Influence-based Network-oblivious Community Detection

Nicola Barbieri
Yahoo Labs, Barcelona, Spain
barbieri@yahoo-inc.com

Francesco Bonchi
Yahoo Labs, Barcelona, Spain
bonchi@yahoo-inc.com

Giuseppe Manco
ICAR-CNR, Rende, Italy
manco@icar.cnr.it

***Abstract*—How can we detect communities when the social graphs is not available? We tackle this problem by modeling social contagion from a log of user activity, that is a dataset of tuples (u, i, t) recording the fact that user u "adopted" item i at time t. This is the only input to our problem.**

We propose a stochastic framework which assumes that item adoptions are governed by un underlying diffusion process over the unobserved social network, and that such diffusion model is based on community-level influence. By fitting the model parameters to the user activity log, we learn the community membership and the level of influence of each user in each community. This allows to identify for each community the "key" users, i.e., the leaders which are most likely to influence the rest of the community to adopt a certain item.

The general framework can be instantiated with different diffusion models. In this paper we define two models: the extension to the community level of the classic (discrete time) *Independent Cascade* model, and a model that focuses on the time delay between adoptions.

To the best of our knowledge, this is the first work studying community detection without the network.

I. Introduction

While the literature on community detection methods is wide (see, e.g., [1]), the applications are mainly limited to simple data analysis for social sciences. This is due to a fundamental observation: the players which would benefit more from knowing the community structure of a social network, e.g., companies advertising or developing web applications, often *do not have access to the whole network!* It is a matter of fact that the social network platforms are owned by third party such as Facebook or Twitter, which have realized that their proprietary social graph is an asset of inestimable value [2]. Thus they keep it secret, for sake of commercial competitive advantage, as well as due to privacy legislation.

In this paper we tackle the ambitious problem of inferring the community structure when the social graph is not available. A first step towards this goal, is to analyze the alternative dynamics and data that we can exploit. A company advertising or developing applications over an on-line social network owns the log of user activity that it produces. In general, we might think of the activity log $\mathbb{D}$ as a set of tuples (u, i, t) which records the timestamp at which the user u "acted on" or "adopted" the item i: for instance, user u bought song i, user u clicked on ad i, user u rated movie i, user u liked photo i.

The key idea at the basis of this paper, is to exploit the phenomenon of social contagion to detect communities by analyzing, exclusively, the activity log $\mathbb{D}$.

The basic assumptions are that (i) information can spread only by exploiting the social connections among users, and that (ii) the network has a community structure, where communities are densely connected internally, and loosely connected externally. As a consequence, social contagion acts mainly locally inside each community: if we see a group of users acting on item i in a short time frame, and we observe this occurring on various different items, then we can infer that these users are connected in some social network, that they communicate and can influence each other.

One possible approach to solve the network-oblivious community detection problem could be to first use the log $\mathbb{D}$ to infer the overall structure of the network, and then apply some standard community detection techniques over the reconstructed social network. However, methods for network reconstruction [3], [4] are inherently quadratic in the number of nodes and thus not easily scalable. Moreover, if the community detection is the ultimate goal then, as we show in our experiments in Section VI, it is more effective to go directly for it, without passing from network reconstruction.

In this paper, we propose a general framework for directly detecting communities in a network-oblivious setting, without attempting to reconstruct the network. In particular, our proposal assumes that item adoptions are governed by un underlying stochastic diffusion process over the unobserved social network, and that such diffusion model is based on community-level influence. By fitting the model parameters to the user activity log $\mathbb{D}$, we learn the community membership and the influence level of each user in each community.

The general framework can be instantiated with different choices of diffusion models. In this paper we propose two models. First we extend to the community level of the classic (discrete time) *Independent Cascade* model [5]. The key idea is to assume that each user exerts the same degree of influence over a whole community. Then, we provide a fine-grained modeling, by directly focusing on activation times. Here we assume that each user induces a fixed delay on the activation times of social peers within the same community.

Finally, our method not only is aimed at detecting communities by exploiting social influence evidence, as a by-product it also defines the level of influence of each user in each community. This allows to identify for each community the "key" users, i.e., the leaders which are most likely to influence the rest of the community to adopt a certain item. Those might be the users to target in a *viral marketing* campaign [5].

II. Related Work

Social contagion. The term *social contagion* refers to the spread of new practices, beliefs, technologies and products trough a population, driven by *social influence*. It is a very central theme in social sciences and recently it has attracted a lot of interest in the data mining community, mainly fueled by the seminal work by Domingos and Richardson [6] and Kempe et al. [5] which studied how to exploit social influence for "word-of-mouth" driven viral marketing applications.

Other researchers have considered the social network and the log of past user activity jointly, and studied important problems such as learning the strength of social influence along each arc [7], or how to distinguishing real social influence from "homophily" [8].

More related to our work, is the work by Gomez-Rodriguez et al. [4]. Here, the social network is not given, and the problem how to reconstruct the unobserved network starting from the log of users activity. In the proposed algorithm (NetRate), if the node u succeeds in activating v, then the contagious happens after an incubation time sampled from a chosen distribution, which is defines the conditional likelihood of transmission between each pair of nodes.

Communities and social contagion. The study of social contagion is intrinsically connected to the problem of understanding the community structure of networks. In fact, individuals tend to adopt the behavior of their social peers, so that social contagion happens first locally, within close-knit communities, and spreads virally only when it is able cross the boundaries of these densely connected clusters of people. Regardless the wide literature on community detection algorithms (see [1]), there has not been much research at the intersection of community detection and social contagion.

Wang et al. [9] study the problem of finding the top-k influential users in *mobile* social networks. They propose to first detect communities and then assume that the influence of a user is limited to her community. In this work communities are only a way to reduce the search space of the problem of finding influential users, not the goal. Moreover the setting is very different from ours: in their framework both the social network and the influence strength are given in input, while in our case none of the two is known a priori.

In our previous work [10] we study the following problem: given both the social graph and the log of user activity as input, the goal is to detect communities that "explain" well the two pieces of input. In simpler terms, the idea is to do better community detection using the additional information contained in the activity log. Towards this goal, we propose the Community-Cascade Network (CCN) model, a stochastic mixture membership generative model that can fit, at the same time, the social graph and the log of user activity. The present work also collocates itself in the intersection of community detection and social contagion, but it differs from [10], as here we tackle the problem of community detection *without the network*.

III. A General Framework for Influence-based Network-oblivious Community Detection

Notation. We are given a log of past user activity $\mathbb{D}$ defined as a relation $(User, Item, Time)$ where each tuple (u, i, t) indicates that the node u acted on item i at the time t. We let V denote the set of all users, i.e., the projection of $\mathbb{D}$ over the first column, $\mathcal{I}$ denote the universe of items, i.e., the projection of $\mathbb{D}$ on the second column, and we assume the time is an integer $t \in [0, T]$. We also use D_i to denote the overall activity on item i, that is the selection of the tuples of $\mathbb{D}$ where $Item = i$. We call it the *propagation trace* of i. The projection of D_i on the first column is denoted as C_i.

Let $t_u(i)$ represent the action time of the user u for the trace D_i; with $t_u(i) = \infty$ if u does not act on i by time T. We denote the time delay between the action of two users u, v on the item i as $\Delta_{u,v}(i) = t_u(i) - t_v(i)$. We also define $\Delta_u(i) = T - t_u(i)$. When i is clear from the context, we simply write $\Delta_{u,v}$ and Δ_u. Finally, let $C_{i,t}$ denote the set of users who have acted on the trace i by time t, i.e $C_{i,t} = \{u \in V : t_u(i) < t\}$.

Framework overview. Given only the user activity log $\mathbb{D}$, our goal is to detect communities in an unobserved network whose set of nodes correspond to the set of users V of $\mathbb{D}$. By communities we mean – as usual in the literature – clusters of nodes of a social network that exhibit high internal and low external link density. While detecting communities, we also aim at learning, for each community, the "key" users who are most likely to influence the adoption of a certain item i.

One possible approach is to forget about the existence of an underlying unobserved social network. Instead, just tackle the problem with a standard clustering approach: V is the set of objects to be clustered, and the actions of each user in V (which item i is adopted and at which time t) is its description. Another possible approach, as already stated in the introduction, is to focus on social influence to reconstruct the social network from the user activity log $\mathbb{D}$, following the framework of [3], [4]. Then apply some standard community detection algorithm to the reconstructed network.

We can consider these two approaches as the two opposite extremes of the spectrum: One totally ignoring the existence of the network effect, the other one explicitly reconstructing the network. Our proposal collocates in between these two extremes: although it does not attempt a direct reconstruction of the network, our approach is aware that information spreads over social connections. Thus the assumption behind our framework is that the unobserved network naturally shapes the process of information diffusion. A high-level overview of our framework is as follows:

- We assume the existence of an unobserved social network having a modular structure (as typical of social networks). This means that communities exist, densely connected internally, and loosely connected with other communities.
- We assume that item adoptions are governed by un underlying stochastic diffusion process over the unobserved social network. In particular, the diffusion model is based on community-level influence.

- Each user is associated with a level of membership and a level of influence in each community. These are the parameters of the diffusion model. The adoption of an item i by a user u depends on the influence exerted by other members of the community on u for adopting i.
- By fitting the model parameters to the activity log $\mathbb{D}$, we learn the community membership and influence levels.

This general framework can be instantiated to different community-level influence diffusion models. We will introduce two such models in Section IV and V respectively. We conclude this section by presenting the EM-like algorithm for fitting the model parameters to the user activity log.

Modeling the likelihood. We assume that each propagation trace is independent from the others, and we adopt a maximum a-posteriori perspective. That is, we hypothesize that action probabilities adhere to a mathematical model governed by a set of parameters Θ. Following the standard mixture modeling approach [11], we assume that users' actions can only happen relative to a community of membership. That is, we assume that a hidden binary variable $z_{u,k}$ denotes the membership of user u to community k, with the constraints $\sum_{k=1}^{K} z_{u,k} = 1$. Thus, Θ can be partitioned into $\{\pi_1, \ldots, \pi_K, \Theta_1, \ldots, \Theta_K\}$, where Θ_k represents the parameter set relative to community k, and $\pi_k \equiv P(z_{u,k} = 1)$. We can express the likelihood of the data as $\mathcal{L}(\Theta; \mathbb{D}) = \prod_u \sum_{k=1}^{K} P(u|\Theta_k)\pi_k$, which can be optimized by resorting to the traditional EM algorithm: Consider the complete-likelihood

$$P(\mathbb{D}, \mathbf{Z}, \Theta) = P(\mathbb{D}|\mathbf{Z}, \Theta) \cdot P(\mathbf{Z}|\Theta) \cdot P(\Theta) \tag{1}$$

where

$$P(\mathbb{D}|\mathbf{Z}, \Theta) = \prod_{u \in V} \prod_{k=1}^{K} P(u|\Theta_k)^{z_{u,k}}, \;\; P(\mathbf{Z}|\Theta) = \prod_{u \in V} \prod_{k=1}^{K} \pi_k^{z_{u,k}},$$

and $P(\Theta)$ represents the prior relative to the parameter set Θ. By standard manipulation of Eq. 1, the *Complete-Data Expectation Likelihood* [11] is given by:

$$\begin{aligned} \mathcal{Q}(\Theta; \Theta') &= E_{\mathbf{Z}}[\log P(\mathbb{D}, \mathbf{Z}, \Theta)|\mathbb{D}; \Theta'] \\ &= \sum_{u \in V} \sum_{k=1}^{K} \gamma_{u,k} \left\{\log P(u|\Theta_k) + \log \pi_k\right\} + \log P(\Theta) \end{aligned} \tag{2}$$

where $\gamma_{u,k} \equiv P(z_{u,k} = 1|u, \Theta')$.

Optimizing the latter can be done by means of EM algorithm: starting with an initial random assignment Θ, the algorithm iteratively performs two steps until convergence:

- (E Step) Given Θ, estimate $\gamma_{u,k}$ for each u, k as

$$\gamma_{u,k} = \frac{P(u|\Theta_k)\pi_k}{\sum_{k=1}^{k} P(u|\Theta_k)\pi_k}$$

- (M step) Given $\gamma_{u,k}$, find theΘ maximizing Eq. 2.

This general scheme is parametric to both the prior $P(\Theta)$ and the component $P(u|\Theta_k)$. We model the former in a way similar to [12], in order to allow an automatic estimation of the optimar number K of communities. As for the latter, it depends on the way we model the probability $P(a)$ for given actions $a \equiv (u, i, t)$. We explore two different alternatives.

1) The probability that u adopts i is the result of a bernoullian process on i, i.e., $P(a) \equiv P(i|u, t)$, and time proceeds in discrete steps.
2) The final model does not consider *whether* a user adopts i, but *when* the adoption happens, i.e. $P(a) \equiv P(t|i, u)$.

We next explore each strategy in turn. We consider only binary activations: at a given timestamp, each user is either active or inactive, and an active user cannot become inactive again.

IV. Community-Level Independent Cascade Model

When the social relationships are explicit, it is possible to define a propagation model which describes how adoptions spread across the network [5] and to model information propagation and community structure suitably [10]. In these models, a users tendency to become active increases monotonically as more of its social peers become active. We next adapt this concept to a network-oblivious situation, where we assume that the user's tendency to become active depends on the influence exterted within the community of membership.

The *Community-Independent Cascade (C-IC)* model draws from the Independent Cascade model (IC) [5], and models the idea that each user exerts the same degree influence over members of each community. Time unfolds in discrete timestamps. As in IC, when a user v becomes active, say at time t, it is considered contagious and has a single chance of influencing each inactive neighbor u, independently of the history thus far. We assume that v exerts her influence "globally", with a strength $p_v^k \in [0, 1]$ which depends on the community k of the targeted node. The idea is that the community-level influence of each user v is higher in the community she belongs to. According to this principle, we assume that information mainly propagate locally and spread across communities thanks to the presence of users who exhibit high degree of "external" influence.

Following [13] we adopt a delay threshold Δ to define influencers. Specifically, we define $\mathcal{F}_{i,u}^{+} = \{v \in V | 0 \leq t_u(i) - t_v(i) \leq \Delta\}$ as the set of users who potentially influenced u in the adoption of i. Similarly we define the set $\mathcal{F}_{i,u}^{-} = \{v \in V | t_u(i) - t_v(i) > \Delta\}$ of users who definitely failed in influencing u over i. Then, we can specify $P(u|\Theta_k)$ as

$$P(u|\Theta_k) = \prod_i P_+(i|u, \Theta_k) \cdot P_-(i|u, \Theta_k), \tag{3}$$

where $P_+(i|u, \Theta_k)$ represents the probability that some of the potential influencers activated u and $P_-(i|u, \Theta_k)$ the probability that none of the "out-of-react" influencers succeeded:

$$P_+(i|u, \Theta_k) = 1 - \prod_{v \in F_{i,u}^{+}} (1 - p_v^k)$$

$$P_-(i|u, \Theta_k) = \prod_{v \in F_{i,u}^{-}} (1 - p_v^k)$$

We can then specify the complete-data likelihood through:

$$P(\mathbb{D}|\mathbf{Z}, \Theta) = \prod_{i,u,k} \left[1 - \prod_{v \in F_{i,u}^{+}} (1 - p_v^k)\right]^{z_{u,k}} \cdot \left[\prod_{v \in F_{i,u}^{-}} (1 - p_v^k)\right]^{z_{u,k}}$$

Learning influence weights. The analytical optimization of $\mathcal{Q}(\Theta;\Theta')$ is still difficult. We resort to the explicit modeling of the influencers as hidden data to simplify the optimization procedure. That is, let $w_{i,u,v}$ be a binary variable such that $w_{i,u,v}=1$ if v triggered the adoption of the item i by u, and let $\mathbf{W}$ denote the set of all possible $w_{i,u,v}$ such that $v \in F^+_{i,u}$. Then, we can rewrite the complete-data likelihood relative to $\mathbf{W}$ as

$$P(\mathbb{D},\mathbf{Z},\mathbf{W},\Theta) = P(\mathbb{D},\mathbf{W}|\Theta,\mathbf{Z}) \cdot P(\mathbf{Z}|\Theta) \cdot P(\Theta),$$

where

$$P(\mathbb{D},\mathbf{W}|\Theta,\mathbf{Z}) = \prod_{i,u,k} \prod_{v\in F^-_{i,u}} (1-p^k_v)^{z_{u,k}} \cdot \prod_{i,u,k} \prod_{v\in F^+_{i,u}} \left(p^k_v\right)^{w_{i,u,v}\cdot z_{u,k}} \left(1-p^k_v\right)^{(1-w_{i,u,v})\cdot z_{u,k}}$$

As a consequence, the contribution to $\mathcal{Q}(\Theta;\Theta')$ in the second row of Eq. 2 can be rewritten as

$$\sum_u \sum_k \gamma_{u,k} \left(\log \pi_k + \sum_i \sum_{v\in F^-_{i,u}} \log(1-p^k_v) + \sum_i \sum_{v\in F^+_{i,u}} \eta_{i,u,v,k} \log p^k_v + (1-\eta_{i,u,v,k}) \log(1-p^k_k) \right)$$

where $\eta_{i,u,v,k}$ is the "responsibility" of the user v in triggering u's adoption in the context of the community k:

$$\eta_{i,u,v,k} = P(w_{i,u,v}=1|u,i,z_{u,k}=1,\Theta') = \frac{p^k_v}{1-\prod_{w\in F^+_{i,u}}(1-p^k_w)}.$$

Finally, optimizing $\mathcal{Q}(\Theta;\Theta')$ with respect to p^k_v yields

$$p^k_v = \frac{\sum_{\substack{\langle u,i\rangle \\ v\in F^+_{i,u}}} \gamma_{u,k}\cdot\eta_{i,u,v,k}}{S^+_{v,k}+S^-_{v,k}}, \tag{4}$$

with $S^+_{v,k} = \sum_{\substack{\langle u,i\rangle \\ v\in F^+_{i,u}}} \gamma_{u,k}$ and $S^-_{v,k} = \sum_{\substack{\langle u,i\rangle \\ v\in F^-_{i,u}}} \gamma_{u,k}$.

V. Modeling Temporal Dynamics

C-IC does not explicitly model temporal dynamics, as it focuses on modeling just binary activations by employing a discrete-time propagation model. Here we present an alternative modeling that exploits time delays to characterize the overall diffusion process.

Given an observation window $[0,T]$, the idea is to explicitly model the likelihood of the time at which each user adopted each item, or the likelihood that the considered adoption did not happen within time T. This approach assumes that there is a dependency between the adoption time of the influencer and the one of the influenced. In NetRate [4], previously described in Sec. II, this dependency in modeled by a conditional likelihood $f(t_u|t_v,\alpha_{v,u})$ of transmission, which depends on the delay $\Delta_{v,u}$. The likelihood of a propagation can be formulated by applying standard survival analysis [14], in terms of survival $S(t_u|t_v,\alpha_{v,u})$ (modeling the probability that a user survives uninfected at least until time t_u) and hazard functions $H(t_u|t_v,\alpha_{v,u})$ (modeling instantaneous infections).

We reformulate this framework into a community-based scenario. The *Community-Rate* (C-Rate) propagation model is characterized by the following assumptions:

- User's influence is limited to the community she belongs to. That is, the user is likely to influence/be influenced by members of the same community, while the effect of influence is marginal on members of a different community.
- Information diffusion from the user v to v within the k-th community is characterized by the density $f(t_u|t_v,\alpha_{v,k})$, where $\alpha_{v,k}$ is related to the expected delay on the activations that v triggers within community k. The probability of contagion depends on the time delay $\Delta_{v.u}$.

The parameter $\alpha_{v,k}$ has a direct interpretation in terms of influence: high values of $\alpha_{v,k}$ cause short delays, and as a consequence denote v as strongly influential within k.

On the basis of the above observations, we can adapt the NetRate model to fit the scheme of Sec. III, by plugging

$$P(u|\Theta_k) = \prod_{i:u\notin C_i} \prod_{v\in C_i} S(T|t_v(i),\alpha_{v,k}) \cdot \prod_{i:u\in C_i} \prod_{v\in C_{i,t_u(i)}} S(t_u(i)|t_v(i),\alpha_{v,k}) \sum_{v\in C_{i,t_u(i)}} H(t_u(i)|t_v(i),\alpha_{v,k}) \tag{5}$$

Learning. Again, instead of directly optimizing the above likelihood, we introduce the latent binary variable $w_{i,u,v}$ denoting the fact that u has been infected by v on i. Then, the likelihood can be rewritten by defining

$$P(\mathbb{D},\mathbf{W}|\mathbf{Z},\Theta) = \prod_{\langle u,i\rangle\notin\mathbb{D}} \prod_k \prod_{v\in C_i} S(T|t_v(i),\alpha_{v,k})^{z_{u,k}} \cdot \prod_{\langle u,i\rangle\in\mathbb{D}} \prod_k \prod_{v\in C_{i,t_u(i)}} H(t_u(i)|t_v(i),\alpha_{v,k})^{w_{i,u,v}z_{u,k}} \cdot S(t_u(i)|t_v(i),\alpha_{v,k})^{z_{u,k}}$$

and replacing $P(\mathbb{D}|\mathbf{Z},\Theta)$ with the above component in the likelihood. In the following we adopt the exponential distribution $f(t_u|t_v,\alpha_{v,k}) = \alpha_{v,k}\exp\{-\alpha_{v,k}\Delta_{v,u}\}$, which enables $S(t_u|t_v,\alpha_{v,k}) = \exp\{-\alpha_{v,k}\Delta_{v,u}\}$ and $H(t_u|t_v,\alpha_{v,k}) = \alpha_{v,k}$. [1] Then,

$$Q(\Theta;\Theta') \propto \sum_{u,k}\gamma_{u,k}\log\pi_k - \sum_{\langle u,i\rangle\notin\mathbb{D}} \sum_k \sum_{v\in C_i} \gamma_{u,k}\Delta_v\alpha_{v,k} + \sum_{\langle u,i\rangle\in\mathbb{D}} \sum_k \sum_{v\in C_{i,t_u(i)}} \eta_{i,u,v,k}\gamma_{u,k}\log\alpha_{v,k} - \sum_{\langle u,i\rangle\in\mathbb{D}} \sum_k \sum_{v\in C_{i,t_u(i)}} \gamma_{u,k}\Delta_{u,v}\alpha_{v,k},$$

[1] Similar formulations can be obtained by adopting different densities and are omitted here for lack of space.

and the probability of observing v as an influencer on u, i is given by:

$$\eta_{u,i,v,k} = \frac{H(t_u(i)|t_v(i), \alpha_{v,k})}{\sum_{v' \in C_{i,t_u(i)}} H(t_u(i)|t_v(i).\alpha_{v',k})}.$$

Finally, optimizing $Q(\Theta; \Theta')$ yields

$$\alpha_{k,v} = \frac{\sum_{\substack{\langle u,i \rangle \in \mathbb{D} \\ v \in C_{i,t_u(i)}}} \eta_{i,u,v,k} \gamma_{u,k}}{\sum_{\substack{\langle u,i \rangle \notin \mathbb{D} \\ v \in C_i}} \gamma_{u,k} \Delta_v + \sum_{\substack{\langle u,i \rangle \in \mathbb{D} \\ v \in C_{i,t_u(i)}}} \gamma_{u,k} \Delta_{u,v}} \quad (6)$$

VI. Experimental Evaluation

In this section we report our experimental analysis aimed at assessing the effectiveness of the proposed framework. Due to space limitations, we only use synthetic data with a predefined community structure, and we deal with real-world data in a forthcoming extended version of this paper.

Synthetic Datasets. We generate synthetic data in two steps. First, we generate a network which exhibits a known community structure, as well as structural features typical of real networks. To this aim, we use the generator of benchmark graphs described in [15], which generates directed unweighted graphs with *possibly* overlapping communities. The process of network generation is controlled by the following parameters: number of nodes $(1,000)$; average in-degree (10); maximum in-degree (150); min/max community size $(50/750)$. The four networks differ on the ratio μ, which controls for each node how many v neighbors do not share any community with v. We use four values for μ $(0.001, 0.01, 0.05$ and $0.1)$ obtaining four networks, named $S1$, $S2$, $S3$, and $S4$ respectively.

Given a network $G = (V, E)$, the next step is to generate synthetic propagation cascades spreading over G. For each community k, an initial dummy node is connected to all nodes within the considered community, with a random influence weight sampled from $[0.02, 0.05]$. For each trace we generate a random permutation of the dummy nodes and, after selecting the first one, the $n-th$ community-node is picked randomly with probability β^n. This initialization step determines the degree to which the trace to be generated will be local/global. At time $t = 0$, the dummy nodes determine the activation of real nodes, from which we start the subsequent diffusion process. At this stage, information can spread on the network by exploiting the links. The strength of each link is determined by considering both the outdegree ($\kappa_{\cdot}^{out}$) of the source and the indegree ($\kappa_{\cdot}^{in}$) of the destination:

$$weight(u, v) \propto \lambda \cdot \frac{\kappa_u^{out}}{\kappa^{out}} \frac{\kappa_v^{in}}{\kappa^{in}} + (1 - \lambda) \cdot rand(0.1, 1)$$

where κ^{out} and κ^{in} are the maximum out-degree and in-degree respectively, and λ introduces a random effect.

In the propagation process, the weight of each link represents a bernullian probability of infection. For each link we also generate a typical infection rate $\alpha_{u,v}$, sampled from a Gamma distribution (shape=2, scale=0.3).

To summarize, the synthesized data depends on μ, the degree of propagation overlap β and the number of propagation cascades $|\mathcal{I}|$. In this first experiment, we fix $\lambda = 0.9$, $\beta = 0.2$ and $|\mathcal{I}| = 1,500$, and vary the μ parameter as discussed above. For each network, we randomly generate 5 propagation logs. The main properties of the synthetic generated data are summarized in Tab. I.

TABLE I: Statistics for the synthetic data: four networks corresponding to four values of μ.

	S1	S2	S3	S4
# of communities (K)	9	7	11	6
avg # of adoptions	56k	59k	82k	370k
avg trace length	38	38	54	256
avg % of communities traversed by a trace	17%	24%	24%	82%

Baselines. The C-IC and C-Rate techniques are compared to some baseline models. The first two baselines builds on the idea of network reconstruction. Given a log of past propagation $\mathbb{D}$ we can apply either NetRate or the Independent Cascade inference procedure [16] (assuming the complete graph). Both algorithms provide a set of link weights as output, and higher weights witness the existence of strong connections. We reconstruct the network by applying a sparisification procedure based on the identification of a threshold value, accomplished by analyzing the distribution of the weights. Finally, communities are discovered by applying the Metis algorithm [17], which is reported to achieve good performances and is fast. These baselines are denoted as NetRate/Metis and IC/Metis.

A further baseline is a standard clustering algorithm that groups users according to their adoptions. The algorithm is based on a *multinomial EM* procedure. This baseline can be exploited for detecting communities, but does not measure the degree of influence of a user within a community, like instead the algorithms proposed in this paper do.

Results. We measure the quality of the discovered communities w.r.t. the known ground truth communities using the Adjusted Rand Index [18], as well as the F-Measure and the Normalized Mutual Information [19]. For all the considered approaches, we report the average quality indices, as well as standard deviation relative to the 5 propagation logs, in Fig. 1. As we can see on the figure, both C-IC and C-Rate perform particularly well on all four networks. The performances degrade on the s4 network, where the IC/Metis method is predominant but still comparable to the performances of C-Rate. However, it should be noted that the two baselines (as expected) have running times which are an order of magnitude larger and do not scale to large network. For these methods, the Metis algorithm does not affect the performance significantly, and the computational burden is essentially due to the network inference phase.

Finally, the quality of the Multinomial EM algorithm is extremely unstable, contrary to all other methods. This is a clear sign of the relevant role of the influencers when associating a user to a community: influencers tend to better explain the activation of a user on a given item, and hence tend to reduce the variability in the membership assignments.

In a second batch of experiments, we measure the effects of

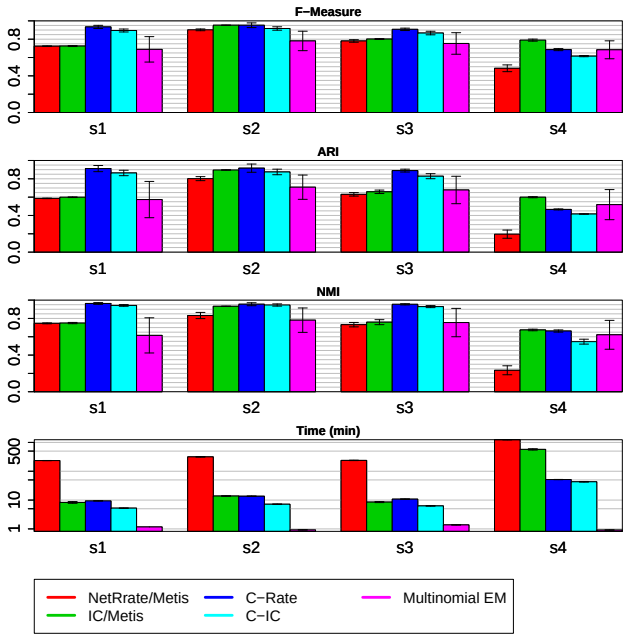

Fig. 1: Summary of the evaluation on synthetic data.

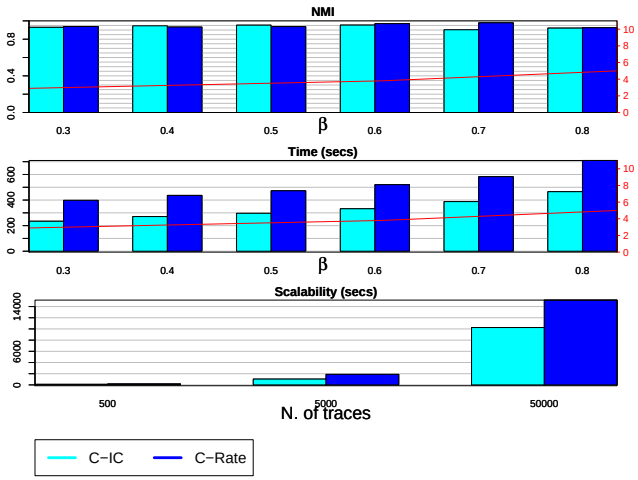

Fig. 2: Robustness and scalability.

the mixing parameter β on C-IC and C-Rate. Higher values of β cause a trace to spread over multiple communities: as a consequence, we can measure the robustness of the algorithms by varying such a parameter. For these experiments, we use the s3 network, and we generate $1,000$ traces by ranging the β within $[0.3, 0.8]$. The first plot of Fig. 2 reports the Normalized Mutual Information. It can be noticed that the performances do not significantly change: the algorithms can still separate communities and associate users with them, even when the average number of communities traversed by a single trace (denoted by the red line in the plot) increase. Rather, a higher mixing parameter affects inference times, as shown by the second plot. The algorithms require more iterations to reach convergence, as a consequence to the fact that separations need to be reconstructed iteratively.

A final experiment measures the scalability of the proposed algorithms for increasing values of $|\mathcal{I}|$. In the last plot of Fig. 2 we show the running times on three log traces of increasing size, relative to the s3 network. Both the algorithms scale linearly on the number of traces, and the general trend, where C-IC seems more efficient than C-Rate, is confirmed.

VII. Conclusions

We proposed a general framework for detecting communities in a network-oblivious setting. The general framework is based on a the assumption that item adoptions are governed by un underlying stochastic diffusion process over the unobserved social network, and that such diffusion model is based on community-level influence. We instantiated the diffusion process by adopting two models which focus on both the influence exerted by a user in a given community, and the likelihood of a user to belong to that community. The experiments show that both models are robust and effective, and be can profitably employed to discover communities and regions of influence in situations where the social connections are not visible.

Acknowledgments. This work was supported by MULTISENSOR project, partially funded by the European Commission, under the contract number FP7-610411.

References

[1] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.
[2] http://techcrunch.com/2013/01/24/my-precious-social-graph/.
[3] M. Gomez-Rodriguez, J. Leskovec, and A. Krause, "Inferring networks of diffusion and influence," in *KDD'10*.
[4] M. Gomez-Rodriguez, D. Balduzzi, and B. Schölkopf, "Uncovering the temporal dynamics of diffusion networks," in *ICML'11*.
[5] D. Kempe, J. M. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *KDD'03*.
[6] P. Domingos and M. Richardson, "Mining the network value of customers," in *KDD'01*.
[7] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, "Learning influence probabilities in social networks," in *WSDM'10*.
[8] A. Anagnostopoulos, R. Kumar, and M. Mahdian, "Influence and correlation in social networks," in *KDD'08*.
[9] Y. Wang, G. Cong, G. Song, and K. Xie, "Community-based greedy algorithm for mining top-k influential nodes in mobile social networks," in *KDD'10*.
[10] N. Barbieri, F. Bonchi, and G. Manco, "Cascade-based community detection," in *WSDM'13*.
[11] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B*, vol. 39, pp. 1–38, 1977.
[12] M. T. Figueiredo and A. Jain, "Unsupervised learning of finite mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 3, pp. 381–396, 2002.
[13] M. Mathioudakis, F. Bonchi, C. Castillo, A. Gionis, and A. Ukkonen, "Sparsification of influence networks," in *KDD'11*.
[14] E. Lee and J. Wang, *Statistical methods for survival data analysis*.Wiley-Interscience, 2003, vol. 476.
[15] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, vol. 80, 2009.
[16] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," in *KES'08*.
[17] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1999.
[18] A. Jain and R. Dubes, *Algorithms for Clustering Data*. Prentice-Hall, 1988.
[19] L. Ana and A. Jain, "Robust data clustering," in *Procs of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2003.