Community Detection in Networks with Node Attributes

Jaewon Yang
Stanford University
jayang@stanford.edu

Julian McAuley
Stanford University
jmcauley@cs.stanford.edu

Jure Leskovec
Stanford University
jure@cs.stanford.edu

***Abstract*—Community detection algorithms are fundamental tools that allow us to uncover organizational principles in networks. When detecting communities, there are two possible sources of information one can use: the network structure, and the features and attributes of nodes. Even though communities form around nodes that have common edges *and* common attributes, typically, algorithms have only focused on one of these two data modalities: *community detection* algorithms traditionally focus only on the network structure, while *clustering* algorithms mostly consider only node attributes. In this paper, we develop Communities from Edge Structure and Node Attributes (CESNA), an accurate and scalable algorithm for detecting overlapping communities in networks with node attributes. CESNA statistically models the interaction between the network structure and the node attributes, which leads to more accurate community detection as well as improved robustness in the presence of noise in the network structure. CESNA has a linear runtime in the network size and is able to process networks an order of magnitude larger than comparable approaches. Last, CESNA also helps with the interpretation of detected communities by finding relevant node attributes for each community.**

I. Introduction

One of the most important tasks when studying networks is that of identifying *network communities*. Fundamentally, communities allow us to discover groups of interacting objects (*i.e.*, nodes) and the relations between them. For example, in social networks, communities correspond to groups of friends who attended the same school, or who come from the same hometown [28]; in protein interaction networks, communities are functional modules of interacting proteins [1]; in co-authorship networks, communities correspond to scientific disciplines [16]. Identifying network communities allows us to discover functionally related objects [15], [16], [41], study interactions between modules [2], infer missing attribute values [4], [10], and predict unobserved connections [9].

Identifying network communities can be viewed as a problem of clustering a set of nodes into communities, where a node can belong multiple communities at once. Because nodes in communities share common properties or attributes, and because they have many relationships among themselves, there are two sources of data that can be used to perform the clustering task. The first is the data about the objects (*i.e.*, nodes) and their attributes. Known properties of proteins, users' social network profiles, or authors' publication histories may tell us which objects are similar, and to which communities or modules they may belong. The second source of data comes from the network and the set of *connections* between the objects. Users form friendships, proteins interact, and authors collaborate.

However, clustering methods typically focus only one of these two data modalities. In terms of attributes, *clustering* algorithms [6], [20] identify sets of objects whose attributes are similar, while ignoring relationships between objects. On the other hand, *community detection* algorithms aim to find communities based on the network structure, *e.g.*, to find groups of nodes that are densely connected [14], [39], but they typically ignore node attributes.

By considering only one of these two sources of information independently, an algorithm may fail to account for important structure in the data. For example, attributes might tell us to which community a node with very few links belongs to; this would be difficult to determine from the network structure alone. Conversely, the network might tell us that two objects belong to the same community, even if one of them has no attribute information. Thus, it is important to consider both sources of information together and consider network communities as sets of nodes that are densely connected, but which *also* share some common attributes. Node attributes can complement the network structure, leading to more precise detection of communities; additionally, if one source of information is missing or noisy, the other can make up for it. However, considering both node attributes and network topology for community detection is also challenging, as one has to combine two very different modalities of information.

Only recently have approaches for detecting communities based on both sources of information been developed [4], [28] (Table I). Many existing methods that combine network and node attribute information use single-assignment clustering [3], [12], [31], [35], [43]; however, the applicability of these methods is limited, as they cannot detect overlapping communities. Approaches based on topic models [4], [27], [38], [40] allow overlapping communities to be detected. However, they assume "soft" node-community memberships, which are not appropriate for modeling communities because they do not allow a node to have high membership strength to multiple communities simultaneously [42]. Finally, all existing methods are only able to handle relatively small networks: the networks typically analyzed consist only of thousands of nodes [9], [27], [28], [38].

Present work: Community detection in networks with node attributes. Here, we develop a high-performance (accurate and scalable) overlapping community detection method for networks with node attribute information. We present *Com-*

Method class	O	H	D	N
Heuristics [3], [12], [31], [35], [43]	✗	✓	✗	100,000
LDA-based [4], [9], [27], [38], [40]	✓	✗	✓	85,000
Clique-based heuristics[18], [19]	✓	✓	✗	100,000
Social circles [28]	✓	✓	✗	5,000
CESNA	✓	✓	✓	**1,000,000**

TABLE I. METHODS FOR COMMUNITY DETECTION IN NETWORKS WITH NODE ATTRIBUTES. O: DETECTS OVERLAPPING COMMUNITIES?, H: ASSIGNS HARD NODE-COMMUNITY MEMBERSHIPS?, D: ALLOWS FOR DEPENDENCE BETWEEN THE NETWORK AND THE NODE ATTRIBUTES? (FIG. 1), N: LARGEST NETWORK THAT CAN BE PROCESSED IN 10 HOURS (FIG. 4). REFER TO SEC. II FOR FURTHER DETAILS.

munities from Edge Structure and Node Attributes (*CESNA*), which is based on a generative model for networks with node attributes. Our model advances existing approaches (summarized in Table I) by making several innovations that ultimately lead to better performance both in terms of accuracy as well as scalability. First, our model allows us to detect overlapping communities by employing hard node-community memberships. This way, we can avoid the assumption of soft-membership methods that nodes sharing multiple common communities are less likely to be connected [42]. Second, in contrast to a line of previous work [18], [28], which assumed that communities and attributes are marginally independent, we assume that communities "generate" both the network as well as attributes (Figure 1). This way we allow for dependence between the network and the attributes. Third, to fit the model and thus discover communities, we develop a block-coordinate ascent method where we can update all model parameters in time *linear* in the number of edges in the network [41]. This makes our method scale to networks an order of magnitude larger than what was possible by previous methods.

To the best our knowledge, CESNA is the first overlapping community detection method that models both hard node-community memberships and the dependency between the communities and attributes. Moreover, CESNA can detect overlapping, non-overlapping, as well as hierarchically nested communities in networks, while considering *both* node attributes and graph structure.

We evaluate CESNA on six online social, information, and content-sharing networks: Facebook, Google+, Twitter, Wikipedia, and Flickr. We quantify CESNA's accuracy in detecting communities by comparing its predictions to hand-labeled ground-truth communities. We compare CESNA to state-of-the-art community detection methods, including those that detect communities based only on the network structure, methods based only on node attributes, and methods that model both network structure and attributes jointly. Overall, CESNA achieves a 47% improvement in the accuracy of detected communities over the baselines we consider. We also examine whether node attributes can boost the performance of community detection algorithms in cases where the network is noisy or not fully observed. We add noise to the network and we find that the performance gap between CESNA over competing methods increases as the network structure becomes noisier and therefore less reliable. This means that CESNA is able to successfully leverage node attributes to compensate for missing or noisy information in the network structure.

To quantify the scalability of CESNA we measure its running time on synthetic networks of increasing size. Compared to existing methods, the size of networks that CESNA can process far exceeds the current state-of-the-art: CESNA can handle networks 100 times larger than LDA-based methods [4] given the same runtime budget. Even when compared to methods that consider only the network structure (*i.e.*, which handle strictly *less* information), CESNA is faster than most baselines.

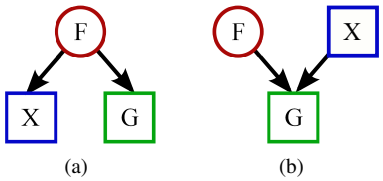


Fig. 1. Two ways of modeling the statistical relationship between a graph G, attributes X, and communities F. Circles represent latent variables that need to be inferred and squares represent manifest (observed) variables.

Last, we also inspect communities detected by CESNA on Facebook networks, and on a network of Wikipedia articles about famous philosophers. We find that, on Facebook data, CESNA automatically learns that education-based attributes ("School name" or "Major") are very highly correlated with a communities, whereas other people's attributes, such as "Work start date" and "Work end date" are not related to community structure. On philosophers data, CESNA learns natural attributes for communities: *e.g.*, subjects about Islamic culture are associated with a community of Islamic philosophers. While methods that ignore node attributes assign very influential philosophers (*e.g.*, Aristotle) to most communities, CESNA circumvents this issue by modeling attributes, and discovering that Aristotle, while well connected to many philosophers, does not share common attributes with all of them.

The rest of the paper is organized as follows. Section II briefly surveys related work. In Section III, we describe the statistical model of CESNA, and in Section IV, we discuss the parameter fitting procedure. We proceed by describing experimental evaluation in Section V and conclude in Section VII.

II. RELATED WORK

We summarize the related work in Table I and group it along two dimensions. First, we consider how the methods model statistical dependency between communities, node attributes, and the underlying network (column D of Table I). Figure 1 shows the two paradigms that are typically used. In Figure 1(a), community memberships F generate both the graph G and attributes X, while in Figure 1(b), F and X are given independently, and then the graph G is generated by the interaction between F and X. Second, we focus on how the methods model the community memberships of individual nodes (columns O and H). Soft-membership models associate a probability distribution with the node's membership to communities, which means the more communities a node belongs to, the less it belongs to each individual community (simply because probabilities have to sum to one). On the other hand, hard-membership models associate an independent binary variable for each node and community pair and, thus, do not suffer from the assumptions made by soft-membership models.

As shown in Table I, heuristic single-assignment clustering methods for networks with node attributes [12], [35], [43] detect hard node-community memberships, however, because

each node can belong to exactly one community, these methods cannot detect overlapping communities.

LDA-based methods [4], [9], [27] aim to find sets of nodes that have similar "topics" of attributes and link among each other. These topic models are based on the paradigm in Figure 1(a) where community memberships nodes generate links and node attributes. However, these methods assume soft community memberships, which leads to unrealistic assumptions about the structure of community overlaps [42]. We note that recently developed methods [38], [40] also assume soft-membership and the paradigm in Fig. 1(a).

III. CESNA Model Description

Here, we develop a probabilistic model that combines community memberships, the network topology, and node attributes. We present the *Communities from Edge Structure and Node Attributes* (CESNA), a probabilistic generative model for networks and node attributes that satisfies the desiderata mentioned above. Our model is based on the following intuitive properties:

- Nodes that belong to the same communities are likely to be connected to each other.
- Communities can overlap, as individual nodes may belong to multiple communities.
- If two nodes belong to multiple common communities, they are more likely to be connected than if they share only a single common community (*i.e.*, overlapping communities are denser [13], [42]).
- Nodes in the same community are likely to share common attributes — for example, a community might consist of friends attending a same school.

We formally describe the generative process of CESNA as follows. We assume that there are N nodes in the network G, each node has K attributes, and there are C communities in total. We denote the network by G, the node attributes by X (X_{uk} is k-th attribute of node u), and community memberships by F. For community memberships F, we assume that each node u has a non-negative affiliation weight $F_{uc} \in [0, \infty)$ to community c. ($F_{uc} = 0$ means that node u does not belong to community c.)

We shall now proceed by describing these components of the model in further detail.

Modeling the links of the network. To model how the network structure depends on node community memberships, we aim to capture the following three intuitions:

1) node community affiliations influence the likelihood that a pair of nodes is connected,
2) the degree of influence (the probability that nodes belonging to the same community are connected) differs per community, and
3) each community influences this connection probability independently.

To achieve these goals, we build on Affiliation Network Models [8], [13], [24], [42], where the graph $G(V, E)$ arises from node community memberships F. To generate the adjacency matrix $A \in \{0,1\}^{N \times N}$ of network G, we employ the probabilistic generative process of the BigCLAM overlapping community detection algorithm [41]. In particular, we assume that two member nodes u, v belonging to a community c are connected with the following probability:

$$P_{uv}(c) = 1 - \exp(-F_{uc} \cdot F_{vc}).$$

Note that if either u or v does not belong to c ($F_{uc} = 0$ or $F_{vc} = 0$), these nodes would not be connected ($P_{uv}(c) = 0$).

We assume that each community c connects nodes u, v independently with probability $1 - \exp(-F_{uc} \cdot F_{vc})$. From this, we can derive the edge probability P_{uv} between nodes u and v. In order for u, v to be unconnected, the nodes u and v should not be connected in *any* community c:

$$1 - P_{uv} = \prod_c (1 - P_{uv}(c)) = \exp(-\sum_c F_{uc} \cdot F_{vc}).$$

In summary, we assume the following generative process for each entry $A_{uv} \in \{0, 1\}$ of the network's adjacency matrix:

$$\begin{aligned} P_{uv} &= 1 - \exp(-\sum_c F_{uc} \cdot F_{vc}), \\ A_{uv} &\sim \text{Bernoulli}(P_{uv}). \end{aligned} \tag{1}$$

Note that the above generative process satisfies our three aforementioned requirements. The network edges are created due to shared community memberships (Requirement (1)). Furthermore, each membership F_{uc} of a node u is regarded as an independent variable to allow a node to belong to multiple communities simultaneously (Requirement (2)). This is in stark contrast to "soft-membership" models (such as mixed membership stochastic block models [2]), which add constraints $\sum_c F_{uc} = 1$ so that F_{uc} is a probability that a node u belongs to a particular community. Finally, because each community c generates connections between its members independently, nodes belonging to multiple common communities have a higher probability of connecting than if they share just a single community (Requirement (3)).

Modeling node attributes. Just as community affiliations can be used to model network edges, they can also be used to model node attributes. We next describe how node attributes are generated from community memberships.

We assume binary-valued attributes where for each attribute X_{uk} of a node u, we consider a separate logistic model. Our intuition is that, based on a node's community memberships, we should be able to predict the value of each of the node's attribute values. Thus, we regard group memberships $F_{u1}, \ldots, F_{uC}$ as input features of the logistic model with the associated logistic weight factor W_{kc} (for each attribute k and community c). We also add an intercept term $F_{u(C+1)} = 1$ to the input feature of each node u:

$$\begin{aligned} Q_{uk} &= \frac{1}{1 + \exp(-\sum_c W_{kc} \cdot F_{uc})}, \\ X_{uk} &\sim \text{Bernoulli}(Q_{uk}) \end{aligned} \tag{2}$$

where W_{kc} is a real-valued logistic model parameter for community c to the k-th node attribute and $W_{k(C+1)}$ is a bias term. The value of W_{kc} represents the relevance of each group membership c to the presence of a particular node attribute k.

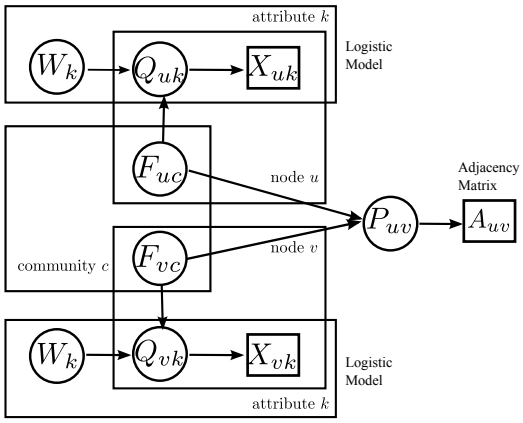


Fig. 2. Plate representation of CESNA. X_{uk}: k-th attribute of node u; W_k: Logistic weight vector for attribute k; Q_{uk}: Probability that $X_{uk} = 1$; F_{uc}: Membership strength of node u to community c; A_{uv}: Indicator for whether the nodes u and v are connected; P_{uv}: Probability that $A_{uv} = 1$.

Figure 2 illustrates the CESNA model. Rectangles (X_{uk}, A_{uv}) are the node attributes and the network adjacency matrix that we observe. Circles denote latent variables: community memberships F and logistic weights W. We explain how to estimate community memberships from node attributes and the network structure (*i.e.*, how we infer F from X and A) in the following section.

Last, we also note that depending on the type of attribute, there are also other choices for modeling attributes X based on F. For example, for real-valued attributes linear regression could be used. Also, note that we assume that the number of attributes is relatively small compared to the number of nodes; as such, we can use a separate logistic model for each attribute. In the case of many attributes, one could consider methods that group attributes as well as nodes [22].

IV. Inferring Communities with CESNA

Next, we shall describe how we detect network communities by estimating CESNA model parameters from given data. We are given an undirected graph $G(V, E)$ with binary node attributes X. We aim to detect C communities as well as the relation between communities and attributes. For now, we shall assume the number of communities C is given. Later, we will describe how to automatically estimate C.

We aim to infer the values of latent variables F and W based on the observed network and the attributes. This means we need to estimate $N \cdot C$ community memberships (*i.e.*, $\hat{F} \in \mathbb{R}^{N \times C}$), and $K \cdot (C+1)$ logistic weight parameters (*i.e.*, $\hat{W} \in \mathbb{R}^{K \times (C+1)}$).

We find the optimal $\hat{F}$ and $\hat{W}$ by maximizing the likelihood $l(F, W) = \log P(G, X|F, W)$ of the observed data G, X:

$$\hat{F}, \hat{W} = \underset{F \geq 0, W}{\operatorname{argmax}} \log P(G, X|F, W). \tag{3}$$

Because G and X are conditionally independent given F and W, we can decompose the log-likelihood $\log P(G, X|F, W)$ as follows:

$$\log P(G, X|F, W) = \mathcal{L}_G + \mathcal{L}_X$$

where $\mathcal{L}_G = \log P(G|F)$ and $\mathcal{L}_X = \log P(X|F, W)$. We compute $\mathcal{L}_G$ and $\mathcal{L}_X$ simply using Equations 1 and 2:

$$\mathcal{L}_G = \sum_{(u,v) \in E} \log(1 - \exp(-F_u F_v^T)) - \sum_{(u,v) \notin E} F_u F_v^T$$
$$\mathcal{L}_X = \sum_{u,k} (X_{uk} \log Q_{uk} + (1 - X_{uk}) \log(1 - Q_{uk})),$$

where F_u is a vector $\{F_{uc}\}$ for node u and Q_{uk} is defined in Equation 2.

Last, we also invoke l_1-regularization on W to avoid overfitting and to learn sparse relationships between communities and attributes. Thus, our optimization problem that we aim to solve is:

$$\hat{F}, \hat{W} = \underset{F \geq 0, W}{\operatorname{argmax}} \mathcal{L}_G + \mathcal{L}_X - \lambda |W|_1, \tag{4}$$

where λ is a regularization hyperparameter.

To solve the problem in Eq. 4, we adopt a block coordinate ascent approach. We update F_u for each node u by fixing both W and the community membership F_v of all other nodes v. After updating F_u for all nodes, we then update W while fixing community memberships F. This way, we can decompose the non-convex optimization problem of Eq. 4 into a set of *convex* subproblems. We describe our solution to each of these subproblems next.

Updating community memberships. To update community memberships, we build on the optimization procedure used in BigCLAM [41]. However, we modify the procedure to consider node attributes (which BigCLAM ignores). We update the membership F_u of an individual node u while fixing all other parameters (the membership F_v of all other nodes, and logistic model parameters W).

We solve the following subproblem for each u:

$$\hat{F_u} = \underset{F_{uc} \geq 0}{\operatorname{argmax}} \mathcal{L}_G(F_u) + \mathcal{L}_X(F_u), \tag{5}$$

where $\mathcal{L}_G(F_u)$ and $\mathcal{L}_X(F_u)$ are the parts of $\mathcal{L}_G, \mathcal{L}_X$ involving F_u, *i.e.*,

$$\mathcal{L}_G(F_u) = \sum_{v \in \mathcal{N}(u)} \log(1 - \exp(-F_u F_v^T)) - \sum_{v \notin \mathcal{N}(u)} F_u F_v^T$$
$$\mathcal{L}_X(F_u) = \sum_{k} (X_{uk} \log Q_{uk} + (1 - X_{uk}) \log(1 - Q_{uk}))$$

where $\mathcal{N}(u)$ is a set of neighbors of u. Note that this problem is convex: $\mathcal{L}_G(F_u)$ is a concave function of F_u [41], [30] and $\mathcal{L}_X(F_u)$ is a logistic function of F_{uc} when W is fixed.

To solve this convex problem, we use projected gradient ascent. The gradient can be computed straightforwardly:

$$\frac{\partial \mathcal{L}_G(F_u)}{\partial F_u} = \sum_{v \in \mathcal{N}(u)} F_{vc} \frac{\exp(-F_u F_v^T)}{1 - \exp(-F_u F_v^T)} - \sum_{v \notin \mathcal{N}(u)} F_{vc}$$
$$\frac{\partial \mathcal{L}_X(F_u)}{\partial F_u} = \sum_{k} (X_{uk} - Q_{uk}) W_{kc}.$$

We then update each F_{uc} by gradient ascent and then project onto a space of non-negative real numbers $[0, \infty)$:

$$F_{uc}^{new} = \max(0, F_{uc}^{old} + \alpha(\frac{\partial \mathcal{L}_G(F_u)}{\partial F_u} + \frac{\partial \mathcal{L}_X(F_u)}{\partial F_u})) \tag{6}$$

where α is a learning rate which we set using backtracking line search [7].

Updating logistic parameters. We update parameters W of the logistic model by keeping community memberships F fixed. To compute this, we first notice that we can ignore $\mathcal{L}_G$ in Eq. 4, as G does not depend on W. Next, we also include l_1-regularization on W, as we aim to learn *sparse* relationships between community memberships and node attributes:

$$\hat{W} = \operatorname*{argmax}_W \sum_{u,k} \log P(X_{uk}|F, W) - \lambda|W|_1.$$

Furthermore, as we employ an independent logistic model for each attribute, we only need to consider the k-th attribute when updating the weight vector W_k:

$$\operatorname*{argmax}_{W_k} \sum_u \log P(X_{uk}|F, W_k) - \lambda|W_k|_1. \tag{7}$$

Note that this is l_1-regularized logistic regression with input features F and output variable X. Again, we simply apply a gradient ascent method:

$$\frac{\partial \log P(X_{uk}|F, W_k)}{\partial W_{kc}} = (X_{uk} - Q_{uk})F_{uc},$$

$$W_{kc}^{new} = W_{kc}^{old} + \alpha(\sum_u \frac{\partial \log P(X_{uk}|F, W_k)}{\partial W_{kc}} - \lambda \cdot \text{Sign}(W_{kc})),$$

where α is a step size as in Eq. 6.

Now, we iteratively update F_u for each u and then update W_k for each attribute k. We stop iterating once the likelihood does not increase (by at least 0.001%) after a full iteration over all F_u and all W_k.

Determining community memberships. After learning real-valued community affiliations $\hat{F}$, we need to determine whether node u belongs to community c. To do so, we regard u as belonging to c only if the corresponding F_{uc} is above the threshold δ. We set δ so that a node belongs to community c if the node is connected to other members of c with an edge probability higher than $1/N$. To determine δ, we need to solve:

$$\frac{1}{N} \leq 1 - \exp(-\delta^2).$$

Solving this inequality, we set the value of $\delta = \sqrt{-\log(1 - 1/N)}$. We have also experimented with other values of δ and found that this value of δ gives good performance in practice.

Choosing the number of communities. To automatically find the number of communities C, we adopt the approach used in [2]. We reserve 10% of node pairs in the adjacency matrix and node-attribute pairs as a holdout set. Varying C, we fit the CESNA with C communities on 90% of node-node pairs and node-attribute pairs and then evaluate the likelihood of CESNA on the holdout set. The K that induces the maximum held-out likelihood will be chosen as the number of communities.

Computational complexity of CESNA. We next analyze the computational complexity of CESNA. In particular, we show that a full iteration of CESNA takes time *linear* in the number of edges and attributes.

For simplicity, let us assume a single community $C = 1$, then updating F_u for a single u takes $N + K$ operations when computed in a naive way. However, we can compute $\frac{\partial \mathcal{L}_G(F_u)}{\partial F_u}$ in $O(|\mathcal{N}(u)|)$. This means that the number of operations required to compute the gradient is proportional to the degree of node u since [30], [41]:

$$\sum_{v \notin \mathcal{N}(u)} F_{vc} = (\sum_v F_{vc} - F_{uc} - \sum_{v \in \mathcal{N}(u)} F_{vc}).$$

By storing $\sum_v F_{vc}$, the second term in $\frac{\partial \mathcal{L}_G(F_u)}{\partial F_u}$ can be computed in $O(|\mathcal{N}(u)|)$. Therefore, updating F_u for all nodes u takes $O(|E| + NK)$ operations. Because updating W_k takes just $O(N)$ for each k, a full iteration of CESNA takes $O(|E| + NK)$ operations, which is linear in the number of edges, nodes and the number of attributes.

Notice that CESNA nicely lends itself to parallelization. In particular, updating W_k naturally allows for parallelization, as we can update W_k for multiple attributes k simultaneously. Because F is fixed, the problems in Eq. 7 are independent for different attributes k. We also update F_u for multiple nodes u in parallel. In this case, updating each u is not necessarily independent for different nodes u. However, as shown by Niu et al. [32], updating F_u in parallel works well in practice, as networks tend to be *sparse*. As we show in the next section, parallelization on a single shared memory machine boosts the speed of CESNA by a factor of 20 (the number of threads).

A parallel C++ implementation of CESNA algorithm is available as a part of the Stanford Network Analysis Platform (SNAP): http://snap.stanford.edu/snap.

CESNA hyperparameter settings. To initialize F, we use locally minimal neighborhoods [17]. A neighborhood $N(u)$ of a node u is *locally minimal* if $N(u)$ has lower conductance than all neighborhoods $N(v)$ of u's neighbors v. Locally minimal neighborhoods have been shown to be a good initialization for community detection methods [17].

Last, notice that the overall model likelihood is a combination of the network likelihood $\mathcal{L}_G$ and the likelihood of node attributes $\mathcal{L}_X$ (Eq. 4). As the two likelihoods can have vastly different ranges we scale them using the parameter α. In particular, we introduce a hyperparameter α that controls the scaling between the two likelihoods:

$$\operatorname*{argmax}_{F \geq 0, W} (1 - \alpha)\mathcal{L}_G + \alpha\mathcal{L}_X - \lambda|W|_1.$$

We choose values of hyperparameters α and λ among $\alpha \in \{0.25, 0.5, 0.75\}, \lambda \in \{0.1, 1.0\}$ based on the held-out data likelihood (*i.e.*, by cross-validation). We note that the performance of CESNA does not change much with the values of hyperparameters. Setting $\alpha = 0.5$ (*i.e.*, the unscaled version of Eq. 4) and $\lambda = 1$ gives reliable performances in most cases.

V. Experimental Evaluation

We quantify the performance of CESNA by comparing it to state-of-the-art community detection methods in various social and information networks. We evaluate the performance of the methods by evaluating the accuracy of the detected communities when compared to the gold-standard, ground-truth

Dataset	N	E	C	K	S	A
Facebook	4,089	170,174	193	175	28,76	1.36
Google+	250,469	30,230,905	437	690	143.51	0.25
Twitter	125,120	2,248,406	3,140	33,569	15.54	0.39
Philosophers	1,218	5,972	1,220	5,770	6.86	6.87
Flickr	16,710	716,063	100,624	1,156	28.91	174.08

TABLE II. DATASET STATISTICS. N: NUMBER OF NODES, E: NUMBER OF EDGES, C: NUMBER OF COMMUNITIES, K: NUMBER OF NODE ATTRIBUTES, S: AVERAGE COMMUNITY SIZE, A: COMMUNITY MEMBERSHIPS PER NODE.

communities. We also evaluate the scalability by measuring the running time as the network size grows.

Dataset description. For our evaluation, we consider five datasets where we have network information as well as node attributes. In addition to networks and attributes, we also have access to explicit *ground-truth* community labels. The availability of such ground-truth allows us to evaluate community detection methods by quantifying the degree of agreement between the detect and the ground-truth communities [35]. Table II lists the networks and their properties.

The networks come from 3 different domains: information network among Wikipedia articles (philosophers) [1], content-sharing network (Flickr) [35], and ego-networks from online social network services (Facebook, Google+, and Twitter) [28]. We next describe each of these networks in further detail.

The philosophers network [1] consists of Wikipedia articles about famous philosophers. Nodes represent Wikipedia articles about philosophers, and undirected edges indicate whether one article links to another. For the attributes of each node u, we use a binary indicator vector of out-links from node u to other non-philosopher Wikipedia articles. For example, we regard a link to a Wikipedia article "Edinburgh" as a binary attribute "Edinburgh." We consider 5,770 attributes, to which at least five philosophers have a link. Moreover, Wikipedia also provides categories (*e.g.*, "Muslim philosophers", or "Early modern philosophers") for each article. We regard each category with more than five philosophers as a ground-truth community.

The Flickr image sharing network [35] consists of nodes which represent Flickr users, and edges indicate follow relations between users. We use tags of images uploaded by a given user as her attributes. In this network, the ground-truth communities are defined as user-created interest-based groups that have more than five members.

The last three networks (Facebook, Google+, and Twitter) are ego-networks that are available from the Stanford Large Network Dataset Collection (http://snap.stanford.edu/data). To obtain ground-truth communities and node attributes, we use the same protocol as in [28]. Ground-truth communities are defined by social circles (or "lists" in Twitter), which are manually labeled by the owner of the ego-network. In Facebook and Google+, node attributes come from user profiles, such as gender, job titles, institutions, and so on. In Twitter, node attributes are defined by hashtags used by the user in her tweets. To reduce the dimensionality of the node attributes, we discard any attribute which the owner of the ego-network does not possess.

Baselines for comparison. We consider the three classes of baseline community detection methods: (1) methods that use only the network structure, (2) methods that user only node attributes, and (3) methods that combine both.

The first class of baselines considers only the network, ignoring node attributes altogether: *Demon* [10] and *BigCLAM* [41] are state-of-the-art overlapping community detection methods.

Second is a class of baselines that focuses on node attributes without considering the network structure. Here, we use *Multi Assignment Clustering (MAC)* [15], which detects overlapping communities based on node attributes alone.

The third class of baselines we consider combines the network structure with node attributes. For this class, we choose three state-of-the-art methods. Based on Table I we select one algorithm from each model type: *Block-LDA* [4] represents soft-membership approaches, while the *CODICIL* [35] represents heuristics for non-overlapping communities, and the *EDCAR* [18] represents heuristics for finding dense subgraphs. Finally, we consider the *Circles* [28] method, which represents overlapping hard-membership approaches.

For all baselines, we use implementations provided by the authors. All baselines except CODICIL require a user to specify the number of communities to detect. We set this parameter so that each model detects the same number of communities as CESNA. CODICIL and EDCAR also has other input parameters, for which we used default values provided by the authors.

Evaluation metrics. We quantify the performance in terms of the agreement between the ground-truth communities and the detected communities. To compare a set of ground-truth communities C^* to a set of detected communities C, we adopt an evaluation procedure previously used in [41]: Every detected community is matched with its most similar ground-truth community. Given this matching, we then compute the performance. We also then take every ground-truth community and match it with a detected community and again compute the performance. Our final performance is the average of these two metrics. We average the two scores because matching only from one side leads to degenerate optimal performance (for example, outputting all possible subsets of nodes as detected communities would achieve perfect matching ground-truth communities to the detected ones).

More formally, our evaluation function is:

$$\frac{1}{2|C^*|}\sum_{C_i^*\in C^*}\max_{C_j\in C}\delta(C_i^*,C_j)+\frac{1}{2|C|}\sum_{C_j\in C}\max_{C_i^*\in C^*}\delta(C_i^*,C_j), \tag{8}$$

where $\delta(C_i^*, C_j)$ is some similarity measure between the communities C_i^* and C_j. We consider two standard metrics $\delta(\cdot)$ for quantifying the similarity between a pair of sets, namely the $F1$ score and the Jaccard similarity. Thus, for each method, we obtain a score between 0 and 1, where 1 indicates the perfect recovery of ground-truth communities.

Experiments on recovering ground-truth communities. We evaluate the performance of CESNA and baselines on our five datasets. Table III shows the results where "N/A" means that the method cannot scale to a given network. We make several observations.

Method	Info	$F1$ score					Jaccard similarity					
		Phil	Flickr	Facebook	Google+	Twitter	Phil	Flickr	Facebook	Google+	Twitter	Avg.
Demon	Net	0.244*	0.171*	0.386*	0.323*	0.280*	0.143*	0.098*	0.283*	0.234	0.186*	0.235*
BigCLAM	Net	0.276*	0.166*	0.455	0.341	0.359*	0.156*	0.092*	**0.347**	0.231	0.246*	0.267*
MAC	Attr	0.117*	N/A	0.297*	0.159*	0.246*	0.069*	N/A	0.190*	0.101*	0.154*	0.133*
Block-LDA	Both	0.146*	N/A	0.356*	0.307	0.273*	0.082*	N/A	0.241*	0.204*	0.173*	0.178*
CODICIL	Both	0.277*	0.132*	0.378*	0.247*	0.279*	0.167*	0.079*	0.263*	0.166*	0.190*	0.218*
EDCAR	Both	0.264*	0.112*	0.321*	0.135*	0.258*	0.157*	0.051*	0.222*	0.081*	0.165*	0.177*
Circles	Both	N/A	N/A	0.401*	**0.365**	0.319*	N/A	N/A	0.265*	**0.254**	0.211*	0.183*
CESNA	Both	**0.314**	**0.183**	**0.462**	0.352	**0.362**	**0.192**	**0.106**	**0.347**	0.249	**0.249**	**0.282**

TABLE III. PERFORMANCE OF METHODS ON FIVE DATASETS. *Info* INDICATES THE INFORMATION USED BY A GIVEN METHOD (NETWORK, ATTRIBUTES, OR BOTH). BEST PERFORMING MODELS ARE BOLDED. SYMBOL * INDICATES THAT CESNA OUTPERFORMS A GIVEN BASELINE BY 95% STATISTICAL CONFIDENCE. OVERALL, CESNA STATISTICALLY SIGNIFICANTLY OUTPERFORMS ALL CONSIDERED METHODS.

Comparing CESNA to methods without the node attributes (Demon and BigCLAM), we notice that CESNA achieves better performance, as it combines the information from the node attributes as well as the network. Similarly, CESNA also outperforms MAC, which only focuses on node attributes. In particular, CESNA never performs worse than state-of-the-art methods that use only a single source of data. The strong performance of CESNA is not obvious, as it would be entirely possible that combining two sources of data would confuse the algorithm and degrade the overall performance (in fact, notice that BigCLAM, which uses only the network structure, indeed outperforms most of the methods that use both sources of information). Thus, we believe that the strong performance of CESNA as an indication that CESNA combines the best ingredients from both worlds.

When comparing the performance of CESNA to methods that consider both the network structure and node attributes (CODICIL, Block-LDA, and Circles), we again observe the strong performance of CESNA. On average, CESNA gives 47% relative improvement in the accuracy of detected communities over methods that consider both sources of information.

We also note that CESNA shows a bigger margin in performance against the baselines in an information network such as the philosophers dataset, or a content-sharing network like Flickr than in social networks. In the philosophers network, for example, CESNA achieves a 14% relative gain in the $F1$ score and 15% in the Jaccard similarity compared to the best baseline. A possible explanation for this phenomenon is that in content-sharing and information networks, the properties/content of the nodes plays a much bigger role in link formation.

Overall, we note that across all datasets and evaluation metrics, CESNA yields the best performance in 8 out of 10 cases. In terms of average performance, CESNA outperforms Demon by 20%, BigCLAM by 6%, MAC by 112%, Block-LDA by 58%, CODICIL by 29%, EDCAR by 57%, and Circles by 54%.

Last, we also measure the statistical significance of performance differences of CESNA and the baselines. For each baseline's performance on each data set, we compute the statistical significance of CESNA outperforming the baseline using a one-sided Z-test. We use the symbol * in Table III to indicate a 95% statistical significance level. On the philosophers, Flickr, and Twitter datasets, CESNA outperforms every baseline at a 95% significance level. On Facebook, CESNA outperforms all baselines, at a 95% significance level in all but one case. On Google+, CESNA performs the second best compared to Circles.

Experiments on partially observed networks. Combining network and attribute information into a single method should, in principle, lead to the development of a more robust community detection algorithm. In particular, when networks may be incomplete or partially observed, the performance of CESNA should degrade gently, as it should be able to rely on the node attribute information; this way, it should compensate for the noise in the network structure.

To investigate the robustness of performance under an unreliable network structure, we next explore the problem of detecting communities from partially observed networks where some fraction of edges are missing while the node attributes are fully available. For the sake of evaluation, we remove a fraction γ of edges in the network uniformly at random. Note that we regard a removed edge in the same way as an unobserved edge, because in practice we cannot distinguish between edges that do not exist (*e.g.*, users who aren't friends) and edges that are unobserved (*e.g.*, users who haven't gotten around to declaring their friendship yet).

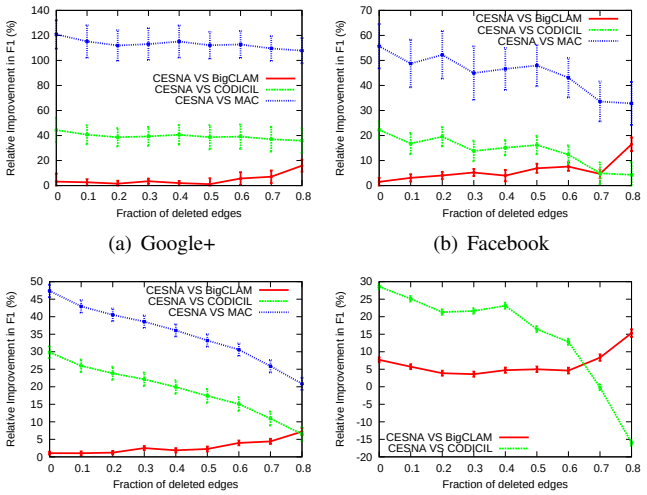
Rather than examining performance of all 6 baselines, we focus on making a comparison over the three top baselines that use either the network or the node attributes: BigCLAM, which considers the network only and is the best baseline in our experiments; MAC, which only considers the node attributes; and CODICIL, which is the best performing baseline that considers both the network and the attributes. For each baseline, we measure the relative performance that CESNA achieves over the baseline:

$$\frac{F1^{\gamma}(\text{CESNA}) - F1^{\gamma}(\text{Baseline})}{F1^{\gamma}(\text{Baseline})}$$

where $F1^{\gamma}$ is the $F1$ score in Eq. 8 for the network with γ fraction of edges removed.

In Figure 3, we display experimental results (with standard deviation) as we vary from $\gamma = 0$ to $\gamma = 0.8$. We consider all datasets except philosophers (for which, results are too noisy due to the small network size). For Flickr, we omit performance of MAC, as the algorithm was not able to process it due to too high time and space complexity.

In all cases, we note similar behavior (Figure 3). As the network becomes more unreliable, the improvement of CESNA over BigClam increases. On the other hand, for methods that use node attributes (and the network structure), we note that in Google+, the performance improvement of CESNA remains constant, while in Facebook and Twitter, the performance improvement of CESNA slowly shrinks as more and more of the network structure gets removed.



(a) Google+ (b) Facebook

(c) Twitter (d) Flickr

Fig. 3. Relative gain in F1 over the best method with network information only (BigCLAM), the method with node attributes only (MAC), and the method with both network and attributes (CODICIL) when edges are randomly removed.

The results are intuitive: Even though the network contains many missing edges, CESNA still outperforms other methods by better leveraging the information present in the node attributes. The results with MAC and CODICIL, which are decreasing functions of γ, nicely shows that the performance gain from the network structure diminishes as we remove more edges.

Last, we also briefly note that similar results are observed with the relative improvement in Jaccard similarity, and that CESNA consistently outperforms the other four baselines not shown in Figure 3 for every value of γ.

Evaluating scalability. We evaluate the scalability of community detection methods by measuring each method's running time on synthetic networks as we increase the network size. Using the Forest Fire model [25], we generate synthetic networks with the forward and backward probabilities set to 0.36 and 0.32, respectively. For attributes, we generate $K = 10$ attributes for each node with independent Bernoulli random variables with probability 0.5.

Figure 4 shows the running time of methods versus the network size. Among the four baselines that consider both network and the node attributes (*i.e.*, Block-LDA, CODICIL, EDCAR, Circles), we show CODICIL since it is the fastest among the four. We also consider a parallelized version of CESNA (CESNA (24 threads)).

Overall, we notice that CESNA is the second-fastest method overall, next to BigCLAM. However, we note that BigCLAM is expected to be faster than CESNA, as it uses a similar optimization procedure as CESNA yet without considering node attributes. MAC is the slowest, and CODICIL is the second-slowest method. DEMON is faster than CESNA for small networks (up to 100,000 nodes), though CESNA is faster when the network becomes larger.

We obtain even further speedup by considering a parallel implementation of CESNA. Using 24 threads on a single machine, CESNA takes just 10 minutes to process a 300,000 node network.

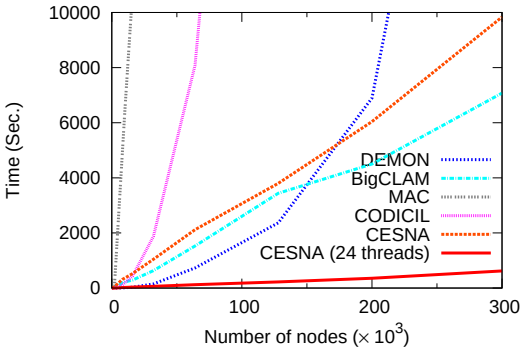


Fig. 4. Algorithm runtime comparison. Block-LDA and Circles are omitted as they took more time than 10,000 seconds for networks larger than 1% of the X-axis (3,000 nodes).

Last, we also note that all the baselines shown in Fig. 4 solve "simpler" problems than CESNA. For example, CODICIL detects non-overlapping communities, which is simpler than detecting overlapping communities. Demon and BigCLAM consider only network information, ignoring node attributes. Nevertheless, CESNA is faster than CODICIL and Demon, and it takes about 30% more time than BigCLAM. Comparing CESNA to methods that achieve the same goal — that is, overlapping community detection with node attributes (*i.e.*, Block-LDA, EDCAR, and Circles) — CESNA has a considerable advantage in scalability, as it is about an order of magnitude faster.

VI. Analysis of Detected Communities

Incorporating node attributes into community detection gives two direct advantages. The first advantage is the improved accuracy in community detection, which we observed in the previous section. The second advantage is that the node attributes provide cues for interpreting detected communities. For example, a community in a Facebook ego-network might consist of a set of high-school friends, and the homogeneity of a particular attribute in a given community might help us to interpret and explain its existence. Such interpretations are an important part of community detection [1], [2], [33], yet finding them is very time-consuming and may require domain knowledge, as in traditional settings, one has to infer the meaning of a given community based only on the identities of its members. By incorporating node attributes, however, CESNA allows us to characterize a community by examining the attributes associated with high logistic weights in the model.

In this section, we qualitatively analyze our results in the Facebook network and the philosophers network to provide insights as to how CESNA brings the two advantages (better interpretability and higher accuracy). In both networks, we find that CESNA is able to find the attributes that are naturally related to the communities. On philosophers data, we also show how CESNA can improve the accuracy of detected communities by incorporating node attributes.

Analysis of Facebook communities. CESNA learns the logistic model weight W_{kc} for each attribute k and community

c. Highly positive values of W_{kc} mean that members of community c are likely to have attribute k, and a highly negative value means the opposite (members are likely *not* to have the attribute). Not every attribute will be associated with community memberships, as some attributes may be irrelevant for a given community. To characterize the level of association between communities and attributes k, we measure the l_2 norm $\|W_k\|$ of its logistic weight $W_k = \{W_{k\cdot}\}$.

To examine which attributes are related to communities (either positively or negatively), we examine detected communities in Facebook ego-networks. We find that the top attributes are related to schools, including the schools attended, the types of education that users received, and the major. On the other hand, the bottom five attributes include work start dates, work end dates, and locale. None of them act as social factors around which communities on Facebook form.

Analysis of Philosophers communities. To analyze the member nodes of communities along with their related attributes, we examine the communities in the Philosophers network.

First, using CESNA, we identify communities, and then for each community we identify the top ten positively related attributes. In Figures 5(a), 5(c) we show two of the detected communities. The figure displays the titles of the corresponding Wikipedia articles. Moreover, we also show the attributes associated with the two communities in Figure 6. In this figure, word sizes are proportional to the value of the logistic weight W_{kc}, *i.e.*, more relevant attributes are larger. Note that node attributes in this network represent Wikipedia articles other than philosophers to which the node links, *e.g.*, the attributes include famous non-philosophical figures, abstract concepts, historic events, places, and so on.

First, based on the names of important attributes, *e.g.*, "Early Islamic Philosophy," we observe that the community in Figure 6(a) represents Islamic philosophers, even without querying for the names of the philosophers in Figure 5(a). These attributes also include some non-philosophical people related to Islam (*e.g.*, René Guénon).

Similarly, Figures 5(c) and 6(b) show the members of the second community detected by CESNA and the top ten related node attributes. Again, "Catechism of the Catholic Church" tells us that this community consists of theologians. The node attributes also include many priests (*e.g.*, Lawrence of Brindisi, Bede, Hilary of Poitiers, Petrus Canisius, and Francis de Sales).

We also compare these communities to those detected by the BigCLAM. For each community detected by CESNA in Figs. 5(a) and 5(c), we identify the most similar BigCLAM community based on the F_1 score. Figures 5(b) and 5(d) show these communities as detected by BigCLAM.

Interestingly, we note that the communities detected by BigCLAM contain some philosophers (in red) who are not Islamic philosophers/theologians. The reason is that these philosophers (in red) are so influential that they are very well connected to other members of the community. For example, Aristotle is connected to 229 philosophers (about one fifth of all the nodes); thus, he appears in both BigCLAM communities in Figure 5. However, by leveraging node attributes, CESNA does not make this mistake and finds that Aristotle does not share the same attributes as any Islamic philosophers or theologians, which, thus, excludes him.

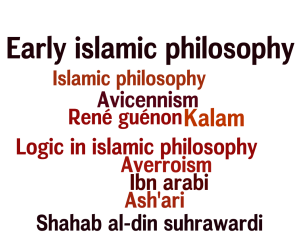
(a) Islamic (CESNA)

(b) Islamic (BigCLAM)

(c) Theologians (CESNA)

(d) Theologians (BigCLAM)

Fig. 5. Communities of philosophers found by CESNA (left) and equivalent communities detected by BigCLAM (right). Top: Community of Islamic philosophers, Bottom: Community of theologians, BigCLAM regards some notable philosophers in red letters as belonging to the communities, even though these philosophers have little to do with theology / Islam. CESNA does not make such mistakes, as CESNA jointly learns attributes associated with the community. (Attributes are in Fig. 6.)



(a) Islamic (Attributes)

(b) Theologians (Attributes)

Fig. 6. The node attributes which CESNA learns to be associated with the communities. Left: For the community of Islamic philosophers, Right: For the community of theologians,

VII. Conclusion

In this paper, we developed CESNA, a scalable method for overlapping community detection in networks with node attributes. Its comparison to the state-of-the-art baselines reveals that CESNA exhibits improved performance both in terms of the accuracy of the detected communities as well as in scalability. CESNA has a linear runtime in the network size and is able to process networks an order of magnitude larger than comparable approaches. Moreover, CESNA also helps with the

interpretation of detected communities by finding relevant node attributes for each community.

There are many possible directions for future work. One direction is to extend CESNA to handle more general types of attributes. Similarly, extending the method to cluster the attributes into "topics," while also identifying communities would likely lead to even easier interpretation of detected communities. Finally, incorporating other sources of information than node attributes, such as information diffusion [5] or edge attributes [4], would also be possible.

Acknowledgements. We thank Yiye Ruan for sharing the CODICIL code and the Flickr data. This research has been supported in part by NSF IIS-1016909, CNS-1010921, CAREER IIS-1149837, IIS-1159679, ARO MURI, DARPA GRAPHS, ARL AHPCRC, Okawa Foundation, Docomo, Boeing, Allyes, Volkswagen, Intel, Alfred P. Sloan Fellowship, and the Microsoft Faculty Fellowship.

REFERENCES

[1] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multi-scale complexity in networks. *Nature*, 2010.

[2] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *JMLR*, 2007.

[3] L. Akoglu, H. Tong, B. Meeder, and C. Faloutsos. PICS: Parameter-free Identification of Cohesive Subgroups in Large Attributed Graphs. *SDM '12*, 2012.

[4] R. Balasubramanyan and W. W. Cohen. Block-lda: Jointly modeling entity-annotated text and entity-entity links. In *SDM '11*, 2011.

[5] N. Barbieri, F. Bonchi, and G. Manco. Cascade-based community detection. In *WSDM '13*, 2013.

[6] D. M. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*, 2004.

[8] R. L. Breiger. The duality of persons and groups. *Social Forces*, 1974.

[9] J. Chang and D. M. Blei. Relational topic models for document networks. In *AISTATS '09*, 2009.

[10] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi. Demon: a local-first discovery method for overlapping communities. In *KDD '12*, 2012.

[11] D. Crandall, D. Cosley, D. Huttenlocher, J. Kleinberg and S. Suri. Feedback effects between similarity and social influence in online communities. In *KDD '08*, 2008.

[12] M. Ester, R. Ge, B. Gao, Z. Hu, and B. Ben-Moshe. Joint Cluster Analysis of Attribute Data and Relationship Data: the Connected k-Center Problem. In *SDM '06*, 2006.

[13] S. L. Feld. The focused organization of social ties. *American J. of Sociology*, 1981.

[14] S. Fortunato. Community detection in graphs. *Physics Reports*, 2010.

[15] M. Frank, A. P. Streich, D. Basin, and J. M. Buhmann. Multi-assignment clustering for boolean data. *JMLR*, Mar. 2012.

[16] M. Girvan and M. Newman. Community structure in social and biological networks. *PNAS*, 2002.

[17] D. F. Gleich and C. Seshadhri. Neighborhoods are good communities. In *KDD '12*, 2012.

[18] S. Günnemann, B. Boden, I. Färber, and T. Seidl Efficient Mining of Combined Subspace and Subgraph Clusters in Graphs with Feature Vectors. In *PAKDD '13*, 2013.

[19] S. Günnemann, I. Färber, B. Boden, and T. Seidl Subspace Clustering Meets Dense Subgraph Mining: A Synthesis of Two Paradigms. In *ICDM '10*, 2010.

[20] S. Johnson. Hierarchical clustering schemes. *Psychometrika*, 1967.

[21] G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *J. of Parallel and Distributed Computing*, 1998.

[22] M. Kim and J. Leskovec. Latent multi-group membership graph model. In *ICML '12*, 2012.

[23] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. *PLoS ONE*, 2011.

[24] S. Lattanzi and D. Sivakumar. Affiliation networks. In *STOC '09*, 2009.

[25] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD '05*, 2005.

[26] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM '03*, 2003.

[27] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link lda: joint models of topic and author community. In *ICML '09*, 2009.

[28] J. McAuley and J. Leskovec. Learning to discover social circles in ego networks. In *NIPS '12*, 2012.

[29] K. Miller, T. Griffiths, and M. Jordan. Nonparametric Latent Feature Models for Link Prediction. In *NIPS '09*, 2009.

[30] M. Mørup, M. N. Schmidt, and L. K. Hansen. Infinite multiple membership relational modeling for complex networks. *CoRR*, 2011.

[31] F. Moser, R. Colak, A. Raey, and M. Ester. Mining Cohesive Patterns from Graphs with Feature Vectors. In *SDM '09*, 2009.

[32] F. Niu, B. Recht, C. Ré, and S. J. Wright. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *NIPS '11*, 2011.

[33] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 2005.

[34] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *PNAS*, 2008.

[35] Y. Ruan, D. Fuhry, and S. Parthasarathy. Efficient community detection in large networks using content and links. In *WWW '13*, 2013.

[36] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. In *KDD '09*, 2009.

[37] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *KDD '09*, 2009.

[38] Y. Sun, C. Aggarwal, and J. Han. Relation Strength-Aware Clustering of Heterogeneous Information Networks with Incomplete Attributes. In *VLDB '12*, 2012.

[39] J. Xie, S. Kelley, and B. K. Szymanski. Overlapping community detection in networks: the state of the art and comparative study. *ACM Computing Surveys*, 2013.

[40] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng. A model-based approach to attributed graph clustering In *SIGMOD '12*, 2012.

[41] J. Yang and J. Leskovec. Overlapping community detection at scale: A non-negative factorization approach. In *WSDM '13*, 2013.

[42] J. Yang and J. Leskovec. Structure and overlaps of communities in networks. *ACM TIST*, 2013.

[43] Z. Yang, H. Cheng, and J. Yu. Graph clustering based on structural/attribute similarities. *VLDB '09*, 2009.