

Local community detection in multilayer networks

Roberto Interdonato¹ · Andrea Tagarelli¹  · Dino Ienco² ·
Arnaud Sallaberry³ · Pascal Poncelet⁴

Received: 7 October 2016 / Accepted: 22 June 2017 / Published online: 6 July 2017
© The Author(s) 2017

Abstract The problem of local community detection in graphs refers to the identification of a community that is specific to a query node and relies on limited information about the network structure. Existing approaches for this problem are defined to work in dynamic network scenarios, however they are not designed to deal with complex real-world networks, in which multiple types of connectivity might be considered. In this work, we fill this gap in the literature by introducing the first framework for local community detection in multilayer networks (ML-LCD). We formalize the ML-LCD optimization problem and provide three definitions of the associated objective function, which correspond to different ways to incorporate within-layer and across-layer topological features. We also exploit our framework to generate multilayer global com-

Responsible editor: Kurt Driessens, Dragi Kocev, Marko Robnik-Šikonja Myra Spiliopoulou.

✉ Andrea Tagarelli
andrea.tagarelli@unical.it

Roberto Interdonato
rinterdonato@dimes.unical.it

Dino Ienco
dino.ienco@irstea.fr

Arnaud Sallaberry
arnaud.sallaberry@lirmm.fr

Pascal Poncelet
pascal.poncelet@lirmm.fr

¹ DIMES - University of Calabria, Rende, CS, Italy

² IRSTEA, UMR TETIS, Montpellier, France

³ LIRMM, Université Paul Valéry, Montpellier, France

⁴ LIRMM, Université de Montpellier, Montpellier, France

munity structures. We conduct an extensive experimentation using seven real-world multilayer networks, which also includes comparison with state-of-the-art methods for single-layer local community detection and for multilayer global community detection. Results show the significance of our proposed methods in discovering local communities over multiple layers, and also highlight their ability in producing global community structures that are better in modularity than those produced by native global community detection approaches.

Keywords Node-centric community detection · Personalized community detection · Complex systems · Multiplex networks

1 Introduction

Community detection is a classic problem in network science and related fields, which has been traditionally addressed with the aim of determining an organization of a given network into subgraphs that express dense groups of nodes well-connected to each other (Newman and Girvan 2004). This corresponds to an optimization problem that is global as it requires knowledge on the *whole* network structure. The problem is known to be computationally difficult to solve, while its approximate solutions have to cope with both accuracy and efficiency issues that become more severe as the network increases in size. Large-scale, web-based environments have indeed traditionally represented a natural scenario for the development and testing of effective community detection approaches. Further challenges correspond to the emergence of processing complex real-world network systems, which are indeed pervasive in many fields of science (Mucha et al. 2010; Carchiolo et al. 2010; Tang et al. 2012; Kivela et al. 2014; Loe and Jensen 2015; Kim and Lee 2015). In this regard, *multilayer network* models provide a powerful and more realistic tool for the analysis of such complex systems, enabling an in-depth understanding of the characteristics and dynamics of multiple, interconnected types of node relations and interactions (Cai et al. 2005; Berlingerio et al. 2013; Dickison et al. 2016).

However, one important aspect to consider is that we might often want to identify the personalized network of social contacts of interest to a single user only: to this aim, we would seek to determine the expanded neighborhood of that user which forms a densely connected, relatively small subgraph. This is known as *local*, or *node-centric*, community detection problem (Clauset 2005; Chen et al. 2009), whose general objective is, given limited information about the network, to identify a community structure which is centered on one or few seed users. The development of methods that can identify query-dependent local communities is beneficial for any scenario in which computing a global community structure is not feasible (e.g., because the whole network information is not available at processing time), or it is not required (e.g., the communities are to be computed only for a subset of target users). As an intuitive practical example, if we want to simply check whether two users in a network belong to the same community, a global community detection method would process the whole network, while a local approach can efficiently discover the two communities of interest by accessing and manipulating only a relatively small portion of the net-

work. This reduction of memory-footprint requirements by local community detection methods also enables an efficient processing of *dynamic networks*, whose structure may change over time (e.g., the insertion of new nodes and edges can be handled at each step of exploration); yet, local approaches can be useful to cope with *privacy* and *access restriction* issues that typically arise from policies adopted in most online social networks (e.g., limitation in the number of queries per day, permission to extract only the ego network of a limited number of seed nodes, and so on).

In the last few years, we have been witnessing an increasing interest towards the local community detection problem (Chen et al. 2009; Branting 2012; Fagnan et al. 2014; Zakrzewska and Bader 2015; Li et al. 2015). Surprisingly, the problem has been mainly investigated by focusing on networks that are built on a single node-relation type or context. However, this is not the case in many situations. For instance, in social computing, an individual often has multiple accounts across different social networks, and in fact it has nowadays become important to link distributed user profiles belonging to the same user from multiple platforms (Kim and Lee 2015; Loe and Jensen 2015). An alternative scenario is obtained by considering that relations of different types can be available for the same population of a social network (Dickison et al. 2016); these might include online as well as offline (i.e., real-life) relations, such as followship, like/comment interactions, working relationship, lunch relationship, etc. Both scenarios can effectively be represented using a *multilayer network model*. Dealing with multiple graph-relation dimensions for a set of entities makes the previously discussed emergence of using a local community detection approach, as well as the reduction of memory requirements, even more evident and important. In general, several questions may arise, such as:

- *How can we profitably use the various relations in which an individual is involved to discover her/his own community?*
- *How do the different relations affect the size and form of a multilayer local community being discovered?*
- *What advantages does the development of a method for multilayer local community detection may bring with respect to single-layer local community detection as well as to multilayer global community detection?*

Contributions. In this work we aim to answer the above questions, by contributing a framework for the novel problem of local community detection in a multilayer network. To the best of our knowledge, we are the first to bring the local community detection problem into the context of multilayer networks since all previous works address the multilayer community detection task from a global point of view (e.g., (Mucha et al. 2010; Tang et al. 2012; Papalexakis et al. 2013; Kuncheva and Montana 2015; Kim and Lee 2015; Loe and Jensen 2015)). More in detail, we summarize our contributions as follows.

- We introduce and formalize the problem of local community detection for multilayer networks (ML-LCD), following an unsupervised paradigm.
- We provide three definitions of the objective function in the ML-LCD problem, which correspond to different ways to incorporate within-layer and across-layer topological features.

- To enable comparative evaluation with global community detection methods in multilayer networks, we exploit our ML-LCD framework to generate multilayer global community structures.
- We present an extensive experimentation of ML-LCD, using seven real-world multilayer networks. We assess the meaningfulness of our methods and analyze their different behaviors in identifying multilayer local communities. Moreover, we have conducted two stages of comparative evaluation with state-of-the-art methods for (single-layer) local community detection as well as for global community detection in multilayer networks; remarkably, in the latter case, ML-LCD has shown to produce communities with comparable or even better multilayer modularity.

Remarkably, the proposed ML-LCD methods are designed to provide community solutions that can meet several criteria of significance and quality. More specifically, under the classification framework discussed in the survey work on multilayer community detection methods by [Kim and Lee \(2015\)](#), our methods meet all (but one, i.e., algorithm insensitivity) of the desired *properties* for multilayer community detection methods, namely: multiple layer applicability; consideration of each layer's importance (this is in particular embedded in the first of our methods, but it can in principle be brought to the other methods as well); flexible layer participation (every local community has in general a different coverage of the layers' structure); no-layer-locality assumption (our local communities do not depend on initializations steps biased by a particular layer); independence from the order of layers; and overlapping layers (two or more local communities can share substructures over different layers).

The rest of the paper is organized as follows. Sect. 2 overviews related work, Sect. 3 describes our proposed local community detection methods for multilayer networks, Sects. 4 and 5 present experimental evaluation, Sect. 6 concludes the paper and provides pointers for future research.

2 Related work

We organize a brief discussion on related work into two parts: the first is devoted to community detection in multilayer networks, the second concerns local community detection methods.

Community detection in multilayer networks. Identifying a community structure in multilayer networks is a research topic that has gained considerable attention in the last few years ([Kim and Lee 2015](#); [Loe and Jensen 2015](#); [Kivela et al. 2014](#)). Early studies have focused on adaptations of the notion of modularity ([Newman and Girvan 2004](#)) to multilayer networks ([Mucha et al. 2010](#); [Carchiolo et al. 2010](#)). Alternatively, new evaluation criteria have been designed specifically for multilayer networks (e.g., redundancy ([Berlingerio et al. 2011](#))).

[Tang et al. \(2009\)](#) utilize structural feature extraction and cross-dimension integration to find a concise representation of features from the various layers (dimensions). A basic k -means clustering method is applied on the computed embedding to produce a community structure over the multilayer graph. In a subsequent work of the same authors ([Tang et al. 2012](#)), a utility integration criterion is introduced for computing utility matrices of a community detection method for each layer separately. Then it

optimizes an objective function for the aggregated multilayer utility matrix. [Kuncheva and Montana \(2015\)](#) propose a community detection method for multilayer networks based on a multilayer random walk model. It first runs a different random walk for each layer, then a dissimilarity measure between nodes is obtained leveraging the per-layer transition probabilities, finally a hierarchical clustering method is used to produce the communities. The approach proposed by [Hmimida and Kanawati \(2015\)](#) leverages the concept of leaders in a network ([Yakoubi and Kanawati 2014](#)), i.e., nodes having higher degree centrality than most of their direct neighbors. These nodes, which first need to be identified over the whole network, form their own communities. An iterative local preference-merging method is then performed to compute and update the membership-preference vector of each node in function of preferences of its neighbors, so that each node will be assigned to the community defined by the leader ranked first in its preference vector. Other works have resorted to representation models such as third-order tensors ([Papalexakis et al. 2013](#)), hypergraphs ([Michoel and Nachtergaele 2012](#)), or transactional representation to support frequent pattern mining ([Berlingerio et al. 2013](#)).

Note that, by definition, all the above approaches address the problem of community detection from the conventional, global perspective, i.e., they assume to access the entire network structure in order to produce a partitioning of the network graph into a set of communities.

Local community detection. One of the earliest contribution to local community detection is the Clauset's framework ([Clauset 2005](#)), which is designed to explore the graph through local expansion starting from a seed node. [Chen et al. \(2009\)](#) exploit the Clauset framework to determine the quality of a community by comparing its internal versus external connectivity. [Branting \(2012\)](#) compares different local community detection methods, which are organized into two broad categories, namely xenophobic and non-xenophobic approaches: the former try to maximize (resp. minimize) the internal (resp. external) connectivity, while the non-xenophobic algorithms discard the external connectivity. [Fagnan et al. \(2014\)](#) propose a local community strategy that accounts for the number of internal and external triads. While still exploiting the Clauset framework, it finally produces a non-overlapping global community structure. The approach presented by [Zakrzewska and Bader \(2015\)](#) employs a seed set expansion procedure that incrementally updates the community as the underlying graph changes. [Kanawati \(2015\)](#) evaluates the impact of applying different ways of combining multiple local community functions to identify the node-centric communities. The Lemon algorithm proposed by [Li et al. \(2015\)](#) exploits truncated random walks and approximate invariant subspace to discover a local community for any given seed set. The number of random walk steps is a key parameter, which should be set high enough to reach all the nodes in the target community, and at the same time low enough to not spread to an unnecessary bigger graph.

It should be emphasized that none of the above works is designed to deal with multilayer networks. Different mention concerns the work by [Jeub et al. \(2015\)](#), where the solution of a personalized PageRank is approximated for a local partition of the multilayer network in order to find communities. However, the approach assumes complete knowledge about the network structure, and the local perspective is intended as the way the random walk is personalized, which differs from identifying local communi-

ties using multilayer features. In this work, we aim to fill this gap in the literature by contributing the first framework for multilayer local community detection.

3 Multilayer local community detection

In this section, we first describe the graph model used to represent a multilayer network. We formally state the Multilayer Local Community Detection (ML-LCD) problem. Then, we formulate the multilayer local community functions and describe the algorithmic framework for identifying a local community centered around an input seed node. We finally discuss computational complexity aspects of the proposed ML-LCD methods.

3.1 Multilayer network model

We refer to the multilayer network model described in (Kivela et al. 2014). Let $\mathcal{L} = \{L_1, \dots, L_\ell\}$ be a set of layers. Each layer corresponds to a given type of entity relation, or edge-label. Consider a set \mathcal{V} of entities (e.g., users), then for each choice of entity in \mathcal{V} and layer in \mathcal{L} , we need to indicate whether the entity is present in that layer. We denote with $V_{\mathcal{L}} \subseteq \mathcal{V} \times \mathcal{L}$ the set containing the entity-layer combinations in which an entity is present in the corresponding layer. The set $E_{\mathcal{L}} \subseteq V_{\mathcal{L}} \times V_{\mathcal{L}}$ contains the undirected links between such entity-layer pairs. We hence denote with $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ the *multilayer network* graph with set of nodes \mathcal{V} .

For every layer $L_i \in \mathcal{L}$, let $V_{L_i} = \{v \in \mathcal{V} \mid (v, L_i) \in V_{\mathcal{L}}\} \subseteq \mathcal{V}$ be the set of nodes in the graph of L_i , and $E_{L_i} \subseteq V_{L_i} \times V_{L_i}$ be the set of edges in L_i . To simplify notations, we will also refer to V_{L_i} and E_{L_i} as V_i and E_i , respectively. Note that while entities (i.e., elements of \mathcal{V}) are not required to participate to all layers, however each entity has to appear in at least one layer, i.e., $\bigcup_{i \in 1 \dots \ell} V_{L_i} = \mathcal{V}$. Moreover, the only inter-layer edges are regarded as “couplings” of nodes representing the same entity between different layers; in other terms, $E_{\mathcal{L}}$ can be seen as partitioned into the set of intra-layer edges and the set of inter-layer coupling edges, i.e., $E_{\mathcal{L}} = \{(u, L_i), (v, L_j) \mid u, v \in \mathcal{V} \wedge L_i, L_j \in \mathcal{L} \wedge i = j\} \cup \{(v, L_i), (v, L_j) \mid v \in \mathcal{V} \wedge L_i, L_j \in \mathcal{L} \wedge i \neq j\}$.

3.2 Problem statement

Local community detection approaches generally implement some strategy that at each step considers a node from one of three sets, namely: the community under construction (initialized with the seed node), the *shell* of nodes that are neighbors of nodes in the community but do not belong to the community, and the *unexplored portion* of the network. Figure 1 illustrates an example of multilayer local community identified over a multilayer network; in the figure, we also utilize terms *boundary* and *core* nodes to differentiate between the within-community nodes that have and do not have neighbors, respectively, in the shell set.

A key aspect in the task at hand is how to select the *best* node in the shell to add to the community to be identified. Most algorithms, which are designed to deal with simple

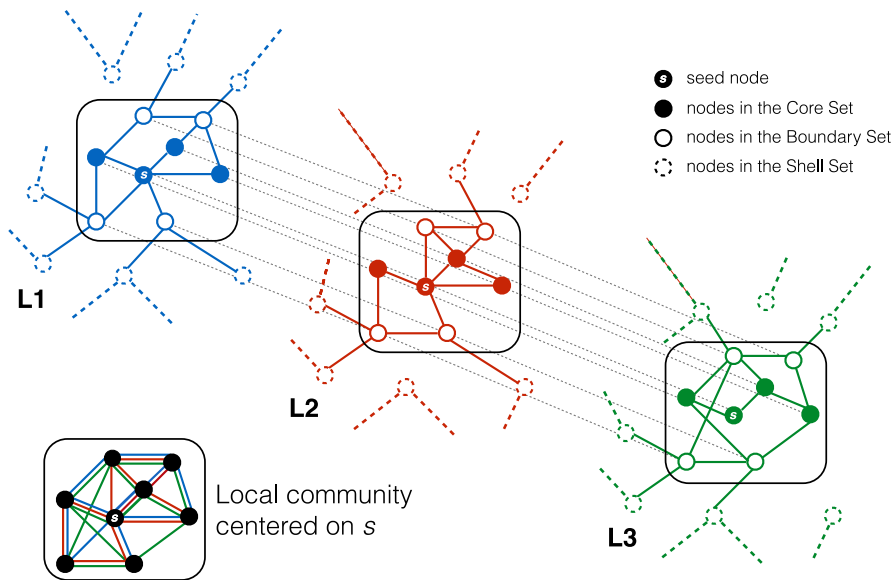


Fig. 1 A local community identified over an example 3-layer network. The local community (delimited by a rounded rectangle) is centered on a seed node s , while core nodes, boundary nodes, and shell nodes are denoted with filled, empty, and dotted circles, respectively. Dotted edges starting from the shell nodes point to unknown portions of the network. (Best viewed in color version, available in electronic format)

(i.e., single-layer) network graphs, try to maximize a function in terms of the *internal* edges, i.e., edges that involve nodes in the community, and to minimize a function in terms of the *external* edges, i.e., edges to nodes outside the community. By accounting for both types of edges, nodes that are candidates to be added to the community being constructed are penalized in proportion to the amount of links to nodes external to the community (Clauset 2005; Chen et al. 2009; Branting 2012; Fagnan et al. 2014).

In this work we follow the above general approach and extend it to identify local communities over a multilayer network as presented in the problem statement reported next.

Definition 1 (*Multilayer local community detection problem*) Given a multilayer graph $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ with set of nodes \mathcal{V} , and a seed node $v_0 \in \mathcal{V}$, find a subgraph $G_{\mathcal{L}}^{v_0} \subseteq G_{\mathcal{L}}$ that contains v_0 and maximizes the *multilayer local community* function LC :

$$G_{\mathcal{L}}^{v_0} = \underset{\substack{G=(V,E,\mathcal{V},\mathcal{L}) \subseteq G_{\mathcal{L}} \\ \wedge v_0 \in V}}{\operatorname{argmax}} LC(G) = \underset{\substack{G=(V,E,\mathcal{V},\mathcal{L}) \subseteq G_{\mathcal{L}} \\ \wedge v_0 \in V}}{\operatorname{argmax}} \frac{LC^{int}(G)}{LC^{ext}(G)} \tag{1}$$

where $LC^{int}(G)$ is a function proportional to the density of links among nodes within G , and $LC^{ext}(G)$ is a function proportional to density of links between nodes within G and nodes outside G .

It should be emphasized that our formulation accounts for the internal-to-external *connection density* ratio rather than the absolute amount of internal and external links to the community. This is an important aspect since, as first analyzed in (Chen et al. 2009), it allows for alleviating the issue of inserting many weakly-linked nodes (i.e., *outliers*) into the local community being discovered.

In our setting, we also have to cope with the complexity of a multilayer network model. In this regard, in the following section we shall provide different definitions of our multilayer local community functions $LC^{int}(G)$ and $LC^{ext}(G)$.

3.3 Multilayer local community functions

Given $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ and a seed node v_0 , we denote with $C \subseteq \mathcal{V}$ the node set and with $E^C \subseteq E_{\mathcal{L}}$ the edge set of subgraph $G_{\mathcal{L}}^{v_0}$ corresponding to the local community built around node v_0 ; moreover, when the context is clear, we will use C to refer to the local community subgraph. Symbol $E_i^C = \{(u, v) | \exists((u, L_i), (v, L_i)) \in E^C\}$ will be used to specialize E^C for edges in the community that correspond to a given layer L_i .

As discussed in the previous section, for a local community being constructed, the *shell* set refers to nodes external to the community that are neighbors of nodes in the community, and these within-community neighbors of shell nodes are also called *boundary* nodes. We define the shell set of C as:

$$S = \{v \in \mathcal{V} \setminus C \mid \exists((u, L_i), (v, L_j)) \in E_{\mathcal{L}} \wedge u \in C\}$$

and the boundary set of C as:

$$B = \{u \in C \mid \exists((u, L_i), (v, L_j)) \in E_{\mathcal{L}} \wedge v \in S\}.$$

Moreover, we denote with $E^B = \{(u, v) \mid ((u, L_i), (v, L_j)) \in E_{\mathcal{L}} \wedge u \in B \wedge v \in S\}$ the set of edges outgoing from C , and for any layer L_i , $E_i^B = \{(u, v) \mid \exists((u, L_i), (v, L_i)) \in E^B\}$ as the portion of E^B corresponding to edges of layer L_i .

We devise different ways for completely specifying the internal-to-external connection density ratio expressed in the objective function of the problem in Def. 1. In particular, we will consider two intuitive criteria:

- *Extension of the local community functions to handle multiple layers.* In Sect. 3.3.1, we accomplish this by linearly combining the layer-specific contributions to the connection density ratio.
- *Integration of multilayer-aware, “homophilic” bias to control the community expansion.* Homophily explains the tendency of individuals to associate and bond with similar others. Therefore, we might also want to make the evaluation of a candidate node sensitive to the affinity of the node with some of the community members. The rationale here is to bias the choice of a node in terms of its (direct or indirect) homophily with the community, rather than only considering its contribution to the internal-to-external connection density ratio. In Sects. 3.3.2

and 3.3.3, we pursue this goal by modeling a within-layer similarity-based factor and a cross-layer similarity-based factor, respectively, into the objective function.

3.3.1 Layer-weighting-based local community functions

In our first specification of the problem in Def. 1, we incorporate multilayer features in the local community functions. One intuitive way to do this is to account for the relevance of each of the layers, which brings to the following definition of *layer-weighting-based* local community functions.

Definition 2 (*Layer-weighted local community functions*) Given $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ and a local community C , the layer-weighting-based local community internal function is defined as:

$$LC^{int}(C) = \frac{1}{|C|} \sum_{v \in C} \sum_{L_i \in \mathcal{L}} \omega_i |E_i^C(v)| \quad (2)$$

The layer-weighting-based local community external function is defined as:

$$LC^{ext}(C) = \frac{1}{|B|} \sum_{v \in B} \sum_{L_i \in \mathcal{L}} \omega_i |E_i^B(v)| \quad (3)$$

where ω_i (for every $L_i \in \mathcal{L}$) are non-negative real-valued coefficients, with $\sum_{L_i \in \mathcal{L}} \omega_i = 1$, which define a weighting scheme over the layers.

The layer weighting scheme based on coefficients ω_i is set by default to a uniform distribution. Alternatively, it can be specified following either in an unsupervised or a supervised way. For instance, assuming to have some external knowledge on the relevance of the layers, the weights would be defined proportionally, thus following a supervised criterion. On the other hand, using any other distribution reflecting some known or computable graph property of the nodes/edges in each layer, would correspond to an unsupervised approach.

3.3.2 Within-layer similarity-based local community functions

While accounting for the presence of multiple layers, we believe it is also important to control the community expansion based on a notion of similarity in the node linkage, which might express a homophilic factor in the definition of multilayer local community functions.

To this end, two major requirements are: (1) how to choose the analytical form of the similarity function, and (2) how to deal with the different, layer-specific connections that any two nodes might have in the multilayer graph. We address the first point in an unsupervised fashion, by resorting to any similarity measure that can express the topological affinity of two nodes in the graph. Concerning the second point, one straightforward solution is to determine the similarity between any two nodes focusing on each layer at a time. The above points are formally captured by the following definition.

Definition 3 (*Within-layer similarity-based local community functions*) Given $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ and a local community C , the within-layer similarity-based community internal relation is defined as:

$$LC^{int}(C) = \frac{1}{|C|} \sum_{v \in C} \sum_{L_i \in \mathcal{L}} \sum_{\substack{(u,v) \in E_i^C \\ \wedge u \in C}} sim_i(u, v) \quad (4)$$

The within-layer similarity-based community external relation is defined as:

$$LC^{ext}(C) = \frac{1}{|B|} \sum_{v \in B} \sum_{L_i \in \mathcal{L}} \sum_{\substack{(u,v) \in E_i^B \\ \wedge u \in S}} sim_i(u, v) \quad (5)$$

where $sim_i(u, v)$ denotes the similarity between nodes u, v contextually to layer L_i .

As previously mentioned, we adopt an unsupervised approach to the evaluation of the similarity between two nodes u, v , by measuring the *topological affinity* of u and v . We choose to accomplish this by resorting to any topology-based similarity measure between two node sets, or more generally, between their induced subgraphs. In this work, we regard the generic form of $sim_i(u, v)$ as $f(N_i(u), N_i(v))$, where f denotes a function proportional to the similarity of two node sets $N_i(u), N_i(v)$, with $N_i(x) = \{y \in \mathcal{V} | (y, x) \in E_i\}$ as the set of neighbors of a node x in layer L_i . In Sect. 4 we will discuss similarity measures used in our experimental evaluation.

3.3.3 Cross-layer similarity-based local community functions

One evident simplification in the previous definition is that it expresses the homophily between two nodes by quantifying the strength of their connections considering the layers on which they lay as separately to each other. This might also be a limitation in that the within-layer similarity-based method is not able to capture *indirect homophily*, i.e., ties between nodes that are not directly connected through the network: nevertheless, two nodes can still have some topological affinity even without the presence of an explicit edge in the network (e.g., having lots of friends in common while being unknown to each other). Based on this simple intuition, we provide our definition of *cross-layer* similarity-based local community functions.

Definition 4 (*Cross-layer similarity-based local community functions*) Given $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ and a local community C , the cross-layer similarity-based community internal function is defined as:

$$LC^{int}(C) = \frac{1}{|C|} \sum_{u,v \in C} \sum_{\substack{L_i, L_j \in \mathcal{L} \\ \wedge u \in V_i, v \in V_j}} sim_{i,j}(u, v) \quad (6)$$

The cross-layer similarity-based community external relation is defined as:

$$LC^{ext}(C) = \frac{1}{|B|} \sum_{v \in B, u \in S} \sum_{\substack{L_i, L_j \in \mathcal{L} \\ \wedge u \in V_i, v \in V_j}} sim_{i,j}(u, v) \quad (7)$$

where $sim_{i,j}(u, v)$ denotes the similarity between u in layer L_i and v in layer L_j .

Analogously to the previous definition, $sim_{i,j}(u, v)$ corresponds to a topology-based similarity measure between the sets of neighbors of u and v , which however in this case are located in different layers.

3.4 Multilayer local community identification algorithm

Algorithm 1 reports the pseudo-code of the general scheme of our proposed **MultiLayer Local Community Detection** (ML-LCD) methods.

The scheme takes as input the multilayer graph $G_{\mathcal{L}}$ and a seed node v_0 , and computes the local community C associated to v_0 . Recall that knowledge on the topology of the multilayer graph is only partial in our setting; precisely, at any give time step, only the direct neighbors of a node are known in advance. Note also that the general scheme is actually instantiated in three algorithms, according to Def. 2, Def. 3, or Def. 4. We will use notation **ML-LCD-*lw***, **ML-LCD-*wlsim***, and **ML-LCD-*clsim*** to refer to the ML-LCD algorithm equipped with layer-weighting-based, within-layer similarity-based, and cross-layer similarity-based local community functions, respectively.

At the beginning, the boundary set (B) and the community (C) are initialized with the starting seed, while the shell set (S) is initialized with the neighborhood set of v_0 considering all the layers in \mathcal{L} . The initial value of $LC(C)$ is computed (lines 3–4) according to one of the three approaches for the definition of the local community functions. The algorithm then starts expanding the node set C (lines 5–19). First, it evaluates nodes v belonging to the current shell set S , and selects the vertex v^* that corresponds to the maximum value of objective function LC if the node would be added to C (line 6). The choice of candidate node v^* impacts on both S and B (lines 7–8): node v^* is removed from S (since it has been examined), whereas all nodes in B that have no other neighbors in S than v^* are removed from B . Since adding node v^* to C might not increase the current value of the objective function, the algorithm checks if (1) v^* actually increases the quality of C (i.e., $LC(C \cup \{v^*\}) > currLC$) and (2) v^* helps to strength the internal connectivity of the community (i.e., $LC^{int}(C \cup \{v^*\}) > currLC^{int}$). If both conditions are satisfied (line 9), node v^* is added to C and the following actions are taken: (1) v^* is added to B as long as it has neighbors that are not in C (line 13), and the shell set is updated to contain these nodes (line 14); (2) a further update to B might be requested in case there are nodes in C that now have neighbors in S (line 15); (3) $LC(C \cup \{v^*\})$ becomes the new value of $currLC$. Otherwise, if node v^* is not added to C , only $LC^{ext}(C)$ is updated according to the changes in S and B (performed at lines 7–8). The algorithm terminates when no further improvement in $LC(C)$ is possible.

Algorithm 1 Multilayer Local Community Detection (ML-LCD) scheme

Input: Multilayer graph $G_{\mathcal{L}} = (V_{\mathcal{L}}, E_{\mathcal{L}}, \mathcal{V}, \mathcal{L})$ (only partially known),
 seed node $v_0 \in V_{\mathcal{L}}$.
Output: Local community C for v_0 .
 1: $B \leftarrow \{v_0\}, C \leftarrow B$
 2: $S \leftarrow \{v | (v, v_0) \in E_L \ \forall L \in \mathcal{L}\}$
 3: $currLC^{int} \leftarrow LC^{int}(C), currLC^{ext} \leftarrow LC^{ext}(C)$ //using Def. 2, Def. 3, or Def. 4
 4: $currLC \leftarrow LC(C) = currLC^{int} / currLC^{ext}$
 5: **repeat**
 6: $v^* \leftarrow \operatorname{argmax}_{v \in S} LC(C \cup \{v\})$ //using update rules in Eq. 8–11
 7: $S \leftarrow S \setminus \{v^*\}$
 8: $B \leftarrow B \setminus \{u \in B | v^* \in N(u) \wedge \nexists (u, v) : v \in S\}$
 9: **if** $LC(C \cup \{v^*\}) > currLC \wedge LC^{int}(C \cup \{v^*\}) > currLC^{int}$
 10: $C \leftarrow C \cup \{v^*\}$
 11: $N_{-C} \leftarrow N(v^*) \setminus C$
 12: **if** $N_{-C} \neq \emptyset$
 13: $B \leftarrow B \cup \{v^*\}$
 14: $S \leftarrow S \cup N_{-C}$
 15: $B \leftarrow B \cup \{u \in C \setminus B | N(u) \subseteq S\}$
 16: $currLC^{int} \leftarrow LC^{int}(C), currLC^{ext} \leftarrow LC^{ext}(C),$
 $currLC \leftarrow currLC^{int} / currLC^{ext}$
 17: **else**
 18: $currLC^{ext} \leftarrow LC^{ext}(C)$
 19: **until** $LC(C)$ cannot be further maximized
 20: **return** C

LC^{int} and LC^{ext} update formulas. Testing a candidate node $v \in S$ for possible insertion into C (line 6) requires evaluation of LC for an updated community $C \cup \{v\}$. To do this, we can avoid computing LC^{int} and LC^{ext} from scratch by using the incremental update formulas reported next; in this regard, we use symbols B_v and S_v , where $B_v \subseteq B$ contains neighbors of v in B that have no other neighbors in S , and $S_v = N(v) \setminus C$ contains neighbors of v that are not in C .

– ML-LCD-*lw* and ML-LCD-*wlsim*:

$$LC_v^{int} = |C|LC^{int} + \sum_{u \in C} \sum_{\substack{L_i \in \mathcal{L} \\ \wedge (u,v) \in E_i^C}} \Gamma \tag{8}$$

$$LC_v^{ext} = |B|LC^{ext} + \sum_{u \in S_v} \sum_{\substack{L_i \in \mathcal{L} \\ \wedge ((u,L_i),(v,L_i)) \in E_{\mathcal{L}}}} \Gamma - \sum_{u \in B_v} \sum_{\substack{L_i \in \mathcal{L} \\ \wedge (u,v) \in E_i^B}} \Gamma \tag{9}$$

where Γ is a reference such that $\Gamma \equiv \omega_i$ in the case of ML-LCD-*lw* and $\Gamma \equiv sim_i(u, v)$ in the case of ML-LCD-*wlsim*.

– ML-LCD-*clsim*:

$$LC_v^{int} = |C|LC^{int} + \sum_{u \in C} \sum_{\substack{L_i, L_j \in \mathcal{L} \\ \wedge ((u,L_i),(v,L_j)) \in E_{\mathcal{L}}}} sim_{i,j}(u, v) \tag{10}$$

$$\begin{aligned}
 LC_v^{ext} = & |B|LC^{ext} + \sum_{u \in S_v} \sum_{s \in S} \sum_{\substack{L_i, L_j \in \mathcal{L} \\ \wedge ((u,i), (s,j)) \in E_{\mathcal{L}}}} sim_{i,j}(u, s) + \\
 & - \sum_{u \in B_v} \sum_{s \in S} \sum_{\substack{L_i, L_j \in \mathcal{L} \\ \wedge ((u,i), (s,j)) \in E_{\mathcal{L}}}} sim_{i,j}(u, s)
 \end{aligned} \tag{11}$$

3.5 Computational complexity aspects

We discuss here computational complexity aspects of ML-LCD. We introduce symbol Φ to denote the computational cost of any of the proposed local community functions, and symbol d to indicate the maximum degree of a node in the network. We recall that $\ell = |\mathcal{L}|$ denotes the number of layers and $|C|$ denotes the size of the community being discovered.

In Algorithm 1, at each iteration the community size increases by one node until it reaches a certain size $|C|$, whereas the shell set S size may vary over the iterations. In this regard, we introduce the quantity $(k \times d)$ as an upper bound of the size of S at the current iteration, where k corresponds to the size of the current community (proportional to the number of the current iteration). Note that this upper bound corresponds to the worst case in which all nodes in the current community are assumed to belong to the boundary set B and all of the neighbor sets are disjoint.

Since the algorithm terminates in a number of iterations which is proportional to the size of the generated community, we state the following upper bound of the overall computational complexity of Algorithm 1: $\mathcal{O}(d \times \Phi \times \sum_{k=1}^{|C|} k)$, which can be rewritten as $\mathcal{O}(|C|^2 \times d \times \Phi)$.

Considering ML-LCD-*lw*, the characteristic operation has a cost proportional to the neighbors of any node v in C and the number of layers ℓ , since the weights ω are known at constant time. Therefore, $\Phi = \mathcal{O}(\ell \times d)$, which leads to the cost of $\mathcal{O}(|C|^2 \times d^2 \times \ell)$ for ML-LCD-*lw*.

While following the same strategy as ML-LCD-*lw*, method ML-LCD-*wlsim* also requires the computation of similarity between any node v and its neighbors. This implies at least comparison between the set of (direct) neighbors of v and the set of (direct) neighbors of u . If we assume that the neighborhood lists are always ordered (based on their identifiers), the cost of set comparison becomes $\mathcal{O}(d \log d)$ — checking if an element is present in an ordered list costs $\mathcal{O}(\log d)$ and this basic operation is repeated d times. In this case, $\Phi = \mathcal{O}(\ell \times d^2 \log d)$. Therefore, the cost for ML-LCD-*wlsim* is $\mathcal{O}(|C|^2 \times d^3 \log d \times \ell)$.

The topology-unawareness of ML-LCD-*clsim* leads to an increase in the cost Φ . Specifically, the cost becomes proportional to $|C|^2$ multiplied by the sum of the costs of functions LC^{int} and LC^{ext} . The latter is expressed in terms of size of S , which, in turn, is expressed in terms of size of the current community; therefore, in the worst case the cost of LC^{ext} is predominant over LC^{int} , thus we focus on LC^{ext} . At each iteration of Algorithm 1, we might compare each node in the boundary B (in the worst case, the whole C), to each node in the shell set S . Therefore, for ML-LCD-*clsim*, the overall cost is given by: $\mathcal{O}(\sum_{k=1}^{|C|} k \times d \times k \times d \times \ell^2 \times d \log d)$, where the first term

$(k \times d)$ is related to the basic cost, while the second term $(k \times d)$ refers to the specific topology-unawareness characteristic of *ML-LCD-clsim*. Rearranging the terms in the previous formula results in $\mathcal{O}(|C|^3 \times d^3 \log d \times \ell^2)$.

4 Experimental evaluation

In the following, Sect. 4.1 summarizes the evaluation datasets, Sect. 4.2 introduces competing methods, and Sect. 4.3 describes the experimental settings¹.

4.1 Data

We used seven real-world multilayer network datasets, namely *Airlines* (Cardillo et al. 2013), *AUCS* (Dickison et al. 2016), *BIOGRID* (Bonchi et al. 2014), *DBLP* (Boden et al. 2012), *Reality Mining* (Kim and Lee 2015; Bourqui et al. 2016), *RemoteSensing*, and *TW-YT-FF* (Dickison et al. 2016). *Airlines* describes airline companies operating in Europe. Nodes and edges represent airport locations and routes, respectively, and each layer corresponds to a different airline company. *AUCS* models relationships between employees of a University department considering five different aspects: coworking, having lunch together, Facebook friendship, offline friendship, and coauthorship. *Biogrid* is a protein-protein interaction network, where layers correspond to seven different types of interactions between proteins. In *DBLP*, nodes correspond to authors and layers represent the top-50 Computer Science conferences. Two authors are connected on a layer if they co-authored at least two papers together in a particular conference. *Reality Mining* contains human interaction data collected by the MIT Media Lab, where layers represent different media employed to communicate: subjects calling each other, friendship claims, and text message exchanges. *RemoteSensing* is a network derived from a remote sensing satellite image obtained from the SWH website.² Nodes correspond to segments of the image and edges of a certain type exists between two segments if they are adjacent and have the same value for a certain radiometric attribute (layer). *TW-YT-FF* is a cross-platform network built by exploiting the feature of FriendFeed as social media aggregator to align registered users who were also members of Twitter and YouTube. Nodes correspond to users and edges to friendships over the three different platforms as layers.

Table 1 summarizes main characteristics of our evaluation datasets. Node relations in all datasets are treated as symmetric. We denote with A_{deg} the average degree of a node considering multiple edges, and with A_{layer} the average number of layers in which a node is present. Note that *Airlines* and *DBLP* are rich in number of layers (resp. 37 and 50), and that *Airlines*, *RemoteSensing*, *AUCS* and *RealityMining* have nodes, on average, involved in more than two layers (resp. 4.88, 4.19, 3.67 and 2.42).

¹ Source code and evaluation data are available at <http://people.dimes.unical.it/andreatagarelli/mlcd/>

² <https://www.theia-land.fr/en/products/spot-world-heritage>

Table 1 Main characteristics of the multilayer network datasets

Dataset	# Nodes	# Edges	#Layers	Density	A_{deg}	A_{layer}
Airlines	417	3588	37	0.056	17.21	4.88
AUCS	61	620	5	0.114	20.33	3.67
Biogrid	38, 936	342, 599	7	4.8e-4	17.60	1.90
DBLP	83, 901	159, 302	50	8.9e-4	3.80	1.35
RealityMining	88	355	3	0.047	8.07	2.42
RemoteSensing	642	4341	5	0.006	13.52	4.19
TW-YT-FF	6407	74, 862	3	2.35e-3	23.37	1.86

4.2 Competing methods

Our evaluation focus is on understanding how the proposed local community detection methods perform on multilayer networks. Nevertheless, given the unavailability of competing methods for local community detection in multilayer networks, we devised two stages of comparative evaluation: (1) comparison with *single-layer, local community detection*, and (2) comparison with *multilayer, global community detection* approaches.

For the first stage of comparative evaluation, we resorted to the methods LCD (Chen et al. 2009) and Lemon (Li et al. 2015). The LCD approach is chosen since its strategy of identification of a local community is close to ours, while Lemon is a more recent, state-of-the-art method for local community detection. Since the methods are designed to deal with single-layer graphs, we applied each of them on the *aggregate graph* derived from the input multilayer network $G_{\mathcal{L}}$, i.e., a graph with set of nodes \mathcal{V} and such that an edge exists between any two nodes that are connected in at least one layer in $G_{\mathcal{L}}$. While LCD is completely parameter free, for Lemon we used the default setting,³ also disabling the ground-truth option and the sampling for graphs with less than 1k nodes.

For the second stage of comparative evaluation, we used Locally Adaptive Random Transitions (LART) (Kuncheva and Montana 2015), Principal Modularity Maximization (PMM) (Tang et al. 2009) and Generalized Louvain (GL) (Mucha et al. 2010). Note that all such methods produce *non-overlapping* global community structures, while the local communities produced by ML-LCD for various seed nodes can of course overlap to each other. Note also that LART and GL deal with node-layer pairs, thus allowing different instances of the same node to belong to more communities; to make the two methods comparable with all the other approaches, we apply a majority voting mechanism, so that a node is assigned to the community containing the majority of its instances.

Inferring multilayer global community structure. In order to compare our ML-LCD methods with global community detection methods, we propose a heuristic, hereinafter

³ As specified in the publicly available implementation from the repository at <https://github.com/YixuanLi/LEMON>

referred to as **LocToGlob**, to infer a global *non-overlapping* community structure from local solutions produced by our methods.

LocToGlob takes as input the multilayer graph $G_{\mathcal{L}}$ and yields a set of non-overlapping communities. The algorithm works by iteratively performing a covering phase, which first computes the local community for a selected seed node, then updates the global community structure and the portion of graph that has not been processed yet. For the selection of seed node at each iteration of **ML-LCD**, we employ the simple heuristic (Fagnan et al. 2014), which first samples at random a node v then selects the node with maximum degree among v and its neighbors. The covering phase terminates when all nodes have been processed. Since this phase may produce singleton communities, a post-processing phase is performed to re-assign each of the nodes belonging to a singleton community to the (non-singleton) community for which adding the singleton produces the highest increase in the local community function.

4.3 Experimental settings

Each of our **ML-LCD** methods was carried out over all nodes in a network, by selecting one node at a time as seed. **ML-LCD-wlsim** and **ML-LCD-clsim** require the definition of a node similarity measure for a given layer ($sim_i(\cdot, \cdot)$) or couple of layers ($sim_{i,j}(\cdot, \cdot)$), respectively (cf. Sect. 3.3). We considered three alternative measures, as listed below:

- *Jaccard similarity*, i.e., $sim_{i,j}(u, v) = \frac{|N_i(u) \cap N_j(v)|}{|N_i(u) \cup N_j(v)|}$, where $N_i(u) = \{v \in \mathcal{V} | (v, u) \in E_i\}$ denotes the set of neighbors of a node u in layer L_i (with $i = j$ for **ML-LCD-wlsim**).
- *Cosine similarity*, whereby the proportionality of shared neighborhood is smoothed to favor unbalanced neighborhoods to be compared: $sim_{i,j}(u, v) = \frac{|N_i(u) \cap N_j(v)|}{\sqrt{|N_i(u)| |N_j(v)|}}$ (with $i = j$ for **ML-LCD-wlsim**).
- *Triad-based similarity*, which we define as a Jaccard similarity calculated on the set of 3-cliques to which any two nodes u, v belong to: $sim_{i,j}(u, v) = \frac{|N_i(u) \cap N_j(v)|}{|T_i(u) \cup T_j(v)|}$, where $T_i(u)$ indicates the set of 3-cliques (or *triads*) to which node u belongs to on layer i ($i = j$ for **ML-LCD-wlsim**). Please note that, since we take into account undirected graphs, the number of triads in common between two adjacent nodes is equivalent to the number of neighbors in common.

We will use notations *jac*, *cos*, and *triads* to refer to the above instantiations of function *sim*. For the layer weighting scheme in **ML-LCD-lw**, we assume uniform weights.

As concerns the evaluation of global community detection, we averaged performance results of **LocToGlob** and **PMM** over a relatively large number of runs (50), in order to reduce the bias due to their non-deterministic behavior. The number of desired communities, which is a further input in **PMM**, was set equal to the number of communities extracted by each of our **LocToGlob** variants based on **ML-LCD**. Per-

formance results by **LocToGlob** variants and competing ones are assessed by means of the multilayer (or multislice) modularity measure proposed in (Mucha et al. 2010).⁴

5 Results

We organize the presentation of experimental results in three subsections: Sect. 5.1 provides an in-depth analysis of the communities identified by our ML-LCD methods, whereas the subsequent two subsections contain comparative evaluation of ML-LCD with local community detection methods on aggregate graphs (Sect. 5.2) and with global, multilayer community detection methods (Sect. 5.3).

5.1 Evaluation of ML-LCD methods

We assessed the behavior of the proposed ML-LCD methods in terms of: (1) size of extracted local communities, (2) structural characteristics of the communities, (3) similarity between communities, (4) distribution of layers involved in each of the local communities, (5) community distribution over number of edges, (6) impact of similarity measures on ML-LCD-*wlsim* and ML-LCD-*clsim*, (7) overlap of communities generated by each of the methods, (8) efficiency analysis. In the following we will present results concerning each of the above evaluation aspects.

5.1.1 Size of local communities

Table 2 compares our ML-LCD methods in terms of size of the local communities produced. (We refer here only to ML-LCD methods, while results corresponding to LCD and Lemon will be discussed later in Sect. 5.2). On average, ML-LCD-*lw* yields the largest communities on all datasets, except *TW-YT-FF*; on this dataset, which has a unique combination of highest A_{deg} and lowest A_{layer} (cf. Table 1), ML-LCD-*clsim* communities slightly prevail in size w.r.t. ML-LCD-*wlsim*, though the latter shows the largest variation. In general, ML-LCD-*wlsim* yields the smallest communities on all datasets (with the exception of *RemoteSensing* on which however it shows the smallest variation). Note also that, on *DBLP*, all methods tend to identify quite small communities, which can be explained due to the nature of the node relation (i.e., co-authorship).

5.1.2 Structure of the communities

Table 3 reports the per-layer *average path length* and *clustering coefficient* of the identified communities. For each of the datasets, ML-LCD methods, and measures, we report mean and standard deviation over the layers, of the mean and maximum values found over all communities. ML-LCD-*lw* communities tend to have the highest

⁴ Experiments were carried out on an Intel Core i7-3960X CPU @3.30GHz, 64GB RAM machine. All our algorithms are written in Python. Original codes of competing methods are in Python (LART) and Matlab (PMM and GL).

Table 2 Mean and standard deviation of the size of the communities identified by ML-LCD methods and by LCD and Lemon

Dataset	ML-LCD- <i>lw</i>	ML-LCD- <i>w/sim</i>	ML-LCD- <i>c/sim</i>	LCD	Lemon
Airlines	33.62 ± 33.57	11.48 ± 15.04	20.42 ± 21.38	9.69 ± 11.29	73.34 ± 15.83
AUCS	11.23 ± 3.40	7.90 ± 2.74	9.10 ± 3.30	11.48 ± 4.28	55.52 ± 11.48
Biogrid	17.65 ± 53.27	4.87 ± 9.14	8.80 ± 38.71	6.45 ± 11.43	<u>61.88</u> ± 23.99
DBLP	4.52 ± 2.74	3.67 ± 2.16	3.72 ± 2.90	3.53 ± 2.43	<u>36.45</u> ± 24.71
RealityMining	8.05 ± 7.08	3.37 ± 1.77	7.02 ± 5.28	5.23 ± 4.31	<u>49.55</u> ± 21.36
RemoteSensing	7.46 ± 3.30	5.66 ± 2.53	5.47 ± 2.82	7.98 ± 3.44	<u>37.83</u> ± 18.88
TW-YT-FF	15.54 ± 29.14	17.53 ± 34.01	17.98 ± 21.05	9.88 ± 17.71	<u>70.75</u> ± 21.00

Bold and underlined values correspond to the maximum (mean) size obtained by a ML-LCD method and all selected methods, respectively

Table 3 Per-layer average path length and clustering coefficient of the communities identified by ML-LCD methods

Dataset	Average path length			Clustering coefficient		
	Mean	<i>SD</i>	Max	Mean	<i>SD</i>	Max
<i>ML-LCD-lw</i>						
Airlines	0.881 ± 0.143		2.259 ± 0.365	0.077 ± 0.088		0.488 ± 0.374
AUCS	1.479 ± 0.201		1.978 ± 0.207	0.560 ± 0.180		0.801 ± 0.138
Biogrid	0.656 ± 0.430		6.049 ± 0.767	0.095 ± 0.102		1.000 ± 0.000
DBLP	0.041 ± 0.029		2.306 ± 0.284	0.016 ± 0.011		1.000 ± 0.000
RealityMining	1.409 ± 0.108		2.545 ± 0.621	0.276 ± 0.112		0.889 ± 0.192
RemoteSensing	1.474 ± 0.048		2.786 ± 0.145	0.471 ± 0.082		1.000 ± 0.000
TW-YT-FF	1.368 ± 1.107		4.515 ± 2.250	0.128 ± 0.118		1.000 ± 0.000
<i>ML-LCD-wlsim</i>						
Airlines	0.222 ± 0.168		1.872 ± 0.397	0.022 ± 0.027		0.508 ± 0.398
AUCS	1.182 ± 0.237		1.880 ± 0.364	0.533 ± 0.238		0.938 ± 0.087
Biogrid	0.314 ± 0.247		3.839 ± 0.655	0.079 ± 0.091		1.000 ± 0.000
DBLP	0.030 ± 0.021		1.875 ± 0.202	0.015 ± 0.011		1.000 ± 0.000
RealityMining	0.778 ± 0.111		1.833 ± 0.153	0.295 ± 0.126		1.000 ± 0.000
RemoteSensing	1.201 ± 0.053		2.541 ± 0.265	0.529 ± 0.094		1.000 ± 0.000
TW-YT-FF	1.091 ± 0.473		2.660 ± 0.121	0.301 ± 0.233		0.917 ± 0.144
<i>ML-LCD-clsim</i>						
Airlines	0.557 ± 0.180		2.208 ± 0.485	0.035 ± 0.052		0.277 ± 0.252
AUCS	1.249 ± 0.221		2.022 ± 0.269	0.447 ± 0.237		0.824 ± 0.176
Biogrid	0.336 ± 0.332		6.593 ± 2.439	0.037 ± 0.044		1.000 ± 0.000
DBLP	0.028 ± 0.020		2.306 ± 0.324	0.010 ± 0.008		1.000 ± 0.000
RealityMining	1.165 ± 0.276		2.743 ± 0.287	0.113 ± 0.083		0.719 ± 0.286
RemoteSensing	1.244 ± 0.183		3.073 ± 0.379	0.232 ± 0.043		1.000 ± 0.000
TW-YT-FF	1.211 ± 0.955		3.771 ± 1.440	0.062 ± 0.058		0.546 ± 0.506

Bold values correspond to the minimum per-dataset (mean) average path length and to the maximum per-dataset (mean) clustering coefficient

mean and maximum values of average path length, followed by *ML-LCD-clsim*. This is likely to be related to the different sizes of communities produced by the methods. Moreover, all methods' communities show very low average path length on *DBLP*, which again depends on the co-authorship type of relations.

Considering the clustering coefficient values, the three methods behave quite similarly to each other on *AUCS*, *DBLP*, and *RealityMining*. Some clique communities are identified on *Biogrid*, *DBLP* and *RealityMining* (all methods), *TW-YT-FF* (*ML-LCD-lw*), and *RealityMining* (*ML-LCD-wlsim*). Also, by coupling with average path length results, roughly small-world communities are observed on *AUCS*, *RealityMining*, *RemoteSensing* and (for *ML-LCD-wlsim*) *TW-YT-FF*. The lowest values of average path length and clustering coefficient is achieved by all methods on *DBLP*, which might be explained due to the very sparse connectivity in this dataset (cf. Table 1).

Table 4 Pairwise comparison of communities produced by ML-LCD methods: mean and standard deviation of Jaccard similarity

Dataset	ML-LCD- <i>lw</i> versus ML-LCD- <i>wlsim</i>	ML-LCD- <i>wlsim</i> versus ML-LCD- <i>clsim</i>	ML-LCD- <i>lw</i> versus ML-LCD- <i>clsim</i>
Airlines	0.345 ± 0.351	0.389 ± 0.397	0.391 ± 0.349
AUCS	0.641 ± 0.314	0.688 ± 0.288	0.638 ± 0.364
Biogrid	0.440 ± 0.400	0.722 ± 0.376	0.425 ± 0.395
DBLP	0.753 ± 0.306	0.807 ± 0.258	0.674 ± 0.317
RealityMining	0.536 ± 0.361	0.542 ± 0.319	0.458 ± 0.308
RemoteSensing	0.536 ± 0.305	0.531 ± 0.234	0.444 ± 0.265
TW-YT-FF	0.254 ± 0.300	0.328 ± 0.325	0.237 ± 0.251

Bold values correspond to the highest per-dataset (mean) similarities

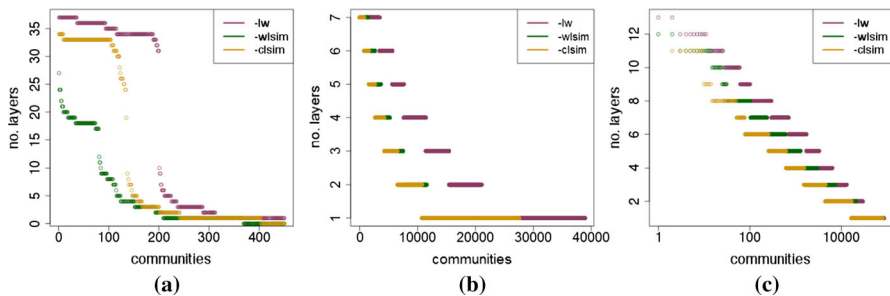


Fig. 2 Distribution of number of layers over communities. (Best viewed in color version, available in electronic format). **a** Airlines **b** Biogrid **c** DBLP

5.1.3 Community similarity

We also compared our methods in terms of Jaccard similarity between the sets of nodes belonging to the local communities respectively extracted for the same seed node. Table 4 summarizes the mean and standard deviation similarities aggregated over all local communities.

The highest average similarity is generally achieved by ML-LCD-*wlsim* vs. ML-LCD-*clsim*, with marginal exceptions for *Airlines* (ML-LCD-*lw* vs. ML-LCD-*clsim*) and *RemoteSensing* (ML-LCD-*lw* vs. ML-LCD-*wlsim*). Also, the comparison between ML-LCD-*lw* and ML-LCD-*clsim* always results in the lowest average similarity with the exception of *Airlines* where ML-LCD-*lw* and ML-LCD-*wlsim* behave more differently from each other.

5.1.4 Layer coverage over communities

We analyzed the number of layers covered by each particular community, for each of the ML-LCD methods. Figure 2 reports results for the datasets with higher number of layers, i.e., *Airlines*, *Biogrid* and *DBLP*. (Communities are sorted by decreasing number of layers.) On *DBLP*, all methods show a stairs-like behavior on the various

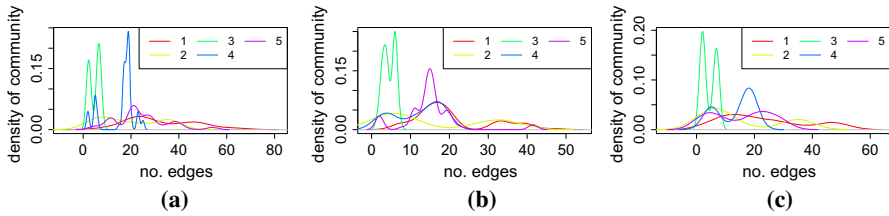


Fig. 3 Kernel density estimates of per-layer community distributions for number of edges, on *AUCS*. (Best viewed in color version, available in electronic format). **a** ML-LCD-*lw* **b** ML-LCD-*wlsim* **c** ML-LCD-*clsim*

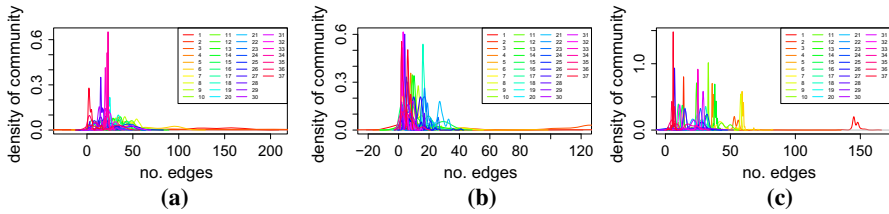


Fig. 4 Kernel density estimates of per-layer community distributions for number of edges, on *Airlines*. (Best viewed in color version, available in electronic format). **a** ML-LCD-*lw* **b** ML-LCD-*wlsim* **c** ML-LCD-*clsim*

layers, with a tendency of ML-LCD-*clsim* to produce more communities that cover a lower number of layers. A less regular form of stairs-like trend is observed for *Biogrid*, with a long tail (more evident for ML-LCD-*clsim*) of single-layer communities. More interestingly, on *Airlines*, a large fraction of communities (above 50%) by ML-LCD-*lw* cover all or most of the layers; ML-LCD-*clsim* here also produces communities that lie on most layers, although for a much smaller fraction of them (about 12%), whereas ML-LCD-*wlsim* produces about 25% of the communities covering 50-70% of layers.

Moreover, considering the datasets with the highest network-specific average ratio of layers per node (i.e., *AUCS*, *RealityMining* and *RemoteSensing*), we observe that all methods are able to produce most of the communities that cover all layers (results not shown). On *TW-YT-FF*, at least two of three layers are always covered by a community.

5.1.5 Distributions of communities for number of edges

In addition to the previous analysis, we also studied the per-layer distribution of communities for number of edges. Due to space limitation, we report here details on *AUCS* and *Airlines*, which can be regarded as representatives of our evaluation datasets in terms of layer coverage and sparseness. Figures 3–4 show the kernel density estimate of the community distribution for number of edges, over each layer in a network.

On *AUCS* (Fig. 3), when using ML-LCD-*lw*, layers #3 and #4 characterize most of the communities with number of edges less than 10 and around 20, respectively. For the other layers there is instead a tendency to be covered by communities of varying size (in the full x-axis, for layer #1). Considering ML-LCD-*wlsim* and ML-LCD-*clsim*, layer #3 characterizes the majority of communities with smaller number of edges,

Table 5 Mean and standard deviation of the size of the communities identified by ML-LCD-*wlsim* and ML-LCD-*clsim* using Cosine and triad-based similarity measures

Dataset	ML-LCD- <i>wlsim</i>		ML-LCD- <i>clsim</i>	
	<i>cos</i>	<i>triads</i>	<i>cos</i>	<i>triads</i>
Airlines	13.348 ± 17.023	8.127 ± 11.968	21.362 ± 22.197	19.273 ± 21.324
AUCS	8.869 ± 3.032	8.328 ± 2.274	12.230 ± 8.538	8.689 ± 4.298
Biogrid	5.383 ± 11.748	4.953 ± 3.919	20.419 ± 57.634	7.338 ± 16.300
DBLP	3.266 ± 2.466	3.642 ± 2.108	3.706 ± 3.930	3.764 ± 2.843
RealityMining	3.523 ± 2.276	3.333 ± 1.805	10.273 ± 10.441	7.632 ± 5.859
RemoteSensing	5.865 ± 2.620	5.926 ± 2.714	7.945 ± 4.276	7.006 ± 3.844
TW-YT-FF	25.785 ± 42.462	16.700 ± 29.853	18.209 ± 17.020	9.080 ± 5.656

Bold values correspond to the highest per-dataset (mean) size

while peaks of layer #5 and #4, respectively, correspond to less than 20 edges per community.

On *Airlines* (Fig. 4), distributions appear very right-skewed w.r.t. the number of edges. For ML-LCD-*lw*, the highest density corresponds to a small group of layers (being #34 the dominant one), in the regime around 25 edges. Much more diversified is the situation for ML-LCD-*wlsim* and ML-LCD-*clsim*, where more layers are characteristic for a relatively large number of edges; in particular, ML-LCD-*clsim* is able to produce communities where more than five layers correspond to high density for a number of edges between ten and forty.

5.1.6 Impact of similarity measures on ML-LCD-*wlsim* and ML-LCD-*clsim*

We investigated the effect of using measures alternative to Jaccard similarity (*jac*), which is chosen as default in ML-LCD-*wlsim* and ML-LCD-*clsim* (cf. Sect. 4.3). Specifically, we discuss results concerning the size, similarity and matching of communities produced by the two methods when equipped with each of the similarity measures.

Looking at Table 5, we observe that the average size of the communities obtained using cosine similarity or triad-based similarity is, in many cases, higher than the one obtained using Jaccard (cf. Table 2), for both ML-LCD methods. One exception is represented by *DBLP*, where all the methods obtain comparable average sizes. This is probably due to the low connectivity and the high number of cliques this dataset contains; moreover, in *DBLP*, almost the same average community size is obtained by using *triads* and *jac*. As a side remark, the method used (i.e., ML-LCD-*wlsim* or ML-LCD-*clsim*) does not seem to have a strong bias on the relative differences among the behaviors of the different similarity measures, for what concerns the size of the communities.

Table 6 reports the average Jaccard similarities between the node sets of the communities obtained by ML-LCD-*wlsim* and ML-LCD-*clsim*, by varying the similarity measures. It can be noted that the most different communities (i.e., the lowest Jac-

Table 6 Average Jaccard similarity of the node sets corresponding to the communities obtained by ML-LCD-*w/sim* and ML-LCD-*c/sim* using Jaccard, Cosine and triad-based similarity measures

Dataset	ML-LCD- <i>w/sim</i>			ML-LCD- <i>c/sim</i>		
	<i>jac</i> versus <i>cos</i>	<i>jac</i> versus <i>triads</i>	<i>cos</i> versus <i>triads</i>	<i>jac</i> versus <i>cos</i>	<i>jac</i> versus <i>triads</i>	<i>cos</i> versus <i>triads</i>
Airlines	0.712 ± 0.286	0.680 ± 0.359	0.554 ± 0.328	0.776 ± 0.274	0.836 ± 0.291	0.717 ± 0.308
AUCS	0.914 ± 0.155	0.861 ± 0.227	0.818 ± 0.235	0.804 ± 0.237	0.802 ± 0.258	0.716 ± 0.246
Biogrid	0.769 ± 0.274	0.680 ± 0.349	0.669 ± 0.366	0.664 ± 0.345	0.905 ± 0.234	0.650 ± 0.229
DBLP	0.774 ± 0.221	0.900 ± 0.213	0.725 ± 0.239	0.739 ± 0.221	0.899 ± 0.214	0.714 ± 0.229
RealityMining	0.873 ± 0.218	0.946 ± 0.136	0.839 ± 0.237	0.682 ± 0.251	0.745 ± 0.255	0.659 ± 0.248
RemoteSensing	0.872 ± 0.219	0.810 ± 0.251	0.809 ± 0.246	0.661 ± 0.256	0.660 ± 0.301	0.671 ± 0.279
TW-YT-FF	0.734 ± 0.333	0.824 ± 0.231	0.701 ± 0.294	0.683 ± 0.272	0.728 ± 0.276	0.625 ± 0.292

Mean and standard deviation values are reported. Bold values correspond to the highest per-dataset (mean) similarities

card similarity values) are obtained when comparing communities detected by *cos* and *triads*, for both ML-LCD-*wlsim* and ML-LCD-*clsim* (with the only exception of ML-LCD-*clsim* on *RemoteSensing*). As regards the most similar communities, on ML-LCD-*clsim* higher similarity values are generally achieved comparing *jac* and *triads* measures. Even when this comparison does not correspond to the highest similarity of community, i.e., on *AUCS* and *RemoteSensing*, it is very close to the maximum value, which is therein obtained by comparing *jac* and *cos*. This is not surprising since *jac* and *triads* are both Jaccard-based measures, and on some specific networks the number of shared 3-cliques can be very similar to the number of shared neighbors, making the effects due to these two measures close to each other. As regards ML-LCD-*wlsim*, the highest similarity is achieved again when comparing *jac* and *triads* on *DBLP*, *RemoteSensing* and *TW-YT-FF*, and when comparing *jac* and *cos* on the other datasets. The latter would indicate that in the rest of the datasets the neighborhood of nodes to be compared tend to be of comparable sizes.

We further studied the effect of alternative measures by observing the percentage of identical communities obtained by ML-LCD-*wlsim* and ML-LCD-*clsim*. Looking at Table 7, this ranges from 24.60% to 81.60% for ML-LCD-*wlsim*, and from 21.80% to 76.60% for ML-LCD-*clsim*. Generally, high percentages are achieved on *DBLP*, which again is explained by the particular structure of this network. Considering ML-LCD-*clsim*, this appears to be more sensitive to the choice of similarity measure since the percentage of identical communities is general lower than the corresponding values obtained by ML-LCD-*wlsim*. In both cases, highest values are achieved considering the comparison between *jac* and *triads* measures. As regards ML-LCD-*wlsim*, relatively high percentages are obtained on *RemoteSensing* and *RealityMining*, i.e., datasets which are relatively small in size and number of layers. Conversely, the fraction of identical communities decreases considering networks with more layers and higher connectivity (e.g., on *Airlines*).

5.1.7 Community structure overlap

In this analysis we evaluated the amount of overlap among the communities identified by each of the ML-LCD methods on the various networks. To this purpose, we computed the Jaccard coefficients of the node sets over all pairs of local communities identified by any particular ML-LCD method. Results, shown in Table 8, indicate that each of the methods, on all datasets, produces local communities sharing, on average, relatively few nodes with the other communities in the network. In all cases, with the exception of all methods on *AUCS* and ML-LCD-*lw* on *Airlines*, the mean Jaccard coefficient is below 0.1, while the standard deviations are generally higher than the corresponding means, up to 0.3 on *AUCS*.

This might be explained considering the network characteristics reported in Table 1, and the semantics of our evaluation data (as described in Sect. 4.1). From a structural point of view, networks on which ML-LCD methods generate communities with more overlap are relatively smaller in terms of nodes (*AUCS*, *Airlines* and *RealityMining*), while less overlapping communities are identified for sparser and larger networks (*RemoteSensing*, *Biogrid* and *TW-YT-FF*). The extreme case is represented by the *DBLP* network where the extracted communities roughly form a partition of the net-

Table 7 Percentage of identical communities obtained by ML-LCD-*w/sim* and ML-LCD-*c/sim* using Jaccard, Cosine and triad-based similarity measures

Dataset	ML-LCD- <i>w/sim</i>		ML-LCD- <i>c/sim</i>	
	<i>jac</i> versus <i>cos</i>	<i>jac</i> versus <i>triads</i>	<i>cos</i> versus <i>triads</i>	<i>jac</i> versus <i>cos</i>
	<i>jac</i> versus <i>cos</i>	<i>jac</i> versus <i>triads</i>	<i>cos</i> versus <i>triads</i>	<i>jac</i> versus <i>cos</i>
Airlines	32.80%	46.50%	24.60%	28.40%
AUCS	59.00%	55.70%	34.40%	36.10%
Biogrid	50.70%	47.88%	29.57%	30.98%
DBLP	41.50%	76.10%	33.60%	32.00%
RealityMining	70.10%	81.60%	63.20%	23.00%
RemoteSensing	65.10%	49.00%	49.00%	22.50%
TW-YT-FF	27.31%	38.76%	25.10%	23.69%

Bold values correspond to the highest per-dataset percentages

Table 8 Community structure overlap: mean and standard deviation of the Jaccard coefficients of the node sets over all pairs of local communities generated by each of the ML-LCD methods

Dataset	Airlines	AUCS	Biogrid	DBLP	RealityMining	RemoteSensing	TW-YT-FF
<i>lw</i>	0.136 ± 0.252	0.184 ± 0.347	0.001 ± 0.018	0 ± 0.002	0.080 ± 0.203	0.011 ± 0.081	0.003 ± 0.030
<i>wlsim_jac</i>	0.037 ± 0.155	0.122 ± 0.310	0.000 ± 0.012	0 ± 0.002	0.040 ± 0.176	0.008 ± 0.077	0.004 ± 0.050
<i>wlsim_cos</i>	0.040 ± 0.153	0.142 ± 0.323	0.000 ± 0.012	0 ± 0	0.039 ± 0.175	0.009 ± 0.080	0.004 ± 0.055
<i>wlsim_triads</i>	0.019 ± 0.102	0.131 ± 0.321	0.009 ± 0.079	0 ± 0.002	0.035 ± 0.160	0.009 ± 0.079	0.003 ± 0.056
<i>clsim_jac</i>	0.090 ± 0.260	0.135 ± 0.307	0.000 ± 0.010	0 ± 0.002	0.058 ± 0.167	0.007 ± 0.064	0.008 ± 0.071
<i>clsim_cos</i>	0.092 ± 0.268	0.158 ± 0.317	0.001 ± 0.025	0 ± 0	0.087 ± 0.207	0.010 ± 0.077	0.008 ± 0.066
<i>clsim_triads</i>	0.091 ± 0.261	0.125 ± 0.301	0.014 ± 0.084	0 ± 0.002	0.066 ± 0.173	0.009 ± 0.071	0.012 ± 0.089

Bold values correspond to the highest per-dataset (mean) community overlaps

work. We remind that this network contains many small connected components and it is characterized by high sparsity and the lowest average degree per node (A_{deg} equals to 3.8). The slightly higher overlap of community structure observed on smaller datasets (i.e., *AUCS*, *Airlines* and *RealityMining*) also finds an intuitive explanation in the nature of the multilayer data, and in particular in the presence of *hubs* or *bridges* in the network. For instance, in the flight routing scenario corresponding to *Airlines*, some nodes may represent airport locations that are main hubs in a certain geographical area (e.g., major European capital cities). These hubs would be connected to the majority of smaller airports in the area, and with other main hubs, for almost all airline companies flying in that area, therefore they will be more likely to be included in several local communities. As concerns *AUCS* and *RealityMining*, since these networks represent different online/offline interactions between (relatively) small sets of individuals, some highly popular individuals may act as “bridges” across layers, causing a certain overlap in the local communities built around them and their neighbors. For instance, the chief of the department represented in *AUCS* is likely to appear, together with full professors, in several local communities. A similar scenario can be figured out for *RealityMining* where members of the MIT Media Laboratory and members of the MIT Sloan business school can belong to different communities since they attend the same master classes and/or sport activities in the campus.

5.1.8 Efficiency analysis

We discuss here the time performances of ML-LCD. Table 9 shows, for each dataset and method, the average, standard deviation and maximum execution times, over all nodes in the network. Note that minimum time values are not reported, since they are of the order of magnitude of $1.0e-4$ in all cases, except for *TW-YT-FF* where they correspond to few seconds for all methods.

Considering first ML-LCD-*lw* and the default versions of ML-LCD-*wlsim* and ML-LCD-*clsim* (*jac* option), it can be noted that, when the methods produce communities of comparable size (like in *AUCS*, *DBLP* and *RemoteSensing*; cf. Table 2), ML-LCD-*lw* and ML-LCD-*wlsim* have similar time performance while ML-LCD-*clsim* tends to be slower; in particular, on *DBLP*, this behavior appears to be more evident considering the maximum time, since the average time is biased by the very sparse connectivity of this network and the presence of many, small, connected components in which the size of the shell set plays a minor role than in the other networks. Conversely, on *Biogrid* and *RealityMining*, ML-LCD-*lw* tends to run more slowly than the other two, whereas on *Airlines* the time performances of ML-LCD-*lw* are between those of ML-LCD-*wlsim* and ML-LCD-*clsim*; also, on *TW-YT-FF*, ML-LCD-*clsim* can perform better than ML-LCD-*wlsim*. The above findings are clearly related to both the computational complexity of each of the ML-LCD methods as well as to the size of the communities extracted by the methods; for instance, on *Airlines*, *Biogrid* and *RealityMining*, ML-LCD-*lw* communities are bigger (from two to four times) than those extracted by the other methods; also, on *TW-YT-FF*, the size obtained by ML-LCD-*wlsim* with *triads* option is nearly double the size of ML-LCD-*clsim* communities (Table 5).

It should be noted that the time performance of the various methods on *TW-YT-FF* is several orders of magnitude higher than in the other datasets (up to an average execution

Table 9 Time performance (in seconds) of ML-LCD-*lv*, ML-LCD-*wlsim* and ML-LCD-*clsim* using different similarity measures

Dataset	<i>lv</i>		<i>wlsim_jac</i>		<i>wlsim_cos</i>		<i>wlsim_triads</i>		<i>clsim_jac</i>		<i>clsim_cos</i>		<i>clsim_triads</i>	
	Mean SD	Max	Mean SD	Max	Mean SD	Max	Mean SD	Max	Mean SD	Max	Mean SD	Max	Mean SD	Max
Airlines	3.872 4.342	18.220	1.351 2.393	11.759	2.184 3.479	15.578	51.101 107.456	607	7.252 10.182	31.335	20.649 28.082	92.297	<u>236</u> 319	824
AUCS	0.042 0.042	0.198	0.042 0.036	0.194	0.065 0.052	0.232	0.638 0.295	1.258	0.115 0.097	0.430	0.359 0.613	2.832	<u>1.451</u> 1.676	7.952
Biogrid	68.8 1196	61,789	13.5 106	5188	10.2 65.9	598	<u>1670</u> 7840	62,780	43.2 637	38,884	42.1 490	8378	1073 5348	63,307
DBLP	0.005 0.020	0.339	0.004 0.011	0.185	0.006 0.022	0.341	0.115 0.607	13.329	0.005 0.037	1.179	0.016 0.139	3.826	<u>0.137</u> 1.504	39,538
RealityMining	0.049 0.087	0.317	0.010 0.010	0.045	0.011 0.011	0.039	0.467 0.406	1.800	0.040 0.052	0.214	0.090 0.163	0.845	<u>0.678</u> 0.832	3.395
RemoteSensing	0.019 0.024	0.133	0.021 0.028	0.265	0.022 0.018	0.098	0.064 0.059	0.480	0.038 0.058	0.364	0.094 0.126	0.776	<u>0.213</u> 0.337	2.632
TW-YT-FF	601 1682	9065	123 578	3651	1493 2270	6266	133,514 <u>105,151</u>	158,704	13,493 38,922	163,299	100 239	856	11,952 13,190	32,250

Mean, standard deviation and maximum values are computed over the execution times of each ML-LCD method over all nodes in each network, by selecting one node at a time as seed. Bold values and underlined values correspond to the lowest and the highest mean time, respectively

time of 133 514 seconds for *wlsim_triads* and absolute maximum time for *clsim_jac*). This is explained since, as reported in Table 1, *TW-YT-FF* is the dataset having the highest average degree of node and, among the largest datasets, is also the densest one (one order of magnitude denser than *Biogrid* and *DBLP*). These characteristics make *TW-YT-FF* our hardest benchmark in terms of time performance obtained by our methods.

Concerning the impact of the different similarity measures on the time performances, the execution time of the methods tend to become much slower when *triads* is used, given the generally higher cost of computing all the 3-cliques in which a node is involved than analyzing its neighbors. Moreover, options *jac* and *cos* generally lead ML-LCD-*wlsim* and ML-LCD-*clsim* to behave similarly (which is quite expected since the two similarity measures have the same cost), in all cases with the exception of *TW-YT-FF* — for the characteristics of this dataset above discussed, the Jaccard measure could lead to particularly slow performance, as it may yield high values (i.e., the nodes may share a large portion of their neighborhood), thus being quite inclusive when building a local community.

5.1.9 Discussion

We summarize here the main results from the evaluation of our ML-LCD methods. First, ML-LCD-*lw* followed by ML-LCD-*clsim* have shown to produce larger communities on all datasets. This indicates that in general the approach underlying ML-LCD-*lw* (i.e., fixed layer-based weighting scheme) tends to be less “xenophobic” than ML-LCD-*clsim*, and this in turn less than ML-LCD-*wlsim*. This is actually explained since the impact of the LC^{ext} term is generally smaller for the ML-LCD-*lw* formulation allowing it to grow more the local community for a particular seed node. A different situation corresponds to ML-LCD-*clsim* where the LC^{ext} function is computed between the boundary nodes and all their neighbors without considering the layers on which they lay.

Related to the above aspect is that ML-LCD-*lw* followed by ML-LCD-*clsim* provide communities having the highest mean and maximum values of average path length. Moreover, all methods are able to produce roughly small-world communities (on *AUCS*, *RealityMining*, and ML-LCD-*wlsim* on *TW-YT-FF*).

ML-LCD-*wlsim* and ML-LCD-*clsim* are more likely to behave similarly, in terms of Jaccard similarity between the extracted communities for the same node. This is not surprising, since these two formulations are based on a similar notion of node similarity, while differing in how layer features are considered. Moreover, for both methods, cosine similarity and triad-based similarity show a generally more aggregating behavior than the default Jaccard similarity in most cases.

ML-LCD-*lw* and ML-LCD-*clsim* generally produce communities that cover all or most of the layers. This holds consistently with various distributions of the number of layers per node; when the latter is quite high (e.g., above 80%), all ML-LCD methods are equally able to mostly produce communities that cover all or most of the layers.

Each of the ML-LCD methods produces local communities sharing, on average, relatively few nodes with the other communities identified in the same network dataset.

As concerns the efficiency of the proposed ML-LCD methods, the best performance is obtained by ML-LCD-*wlsim* (with Jaccard or cosine similarity) in most cases. Yet, this can be explained since, despite the ML-LCD-*lw* has the lowest computational complexity, it tends to discover larger local communities than the other methods, thus affecting the number of iterations to terminate. This efficiency result would make ML-LCD-*wlsim* a preferred choice for the ML-LCD problem, supporting our initial intuition on beneficial effects that might be obtained by incorporating node similarity over the layers into the objective function of the ML-LCD problem.

5.2 Comparison with local community detection methods on the aggregate graphs

In this evaluation stage, our goal was to assess the difference in size and in composition between the local communities produced by ML-LCD and the ones (corresponding to the same seeds) produced by the LCD and Lemon methods (cf. Sect. 4.2).

Size values of the communities identified by LCD and Lemon are reported in Table 2, along with values corresponding to the ML-LCD methods. From the comparison, we observe that the mean size of communities by LCD are generally lower than the size of at least one of the ML-LCD methods; on *AUCS* and *RemoteSensing*, where LCD yields the largest communities on average, the corresponding values are however very close to those by ML-LCD-*lw*, yet with higher standard deviation. Different situation concerns Lemon, which in absence of ground-truth information on the communities to be identified (as in our case), tends to generate communities with significantly larger size than approaches based on the optimization of the internal-to-external connection density ratio.

Tables 10–11 summarize results relating to the comparison with LCD and Lemon, respectively, in terms of Jaccard similarity (mean and standard deviation values over all seed nodes) and fraction of identical communities (i.e., Jaccard similarity equal to one). Considering Table 10 and looking at the mean similarity values w.r.t. LCD, it appears that on all datasets, but *DBLP*, they are always below 0.5, with maximum values per dataset that are reached when the *cos* option is used. Also, the percentage of identical communities w.r.t. LCD is generally very low or even null (again, with the exception of *DBLP* where ML-LCD-*wlsim* and ML-LCD-*clsim* equipped with *cos* obtain about 40% of same communities as those by LCD); clearly, the situation observed for *DBLP* depends on the presence of many small, highly-connected components in this network. Concerning the comparison with Lemon (Table 11), it is evident and even more significant the lack of similarity between ML-LCD and the competing one: mean similarities are occasionally above 0.2, while percentages of identical communities are zero in nearly all cases, again with the exception of *DBLP* (in which case, they are around or below 5%).

Overall, not surprisingly, the above results confirm that accounting for the multiple available layer information, as our ML-LCD methods do, brings to the identification of local communities that are quite different from the ones obtained by a single-layer LCD method applied on the aggregate graph derived from the original multilayer network.

Table 10 Comparison of ML-LCD methods with LCD: mean and standard deviation of Jaccard similarity of communities and percentage of identical communities (%*idc*)

Dataset	<i>lw</i>		<i>wlsim_jac</i>		<i>wlsim_cos</i>		<i>wlsim_trtads</i>		<i>clsim_jac</i>		<i>clsim_cos</i>		<i>clsim_trtads</i>	
	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>
Airlines	0.28	2.17%	0.27	0.24%	0.36	17.59%	0.29	1.45%	0.26	0.00%	0.34	17.11%	0.26	0.00%
	0.25		0.22		0.35		0.23		0.23		0.36		0.23	
AUCS	0.46	4.92%	0.41	4.92%	0.42	4.92%	0.43	4.92%	0.41	0.00%	0.42	0.00%	0.39	0.00%
	0.32		0.31		0.32		0.31		0.30		0.30		0.29	
Biogrid	0.34	3.34%	0.31	0.58%	0.44	26.14%	0.23	0.00%	0.29	0.11%	0.26	11.07%	0.36	0.00%
	0.25		0.20		0.37		0.17		0.19		0.30		0.18	
DBLP	0.59	10.72%	0.55	6.08%	0.69	41.20%	0.55	6.04%	0.53	3.70%	0.65	36.40%	0.53	3.80%
	0.23		0.23		0.32		0.22		0.21		0.31		0.21	
RealityMining	0.41	2.30%	0.40	1.15%	0.46	14.94%	0.40	1.15%	0.35	0.00%	0.40	14.94%	0.34	0.00%
	0.24		0.22		0.30		0.21		0.19		0.30		0.18	
RemoteSensing	0.39	1.10%	0.39	1.25%	0.41	5.79%	0.38	0.16%	0.34	0.00%	0.38	4.85%	0.33	0.00%
	0.21		0.22		0.25		0.20		0.17		0.22		0.16	
TW-YT-FF	0.19	0.55%	0.20	0.63%	0.24	3.59%	0.16	0.00%	0.21	1.30%	0.25	6.09%	0.31	12.00%
	0.20		0.20		0.21		0.34		0.20		0.22		0.32	

Bold values correspond to the highest per-dataset (mean) similarities and percentages

Table 11 Comparison of ML-LCD methods with Lemon: mean and standard deviation of Jaccard similarity of communities and percentage of identical communities (%*idc*)

Dataset	<i>lw</i>		<i>wlsim_jac</i>		<i>wlsim_cos</i>		<i>wlsim_triads</i>		<i>clsim_jac</i>		<i>clsim_cos</i>		<i>clsim_triads</i>	
	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>	Mean	% <i>idc</i>
Airlines	0.27	0.00	0.12	0.00	0.14	0.00	0.09	0.00	0.18	0.00	0.18	0.00	0.17	0.00
	0.23		0.14		0.16		0.11		0.18		0.18		0.18	
AUCS	0.21	0.00	0.15	0.00	0.42	0.17	0.16	0.00	0.17	0.00	0.22	0.00	0.16	0.00
	0.08		0.06		0.07		0.06		0.08		0.13		0.10	
Biogrid	0.09	0.00	0.07	0.00	0.07	0.00	0.09	0.00	0.07	0.00	0.09	0.00	0.06	0.00
	0.10		0.08		0.09		0.11		0.09		0.12		0.06	
DBLP	0.21	4.25	0.20	5.18	0.19	4.23	0.20	5.19	0.20	5.08	0.21	4.46	0.20	5.01
	0.22		0.24		0.24		0.24		0.24		0.24		0.24	
RealityMining	0.16	0.00	0.08	0.00	0.09	0.00	0.08	0.00	0.16	0.00	0.21	0.00	0.18	0.00
	0.14		0.06		0.07		0.06		0.14		0.18		0.16	
RemoteSensing	0.22	0.00	0.18	0.00	0.18	0.00	0.18	0.00	0.17	0.00	0.23	0.00	0.20	0.00
	0.11		0.09		0.10		0.10		0.10		0.13		0.12	
TW-YT-FF	0.07	0.03	0.08	0.08	0.08	0.11	0.04	0.00	0.11	0.00	0.11	0.00	0.07	0.00
	0.09		0.12		0.13		0.04		0.12		0.13		0.09	

Bold values correspond to the highest per-dataset (mean) similarities and percentages

Table 12 Number of communities identified by LocToGlob (mean and standard deviation) and competing methods

Dataset	LocToGlob- <i>lw</i>	LocToGlob- <i>wlsim</i>	LocToGlob- <i>clsim</i>	LART	GL
Airlines	50.38 ± 4.28	67.36 ± 5.18	56.20 ± 6.14	<u>381</u>	12
AUCS	8.46 ± 1.17	10.24 ± 1.11	8.66 ± 1.05	<u>27</u>	6
Biogrid	5959.0 ± 105.53	7323.60 ± 229.32	6037.20 ± 272.1	NA	<u>16151</u>
RemoteSensing	116.44 ± 5.28	134.76 ± 5.10	152.62 ± 5.32	<u>218</u>	19
RealityMining	20.64 ± 2.30	23.82 ± 2.02	20.42 ± 2.55	<u>62</u>	13
TW-YT-FF	1076.44 ± 25.17	1255.71 ± 38.33	1230.71 ± 45.33	NA	761

Results of LocToGlob are aggregated over fifty runs. Bold values and underlined values correspond to the highest per-dataset community numbers considering ML-LCD methods and all methods, respectively

5.3 Comparison with global community detection methods

Table 12 summarizes statistics on the number of communities identified by LocToGlob (with suffixes *lw*, *wlsim*, and *clsim* to distinguish among the different variants) and by the competing methods — please note that we do not report results of PMM since, for this method, the number of communities is an input parameter. We observe that, among our proposed methods, LocToGlob-*lw* and LocToGlob-*wlsim* tend to produce the lowest and highest number of communities, respectively (which is in line with results previously presented). Considering also LART and GL, the former yields a number of communities that is significantly higher than that of other methods; however, on *Biogrid* and *TW-YT-FF*, it was not able to terminate as it incurred an out-of-memory issue.⁵ Note also that on *Biogrid*, all the methods produced the largest number of communities, which might be explained since this network is characterized by the highest sparsity among all the employed datasets (cf. Table 1).

Table 13 reports multilayer modularity values corresponding to the global, non-overlapping community structures obtained by LocToGlob and competing methods. Notation PMM-*i*, with $i = \{1, 2, 3\}$, refers to the setting of number of communities corresponding to each of the LocToGlob methods (cf. Sect. 4.3).⁶

A first important remark is that, on all network datasets, LocToGlob-*lw* always leads to better modularities than the other two approaches, with LocToGlob-*wlsim* and LocToGlob-*clsim* showing similar behaviors. Compared to the competitors, all LocToGlob methods outperform, in terms of both average and maximum modularity, LART and the three variants of PMM. In the case of GL, LocToGlob-*lw* obtains comparable average modularity; moreover, LocToGlob-*lw* achieves higher maximum modularity on *AUCS*, *Airlines* and *TW-YT-FF* with respect to GL.

The obtained results highlight that our proposed framework is able to reach *comparable or even better modularity* results against state-of-the-art global community detection methods. We stress here that this evaluation setting is actually a severe com-

⁵ We do not include *DBLP* in this evaluation because the competitors incurred out-of-memory issues, though our methods did not.

⁶ For LART and GL we report only one performance score since they have a deterministic behavior.

Table 13 Multilayer modularity performance of global community detection methods

Dataset	LocToGlob- <i>lw</i>		LocToGlob- <i>wlsim</i>		LocToGlob- <i>clsim</i>	
	Mean	Max	Mean	Max	Mean	Max
Airlines	0.033	0.040	0.023	0.028	0.027	0.033
AUCS	0.225	0.252	0.212	0.233	0.214	0.239
Biogrid	0.113	0.135	0.074	0.076	0.085	0.088
RemoteSensing	0.196	0.201	0.185	0.190	0.151	0.157
RealityMining	0.243	0.281	0.209	0.226	0.187	0.225
TW-YT-FF	0.167	0.185	0.118	0.121	0.117	0.122

Dataset	LART	GL	PMM-1		PMM-2		PMM-3	
			Mean	Max	Mean	Max	Mean	Max
Airlines	0.013	<u>0.037</u>	0.014	0.018	0.014	0.016	0.014	0.015
AUCS	0.154	<u>0.249</u>	0.167	0.222	0.157	0.187	0.166	0.218
Biogrid	NA	<u>0.225</u>	0.038	0.039	0.037	0.039	0.038	0.039
RemoteSensing	0.069	<u>0.269</u>	0.159	0.167	0.152	0.161	0.143	0.150
RealityMining	0.101	<u>0.288</u>	0.175	0.217	0.165	0.200	0.171	0.237
TW-YT-FF	NA	0.022	0.053	0.580	0.060	0.069	0.053	0.053

Bold values and underlined values correspond to the highest per-dataset modularities considering ML-LCD methods and all methods, respectively

parison for our approach since: all the competitors exploit a complete knowledge of the graph to optimize their internal objective function and, the multilayer modularity we employ as evaluation criterion is also the function internally optimized by GL.

6 Conclusion

We addressed the novel problem of local community detection in multilayer networks, and presented the first algorithmic framework to solve it. Our ML-LCD formulation relies on the optimization of a function based on internal and external connectivity of a multilayer local community, and is instantiated into three methods that differ in the way within- and across-layer features are taken into account to discover a local community for a given seed node. An extensive analysis over seven real-world multilayer networks has shown significance and ability of our methods in detecting multilayer local communities. In addition, by inferring a global non-overlapping community structure from the local communities identified for a given set of seed nodes in the network, we demonstrated that our ML-LCD methods can produce higher-modularity communities than state-of-the-art methods designed for multilayer global community detection.

Our future research directions would move along paths involving both theoretical and experimental aspects. On the former, we would like to study alternative objective functions for the ML-LCD problem and theoretical guarantees on the sub-optimality of the solutions. Also, an interesting aspect to understand is how outlier and hub nodes

can be efficiently detected and handled during the construction of a local community. Considering the experimental evaluation, it would be an important addition the construction of a ground-truth for the local communities to be identified in a multilayer network. Finally, we envisage a number of application problems for which ML-LCD methods can profitably be used, such as friendship prediction, targeted influence propagation, and more in general, mining in incomplete networks.

Acknowledgements D. Ienco would like to acknowledge the National French Center for Spatial Study (CNES), in the framework of the project “DYNAMITEF TOSCA 2016”, to support this study with the *RemoteSensing* dataset.

References

- Berlingerio M, Coscia M, Giannotti F (2011) Finding and characterizing communities in multidimensional networks. In: Proc. IEEE/ACM Int. Conf. Adv Soc Netw Anal Min (ASONAM), 490–494
- Berlingerio M, Pinelli F, Calabrese F (2013) ABACUS: frequent pattern mining-based community discovery in multidimensional networks. *Data Min Knowl Discov* 27(3):294–320
- Boden B, Günnemann S, Hoffmann H, Seidl T (2012) Mining coherent subgraphs in multi-layer graphs with edge labels. In: Proc. ACM SIGKDD Int. Conf. Knowl Discovery and Data Min (KDD), 1258–1266
- Bonchi F, Gionis A, Gullo F, Ukkonen A (2014) Distance oracles in edge-labeled graphs. In: Proc. Int. Conf. Ext Database Technol (EDBT), 547–558
- Bourqui R, Ienco D, Sallaberry A, Poncelet P (2016) Multilayer graph edge bundling. In: Proc. IEEE Pac Visualization Symp (PacificVis), 184–188
- Branting K (2012) Context-sensitive detection of local community structure. *Soc Netw Anal Min* 2(3):279–289
- Cai D, Shao Z, He X, Yan X, Han J (2005) Community Mining from Multi-relational Networks. In: Proc. Europ Conf. Princ Pract Knowl Discov Databases (PKDD), 445–452
- Carchiolo V, Longheu A, Malgeri M, Mangioni G (2010) Communities unfolding in multislice networks. In: Proc. Comp Netw, 187–195
- Cardillo A, Gomez-Gardenes J, Zanin M, Romance M, Papo D, del Pozo F, Boccaletti S (2013) Emergence of network features from multiplexity. *Sci Rep* 3:1344
- Chen J, Zaïane OR, Goebel R (2009) Local community identification in social networks. In: Proc. IEEE/ACM Int. Conf. Adv Soc Netw Anal Min (ASONAM), 237–242
- Clauset A (2005) Finding local community structure in networks. *Phys Rev E* 72(2):026132
- Dickison ME, Magnani M, Rossi L (2016) Multilayer social networks. Cambridge University Press, UK
- Fagnan J, Zaïane OR, Barbosa D (2014) Using triads to identify local community structure in social networks. In: Proc. IEEE/ACM Int. Conf. on Adv Soc Netw Anal Min (ASONAM), 108–112
- Hmimida M, Kanawati R (2015) Community detection in multiplex networks: a seed-centric approach. *Netw Heterogen Media* 10(1):71–85
- Jeub LGS, Mahoney MW, Mucha PJ, Porter MA (2015) A local perspective on community structure in multilayer networks. *Netw Sci*, 1–20
- Kanawati R (2015) Empirical evaluation of applying ensemble methods to ego-centred community identification in complex networks. *Neurocomputing* 150:417–427
- Kim J, Lee J (2015) Community detection in multi-layer graphs: a survey. *SIGMOD Rec* 44(3):37–48
- Kivela M, Arenas A, Barthélemy M, Gleeson JP, Moreno Y, Porter MA (2014) Multilayer networks. *J Complex Netw* 2(3):203–271
- Kuncheva Z, Montana G (2015) Community detection in multiplex networks using locally adaptive random walks. In: Proc. IEEE/ACM Int. Conf. on Adv Soc Netw Anal Min (ASONAM), 1308–1315
- Li Y, He K, Bindel D, Hopcroft JE (2015) Uncovering the small community structure in large networks: a local spectral approach. In: Proc. ACM Conf. on World Wide Web (WWW), 658–668
- Loe CW, Jensen HJ (2015) Comparison of communities detection algorithms for multiplex. *Phys A* 431:29–45
- Michael T, Nachtergaele B (2012) Alignment and integration of complex networks by hypergraph-based spectral clustering. *Phis. Rev. E* 85:056111

- Mucha PJ, Richardson T, Macon K, Porter MA, Onnela J-P (2010) Community structure in time-dependent, multiscale, and multiplex networks. *Science* 328(5980):876–878
- Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. *Phys. Rev. E* 69(2):026113
- Papalexakis EE, Akoglu L, Ienco D (2013) Do more views of a graph help? community detection and clustering in multi-graphs. In: *Proc. Int. Conf. on Inform Fus*, 899–905
- Tang L, Wang X, Liu H (2009) Uncovering groups via heterogeneous interaction analysis. In: *Proc. ICDM*, 503–512
- Tang L, Wang X, Liu H (2012) Community detection via heterogeneous interaction analysis. *Data Min Knowl Discov* 25:1–33
- Yakoubi Z, Kanawati R (2014) LICOD: a leader-driven algorithm for community detection in complex networks. *Vietnam J Comput Sci* 1(4):241–256
- Zakrzewska A, Bader DA (2015) A dynamic algorithm for local community detection in graphs. In: *Proc. IEEE/ACM Int. Conf. Adv Soc Netw Anal Min (ASONAM)*, 559–564