as modularity, conductance, permanence, and so forth. We wanted to evaluate the competing methods w.r.t. the ground-truth community structure, not by the intrinsic measures.

4.2 Baseline Algorithms

A community structure can be depicted in many different ways via various community detection techniques. In our method, the collection of algorithms given below have been considered as the baseline methods. Here, they are clustered based on the idea they apply to detect communities as per [76]: (i) **Modularity-based approaches**: FastGreedy (`FstGrdy`) [72], Louvain (`Louvain`) [6], and `CNM` [21]; (ii) **Vertex similarity-based approaches**: WalkTrap (`WalkTrap`) [81]; (iii) **Compression-based approaches**: InfoMap (`InfoMap`) [88]; (iv) **Diffusion-based approaches**: Label Propagation (`LabelPr`) [83]; (v) **Ensemble approaches**: The most recent ensemble-based disjoint community detection—Consensus Clustering (`ConsCl`) [58] and `EnDisCo` [13].

A point to note here is that the above baselines have also been used in the ensemble methods in $\mathcal{AL}$ as base algorithms, except consensus clustering and `EnDisCo`.

4.3 Evaluation Metrics

Since we already have the community structure which corresponds to the ground-truth for each network, we make use of two standard measures to evaluate our obtained community structure with the ground-truth: Normalized Mutual Information (NMI) [24] and Adjusted Rand Index (ARI) [46]. The larger the value of NMI and ARI, the better the matching between two community structures.

4.4 Experimental Results

We ran a set of experiments to identify the appropriate parameters for `MeDOF` for disjoint community detection and then compare them with competing algorithms.

4.4.1 Parameter Dependency. We start by experimenting with different values of μ on the LFR Networks. When the `MeDOF` algorithm is run, Figure 2(a) shows that, when compared to the average precision, the Jaccard coefficient clearly has a better performance. When $\mu < 0.6$, the weighted association function is far superior than the other while when $\mu \geq 0.6$, the performance exhibited is nearly the same. This behavior is shown in Figure 2(b). Further, we also change the number of iterations parameter in order to get community samples with distinct vertex orderings. The accuracy levels off at $K = 0.2|V|$ in cases of graphs that have strong community structures embedded within. Examples of such networks include LFR ($\mu = 0.1$), Football and the accuracy trend can be seen in Figure 2(c). However, a bigger value of K is needed to have the accuracy leveling off with increasing μ. The patterns observable for the LFR Network are valid for all other networks as

[1] http://konect.uni-koblenz.de/networks/ucidata-zachary.
[2] http://www.orgnet.com/.
[3] http://networkrepository.com/polblogs.php.

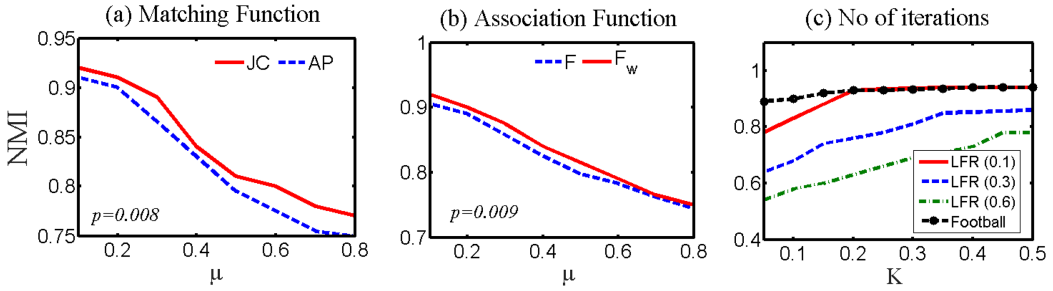


Fig. 2. Dependencies of the performance of MeDOF determined by running it using a grid of parameters. The mixing coefficient which determines the quality of the community is varied between 0.1 and 0.8 (in these runs, we ensure that other parameters of the LFR are not disturbed). The runs are evaluated using NMI measure. The third plot shows the performance accuracy of three different LFR synthetic and Football networks while varying K (the results are similar for the other real networks and are not shown). Figures are reported in such a way that the final value shown is the mean for exhaustive combinations of all the other parameters.

well. And so, unless explicitly mentioned, we will continue to consider the performance of MeDOF experiments with the given parameters for disjoint community detection: $K = 0.2|V|$, JC, F_w.

4.4.2 Impact of Base Community Detection Algorithms. To determine the importance of each baseline community detection algorithm in our ensemble system, we conduct a run of MeDOF with the particular baseline removed from the collective model. We measure the accuracy of each such run and record it in Table 3. We observe that for LFR and real-world networks, InfoMap has the most impact on accuracy based on both the evaluation measures (NMI and ARI) for MeDOF. We also observe that the overall accuracy of the ensemble algorithm decreases after removing each base algorithm. Therefore, irrespective of the quality of the base algorithms, we might need all of them to obtain high-quality community structures as output.

However, the following issues are not clear: (i) whether we need all the outputs obtained from K vertex orderings of a base algorithm and (ii) whether a certain combination of the base algorithms would produce the same accuracy as that obtained from using all the base algorithms. We will discuss these issues further in Section 7.

4.4.3 Impact of Re-Clustering Algorithms on MeDOF. The final step in MeDOF involves a run of a re-clustering algorithm. We also conducted experiments to determine the best re-clustering algorithm. Table 4 shows that for real networks as well as LFR networks, InfoMap is the best re-clustering algorithm.

4.4.4 Comparative Evaluation. Table 5 and Table 6 report the performance of our approach on synthetic and real-world networks, respectively, for the usage of varying algorithms in the last stage of two baseline ensemble algorithms, namely, ConsCl, EnDisCo, and our algorithm MeDOF for synthetic networks. The numbers denote that when that algorithm is used as the final algorithm of the ensemble, there is relative performance improvement in all three algorithms, ConsCl, EnDisCo, and MeDOF. For instance, the accuracy for MeDOF on LFR ($\mu = 0.6$) (when LabelPr is the algorithm used for the re-clustering process in the last stage) averaged over NMI and ARI is 7.82% greater than in a case where stand-alone LabelPr is used. This is shown in the last entry (last row and column) of Table 5. Similarly, MeDOF ties with Infomap on Football network (0% improvement) but beats InfoMap on the other datasets. The full and exact values are present in Table 7. The table

Table 3. The Table Shows the Importance of Each Base Algorithm on MeDOF

(a) LFR Network

No.	Base Algorithm	Disjoint		Overlapping	
		NMI	ARI	ONMI	Ω
(1)	All	0.87	0.90	0.84	0.87
(2)	(1) – FstGrdy	0.84	0.88	0.83	0.85
(3)	(1) – Louvain	0.85	0.86	0.81	0.84
(4)	(1) – CNM	0.83	0.87	0.82	0.85
(5)	(1) – InfoMap	0.81	0.82	0.80	0.81
(6)	(1) – WalkTrap	0.85	0.81	0.83	0.86
(7)	(1) – LabelPr	0.86	0.87	0.83	0.85

(b) Real-World Network

No.	Base Algorithm	Football		Senate	
		NMI	ARI	ONMI	Ω
(1)	All	0.92	0.93	0.81	0.85
(2)	(1) – FstGrdy	0.91	0.90	0.80	0.83
(3)	(1) – Louvain	0.88	0.91	0.78	0.80
(4)	(1) – CNM	0.88	0.91	0.79	0.82
(5)	(1) – InfoMap	0.85	0.83	0.76	0.79
(6)	(1) – WalkTrap	0.89	0.88	0.80	0.81
(7)	(1) – LabelPr	0.91	0.90	0.80	0.81

The values are recorded on default LFR and real networks. All the other parameters have a constant default value. InfoMap has been used as the re-clustering algorithm. To get a fair idea about all the base algorithms, each one of them is dropped one by one when the ensemble matrix for the run is built.

shows that MeDOF always has a better quality of communities detected as compared to any other classical community structure detection algorithm. We further observe that with the decrease in quality of the community structure by increasing the value of μ, there is a greater increase in the improvement brought about by all considered re-clustering algorithms. This indicates that ensemble-based approaches are even more useful when the underlying community structure is not well-separated.

We further compare MeDOF with the performance of the recent DCD_RAM [93] algorithm for disjoint community detection. Table 7 shows the accuracy of the methods. Table 6 presents the same patterns observed for synthetic networks. Once again, MeDOF outperforms both ConsCl and EnDisCo, and the other stand-alone algorithms. For the Football network, our algorithm performs as well as other baseline algorithms, because the underlying community structure is very clear in the Football network [21]. Interestingly, MeDOF exhibits better performance for the Coauthorship network which has weaker community structure [16].

Both the results on synthetic and real-world networks lead to the conclusions that (i) MeDOF is quite competitive w.r.t. the stand-alone algorithms for those networks which have prominent community structure and (ii) MeDOF is more effective than the stand-alone algorithms for those networks whose underlying communities are weakly separated and difficult to detect by any traditional community detection algorithm.

Table 4. The Table Presents the Effect of Re-Clustering Algorithms Used in the Last Step of MeDOF

(a) LFR Network

Re-clustering Algorithm	Disjoint		Overlapping	
	NMI	ARI	ONMI	Ω
FstGrdy	0.80	0.83	0.81	0.84
Louvain	0.83	0.86	0.82	0.83
CNM	0.83	0.86	0.81	0.80
InfoMap	0.87	0.90	0.84	0.87
WalkTrap	0.77	0.82	0.76	0.79
LabelPr	0.78	0.80	0.75	0.77

(b) Real-World Network

Re-clustering Algorithm	Football		Senate	
	NMI	ARI	ONMI	Ω
FstGrdy	0.87	0.91	0.78	0.77
Louvain	0.88	0.89	0.79	0.84
CNM	0.89	0.90	0.80	0.81
InfoMap	0.92	0.93	0.81	0.85
WalkTrap	0.84	0.82	0.76	0.79
LabelPr	0.80	0.82	0.75	0.78

The values are noted on (a) default LFR network and (b) two real-world networks with the standard parameter values of the suggested techniques.

Table 5. Relative Percentage Improvement (Averaged Over NMI and ARI) of ConsCl, EnDisCo, and MeDOF over the Baseline Algorithms for Disjoint Community Detection from Synthetic Networks

Algorithm	Synthetic Network								
	LFR ($\mu = 0.1$)			LFR ($\mu = 0.3$)			LFR ($\mu = 0.6$)		
	ConsCl	EnDisCo	MeDOF	ConsCl	EnDisCo	MeDOF	ConsCl	EnDisCo	MeDOF
FstGrdy	1.92	2.39	2.93	1.96	2.71	3.02	1.90	3.81	3.91
Louvain	1.86	1.97	2.04	1.90	2.22	2.40	1.97	3.41	3.86
CNM	1.98	2.07	2.46	2.03	2.14	2.83	2.01	3.22	3.50
InfoMap	0	0	0	0.98	1.44	1.62	1.62	2.01	2.46
WalkTrap	3.43	4.43	4.97	3.91	4.86	5.08	5.05	6.98	7.42
LabelPr	3.90	5.06	5.72	4.01	5.12	5.39	4.96	7.50	7.82

Each row corresponds to an algorithm *Al* and the value indicates the performance improvement of the ensemble approach with *Al* as the re-clustering algorithm over the isolated performance of *Al* without ensemble.

5 EXPERIMENTS: RESULTS FOR OVERLAPPING COMMUNITY DETECTION

In this section, we evaluate MeDOF for overlapping community detection. We start by explaining the datasets used in this experiment, followed by the baseline algorithms used to compare our method and the evaluation metrics used to match the detected communities with the available gold standard. We then show how to choose the best parameters followed by the comparative evaluation.

Table 6. Relative Percentage Improvement (Averaged Over NMI and ARI) of ConsCl, EnDisCo, and MeDOF Over the Baseline Algorithms for Disjoint Community Detection from Real-World Networks

Algorithm	Real-World Network											
	Football			Railway			University			Coauthorship		
	ConsCl	EnDisCo	MeDOF	ConsCl	EnDisCo	MeDOF	ConsCl	EnDisCo	MeDOF	ConsCl	EnDisCo	MeDOF
FstGrdy	0	0	0	1.01	1.22	1.43	1.92	2.20	2.86	2.23	3.98	4.60
Louvain	0	0	0	0.96	1.17	1.43	1.76	2.12	2.30	1.87	2.21	2.39
CNM	0.84	1.23	1.46	1.01	1.49	1.92	1.54	2.39	2.40	1.32	2.92	3.41
InfoMap	0	0	0	1.10	1.22	1.56	1.76	2.01	2.20	1.98	2.31	2.98
WalkTrap	1.42	2.21	2.46	2.13	3.21	3.49	3.01	4.22	4.49	4.02	5.06	5.51
LabelPr	2.23	3.01	3.29	2.21	3.46	3.79	4.32	6.21	6.80	4.32	6.21	6.98

Each row corresponds to an algorithm *Al* and the value indicates the performance improvement of the ensemble approach with *Al* as the re-clustering algorithm over the isolated performance of *Al* without ensemble.

Table 7. Actual Accuracy Values (in Terms of NMI and ARI) of MeDOF (with Default Parameter Setting), EnDisCo, ConsCl, and DCD_RAM on Both Synthetic and Real-World Networks

Network	ConsCl		EnDisCo		MeDOF		DCD_RAM	
	NMI	ARI	NMI	ARI	NMI	ARI	NMI	ARI
LFR ($\mu = 0.1$)	0.89	0.91	0.93	**0.96**	**0.94**	**0.96**	0.87	0.88
LFR ($\mu = 0.3$)	0.85	0.86	0.89	0.88	**0.90**	**0.91**	0.80	0.82
LFR ($\mu = 0.6$)	0.76	0.79	0.82	0.84	**0.84**	**0.86**	0.78	0.75
Football	0.90	0.92	0.90	0.92	**0.92**	**0.93**	0.90	0.89
Railway	0.71	0.73	0.78	0.80	**0.79**	**0.83**	0.72	0.79
University	0.75	0.79	0.83	0.86	**0.86**	**0.87**	0.76	0.75
Coauthorship	0.61	0.65	0.67	0.68	**0.70**	**0.76**	0.59	0.55

5.1 Datasets

Here, we briefly describe the synthetic and real-world networks that we use in this experiment.

5.1.1 Synthetic Networks. We utilize the synthetic LFR dataset as a standard to build synthetic graphs with an overlapping structure available within them considering the following default parameter settings as mentioned in [40, 60]: $n = 10{,}000$, $\bar{k} = 50$, $k_{max} = 150$, $\mu = 0.3$, $O_n = 20\%$, $O_m = 20$, $c_{max} = 100$, $c_{min} = 20$. We construct 50 different LFR networks considering every configuration of parameters; the results have been reported as per the experiments which have been averaged over these 50 networks. We further vary μ (0.1–0.8 with increments of 0.05), O_m, and O_n (both from 15% to 30% with increments of 1%) based on our experimental requirements.

5.1.2 Real-World Networks. A similar kind of experiment is done using six real datasets noted in Table 8.

Senate network: This network combines the voting patterns of 110 U.S. senators [50, 55]. Each vertex represents a senator and the senators are connected in that session to their three nearest neighbors measured by voting similarities. The ground-truth communities are marked based on the senators who served in the same instance of the senate, i.e., senators who served during the same term.

Flickr: The nodes in this data are the images which have a metadata from Flickr underneath and the edges are drawn between images which have a similar metadata [97]. For instance, metadata similarity could be considered to be the same location from which the image is, or a common

Table 8. Properties of the Real-world Networks with Overlapping Community Structure

Network	N	E	C	ρ	S	O_m
Senate	1,884	16,662	110	0.45	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
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

N: # of vertices, E: # of edges, C: # of communities, ρ: average density of links per community, S: average size of a community, O_m: average # of community memberships per vertex.

gallery/group to which the image has been submitted, or pictures that have been posted by friends, and so forth. Communities are the user-specified groups.

Coauthorship: The coauthorship network [79] here is exactly the same as mentioned before for disjoint community structure in Section 4.1.2, except the ground-truth community marking which in this case is the publication venues (conferences or journals).

LiveJournal: LiveJournal is a free online blogging community where users declare their friends. The website allows users to form groups which their friends can join. The user-defined groups are taken to be the ground-truth communities. [104] provided the LiveJournal friendship social network and ground-truth communities.

Orkut: In this dataset, users in Orkut social networking site are nodes, and links are their friendships. Ground-truth communities are user-defined groups. [104] provided the Orkut friendship social network and ground-truth communities.

5.2 Baseline Algorithms

There are several stand-alone overlapping community detection algorithms, which are different based on the underlying working principle. We take six state-of-the-art algorithms from three different categories mentioned in [100]: (i) **Local expansion:** OSLOM [60] and EAGLE [43]; (ii) **Agent-based dynamical algorithms:** COPRA [40] and SLPA [102], and (iii) **Detection using mixture model:** MOSES [65] and BIGCLAM [104].

5.3 Evaluation Metrics

In order to measure the similarity between the calculated overlapping community structure and the ground-truth, we consider two standard validation metrics: Overlapping Normalized Mutual Information (ONMI)[4] [59, 66] and Omega Index (Ω Index) [22, 69]. The greater the ONMI and Omega index, it can then be said that there is a greater match between the above structures.

5.4 Experimental Results

In this section, we present a detailed description of the comparative evaluation of the competing algorithms for overlapping community detection. We first describe the parameter selection process for `MeDOF`, followed by the results showing the impact of the base algorithms on `MeDOF`. We then present the performance of the competing algorithms.

5.4.1 Parameter Settings. The first step involves determining the most appropriate configuration of the parameters for `MeDOF`. These are matching function W, association function $\mathcal{F}$, number of iterations K, and threshold τ. Figure 3 depicts the outcome using LFR networks by varying μ,

[4] https://github.com/aaronmcdaid/Overlapping-NMI.

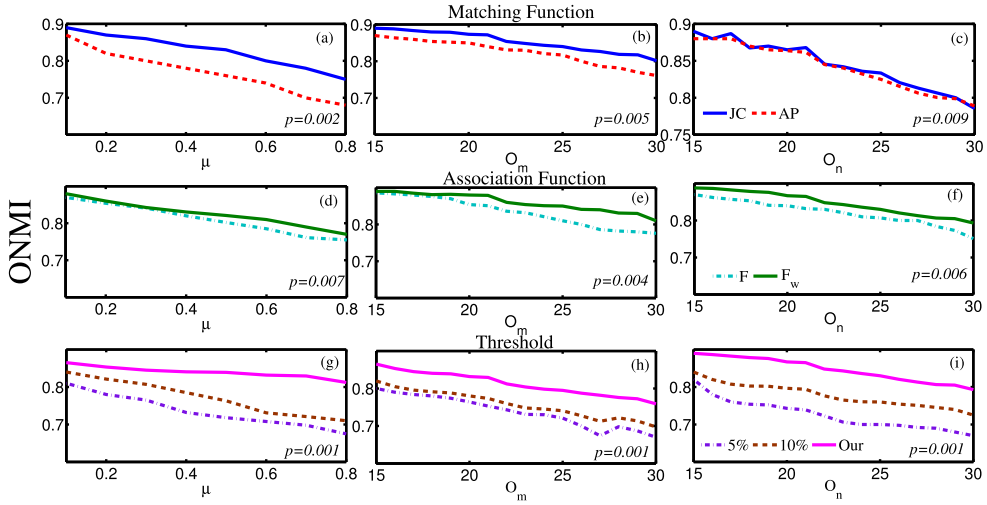


Fig. 3. Dependency of MeDOF on different algorithmic parameters. Three parameters are changed to give the results on overlapping LFRs: μ, O_m, and O_n. In order to perform an effective thresholding, corresponding to each node, the top 5%–10% communities which are probable are chosen. This is compared with our threshold model. The reported figures are statistically significant and the result corresponding to one parameter is reported by taking a mean over all the other possible value combinations for all the other parameters.

O_m, and O_n. The results are almost identical with that of disjoint setting shown in Figure 2. The Jaccard coefficient seems to be the best matching function and in the case of the association function a weighted measure is better. Figure 2(f) shows the choice of K; accuracy almost levels off at $K = 0.2|V|$. Two types of experiments are performed according to the thresholds decided, i.e., the top 5% and 10% highly probable communities for each vertex are considered and compared with the threshold model described in Section 3.2. Figures 3(g) and 3(i) show that even if the network parameter selection is done arbitrarily, our threshold selection choice works better in all cases. It can be observed from Table 4 that the best model for re-clustering is InfoMap. Therefore, unless otherwise stated, in all the remaining experimentation, we run MeDOF with $K = 0.2|V|$, JC, F_w, InfoMap, and τ (which is the value obtained from our method).

5.4.2 Impact of Baseline Algorithms for Overlapping Community Detection. As we saw in the case of disjoint community detection earlier, here also the impact of base algorithms on MeDOF's performance is somewhat similar. As can be seen in Table 3, the results deteriorate the most when we consider leaving InfoMap out from the base algorithm, followed by when we remove Louvain and CNM.

5.4.3 Comparative Evaluation. MeDOF is run with the original configuration on five real-world networks and three synthetic LFR networks. Results from MeDOF are weighed up against the values obtained from the six base overlapping community detection algorithms. Table 9 depicts the results corresponding to contrasted methods with respect to better ONMI and Ω index for synthetic network. As observed, MeDOF performs the best in all cases, with the results indicating a sizable margin compared to the second best. The absolute average of ONMI (Ω) for MeDOF over all synthetic networks is 0.85 (0.87), which is 4.50% (5.66%) higher than BIGCLAM, 6.25% (6.53%) higher than MOSES, 7.14% (8.75%) higher than SLPA, 11.84% (13.97%) higher than COPRA, 12.83% (12.01%) higher than EAGLE, and 12.38% (13.97%) higher than OSLOM. Something else that can be observed is that for synthetic networks, the more the community structure deteriorates with the increase

Table 9. The Below Table Depicts the Accuracy Result for all the Relevant Algorithms Which Detect Overlapping Community Structure in Synthetic Networks

Algorithm	Synthetic Networks					
	LFR ($\mu = 0.1$)		LFR ($\mu = 0.3$)		LFR ($\mu = 0.6$)	
	ONMI	Ω	ONMI	Ω	ONMI	Ω
OSLOM	0.80	0.78	0.74	0.78	0.72	0.73
EAGLE	0.81	0.83	0.75	0.76	0.70	0.74
COPRA	0.80	0.81	0.76	0.74	0.72	0.74
SLPA	0.84	0.86	0.78	0.77	0.76	0.77
MOSES	0.85	0.86	0.80	0.81	0.75	0.78
BIGCLAM	0.86	0.85	0.81	0.83	0.77	0.79
MeDOF	0.88	0.91	0.84	0.87	0.82	0.84

The MeDOF algorithm is run with its default parameter configuration while all the available disjoint algorithms are made use of to generate the multipartite network.

Table 10. The Below Table Depicts the Accuracy Result for all the Relevant Algorithms which Detect Overlapping Community Structure in Real-World Networks

Algorithm	Real-World Networks									
	Senate		Flickr		Coauthorship		LiveJournal		Orkut	
	ONMI	Ω	ONMI	Ω	ONMI	Ω	ONMI	Ω	ONMI	Ω
OSLOM	0.71	0.73	0.68	0.74	0.70	0.71	0.73	0.75	0.71	0.76
EAGLE	0.73	0.74	0.69	0.76	0.71	0.74	0.74	0.76	0.70	0.77
COPRA	0.74	0.77	0.73	0.78	0.75	0.79	0.76	0.82	0.74	0.76
SLPA	0.74	0.76	0.72	0.74	0.76	0.77	0.78	0.85	0.75	0.79
MOSES	0.75	0.78	0.74	0.76	0.79	0.78	0.81	0.82	0.78	0.82
BIGCLAM	0.76	0.79	0.75	0.76	0.80	0.84	0.84	0.87	0.81	0.84
MeDOF	0.81	0.85	0.79	0.84	0.82	0.86	0.86	0.88	0.83	0.86

The MeDOF algorithm is run with its default parameter configuration while all the available disjoint algorithms are made use of to generate the multipartite network.

in μ, the harder it becomes to detect the communities. It is in the "hard to detect" cases that the performance of MeDOF significantly improves compared to the baselines. Additionally, it can be seen that the results become better with the deterioration of community quality. As an example, we look at the good performance of MeDOF when the best baseline algorithm (BIGCLAM) is applied, yielding an improvement of 2.32% (7.06%), 3.70% (4.82%), and 6.49% (6.33%) in ONMI (Ω) terms in the cases where μ ranges between 0.1, 0.3, and 0.6, respectively. This is directly in line with the results seen earlier in Section 4.4.4 that MeDOF is highly effective for those networks where the underlying community structure is not prominent and hard to detect.

In Table 10, we show the performance of the competing algorithms on real-world networks. MeDOF once again outperforms other competing methods. The average absolute ONMI of MeDOF over all networks is 0.82, which is followed by BIGCLAM (0.79), MOSES (0.77), OSLOM (0.76), SLPA (0.75), COPRA (0.74), and EAGLE (0.71). In short, MeDOF performs the best irrespective of the network and validation measure used.

6 EXPERIMENTS: RESULTS OF FUZZY COMMUNITY DETECTION

In the case of overlapping communities, there are two different ways of defining overlap: *crisp overlapping* in which each vertex belongs to one or more communities (the membership is binary—the

idea of membership strength within the community doesn't exist); whereas in the case of *fuzzy overlapping*, a vertex can be present in multiple communities but its connectivity strength to each community may be different. For instance, a person on Facebook might belong to multiple groups, but he may be much more active in one group compared to another. In this case, his degree of membership in that group would be considered to be larger. Detecting the fuzziness of a community structure has its own merit. For example, suppose a telecommunication company is interested to know the name of those customers who are extremely loyal to the company and have a long association with them. At the same time, the company may also want to know those customers who are unstable and often move to other services. If the company is able too measure the membership strength of both these types of customers, they may think of providing a discount/rebate to the former type of customers, and different offers to keep latter type of customers loyal to their network (this is similar to the problem of "churn prediction"). Another application would be to find out researchers who are involved in multiple areas (communities) but cannot be fully involved with all of them due to time and resource constraints. The membership strength obtained from the fuzzy community detection may provide a hint about their association with different communities. Fuzzy community detection is also useful in Bioinformatics research where one may want to know those genes which belong to multiple clusters (genes with similar expression patterns) and the association to each cluster may be useful to decide the "transcription factor" [95].

We earlier observed in Step 4 of MeDOF that it can assign each vertex v to a community C with a membership probability of $\hat{\mathbb{A}}(v, C)$. In this section, we provide a comprehensive analysis of the performance of MeDOF in detecting fuzzy community structure. We start by explaining the datasets used in this experiment, followed by the baseline algorithms and the evaluation metrics. We then describe the results of our comparative experimental evaluation.

6.1 Datasets

We first describe the construction of synthetic networks with fuzzy community structure, followed by the real-world network.

6.1.1 Synthetic Network. The synthetic network generated by the LFR model [57] does not contain the fuzzy community structure. [41] proposed a modified version of the LFR model to generate synthetic fuzzy community structure. Here we adopt their approach to generate synthetic networks. First, we generate crisp overlapping communities from the LFR model. Second, the above generated structures are changed to a fuzzy form by assigning a random membership probability in every case of a present vertex. The membership probabilities are chosen from a uniform distribution. Following this, we attempt to generate a network structure from the fuzzy collection as per the given formula:

$$p_{ij} = s_{ij}p_1 + (1 - s_{ij})p_0, \tag{2}$$

where p_{ij} is the probability of the edge e_{ij} being present, s_{ij} is the co-membership of vertices i and j; and $p_{ij} = p_1 \; if \; \exists c \in C[i \in c \wedge j \in c]; \; else \; p_0$. In the above equation, p_0 and p_1 are chosen in order to maintain the already decided upon average degree ($<k>$) and mixing parameter (μ) in the constructed LFR synthetic network. The network obtained at the end then affirms to all the initial parameters of LFR except the parameter corresponding to the degree distribution (k_{max}), maximum degree, and τ_1, the exponent of the power-law distribution of degrees corresponding to the vertices. However, other parameters of LFR (such as μ, O_m, O_n, etc.) represent the same functionalities in this model. More details can be found in [41].[5] Unless otherwise stated, we generate the

[5]We took the implementation of the synthetic model by the author available at http://www.cs.bris.ac.uk/~steve/networks/.

synthetic networks with the following parameter setting: $n = 10{,}000$, $\bar{k} = 50$, $k_{max} = 150$, $\mu = 0.3$, $O_n = 20\%$, $O_m = 20$, $c_{max} = 100$, $c_{min} = 20$.

6.1.2 Real-World Network. There is no real-world network where fuzzy community memberships of vertices are known. Therefore, the existing fuzzy community detection algorithms were mostly tested either with synthetic networks [41, 96], or by calculating community evaluation metrics such as modularity [91]. Here, we use the metadata information of the coauthorship network mentioned in Section 4.1.2 to construct the ground-truth. We recall that in the coauthorship network, authors are the vertices, edges are drawn based on coauthorship relations, and communities are different research areas. We then assign each author into a community with the community membership indicated by the fraction of papers the author has written on the corresponding research area. It also ensures that the sum of community memberships of each author is 1.

6.2 Baseline Algorithms

Relatively few fuzzy methods have been proposed in the literature thus far. [70] presented "FuzzyClust" that maps the problem to a nonlinear constrained optimization problem and solves it. [106] made use of the fuzzy c-means algorithm to determine community structures after converting the network into the feature space. [82] presented a way that is based on Bayesian NMF. Lastly, FOG [25] clusters "link data," where the networks are considered as a special inclusion into fuzzy communities based on stochastic framework. The algorithm proposed by [41] was the "MakeFuzzy" algorithm, which is used on top of a crisp overlapping algorithm as a possible post-processing method to determine the fuzzy community structure. Apart from this he demonstrated that the given algorithm along with `EAGLE` [43] works better than other available techniques.

In our experiment, we use the FuzzyClust algorithm of [70] and the NMF algorithm of [82] (with default parameters) as two baseline algorithms. We also consider MakFuzzy+`EAGLE` (henceforth, named as "`M-E`") as another baseline to compare with `MeDOF`.

6.3 Evaluation Metric

There exist very few metrics for comparing two fuzzy community structures. To the best of our knowledge, Fuzzy Rand Index (FRI) proposed by [47] is the only metric for this purpose. Since this metric is used infrequently, we briefly explain it here:

$$RI_u(C_1, C_2) = \frac{s(C_1, C_2)}{N}, \tag{3}$$

where $s(C_1, C_2) = N - \sum_{i,j \in V} |f(i, j, C_1) - f(i, j, C_2)|$, and $f(i, j, C_1) = 1$ is vertices v_i and v_j appear in the same community, 0 otherwise. Then the expected Rand Index is defined as $RI_e(C_1, C_2) = \frac{s(C_1)s(C_2)+(N-s(C_1))(N-s(C_2))}{N^2}$, where N is the total number of vertices and $s(C) = \sum_{i,j \in V} f(i, j, C)$.

$f(i, j, C)$ provides a measure of the scope of i and j being present together in a single community (C), that in turn depends on the membership probability of v_i and v_j as follows:

$$f(i, j, C) = 1 - \frac{1}{2} \sum_{c \in C} [\alpha_{ic} - \alpha_{jc}], \tag{4}$$

where α_{ic} is the membership probability of i in community c.

6.4 Experimental Results

We choose the same parameters for `MeDOF` as shown in Figure 3, i.e., $K = 0.2|V|$, JC as pairwise similarly of communities, F_w as weighted association measure, and `InfoMap` as re-clustering algorithm. Further, we consider two setups for `MeDOF`: (i) `MeDOF` is run with the default parameter

Table 11. Accuracy (in Terms of Fuzzy Rand Index) Corresponding to the Available Fuzzy Community Determining Methods for Both LFR and Real-World Structures

Algorithm	Synthetic			Real-world
	LFR ($\mu = 0.1$)	LFR ($\mu = 0.3$)	LFR ($\mu = 0.6$)	Coauthorship
FuzzyClust	0.71	0.68	0.65	0.62
NMF	0.74	0.70	0.68	0.65
M-E	**0.78**	0.74	0.71	0.67
MeDOF	**0.78**	0.75	0.70	**0.68**
MeDOF + MakeFuzzy	**0.78**	**0.76**	**0.73**	**0.68**

Synthetic networks are generated by varying μ and setting other parameters to default values. We run MeDOF in two settings: (i) MeDOF: Algorithm 1 to detect fuzzy communities and (ii) MeDOF+MakeFuzzy: crisp overlapping communities are detected by MeDOF, followed by M-E to post-process the output.

setting to detect the fuzzy community structure and (ii) MeDOF is run to detect the crisp overlapping community first, and then MakeFuzzy is used as a post-processing technique to detect the fuzzing overlapping communities (we call it MeDOF+ MakeFuzzy).

Table 11 presents the results of three baseline algorithms along with two setups of MeDOF. We observe that both the setups of MeDOF tend to be very competitive with the M-E, which seems to be the best baseline algorithm. However, incorporating MakeFuzzy into MeDOF outperforms other competing algorithms with a significant margin.

7 SELECTION OF BASE OUTPUTS

In Section 4.4.2, we observed that removal of each base algorithm from the entire set reduces the overall accuracy of the ensemble algorithms. However, it was not clear (i) whether we need to consider *all K outputs* obtained from running each base algorithm K times and (ii) whether a subset of base algorithms are enough to get similar accuracy. In this section, we address these questions. In particular, we ask a general question: *given a large set of different base solutions, what is the most appropriate combination subset of methods which helps form a concise but more effective ensemble as compared to utilizing all present baselines?*

To this end, we investigate two properties of the detected solutions that have already been identified to be effective in literature [32, 44, 56, 80]: the *quality* and the *diversity*.

7.1 Defining Quality and Diversity

Quality. Since the original communities to which vertices in a network belong are not known *a priori*, it could be prudent to utilize a measure for quality within. Given an ensemble solution E, which is a combination of all base structures $\Gamma = \{\mathbb{C}_m^k\}, \forall m \in M \wedge k \in K$, the following quality function is used to measure the similarity of each solution with the ensemble:

$$Quality\left(\mathbb{C}_m^k, \Gamma\right) = \sum_{m' \in M} \sum_{n=1}^{K} \mathbb{Q}\left(\mathbb{C}_m^k, \mathbb{C}_{m'}^n\right). \tag{5}$$

$\mathbb{Q}$ can be any of the standard evaluation metrics such as NMI, ARI mentioned in Section 4.3 for disjoint communities and ONMI, Omega index in Section 5.3 for overlapping communities. However, we use NMI and ONMI for disjoint and overlapping communities, respectively, as suggested by Strehl and Ghosh [90]. Intuitively, *Quality* gives an indication of how suitable a given base solution is in the context of the pattern available in Γ.

Diversity. There have been many instances in past work when diversity measures were proposed for ensemble models in data mining [62]. However, to make the function consistent with the quality

measure, we consider pairwise NMI/ONMI among the base solutions. In particular, we measure the pairwise similarity of two base solutions as $\mathbb{Q}(\mathbb{C}_m^k, \mathbb{C}_{m'}^{k'})$ and compute the sum of all pairwise similarities $\sum_{i\neq j \wedge i,j\in M \wedge k,k'\in K \wedge \mathbb{C}_i^k,\mathbb{C}_j^{k'}\in\Gamma} \mathbb{Q}(\mathbb{C}_i^k, \mathbb{C}_j^{k'})$. As the value keeps decreasing there is a marked increase in the diversity. This value of diversity is significant as it has previously proved to be an effective measure for cluster ensembles [31].

A point to be taken under close consideration is that there is no limit to any diversity or quality measure in the baseline strategies which we have considered here.

7.2 Selection Strategies

Among the MK solutions obtained from base community detection algorithms, we select S solutions based on the following criteria individually.

7.2.1 Only Quality. Our method immediately ranks solutions on $Quality$ and selects the top S solutions to include in the ensemble algorithm. The solution with the highest $Quality$ essentially indicates that it has high consistency with the rest of the solutions. We expect that if we take only high-quality solutions, due to the high similarity among them this strategy may lead to huge redundancy in the chosen set of solutions. This in turn reduces the ability to obtain improved results for those portions of the data which none of the selected solutions is able to capture properly. This explains the need for diversity amongst the solutions.

7.2.2 Only Diversity. We consider a greedy solution to select S solutions which are highly diverse. The process begins with selecting the method which yields the greatest amount of $Quality$.[6] We then incrementally add one solution at a time such that the resulting ensemble has the highest diversity. This process continues until the required number of solutions are chosen. It is commonly believed that diversifying the base solution is beneficial because mistakes made by one base algorithm may be compensated by another. However, that solutions that are not high quality, may accidentally be added into the ensemble. This is one of the reasons to choose the solution with highest quality first in the greedy strategy.

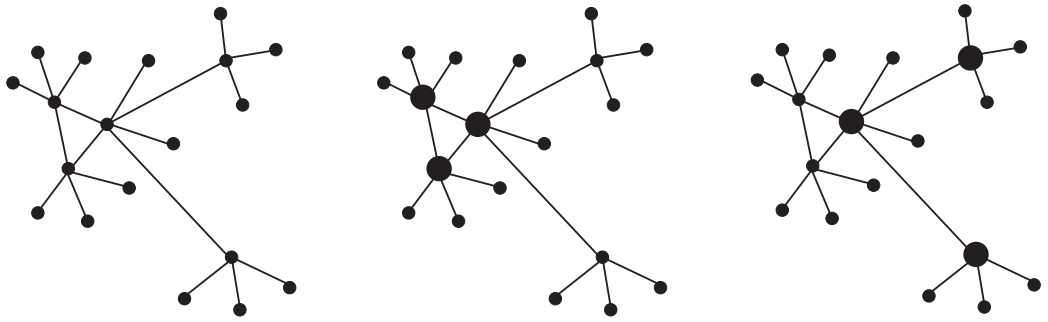
7.2.3 Combining Quality and Diversity. When the base solutions are selected, there is always a tradeoff between diversity and quality. This can essentially be seen as a problem which requires a multi-objective optimization. A traditional way of handling such problems is to combine them into a single aggregate objective function. We choose S such solutions, denoted by $\mathbb{C}_S$, that maximize the following objective function:

$$J = \underbrace{\alpha \sum_{c\in\mathbb{C}_S} Quality(c,\Gamma)}_{\text{Quality}} + \underbrace{(1-\alpha) \sum_{c_i,c_j\in\mathbb{C}_S, c_i\neq c_j,} (1-\mathbb{Q}(c_i,c_j))}_{\text{Diversity}} \tag{6}$$

The parameter α controls the tradeoff between these two quantities. However, it is computationally expensive to select S solutions out of MK solutions [33]. We therefore use a greedy technique. Initially, the method which gives the highest quality is considered and then we keep adding solutions gradually which lead to J being maximized. The default for α is assigned as 0.5. However, we will examine different choices of α in Figure 7.

7.2.4 PageRank-Based Selection. This approach leverages the well-known PageRank algorithm to select the top S solutions which are of high quality but which are as diversely separated in the

[6]However, one can start by selecting the solution which has highest pairwise diversity. However, we observed that it ended up selecting poor solutions in the ensemble, which leads to a decrease in performance.



(a) A hypothetical network (b) Ranking based on PageRank (c) Ranking based on VRRW

Fig. 4. (a) A hypothetical network; (b) top three vertices found using PageRank are highlighted with large circles; and (c) top three nodes found using VRRW are highlighted with large circles. This example is taken from [12].

network as possible. We adopt Vertex Reinforced Random Walk (VRRW), a time heterogeneous random walk process used by [67]. Let us assume that we construct a network, where vertices are the base solutions and the weight of an edge connecting two vertices i and j indicates the pairwise diversity $(1 - \mathbb{Q})$ between corresponding base solutions. Unlike in traditional PageRank where the transition probabilities remain static throughout the iterations, in VRRW they change over time/iteration based on the following equation:

$$p_T(i,j) = (1-\lambda) \cdot p^*(j) + \lambda \cdot \frac{p_0(i,j) \cdot p_T(j)}{D_T(i)}, \tag{7}$$

where

$$D_T(i) = \sum_{k \in V} p_0(i,j) p_T(k). \tag{8}$$

In the above equation, $p_T(i,j)$ is the probability of a link between vertex i and vertex j at time T, $p^*(j)$ is a probability distribution representing the prior for the linkage to j, and $p_0(i,j)$ is the "organic" transition probability prior to any reinforcement. λ is set to 0.9 as suggested by [12]. $p_T(k)$ denotes the probability that the walk is at vertex k at time T: $p_T(k) = \sum_{(n,k) \in E} p_T(n,k) p_{T-1}(n)$. We set $p^*(j) = Quality(C_j, \Gamma)$ and $p_0(i,j)$ as follows:

$$p_0(i,j) = \begin{cases} \alpha \cdot \frac{(1-\mathbb{Q}(\mathbb{C}_i, \mathbb{C}_j))}{Weighted\_deg(\mathbb{C}_i)} & if\ i \neq j \\ 1-\alpha, & otherwise, \end{cases} \tag{9}$$

where $Weighted\_deg(\mathbb{C}_i)$ is the weighted degree of vertex i, representing community structure $\mathbb{C}_i$. A schematic diagram of the VRRW process compared to PageRank is presented in Figure 4.[7] In this strategy, vertices are ranked based on the VRRW score at the stationary state. We then collect the top S highest ranked vertices, which in turn produces S high-quality base communities which are diversely separated in the network.

7.3 Experimental Results

To evaluate the ensemble selection strategies, we apply each strategy on all the datasets separately for disjoint and overlapping community detection. We provide the numbers after making 10 independent runs and averaging the results across these runs. In Figures 5 and 6, we plot the

[7]Readers are encouraged to read the details in [67].

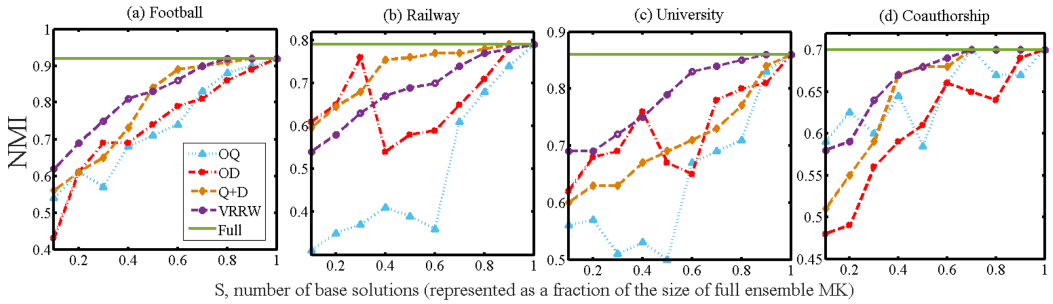


Fig. 5. Comparison of the performance of different selection strategies for disjoint communities: OQ: only quality; OD: only diversity; Q+D: combining quality and diversity with $\alpha = 0.5$; VRRW: vertex reinforced random walk, with the variation of the size of the base solutions for disjoint community detection. The plot contains the comparison of results of these strategies with the full ensemble (Full) represented by the solid green line.

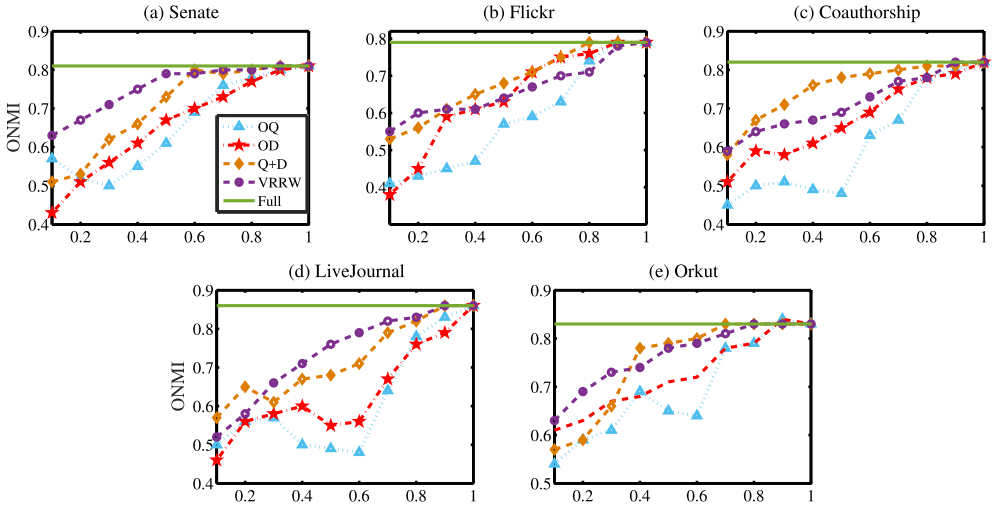


Fig. 6. Comparison of the performance of different selection strategies: OQ: only quality; OD: only diversity; Q+D: combining quality and diversity with $\alpha = 0.5$; VRRW: vertex reinforced random walk, with the variation of the size of the base solutions for overlapping community detection. The plot contains the comparison of results of these strategies with the full ensemble (Full) represented by the solid green line.

performance of different selection strategies as a function of ensemble size S (where S is represented as a fraction of the full ensemble set). We also show the results after considering all the base solutions in the ensemble set using the horizontal green line.

It is evident from both these results that the full ensemble is always best. However, we might achieve the same performance by selecting a subset of the base solutions. Both "combining quality and diversity" and "VRRW" strategies seem to be more consistent and are able to record progress in the direction of our aim, i.e., picking of smaller and more efficient ensembles. Among them, the VRRW-based strategy tends to achieve the maximum accuracy with just 60–80% size of the full ensemble.

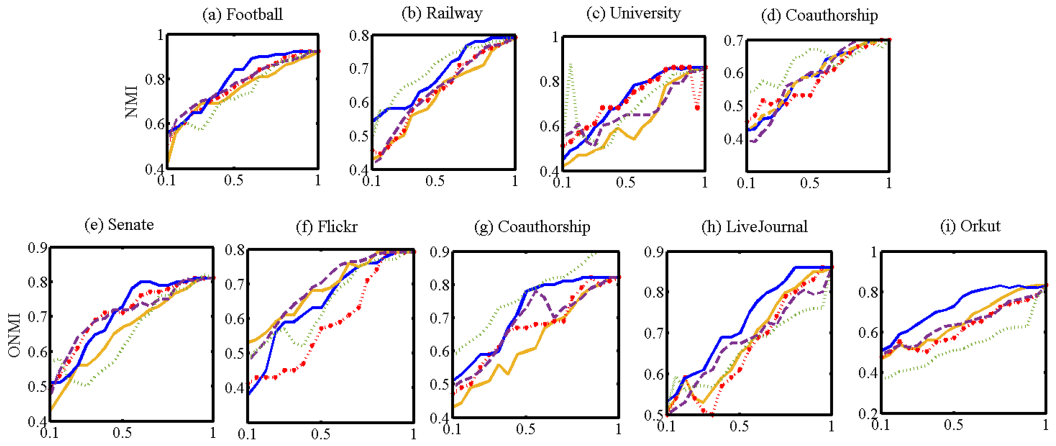


Fig. 7. Sensitivity of the "Combining quality and diversity" strategy by varying the parameter α (0, 0.3, 0.5, 0.7, 1) for different sizes of the selected base solutions (S). S is represented as a fraction of the size of full ensemble (MK), i.e., the size of the base solutions when all base solutions are taken. (a)–(d) Results of MeDOF for disjoint community detection. (e)–(i) Results of MeDOF for overlapping community detection.

Sensitivity of α for "combining quality and diversity" strategy: Thus far, we conducted all the experiments with $\alpha = 0.5$ for "combining quality and diversity". We now examine how this strategy is affected when we vary the value of α. The experiments are done with many different α values which consist of $0, 0.1 \ldots, 0.9, 1$ and the results obtained are compared with "only quality" ($\alpha = 1$) and "only diversity" ($\alpha = 0$). The smaller values of α take "diversity" into account; whereas larger values prefer "quality," In Figure 7, we present the results of EnDisCo and MeDOF for both disjoint and overlapping community detection by varying the size of the selected base solutions and five different values of α $(0, 0.3, 0.5, 0.7, 1)$. Note that each accuracy value shown here is the average of 10 runs. In general, we observe that assigning very high or very low values to α might be beneficial for some cases, but in general the result is more stable and robust for $\alpha = 0.5$. An added observation is that there is never a decline in accuracy with the rise of S and gets saturated quickly for $\alpha = 0.5$.

8 OTHER IMPLICATIONS OF MEDOF

In this section, we present two other useful capabilities of MeDOF. We show that MeDOF can be used to explore the core-periphery structure of communities in a network and to detect stable communities from dynamic networks.

8.1 Exploring Community-Centric Core-Periphery Structure

The association values of individual vertices in a community obtained from MeDOF provide additional information about the membership strength in that community. This phenomenon might be related to the core-periphery organization [7, 107] of the induced subgraph composed of the vertices in a community. We hypothesize that the greater the association of a vertex in a community, the more likely the vertex is to be a core part of the community. To verify this, we first explore the core-periphery structure (also known as k-core decomposition [7]) of each community in a network.

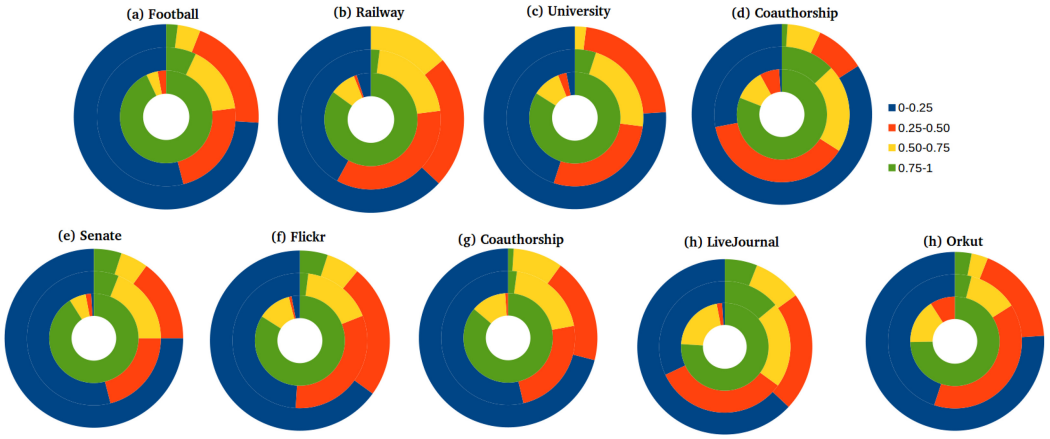


Fig. 8. Core-periphery organization of the disjoint (top panel) and overlapping (bottom panel) communities detected by MeDOF and its relation with the association values. The colors represent different ranges of association values and the space encompassed by each colored region in each k^s-shell shows the fraction of vertices with the association values w.r.t. them occupied in that shell. While the innermost shell is the core region, the outermost layer is the periphery region. In order to aid better visualization, we divide the total number of shells identified from the communities in each network into three broad shells and the range of association values into four buckets; thus the core-periphery structure in each network has three concentric layers and each layer is divided into four association ranges.

Core-periphery organization: For each community detected by MeDOF, we start by creating the induced subgraph composed of vertices in that community. We then recursively remove vertices which have degree one until no such degree-one vertices are left as part of the graph. The removed vertices form the 1-shell of the network (k^s-shell index $k^s = 1$). In the same way, we obtain 2-shell by recursively removing degree-two vertices from the resultant network obtained from the last stage. We keep on increasing the value of k until all the vertices are assigned to at least one shell. Then the k^s-core is formed by taking the union of all the shells with index greater than or equal to k^s.

The idea is to show how the association value of a vertex to a community is related to its community-centric shell index.[8] For each network, we first detect the community using MeDOF and measure the association values of vertices. Then, for each detected community, we extract the core-periphery structure. In Figure 8, we present the correlation between association values of vertices with their positions in the core-periphery structure. For better visualization, we divided the total number of shells into three broad shells, and the range of association value into four buckets. We note that the inner core is highly dominated by the vertices with association value ranging 0.75–1, whereas the outermost layer is dominated by vertices having association value 0–0.25. Further, we observe that as we move from core to periphery, vertices with low association score start dominating. This result indicates that the association value derived from MeDOF has high correlation with the core-periphery origination of the community structure.

8.2 Discovering Stable Communities in Dynamic Networks

Most real-world networks are dynamic; either vertices or edges get added to the network or get deleted over time. As a result, the community structure of the network might change over time.

[8]Note that the average core-periphery structure of communities within a network is different from the core-periphery network structure. Here we are interested in the former case.

Table 12. Similarity of the Stable Communities of Temporal Networks in Two Consecutive Timestamps Obtained from MeDOF for Two Dynamic Networks (Hypertext and School Contact Network)

Network	Measure	$t_1 - t_2$	$t_2 - t_3$	$t_3 - t_4$	$t_4 - t_5$
Hypertext	NMI	0.79	0.76	0.80	0.82
	ARI	0.81	0.78	0.79	0.83
School	NMI	0.76	0.79	0.78	0.80
	ARI	0.75	0.76	0.82	0.83

[20] argued that despite the change in community structure, there are some stable communities that persist. Such stable communities may be important for many reasons. For instance, in an election campaign, vertices in stable communities might be individuals who are strong supporters of a particular candidate. In human interaction networks, the stable communities might be the groups of individuals who are close friends or family members. For instance, there are a number of efforts [29] in which researchers have provided subjects with cell phones (or cell phone apps), and a temporal human interaction network is built by identifying mobile phones that are in close proximity with one another during a given time window. In such cases, communities that are stable between 8am and 6pm might represent people who work together, while communities that are stable during the 8pm to 6am time frame might represent families. We hypothesize that the association values of individual vertices obtained via the MeDOF algorithm can discover stable communities in dynamic networks. In particular, vertices with an association value tending to 1 might form stable communities in a network.

To this end, we collect two dynamic networks with unknown ground-truth community structure[9] [9]: (i) *Hypertext 2009 contact network*: A network which is dynamic and consists of face-to-face proximity samples of more than 100 people who participated at the ACM Hypertext 2009 conference which went on for more than 2.5 days [48]. (ii) *School contact network*: The particular set consists of a contact network between 298 high school students over a time domain of 4 days in a school in Marseilles, France. Every network is subdivided in a manner such that there is an equal division along the temporal dimension and the number of nodes within each small division is same. It is, however, prudent to inform one that the connection of edges may be different for any of the time windows [64].

In every such partition t_i, we run MeDOF on the corresponding network. We allow MeDOF to detect communities with the default parameter setting. Once we obtain the association matrix $\mathbb{A}$ (in Step 4 of Algorithm 1), we considered *only* those vertices whose association score is exactly 1 and group them accordingly (resulting disjoint stable community structure). We believe that these vertices form the stable communities. We repeat the same procedure for each temporal network separately and detect the stable community structure. Following this, we calculate the similarity (dependent on NMI and ARI) between stable community structures that we get from the temporal networks in two consecutive time windows (t_i and t_{i+1}, where i lies between 1 and 4) of each dynamic network. From Table 12, it can be seen that between two successive communities that are stable, there is considerable temporal similarity and it remains nearly constant as time progresses. This indicates that the stable community structure of a network doesn't change much over the successive time periods, and MeDOFmanages satisfactory detection of these stable communities in dynamic networks.

[9] http://www.sociopatterns.org/datasets/.

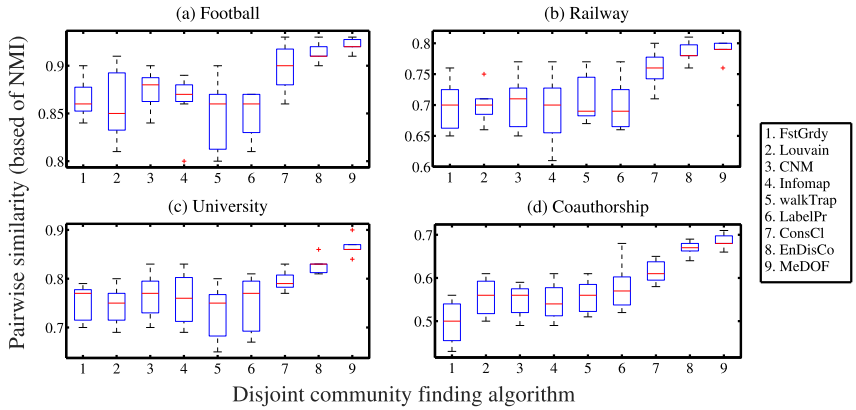


Fig. 9. Box plots which show the difference variation of the procured values from the disjoint community finding algorithms on real-world networks.

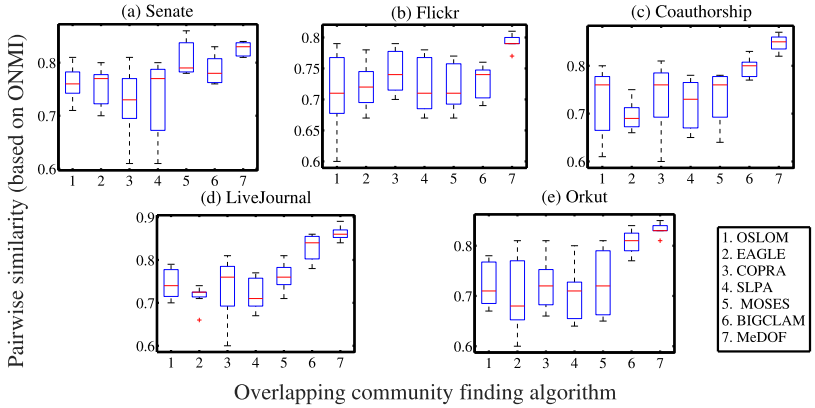


Fig. 10. Box plots which show the difference variation of the procured values from the overlapping community finding algorithms on real-world networks.

9 HANDLING DEGENERACY OF SOLUTIONS

Most community finding algorithms suffer from the well-known "degeneracy of solutions" problem [38] which states that these algorithms typically yield solutions to the order of exponents that have largely alike optimal measure of the objective function (an example for this includes modularity); the point to be noted here is that these obtained solutions are allowed to be structurally different with respect to one another.

We hypothesize that since ensemble algorithms combine multiple views of the underlying community structure of a network, they might suffer less due to the issue of degeneracy of solutions. We test this by considering the real-world networks; we run each competing algorithm by taking 100 distinct orderings of nodes in each network. Following this, we calculate the similarity between each pair of solutions that we get after every algorithm run. The box plots in Figures 9 and 10 show the variation of the solutions for all the competing algorithms on both disjoint and overlapping community detections, respectively. In the case of `MeDOF`, the median similarity that is obtained is high and there is significantly less variation. In fact, all the ensemble algorithms, i.e., `EnDisCo`, `MeDOF`, and `ConsCl`, show low variance. These results suggest that ensemble-based

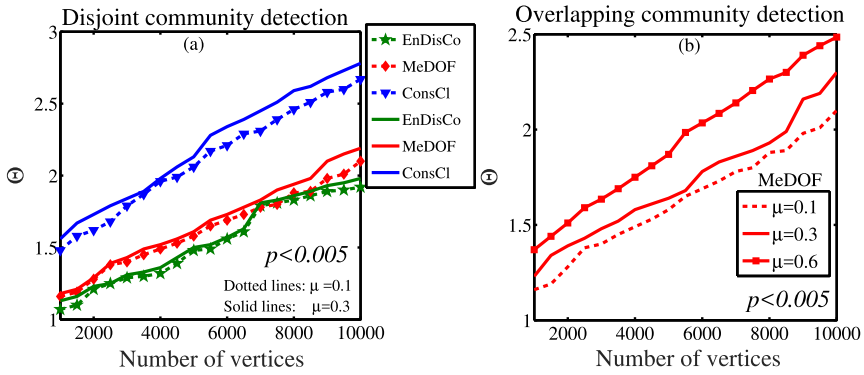


Fig. 11. The figure provides a comparison of MeDOF with EnDisCo and ConsCl. This is done by plotting the value of Θ against the rise in nodes in synthetic LFR. The results are statistically significant (we record the range of p-values with each entry corresponding to one of the many curves).

algorithms always provide more robust results as compared to stand-alone algorithms and help counter the problems which might arise in case of degeneracy.

10 RUNTIME ANALYSIS

Ensembles must run all baseline algorithms either sequentially or in parallel. Hence, an ensemble algorithm is always slow as compared to stand-alone algorithms. However, the ensemble approach suggested by us is faster than existing state-of-the-art ensembles like the consensus clustering and EnDisCo. In order to show this, we report Θ for each ensemble method, that is given by the ratio of the running time corresponding to each ensemble and the sum of the running times of all the baseline algorithms together. We repeat this process for all increasing number of nodes in the synthetic LFR networks. We also alter the quantity of edges in the LFR by adjusting μ (0.1 to 0.3). The runtime of our algorithm is much more faster than the consensus clustering algorithm which is shown in Figure 11. Finally, we provide the results of MeDOF for the problem of overlapping community detection. The process remains the same as in the case of the disjoint structure as it doesn't need any new step except the threshold computation. The runtime of MeDOF on three large real-world datasets—Coauthorship, LiveJournal, and Orkut—is 14,986 sec, 103,456 sec, and 93,765 sec, respectively, on a 2.4GHz machine with 64GB RAM running Ubuntu. Note that while measuring the runtime of MeDOF, we did not consider the time to run the base algorithms as the base solutions can be obtained beforehand and the base algorithms can run in parallel. Rather we only considered the time to perform the ensemble approach.

11 CONCLUSION

In this article, we proposed MeDOF which detects both disjoint, overlapping and fuzzy community structures. MeDOF is the first ever ensemble algorithm for overlapping (and fuzzy) community detection algorithms in the literature.

We managed to provide a thorough analysis by testing our algorithm on both synthetic LFR data as well as with several real-world datasets containing a legitimate community ground-truth structure. We showed that MeDOF is more accurate than existing stand-alone community detection algorithms (though of course, MeDOF leverages them by using the known community detection algorithms in the ensembles) and two other state-of-the-art ensemble algorithms.

We further presented an extensive analysis of how to select a subset of solutions to obtain near-optimal accuracy. The association values of vertices obtained from MeDOF help us to explore the

core-periphery structure of the communities in a network and detect the stable communities in dynamic networks. We showed that ensemble methods generally decrease the impact of degeneracy of solution significantly.

As part of future work, we will attempt to create a theory to show when ensemble methods work more efficiently than baseline approaches. In the traditional ensemble-based classification problem, it has already been shown that an aggregation of several weak base algorithms performs well even though each weak algorithm is just merely better than the random guessing, i.e., the probability of correct classification is $\frac{1}{|C|} + \epsilon$, where ϵ is a small non-zero value and $|C|$ is the number of possible classes [36, 89]. In the case of community detection, one can think of weak base algorithms that find the correct community for a vertex in the probability of $\frac{1}{|C|} + \epsilon$ (boosting condition). After a series of boosting procedures, e.g., boosting by a majority voting, the small ϵ value would help the probability that the community structure is correctly identified converging to a much larger value than $\frac{1}{|C|} + \epsilon$. Of course, there exists a certain probability that the majority vote may not work correctly. If the boosting condition is guaranteed for every vertex and the number of weak algorithms is enough, taking a majority vote may produce stable results. In many practical cases, however, we found that the community detection for a subset of vertices is significantly inaccurate by a majority of weak algorithms, and thus the boosting condition might not be guaranteed for all vertices. We leave this line of research as future agenda.

REFERENCES

[1] Prerna Agarwal, Richa Verma, Ayush Agarwal, and Tanmoy Chakraborty. 2018. DyPerm: Maximizing permanence for dynamic community detection. In *Proceedings of the 22nd Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD'18), Part I.* 437–449. DOI: https://doi.org/10.1007/978-3-319-93034-3_35

[2] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. 2010. Link communities reveal multiscale complexity in networks. *Nature* 466 (2010), 761–764.

[3] David A. Bader, Henning Meyerhenke, Peter Sanders, and Dorothea Wagner (Eds.). 2013. In *Proceedings of the 10th DIMACS Implementation Challenge Workshop on Graph Partitioning and Graph Clustering.* Contemporary Mathematics, Vol. 588. American Mathematical Society.

[4] Jeffrey Baumes, Mark Goldberg, and Malik Magdon-Ismail. 2005. Efficient identification of overlapping communities. In *Proceedings of the 2005 IEEE International Conference on Intelligence and Security Informatics (ISI'05).* Springer-Verlag, Berlin, 27–36.

[5] Jeffrey Baumes, Mark K. Goldberg, Mukkai S. Krishnamoorthy, Malik Magdon-Ismail, and Nathan Preston. 2005. Finding communities by clustering a graph into overlapping subgraphs. In *IADIS AC.* IADIS, 97–104.

[6] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *J. Stat. Mech* (2008), P10008.

[7] Shai Carmi, Shlomo Havlin, Scott Kirkpatrick, Yuval Shavitt, and Eran Shir. 2007. From the cover: A model of internet topology using k-shell decomposition. *Proc. Natl. Acad. Sci. U.S.A.* 104, 27 (2007), 11150–11154. DOI: https://doi.org/10.1073/pnas.0701175104 arXiv:http://www.pnas.org/cgi/reprint/104/27/11150.pdf.

[8] Tanmoy Chakraborty. 2015. Leveraging disjoint communities for detecting overlapping community structure. *J. Stat. Mech.: Theory Exp.* 2015, 5 (2015), P05017. http://stacks.iop.org/1742-5468/2015/i=5/a=P05017

[9] Tanmoy Chakraborty. 2017. On the discovery of invariant groups in complex networks. *J. Complex Networks* 5, 5 (2017), 734–749. DOI: https://doi.org/10.1093/comnet/cnx004

[10] Tanmoy Chakraborty, Ayushi Dalmia, Animesh Mukherjee, and Niloy Ganguly. 2017. Metrics for community analysis: A survey. *ACM Comput. Surv.* 50, 4 (Aug. 2017), Article 54, 37 pages. DOI: https://doi.org/10.1145/3091106

[11] Tanmoy Chakraborty, Saptarshi Ghosh, and Noseong Park. 2019. Ensemble-based overlapping community detection using disjoint community structures. *Knowl.-Based Syst.* 163 (2019), 241–251. DOI: https://doi.org/10.1016/j.knosys.2018.08.033

[12] Tanmoy Chakraborty, Natawar Modani, Ramasuri Narayanam, and Seema Nagar. 2015. DiSCern: A diversified citation recommendation system for scientific queries. In *Proceedings of the 2015 IEEE 31st International Conference on Data Engineering.* 555–566. DOI: https://doi.org/10.1109/ICDE.2015.7113314

[13] Tanmoy Chakraborty, Noseong Park, and V. S. Subrahmanian. 2016. Ensemble-based algorithms to detect disjoint and overlapping communities in networks. In *Proceedings of the 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'16).* 73–80. DOI: https://doi.org/10.1109/ASONAM.2016.7752216

[14] Tanmoy Chakraborty, Sandipan Sikdar, Vihar Tammana, Niloy Ganguly, and Animesh Mukherjee. 2013. Computer science fields as ground-truth communities: Their impact, rise and fall. In *ASONAM*. ACM, 426–433.
[15] Tanmoy Chakraborty, Sriram Srinivasan, Niloy Ganguly, Sanjukta Bhowmick, and Animesh Mukherjee. 2013. Constant communities in complex networks. *Sci. Rep.* 3 (May 2013).
[16] Tanmoy Chakraborty, Sriram Srinivasan, Niloy Ganguly, Animesh Mukherjee, and Sanjukta Bhowmick. 2014. On the permanence of vertices in network communities. In *SIGKDD*. New York, 1396–1405.
[17] Tanmoy Chakraborty, Sriram Srinivasan, Niloy Ganguly, Animesh Mukherjee, and Sanjukta Bhowmick. 2016. Permanence and community structure in complex networks. *ACM Trans. Knowl. Discov. Data* 11, 2 (Nov. 2016), Article 14, 34 pages. DOI: https://doi.org/10.1145/2953883
[18] Duanbing Chen, Mingsheng Shang, Zehua Lv, and Yan Fu. 2010. Detecting overlapping communities of weighted networks via a local algorithm. *Physica A* 389, 19 (2010), 4177–4187.
[19] Wei Chen, Zhenming Liu, Xiaorui Sun, and Yajun Wang. 2010. A game-theoretic framework to identify overlapping communities in social networks. *Data Min. Knowl. Discov.* 21, 2 (Sept. 2010), 224–240.
[20] Cheng Chao Long Ke-zhen Yang De-pin Chen Xiao-qiang, and Zhou Li-hua. 2015. Detecting stable communities in dynamic networks. *J. Chinese Comput. Syst.* 36, 9 (2015), Article 1977, 4 pages. http://xwxt.sict.ac.cn/EN/abstract/article_3050.shtml.
[21] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. 2004. Finding community structure in very large networks. *Phys. Rev. E* 70, 6 (2004), 066111.
[22] L. M. Collins and C. W. Dent. 1988. Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. *Multivar. Behav. Res.* 23, 2 (1988), 231–242.
[23] Johan Dahlin and Pontus Svenson. 2013. Ensemble approaches for improving community detection methods. *CoRR* abs/1309.0242 (2013).
[24] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas. 2005. Comparing community structure identification. *J. Stat. Mech.* 9 (2005), P09008.
[25] George B. Davis and Kathleen M. Carley. 2008. Clearing the FOG: Fuzzy, overlapping groups for social networks. *Soc. Netw.* 30, 3 (2008), 201–212. http://dblp.uni-trier.de/db/journals/socnet/socnet30.html#DavisC08.
[26] Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. 2013. Enhancing community detection using a network weighting strategy. *J. Inf. Sci.* 222 (Feb. 2013), 648–668.
[27] F. Ding, Z. Luo, J. Shi, and X. Fang. 2010. Overlapping community detection by kernel-based fuzzy affinity propagation. In *ISA*. 1–4.
[28] Nan Du, Bai Wang, and Bin Wu. 2008. Overlapping community structure detection in networks. In *CIKM*. ACM, New York, 1371–1372.
[29] N. Eagle, A. Pentland, and D. Lazer. 2007. Inferring social network structure using mobile phone data. *Proc. Natl. Acad. Sci. U.S.A.* (2007).
[30] Illés Farkas, Dániel Ábel, Gergely Palla, and Tamás Vicsek. 2007. Weighted network modules. *New J. Phys.* 9, 6 (2007), 180.
[31] Xiaoli Zhang Fern and Carla E. Brodley. 2003. Random projection for high dimensional data clustering: A cluster ensemble approach. In *ICML*, Tom Fawcett and Nina Mishra (Eds.). AAAI Press, 186–193. http://dblp.uni-trier.de/db/conf/icml/icml2003.html#FernB03a.
[32] Xiaoli Z. Fern and Wei Lin. 2008. Cluster ensemble selection. *Stat. Anal. Data Mining* 1, 3 (2008), 128–141. DOI: https://doi.org/10.1002/sam.10008
[33] Xiaoli Z. Fern and Wei Lin. 2008. Cluster ensemble selection. In *SDM* (2008-06-11). SIAM, 787–797. http://dblp.uni-trier.de/db/conf/sdm/sdm2008.html#FernL08.
[34] Santo Fortunato. 2010. Community detection in graphs. *Phys. Rep.* 486, 3–5 (2010), 75–174. DOI: 10.1016/j.physrep.2009.11.002
[35] Santo Fortunato and Andrea Lancichinetti. 2009. Community detection algorithms: A comparative analysis: Invited presentation, extended abstract. In *Proceedings of the 4th International ICST Conference on Performance Evaluation Methodologies and Tools*. ICST, 27:1–27:2.
[36] Yoav Freund. 1995. Boosting a weak learning algorithm by majority. *Inf. Comput.* 121, 2 (Sept. 1995), 256–285. DOI: https://doi.org/10.1006/inco.1995.1136
[37] M. Girvan and M. E. J. Newman. 2002. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. U.S.A.* 99, 12 (2002), 7821–7826.
[38] B. H. Good, Y. A. De Montjoye, and A. Clauset. 2010. Performance of modularity maximization in practical contexts. *Phys. Rev. E* 81, 4 (2010), 046106.
[39] Steve Gregory. 2007. An algorithm to find overlapping community structure in networks. In *PKDD*. Springer-Verlag, Berlin, 91–102.
[40] Steve Gregory. 2010. Finding overlapping communities in networks by label propagation. *New J. Phys.* 12, 10 (2010), 103018.

[41] Steve Gregory. 2011. Fuzzy overlapping communities in networks. *J. Stat. Mech.: Theory Exp.* 2011, 2 (2011), P02017. http://stacks.iop.org/1742-5468/2011/i=02/a=P02017

[42] Roger Guimera and Luis A. Nunes Amaral. 2005. Functional cartography of complex metabolic networks. *Nature* 433, 7028 (Feb. 2005), 895–900.

[43] K. Cai, H. Shen, X. Cheng, and M. B. Hu. 2009. Detect overlapping and hierarchical community structure in networks. *Phys. A* 388, 8 (2009), 1706–1712.

[44] Stefan T. Hadjitodorov, Ludmila I. Kuncheva, and Ludmila P. Todorova. 2006. Moderate diversity for better cluster ensembles. *Inf. Fusion* 7, 3 (Sept. 2006), 264–275. DOI: https://doi.org/10.1016/j.inffus.2005.01.008

[45] Frank Havemann, Michael Heinz, Alexander Struck, and Jochen Gläser. 2011. Identification of overlapping communities and their hierarchy by locally calculating community-changing resolution levels. *J. Stat. Mech.: Theory Exp.* 2011, 1 (2011), P01023.

[46] L. Hubert and P. Arabie. 1985. Comparing partitions. *J. Class.* 2, 1 (1985), 193–218.

[47] Eyke Hüllermeier, Maria Rifqi, Sascha Henzgen, and Robin Senge. 2012. Comparing fuzzy partitions: A generalization of the Rand index and related measures. *IEEE Trans. Fuzzy Syst.* 20, 3 (2012), 546–556. http://dblp.uni-trier.de/db/journals/tfs/tfs20.html#HullermeierRHS12.

[48] Lorenzo Isella, Juliette Stehlé, Alain Barrat, Ciro Cattuto, Jean-François Pinton, and Wouter Van den Broeck. 2011. What's in a crowd? Analysis of face-to-face behavioral networks. *J. Theor. Biol.* 271, 1 (2011), 166–180.

[49] A. István, Robin Palotai, Máté S. Szalay, and Peter Kovács Csermely. 2010. Community landscapes: An integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics. *PLoS One* 5, 9 (9 2010), e12528.

[50] Lucas G. S. Jeub, Prakash Balachandran, Mason A. Porter, Peter J. Mucha, and Michael W. Mahoney. 2015. Think locally, act locally: Detection of small, medium-sized, and large communities in large networks. *Phys. Rev. E* 91, 1 (2015), 012821.

[51] Rushed Kanawati. 2014. YASCA: A collective intelligence approach for community detection in complex networks. *CoRR* abs/1401.4472 (2014).

[52] Rushed Kanawati. 2014. YASCA: An ensemble-based approach for community detection in complex networks. In *COCOON*. Springer, Cham, 657–666.

[53] Rushed Kanawati. 2015. Ensemble selection for community detection in complex networks. In *SCSM*. Springer, CA, 138–147.

[54] Vikas Kawadia and Sameet Sreenivasan. 2012. Sequential detection of temporal communities by estrangement confinement. *Sci. Rep.* 2 (Nov. 2012).

[55] Kyle Kloster and David F. Gleich. 2015. Personalized pagerank solution paths. *CoRR* abs/1503.00322 (2015). http://arxiv.org/abs/1503.00322

[56] L. I. Kuncheva and S. T. Hadjitodorov. 2004. Using diversity in cluster ensembles. In *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 2. 1214–1219. DOI: https://doi.org/10.1109/ICSMC.2004.1399790

[57] Andrea Lancichinetti and Santo Fortunato. 2009. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Phy. Rev. E* 80 (2009), 016118. 人造网络

[58] Andrea Lancichinetti and Santo Fortunato. 2012. Consensus clustering in complex networks. *Sci. Rep.* 2 (2012).

[59] Andrea Lancichinetti, Santo Fortunato, and János Kertész. 2009. Detecting the overlapping and hierarchical community structure in complex networks. *New J. Phys.* 11, 3 (2009), 033015.

[60] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato. 2011. Finding statistically significant communities in networks. *PLoS One* 6, 4 (2011), e18961.

[61] C. Lee, F. Reid, A. McDaid, and N. Hurley. 2010. Detecting highly-overlapping community structure by greedy clique expansion. In *ACM KDD-SNA*. 33–42.

[62] Jian Li, Ke Yi, and Qin Zhang. 2010. *Clustering with Diversity*. Springer, Berlin, 188–200.

[63] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M. Dawson. 2003. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behav. Ecol. Sociobiol.* 54, 4 (2003), 396–405.

[64] Rossana Mastrandrea, Julie Fournet, and Alain Barrat. 2015. Contact patterns in a high school: A comparison between data collected using wearable sensors, contact diaries and friendship surveys. *PLoS One* 10, 9 (2015), e0136497.

[65] Aaron McDaid and Neil Hurley. 2010. Detecting highly overlapping communities with model-based overlapping seed expansion. In *ASONAM*. Washington, DC, 112–119.

[66] Aaron F. McDaid, Derek Greene, and Neil J. Hurley. 2011. Normalized mutual information to evaluate overlapping community finding algorithms. *CoRR* abs/1110.2515 (2011).

[67] Qiaozhu Mei, Jian Guo, and Dragomir Radev. 2010. DivRank: The interplay of prestige and diversity in information networks. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*. ACM, New York, 1009–1018. DOI: https://doi.org/10.1145/1835804.1835931

[68] Bivas Mitra, Lionel Tabourier, and Camille Roth. 2011. Intrinsically dynamic network communities. *CoRR* abs/1111.2018 (2011).
[69] Gabriel Murray, Giuseppe Carenini, and Raymond Ng. 2012. Using the omega index for evaluating abstractive community detection. In *Proceedings of the Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*. Association for Computational Linguistics, Stroudsburg, PA, 10–18. http://dl.acm.org/citation.cfm?id=2391258.2391260
[70] Tamás Nepusz, Andrea Petróczi, László Négyessy, and Fülöp Bazsó. 2008. Fuzzy communities and the concept of bridgeness in complex networks. *Phys. Rev. E* 77, 1 (Jan. 2008), 016107. https://doi.org/10.1103/PhysRevE.77.016107
[71] M. E. Newman. 2006. Modularity and community structure in networks. *Proc. Natl. Acad. Sci. U.S.A.* 103, 23 (2006), 8577–8582.
[72] M. E. J. Newman. 2004. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* 69, 6 (June 2004), 066133.
[73] M. E. J. Newman. 2013. Community detection and graph partitioning. *CoRR* abs/1305.4974 (2013).
[74] M. E. J. Newman and E. A. Leicht. 2007. Mixture models and exploratory analysis in networks. *Proc. Natl. Acad. Sci. U.S.A.* 104, 23 (2007), 9564–9569.
[75] K. Nowicki and T. A. B. Snijders. 2001. Estimation and prediction for stochastic blockstructures. *J. Amer. Statist. Assoc.* 96, 455 (Sept. 2001), 1077–1087.
[76] Günce Keziban Orman, Vincent Labatut, and Hocine Cherifi. 2012. Comparative evaluation of community detection algorithms: A topological approach. *J. Stat. Mech.: Theory Exp.* 2012, 8 (2012), P08001. http://stacks.iop.org/1742-5468/2012/i=08/a=P08001.
[77] Michael Ovelgönne and Andreas Geyer-Schulz. 2012. An ensemble learning strategy for graph clustering. In *Graph Partitioning and Graph Clustering (Contemporary Mathematics)*, Vol. 588. American Mathematical Society, 187–206.
[78] Gergely Palla, Imré Derényi, Illés Farkas, and Tamás Vicsek. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 7043 (June 2005), 814–818.
[79] Gergely Palla, Illés J. Farkas, Péter Pollner, Imre Derényi, and Tamás Vicsek. 2008. Fundamental statistical features and self-similar properties of tagged networks. *New J. Phys.* 10, 12 (2008), 123026.
[80] Lance Parsons, Ehtesham Haque, and Huan Liu. 2004. Subspace clustering for high dimensional data: A review. *SIGKDD Explor. Newsl.* 6, 1 (June 2004), 90–105. DOI: https://doi.org/10.1145/1007730.1007731
[81] Pascal Pons and Matthieu Latapy. 2006. Computing communities in large networks using random walks. *J. Graph Algorithms Appl.* 10, 2 (2006), 191–218.
[82] Ioannis Psorakis, Stephen Roberts, and Ben Sheldon. 2010. Efficient Bayesian community detection using non-negative matrix factorisation. (2010). arXiv:arXiv/1009.2646
[83] Usha N. Raghavan, Réka Albert, and Soundar Kumara. 2007. Near linear time algorithm to detect community structures in large-scale networks. *Phys. Rev. E* 76, 3 (Sept. 2007), 036106.
[84] Wei Ren, Guiying Yan, Xiaoping Liao, and Lan Xiao. 2009. Simple probabilistic algorithm for detecting community structure. *Phys. Rev. E* 79, 3 (2009), 036111.
[85] Thomas Richardson, Peter J. Mucha, and Mason A. Porter. 2009. Spectral tripartitioning of networks. *Phys. Rev. E* 40 (2009), 027104.
[86] Jason Riedy, David A. Bader, Karl Jiang, Pushkar Pande, and Richa Sharma. 2011. *Detecting Communities from Given Seeds in Social Networks*. Technical Report GT-CSE-11-01. Georgia Institute of Technology.
[87] M. Rosvall and C. T. Bergstrom. 2007. An information-theoretic framework for resolving community structure in complex networks. *Proc. Natl. Acad. Sci. U.S.A.* 104, 18 (2007), 7327.
[88] Martin Rosvall and Carl T. Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. *Proc. Natl. Acad. Sci. U.S.A.* 105, 4 (2008), 1118–1123.
[89] Robert E. Schapire. 1990. The strength of weak learnability. *Mach. Learn.* 5, 2 (July 1990), 197–227. DOI: https://doi.org/10.1023/A:1022648800760
[90] Alexander Strehl and Joydeep Ghosh. 2003. Cluster ensembles—A knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* 3 (March 2003), 583–617. DOI: https://doi.org/10.1162/153244303321897735
[91] J. Su and T. C. Havens. 2014. Fuzzy community detection in social networks using a genetic algortihm. In *Proceedings of the 2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'14)*. 2039–2046.
[92] Peng-Gang Sun, Lin Gao, and Shan Shan Han. 2011. Identification of overlapping and non-overlapping community structure by fuzzy clustering in complex networks. *Inf. Sci.* 181, 6 (2011), 1060–1071.
[93] Kamal Taha. 2018. Disjoint community detection in networks based on the relative association of members. *IEEE Trans. Comput. Soc. Syst.* 5, 2 (2018), 493–507.
[94] Vincent A. Traag, Gautier Krings, and Paul Van Dooren. 2013. Significant scales in community structure. *Sci. Rep.* 3 (2013). http://dblp.uni-trier.de/db/journals/corr/corr1306.html#TraagKD13.

[95] Faramarz Valafar. 2002. Pattern recognition techniques in microarray data analysis. *Ann. N. Y. Acad. Sci.* 980, 1 (2002), 41–64.
[96] Corinna Vehlow, Thomas Reinhardt, and Daniel Weiskopf. 2013. Visualizing fuzzy overlapping communities in networks. *IEEE Trans. Vis. Comput. Graphics* 19, 12 (2013), 2486–2495. DOI: https://doi.org/10.1109/TVCG.2013.232
[97] Xufei Wang, Lei Tang, Huan Liu, and Lei Wang. 2013. Learning with multi-resolution overlapping communities. *Knowl. Inf. Syst.* 36, 2 (2013), 517–535.
[98] Joyce Jiyoung Whang, David F. Gleich, and Inderjit S. Dhillon. 2013. Overlapping community detection using seed set expansion. In *CIKM*. ACM, 2099–2108.
[99] Jierui Xie, Mingming Chen, and Boleslaw K. Szymanski. 2013. LabelRankT: Incremental community detection in dynamic networks via label propagation. *CoRR* abs/1305.2006 (2013).
[100] Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. 2013. Overlapping community detection in networks: The state-of-the-art and comparative study. *ACM Comput. Surv.* 45, 4 (Aug. 2013), Article 43, 35 pages. DOI: https://doi.org/10.1145/2501654.2501657
[101] J. Xie and B. K. Szymanski. 2011. Community detection using a neighborhood strength driven label propagation algorithm. In *IEEE NSW*. 188–195.
[102] Jierui Xie and Boleslaw K. Szymanski. 2012. Towards linear time overlapping community detection in social networks. In *PAKDD*. Malaysia, 25–36.
[103] Rui Xu and D. Wunsch, II. 2005. Survey of clustering algorithms. *Trans. Neur. Netw.* 16, 3 (May 2005), 645–678.
[104] Jaewon Yang and Jure Leskovec. 2013. Overlapping community detection at scale: A nonnegative matrix factorization approach. In *WSDM*. ACM, New York, 587–596.
[105] Mina Zarei, Dena Izadi, and Keivan Aghababaei Samani. 2009. Detecting overlapping community structure of networks based on vertex–vertex correlations. *J. Stat. Mech. Theor. Exp.* 2009, 11 (2009), P11013.
[106] Shihua Zhang, Rui-Sheng Wang, and Xiang-Sun Zhang. 2007. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. *Phys. A: Stat. Mech. Appl.* 374, 1 (2007), 483–490.
[107] Xiao Zhang, Travis Martin, and M. E. J. Newman. 2015. Identification of core-periphery structure in networks. *Phys. Rev. E* 91, 3 (Mar. 2015), 032803. DOI: https://doi.org/10.1103/PhysRevE.91.032803
[108] Yuzhou Zhang, Jianyong Wang, Yi Wang, and Lizhu Zhou. 2009. Parallel community detection on large networks with propinquity dynamics. In *KDD*. ACM, 997–1006.
[109] Kun Zhao, Shao-Wu Zhang, and Quan Pan. 2010. Fuzzy analysis for overlapping community structure of complex network. In *CCDC*. 3976–3981.

Received May 2018; revised September 2018; accepted December 2018