Community Detection Using Diffusion Information

MARYAM RAMEZANI, ALI KHODADADI, and HAMID R. RABIEE, Sharif University of Technology

Community detection in social networks has become a popular topic of research during the last decade. There exist a variety of algorithms for modularizing the network graph into different communities. However, they mostly assume that partial or complete information of the network graphs are available that is not feasible in many cases. In this article, we focus on detecting communities by exploiting their diffusion information. To this end, we utilize the Conditional Random Fields (CRF) to discover the community structures. The proposed method, community diffusion (CoDi), does not require any prior knowledge about the network structure or specific properties of communities. Furthermore, in contrast to the structure-based community detection methods, this method is able to identify the hidden communities. The experimental results indicate considerable improvements in detecting communities based on accuracy, scalability, and real cascade information measures.

CCS Concepts: • **Mathematics of computing → Probability and statistics**; **Factor graphs**; • **Information systems → Clustering**; • **Networks → Network performance evaluation**; • **Human-centered computing → Social networking sites**; • **Computing methodologies → Machine learning algorithms**; • **Applied computing → Sociology**;

Additional Key Words and Phrases: Social networks, community detection, information diffusion, conditional random field, social influence

ACM Reference format:
Maryam Ramezani, Ali Khodadadi, and Hamid R. Rabiee. 2018. Community Detection Using Diffusion Information. *ACM Trans. Knowl. Discov. Data.* 12, 2, Article 20 (January 2018), 22 pages.
https://doi.org/10.1145/3110215

1 INTRODUCTION

The explosion of data in social media has attracted many researchers to model and analyze social networks. One of the most useful features of these networks is the existence of communities in their structures. Communities are the various dense sub-modules with sparse inter-community links among them [39]. In general, the members of a community have common interests that indicate behavioral mimicry among the social network members [12]. Detecting network communities has attracted considerable attention during the last few years. Finding the influential nodes for targets in viral marketing [14], and suggesting friends or items in recommender systems [18, 45] are among the most common applications of community detection.

Access to the network topology is a necessary and prevalent requirement in many of the existing community detection methods [19, 30, 41]. These methods try to detect communities by

Authors' addresses: M. Ramezani, A. Khodadadi, and H. R. Rabiee, Advanced ICT Innovation Center, Department of Computer Engineering, Sharif University of Technology, Tehran, Iran 11365; emails: {m_ramezani, khodadadi}@ce.sharif.edu, rabiee@sharif.edu.

https://doi.org/10.1145/3110215

considering the density of network structure. However, there is an access limitation to the entire network connectivity structure due to reasons such as large amount of data, privacy, and confidentiality policies in popular online social networks that affects the accuracy of these methods. Moreover, the structure-based methods cannot exploit the behavioral features, interests, and manners between the members of communities, while considering these features are important for the following reasons [5, 32]:

(1) Distinguishing between similar and contrasting behavior: A user may follow another user in a social network with a hostile behavior. If we detect communities only by utilizing the network topology, hostile relations will be modeled similar to a friendship or trust relationship. As a result, the probability of being in a community with someone having opposite behavior will confusedly increase.
(2) Considering hidden links: A user may not follow another user to keep himself hidden from him. Therefore, no link will be available directly between them. However, the behavior of the other user may affect his choices and behaviors.

Using a traceable process such as diffusion information over the network can be utilized to overcome the aforementioned problems. Therefore, inferring the links of network from diffusion information can be used to estimate the structure of network, and then a structure-based community detection method can be applied to find the proper communities. However, this approach suffers from high computational complexity [7]. In addition, as discussed in [43], most of the state of the art algorithms for network inference do not efficiently preserve the community structure and are not scalable. Incidentally, the community structure affects the information exchange known as diffusion between different users [3, 36], and the different behaviors of users appear at diffusion process. As a result, diffusion makes various relations between the users apparent in the network. Hence, using diffusion information in the community detection procedure will produce results that are more closer to the reality [6]. Moreover, by propagating news, opinions, or advertisements on social networks, we can gather the actions of users after their exposure, without the limitations and difficulties faced by collecting the topology structures.

In summary, accessibility of the whole network topology is a major assumption in the most related works in community detection areas. In this article, we are concerned with the community detection by only using the diffusion information, while the network topology is completely unknown. During the diffusion process, a contagion spreads over the network and creates a traceable directed path called cascade, and infection time is the time a node takes parts in a certain cascade [12, 21]. To achieve our goal, we use the information from the diffusion process that is a set of cascades that indicates the infection time per user.

To the best of our knowledge, only one recently published research attempts to discover the communities without the information about the network topology [7]. However, this method needs to know the number of communities as *a priori* knowledge; an assumption that is far from reality. Therefore, this is the first work which attempts to detect the network communities from the diffusion information without any prior knowledge. The main contributions of the proposed method, community diffusion (CoDi), can be summarized as follows:

—The set of cascades over the network is the only required information and the network structure is latent to our method.
—We avoid the complexity of using methods for estimating the network structure and then detecting the communities, by discovering the communities directly from the diffusion information without utilizing any intermediate step.

—There is no need to have any prior knowledge about the network structure such as the number of communities or links.
—The proposed method achieves scalability with high accuracy.

The rest of the article is organized as follows. Section 2 presents the related works in selected areas of community detection with a focus on diffusion concepts. Section 3 provides a formal definition of the problem. Section 4 describes the theoretical aspects of the proposed method. Section 5 presents the experimental results on various datasets. Finally, Section 6 concludes the article and provides directions for the future works.

2 RELATED WORK

Despite the existence of a wide range of community detection algorithms [19, 30, 41], most of them do not consider the diffusion information. In this section, we categorize the current diffusion-based approaches according to the available information about the network topology and diffusion process.

Topology conscious and diffusion information: These studies use the whole network topology as well as the information from diffusion process as a real contagion propagation over the network. Barbieri et al. [6] detected communities from the topology and then improved the detected communities by utilizing a set of diffusions to find more realistic communities. Himel et al. [16] created a weighted graph from the interaction of network members (i.e., shared photos, films, or comments). Then, they created another graph by using the common neighbors between each pairs of nodes at the network structure graph. Finally, by utilizing these weighted graphs, and hierarchically separating or merging, the communities are detected. Moreover, there has been a considerable amount of research in using communities for maximizing the spread of influence, specially in marketing applications. One of the best ways to achieve this goal is to introduce products to influential users and let the advertisement propagates virally. Such influential nodes are usually the core nodes and identifying them, can be achieved by detecting communities. Some researchers have used greedy algorithms to determine these core nodes, and utilize diffusion models from core nodes to coherent nodes, in order to spread similar behaviors [24, 42, 49].

Topology oblivious and diffusion information: These methods try to infer communities only by using diffusion information. As previously mentioned, there are two possible solutions for this problem: First, inferring the network links and detecting the communities using inferred links, and second, detecting communities directly by using diffusion information. In the first approach, our previous work on network inference, which is called Diffusion Aware Network Inference Algorithm (DANI), tries to infer the network links only by using the nodes infection times in the diffusion process [43]. Unlike the other inference algorithms [17, 22], DANI preserves the community structure of the network. DANI applies a structure-based community detection algorithm on the inferred networks to detect the communities. The main problem of DANI and other network inference methods is their need to know the number of links in the underlying network. Having the double complexity of an inference and a community detection algorithm is another difficulty of this approach for solving the community detection problem [7]. In the second approach, Barbieri et al. [7] modified the node to node diffusion influence in NETRATE [20] and Independent Cascade (IC) [46] to the efficacy of the community to node information spread and utilized these two modified network inference models to find communities without knowing the network structure. The infection times of nodes in different cascades and the number of communities are the only prior information needed for this algorithm. Their method suffers from a high run time specially when the network is large. Chen et al. [13] proposed an approach for finding communities based

on user interactions in Facebook dataset. They gather the user profile information and diffusion over the status of users.

This research belongs to the second category above. Since the proposed method in [13] is restricted to a dataset and needs users profile information, the method in [7] is the most related work to this article. In this article, our goal is to extract the network communities directly from the diffusion information with low running time. Therefore, we gather the diffusion information from the actions of users during the proper periods.

3 PROBLEM STATEMENT

The goal of this research is to partition the nodes $\mathbf{V} = \{v_1, v_2, \ldots, v_g\}$ of network ($\mathcal{G} = (\mathbf{V}, \mathbf{E})$) into different communities with the assumption that no information about the network topology ($\mathcal{G}$) is available. The only available information is the diffusion information; a set of cascades $\mathbf{O} = \{O_1, O_2, \ldots, O_m\}$ that propagated separately over the network. Each cascade O_i is the trace of a contagion i that spreads over unknown graph $\mathcal{G}$. O_i illustrates the pairs of network nodes and their infection time in the ith cascade ($O_i = \{(v_1(i), t_1(i)), (v_2(i), t_2(i)), \ldots (v_g(i), t_g(i))\}$). When a node v_j does not participate in the ith cascade, then $t_i(j) = \infty$. However, we have only knowledge about the active nodes that take part in the set of cascades ($\mathbf{O}$); hence, the community detection will be done on the set of nodes ($\mathbf{U} = \{v_j | (v_j \in V) \wedge (\forall X_i \in \mathbf{O} : (t_j(i) \prec \infty))\}$), instead of all the network nodes ($v \in \mathbf{V}$). Sometimes for privacy reasons, we may be not aware of the exact value of infection times. However, the proposed method will work by just knowing the ascending infection times in each cascade. Assuming that the network has k different communities, for each active node ($u \in \mathbf{U}$), we want to find its most probable community assignment ($Y(u) = j | j \in \{1, 2, \ldots, k\}$). Hence, our problem becomes equal to finding a community assignment vector named Y (a vector with g elements where each $Y(e)$ indicate the community of node e) for graph $G = (\mathbf{U}, \mathbf{R})$ that maximizes $P(Y|\mathbf{O})$ as shown in Equation (1) ($\mathbf{U}$ represents the active nodes that participate in cascades with corresponding nodes, $Y(u)$s in the CRF graph, and $\mathbf{R}$ represents the links between $Y(u)$s). Therefore, the set $\mathbf{U}$ will be partitioned into k subsets $\mathbf{C} = \{C_1, C_2, \ldots, C_k\}$ with the constraint that $\bigcup_{C_i \in C} C_i = \mathbf{U}$ and $\bigcap_{C_i \in C} C_i = \emptyset$.

$$\hat{Y} = \underset{Y}{\operatorname{argmax}}\, P(\mathbf{y}|\mathbf{O}) \tag{1}$$

4 PROPOSED METHOD

We propose a method to find the vector Y that maximizes the conditional probability in Equation (1). In fact, in a network with a community structure, a node i belongs to a community $Y(i) = a$ when at least one of its neighboring nodes is assigned to that community ($Y(N\{i\}) = a$). If $N\{i\}$ represents the set of neighbors of node i, the "Markov property" [26] is satisfied because the community label $Y(i)$ is only determined by the labels of its neighbors, and does not depend on rest of the network. That is,

$$P(Y(i)|Y(U - \{i\})) = P(Y(i)|Y(N\{i\})). \tag{2}$$

Equation (1) corresponds to a discriminative model, and here we try to utilize a statistical graphical model to solve our problem. Conditional Random Field (CRF) is an undirected graphical model for representing a conditional probability $P(Y|X)$ between the observed random variables X and latent random variables Y with the following properties [48].

(1) $G = (\mathbf{U}, \mathbf{R})$: Is an undirected graph that each of its nodes i corresponds to a label y_i and the affinity between nodes i and j is modeled by an edge $r_{ij} \in R$.

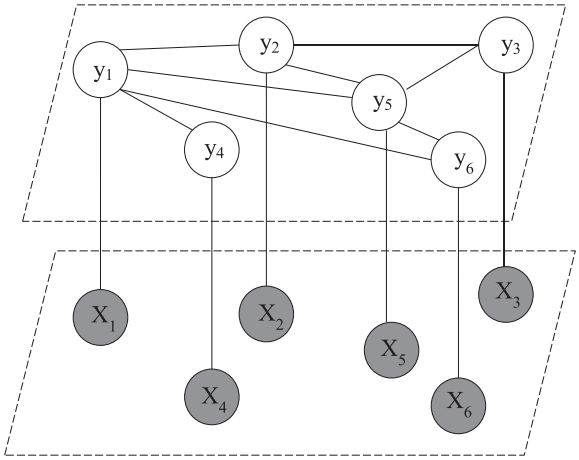


Fig. 1. CRF representation of our problem for a simple network with six nodes. X_is are the observation random vectors coming from the node diffusion behaviors. y_is are the latent random variable representing the community membership of nodes.

(2) Markov property:

$$P(y_i|x_i, y_{U-\{i\}}) = P(y_i|x_i, y_{N\{i\}}). \tag{3}$$

The representation of our problem with a CRF is illustrated in Figure 1. Let assume that m different ICs spread over the network, then the corresponding random variables and graph can be defined as follows:

—X: We consider a $1 \times m$ vector for each node X_i corresponding to the observation of node participation in the cascades. If the ith node had been infected in cascade O_j, $(t_i(j) \prec \infty)$, the jth element of the vector will be set to 1:

$$X_i[j] = \begin{cases} 1 & t_i(j) < \infty \\ 0 & t_i(j) = \infty \end{cases} \tag{4}$$

—Y: The latent random variable y_i represents the community membership of node i. Union of y_is form the latent random vector Y.

—$G = (\mathbf{U}, \mathbf{R})$: A weighted undirected graph between network nodes, y_is, which is obtained with respect to the similarities between the related node observations, X_is. Structure learning is the usual approach for constructing this graph. Here, we create the graph without suffering from the complexity of learning models by using the node features, X, from diffusion observation based on the common concepts of community structure and diffusion process. The nodes of this graph are the active nodes $\mathbf{U}$ from the set of cascades. Since the contagion tends to spread in a community, the nodes in a community participate jointly in cascades. Relying on this fact, a probable dependency between the nodes can be detected by considering the similarities among their diffusion behaviors. There exist an edge $r_{ij} \in \mathbf{R}$ between the nodes i and j from set $\mathbf{U}$, if these nodes jointly get infected in at least one cascade. We define the weight of each edge r_{ij} by using the cosine similarity that demonstrates the amount of similarity in the diffusion behaviors (r_{ij}):

$$r_{ij} = \frac{X_i.X_j}{|X_i||X_j|}. \tag{5}$$

Since, similarity is a symmetric measure, the constructed graph is undirected. Moreover, many pairs of nodes are not involved in any common cascades; hence, the adjacency matrix of this graph is sparse. These two properties lead to fast cosine similarity computations in the proposed method.

Therefore, we may solve the following equation instead of Equation (1):

$$\hat{Y} = \underset{Y}{\operatorname{argmax}} P(Y|X). \tag{6}$$

Now, we need to define the probability distribution $P(Y|X)$ and then try to maximize it based on the random vector Y. To this end, we utilize the Hammersley–Clifford theorem.

Hammersley–Clifford theorem: The random field $P(Y|X)$ is a CRF if and only if its probability distribution function can be defined as a Gibbs distribution [9].

Considering the above theorem, we model $P(Y|X)$ as a Gibbs distribution:

$$P(Y|X) = \frac{1}{T(X)} \exp(-E(Y|X)), \tag{7}$$

where $T(x)$ is a normalization factor $T(X) = \sum_Y \exp(-E(Y|X))$ and $E(Y|X)$ is the energy function that is defined on the cliques of the graph G:

$$E(Y|X) = \sum_i E_i(y_i|X) + \sum_{(i,j)} E_{ij}(y_i, y_j|X) + \dots \tag{8}$$

using Equation (7) we can rewrite Equation (6) as

$$\hat{Y} = \arg\max_Y(exp(-E(Y|X))) = \arg\min_Y(exp(E(Y|X)) \tag{9}$$

As a result, the proposed method decomposes into the following two steps:

— *Energy Function:* In this step, we define the energy function on the cliques by Equation (8).
— *Maximum Likelihood Estimation:* In this step, we solve the maximum likelihood estimation (MLE) of Equation (9) to find the best community assignment.

4.1 Energy Function

In order to utilize CRF, we should define energy functions on the cliques. In the context of social networks, it is common to define the energy function on two nodes [40, 47]. Hence, we select the two-order clique as the maximal clique, and the higher order cliques are built from sum of the two orders. For example, a three order clique is the three times sum of two order cliques. Based on the definition of community, usually dense groups of nodes form the communities. One of the most well known measures for detecting the communities is modularity. We are looking for an energy function with low and stable values, when the modularity of network is high. Therefore, this energy function must be defined for each pair of community labels (y_i, y_j). To achieve this goal, we utilize an approach that is similar to the modularity on weighted constructed graph (G) [38]:

$$E_{ij}(y_i, y_j|X) = -1 \times [r_{ij} - \frac{r_{i,.} r_{j,.}}{b}] \times I(y_i, y_j), \tag{10}$$

where b is the sum of all weighted edges, and r_{ij} represents the weight of edge between nodes i and j. Moreover, $r_{i,.}$ is the weighted degree of node i, and I is the indicator function:

$$I(y_i, y_j) = \begin{cases} 1 & if(y_i = y_j) \\ 0 & else \end{cases} \tag{11}$$

As a consequence, Equation (8) can be represented as

$$E(Y|X) = \sum_{(i,j)} \left(-1 \times \left[r_{ij} - \frac{r_{i,.} r_{j,.}}{b} \right] \times I(y_i, y_j) \right). \tag{12}$$

4.2 Maximum Likelihood Estimation

It is difficult to solve the maximum likelihood estimation of Equation (9). Instead of using the usual methods to estimate the latent random variables, we are going to solve it from the perspective of inference methods for probabilistic graphical models. However, by combing Equations (9) and (12) and applying log on the both sides of this equation, the MLE problem in Equation (13) can be solved efficiently by using optimization techniques:

$$\hat{Y} = \arg\min_{Y} \left(\sum_{(i,j)} \left(-1 \times \left[r_{ij} - \frac{r_{i,.} r_{j,.}}{b} \right] \times I(y_i, y_j) \right) \right). \tag{13}$$

Here, we utilize the Iterated Conditional Mode (ICM), an algorithm based on solving derivative relations for Markov models with exponential distributions. ICM is a local iterative deterministic method that minimizes or maximizes a random variable conditioned on the other variables with a greedy strategy. It starts with limited initial values for the latent random variables and updates them in a greedy approach until convergence to a minimum amount of energy in each step. It is worth noting that the updates are done in a synchronous manner [10].

Here, we first assume the number of communities (k) is given, and continue with the optimization steps. In the next subsection, we will describe how to estimate k. The likelihood function in Equation (13) can be solved by the proposed method through the following steps:

(1) Initialization: We initialize the latent random variables $y_{1:g}$ (g is the number of network active nodes) with discrete values from $\{1, 2, \ldots, k\}$.
(2) Updating labels: For each node i, we calculate the energy function according to each of its neighbors j ($j \in N(i)$). The label of node i, called y_i, will change to the label of its neighbor j, when it reaches the minimum value of $E(y_i, y_j|X)$.
(3) Termination condition: After p iterations of step 2, the algorithm converges to the community label of network nodes, $Y^{(p)}$. When the change between two partitions is not significant ($\text{Diff}(Y^{(p)}, Y^{(p-1)}) < \varepsilon$), it converges and $Y^{(p)}$ represents the vector of assigned communities. Since Y represent the assigned community labels, we cannot use the Euclidean distance as the convergence measure. However, the metric for difference between two consecutive Ys should be related to the concept of community structure. Therefore, we define the Diff function by utilizing the purity measure [25], which shows the fraction of same vertices between $Y^{(p-1)}$ and $Y^{(p)}$. Indeed, the Diff function shows the percentage of the nodes that have changed their communities in partition $Y^{(p)}$ toward $Y^{(p-1)}$, and a low value for it leads to convergence:

$$\text{Diff}(Y^p, Y^{p-1}) = 1 - purity(Y^p, Y^{p-1}). \tag{14}$$

Assume the number of all active nodes of network is $n = |\mathbf{U}|$ and C_1 and C_2 are the number of communities for partitions $Y^{(p-1)}$ and $Y^{(p)}$, respectively. Set $\mathbf{Z}_i^{(p)}$ includes the nodes in partition $Y^{(p)}$ who are in the same community i: ($\mathbf{Z}_i^{(p)} = \{a | Y^{(p-1)}[a] = i\}$). $Q_{C_1 \times C_2}$ is a matrix with C_1 rows and C_2 columns. Each element Q_{ij} of this matrix represents the number of common nodes for each pair of communities $\mathbf{Z}_i^{(p-1)}$ and $\mathbf{Z}_j^{(p)}$. Then, the purity

measure is defined as [25, 34]

$$Purity(Y^p, Y^{p-1}) = \frac{\sum_{i=1}^{C_1} (max_{j=1}^{C_2}(Q_{ij}))}{n}. \quad (15)$$

The pseudo-code of proposed method is shown in Algorithm 1. The algorithm first maps the problem into the CRF model and constructs the corresponding graph and random variables. Then, it finds the final community labels for all the active nodes by applying the ICM procedure, iteratively.

ALGORITHM 1: CoDi-ICM

Input: Set of cascades over network ($\mathbf{O} = \{O_1, O_2, \ldots O_M\}$), Initial community labels (Y^0)
Output: Set of community labels (Y)
for *the* $O_m \in \mathbf{O}$ **do**
  for *the* $(v_m(i), t_m(i)) \in O_m$ **do**
    if $(t_m(i)) < \infty)$ **then**
      $\mathbf{U} = \mathbf{U} \cup v_m(i)$
      $X_i[m] = 1$
    end
  end
end
for $(i\&j \in \mathbf{U} | i \prec j)$ **do**
  $r[i][j]$=Solution to Equation (5) using X_i and X_j
end
$p \leftarrow 1$
while *not converged* $((Y^{(p)}, Y^{(p-1)}) \succ \varepsilon)$ **do**
  for *the* $s \in \mathbf{U}$ **do**
    ${Y_s}^p \leftarrow \arg\min_{Y_{N(s)}}(E({Y_s}^{(p)}, {Y_{N(s)}}^{(p-1)}))$ using Equation (12)
  end
  $p \leftarrow p + 1$
end

4.3 Estimating the Number of Communities

We estimate the optimal number of communities k, by utilizing the framework in [33]. We set the value of k to the constant value of 1, and increase it linearly until the stopping criteria is met as follows. For each value of k^t, we use the procedure introduced in 4.2, and discover the communities C^t. We compute the Normalized Mutual Information (NMI) of two sequential communities C^t and C^{t+1}, and refer to it as $z^{t,t+1}$. In each iteration, we increase the value of k linearly until $z^{t,t+1}$ achieves its local maximum. This value of k is only used at the initializing step of the proposed algorithm and its final value is obtained while updating the community labels in our algorithm. The pseudo-code for this procedure is presented in Algorithm 2.

5 EXPERIMENTAL EVALUATION

In this section, first we introduce the utilized datasets and employ different metrics for evaluating the proposed method. Then, the experimental setup is described for the proposed method[1] with

[1]Code is available at http://cnet.dml.ir/?page_id=728.

ALGORITHM 2: CoDi Algorithm

Input: Set of cascades over network ($\mathbf{O} = \{O_1, O_2, \ldots O_M\}$)
Output: Set of community labels (Y)
$k \leftarrow 1$
T: set of values
while *TRUE* **do**
  $t \leftarrow 1$
  Y^0=Initialize Y with $\{1, 2, \ldots, k\}$
  C^t=CoDi-ICM($\mathbf{O}$, Y^0)
  $\mathbf{T} = \mathbf{T} \cup NMI(C^t, C^{t-1})$
  if *(***T** *reaches local maximum)* **then**
    Return C^t
  end
  $t \leftarrow t + 1$
  $k \leftarrow k + stepsize$
end

two different initialization procedures. Finally, we choose the most related previous works for comparisons and evaluate them against the proposed model on synthetic networks with ground truth community structure having different artificial diffusion models as well as real-world social networks with real diffusion information.

5.1 Datasets

5.1.1 Synthetic Networks. We generated a synthetic network with build-in communities and then used different diffusion models to create cascade information over these networks, as follows.

Network generation: Different parameters for the simulated Lancichinetti–Fortunato–Radicchi (LFR) benchmark network [27] were used. Number of nodes: 1,000, average degrees of nodes: 15, maximum degrees of nodes: 50, exponent of power law distribution for the degree sequence: 2, t2:exponent of power law distribution for the community size 1, and the minimum and maximum sizes of communities: 20 and 50. The mixing parameter μ, defines the network community structure, and we generated five different synthetic networks according to five different values of μ (from 0.1 to 0.5).

Cascade generation: For the synthetic datasets, we used the exponential IC model [21] to simulate diffusion over the network.

5.1.2 Real Networks. We utilized different real network datasets, according to the following classifications.

(1) Cascade information and network connectivity: In the blogs, when a site refers to a post on the other website, an information diffusion process is created. The infection time of websites among 3.3 million sites by specific topics was collected over March 2011 until February 2012 [28]. Each topic contains phrases that are called memes. The spread of each meme corresponds to a cascade. We focused on four topics as described in Table 1.
(2) Cascade information and ground truth communities: Twitter dataset was collected over the period of September until November 2010 [15]. It traced the timestamps when a user retweeted. The information about a tweet includes the list of hash tags, the user source id that started the tweet, and the number of its hyperlinks. By considering these

Table 1. The Characteristics of Four Real Datasets with Diffusion and Structural Information

Network (Topic)	# Sites	# Links	# Cascades	Average length of cascades
Alqueda	1,142	44,911	8,955,039	2.86
NBA	2,056	142,738	11,955,546	2.90
LinkedIn	1,035	22,964	9,215,167	2.95
NewsOftheWorld	1,390	64,619	10,546,071	2.93

information, we processed the set of cascades over the network. The authors in [15] discovered two political communities among these users and partitioned users into right and left communities. We preprocessed this dataset by selecting the users that participated in at least three cascades. If a user retweet a tweet with unique hash tags, source id, and number of hyperlinks more than one time, we only recorded its first infection time. Some attributes of this dataset were considered. Number of users: 6,185, number of users at the right community: 3,217, number of users at the left community: 2,968, number of cascades over the network: 38,886, and average length of cascades: 2.73.

(3) Large dataset with cascade information: We utilized a large twitter dataset [50] over the period of 24 March 2012 to 25 April 2012, to assess the scalability of the proposed method. It contains the information of users, follower/following links, and tweets. We refer to the retweeting of a tweet as a cascade. After preprocessing the dataset and choosing the users with more than five actions on the site, the statistics of dataset are as follows: 132,102 users, 2,703,082 links, 957,685 different cascades with an average length of 2.79.

5.2 Performance Evaluation Metrics

In this article, the methods are compared in terms of accuracy and complexity. To evaluate the accuracy, different metrics are used on different datasets based on their characteristics, and the running time is considered as the complexity metric. In some datasets, the communities are available as ground truth, and the accuracy of detected communities can be measured with them. In some others, the communities are not specified explicitly, but the structure graph is available for them, and hence the accuracy of the detected communities can be evaluated based on these structural information. From the perspective of available information in datasets, the metrics can be classified as follows.

— *Datasets with ground truth communities:* Each detected community will be compared to the most similar community of the network. F-measure, NMI [33, 35], Adjusted Rand Index (ARI) [19, 23], and Error [19] are the most widely used metrics for evaluating the result of a community detection algorithm, when the real communities are known explicitly. The most challenging with these metrics is their weakness in ignoring the number of communities. Studies [44] show that when an algorithm partition the network in more number of communities, it can achieve higher NMI. Hence, we use a new metric named FNMI [2] for fair comparison and focus on number of detected communities, too.
— *Datasets without ground truth communities:* Usually, the real communities in a dataset are not explicitly determined. Modularity [37] and conductance [29] are the metrics that utilize the network connectivity to measure the goodness of detected communities.

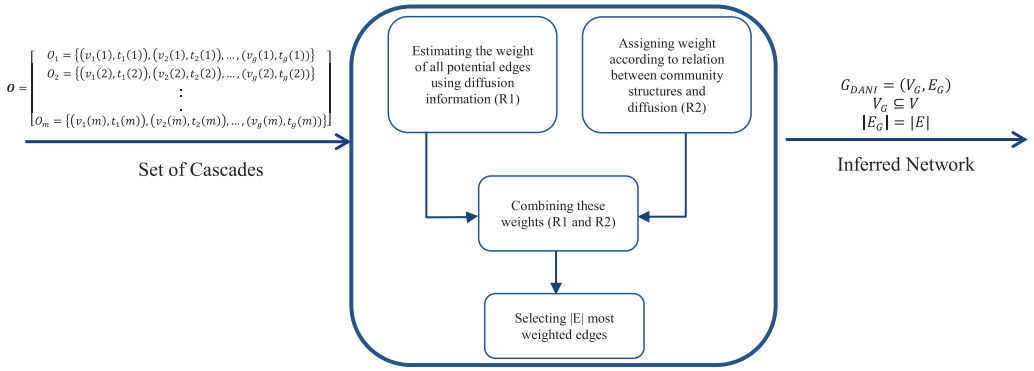


Fig. 2. Overview of DANI algorithm.

5.3 Experimental Setup

We used two different initialization procedures for the proposed method:

(1) R-CoDi (Random-based Community Diffusion):
We used a uniform random function to initialize Ys with the discrete values over the set $\{1, \ldots, k\}$.

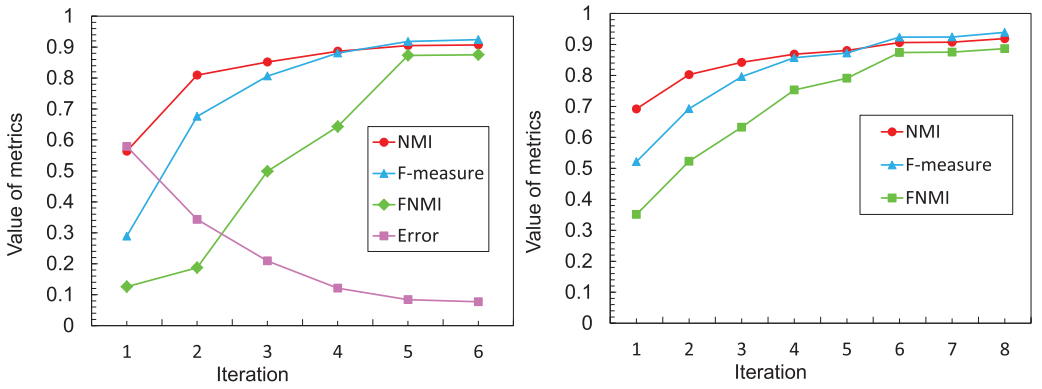
(2) D-CoDi (DANI-based Community Diffusion):
Here, we utilize part of information from the directed weighted probability graph (G_{DANI}) proposed in [43].

The purpose of DANI is to extract the links of a network by utilizing the timestamps appeared in diffusion cascades (**O**). Figure 2 provides an overview of how DANI works. The method contains four main steps: First, it models the problem by a Markov chain. By using the sequence of propagation in each cascade, it estimates the probability of existence for each edge from the difference between the infection labels. Second, according to joint behavior of nodes and their relation with information propagation, DANI assigns another weight to the potential edges. Third, it combines the weights of these two steps and computes the final existence probability for each edge. Up to this step, we have a weighted directed graph named ($G_{DANI} = \{\mathbf{V_G}, \mathbf{E_G}\}$). Fourth, the method chooses $|\mathbf{E}|$s (number of edges in the underlying network) with higher weighted edges and outputs inferred network ($G^{'}$). As a result, compared to other state of the art inference methods, it is shown that by applying structural community detection methods on the inferred network $G^{'}$, the results are more similar to detected communities of underlying network [43].

By running DANI algorithm up to the third step and producing G_{DANI}, we can compute the weighted degree of each node (s_i) as

$$s_i = \sum_{(i,j)\in G_{DANI}} w_{ij} + \sum_{(ji)\in G_{DANI}} w_{ji}. \tag{16}$$

Then, we choose k nodes with the highest value of s_i as the core nodes of k different communities. These k nodes ($H = \{i | i \in \mathbf{V_G}, k$ highest values of $s_i\}$) have the highest degree of participation in exchange of information with other nodes in the same community, and hence are considered to be the core nodes. Therefore, for the initialization step, we first assign the community label of those k nodes (nodes of set H) to the discrete values $\{1, \ldots, k\}$, and then traverse all the other nodes of G_{DANI} in one stage and assign the same community labels to their neighboring nodes ($N(i)$) according to the highest



(a) ICM procedure (b) Detecting the number of communities

Fig. 3. Performance metrics versus iteration plot for LFR synthetic network ($\mu = 0.1$ and #Cascades = 20,000).

weighted edge between them.

$$y^0_{i\in V_G, i\notin H} = \begin{cases} y^0_j & if(\underset{m\in N(i)}{\forall}((w_{ij}+w_{ji}) > (w_{im}+w_{mi})). \\ y^0_i & else \end{cases} \tag{17}$$

As a result only a few nodes may not have any label in initialization, but the other nodes are labeled more accurately than the R-CoDi.

The iterative methods looking for local optimum, can lead to better result if they starts with a good initialization. In total, CoDi method has two iterations: ICM procedure (Algorithm 1) and Detecting the number of communities (Algorithm 2). Figure 3 presents the impact of DANI initialization and the two iterations of CoDi on the total performance of proposed method.

5.4 Baselines for Comparison

The most related work to this article is [7], which proposed two different methods (C-IC and C-Rate) for community detection from diffusion information. We used the implementations provided by the authors that requires the user to specify the number of communities before the algorithms start to detect the communities. We specified the true number of communities for the datasets with known ground truth communities and tried different ranges for the number of communities in the other datasets to achieve best performances for these methods.

As discussed before, using network inference methods such as [43] that preserves the community structure with a community detection method, depends highly on knowing the true number of the links $|\mathbf{E}|$ of the underlying network $G = (\mathbf{V}, \mathbf{E})$ which may vary up to $\binom{|\mathbf{V}|}{2}$ links. We tested the impact of number of links of the inferred network on the performance of a community detection method by using the LFR-benchmark synthetic dataset ($\mu = 0.1$) with 1,000 nodes and 7,965 links when 12,000 different cascades were generated as described in Section 5.1. First, the DANI algorithm [43] was ran on the data, and then a community detection method named Louvain [11] was applied on the output of inferred network. As it is evident from Figure 4, the inference method needs the actual number of network links to obtain the best-matched communities in the underlying network. The high NMI and F-measure with low error occurred when the assumed number of links was close to the actual number of links (7,965) as shown by the dashed lines in the plot. However, none of the existing methods can estimate the true number of network links from

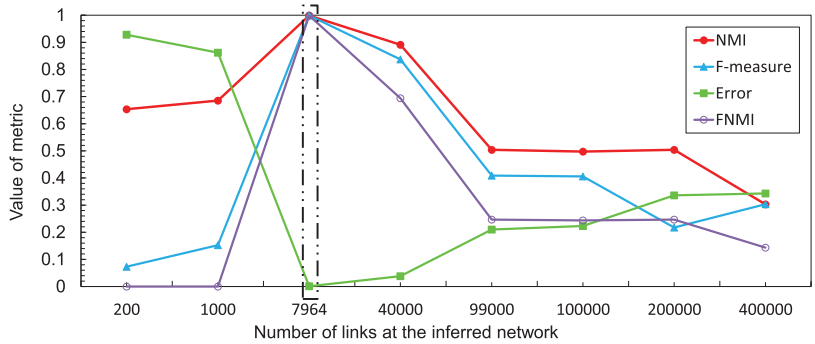


Fig. 4. Comparing the performance of an inference algorithm for different number of links on LFR dataset with $\mu = 0.1$, #Nodes = 1,000, #Links = 7,965 and #Cascades = 12,000. The best match is specified by dashed lines that is close to the actual number of underlying network links.

the diffusion information. Because of the limitations for inference methods that was described in Section 2, it was not feasible to compare the proposed method with the network inference algorithms, and only the most related works (C-IC and C-Rate) were used as comparison baselines. Using network inference methods to reconstruct network structure from the observed cascades is explicitly forbidden in popular social networks [8]. Therefore, in practice using an inference method and then applying a community detection algorithm on its output is not feasible. However, knowing the communities can be useful in many applications [8]. Despite the aforementioned fact, we test two inference methods, i.e., DANI [43] and Netrate [20] on different datasets, in Section 5.5.

5.5 Simulation Results

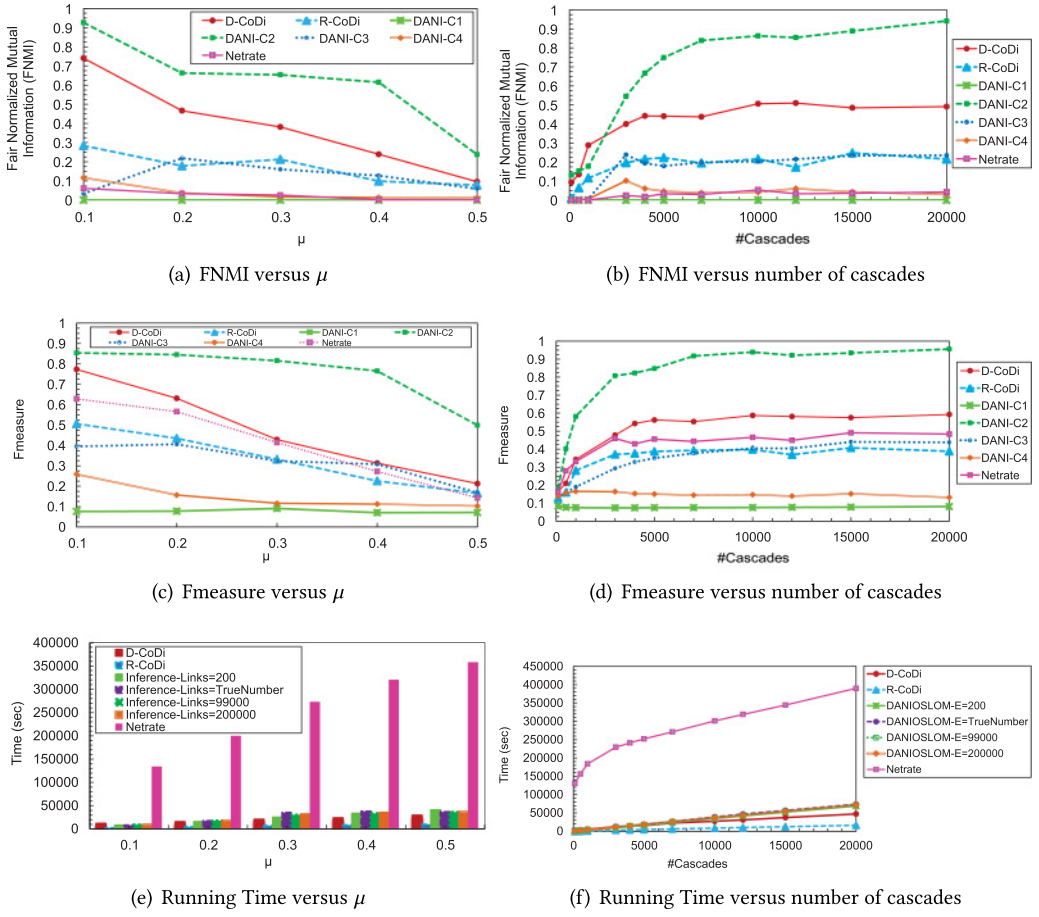
We evaluated the performance of CoDi on different synthetic and real networks by using various community detection measures. The results indicate that the two configurations of proposed method (D-CoDi and R-CoDi) outperform the previous state of the art methods (C-Rate and C-IC). To achieve a fair compression, all the methods were coded in Java and the runs were performed on a server with 20GB of memory, 8 CPUs, and 2 cores. We have reported all the results based on the average of 10 different runs with different random initializations for R-CoDi.

— *Synthetic networks*

— Network inference methods

The community detection algorithms can work well on the output of network inference methods that are preserving the community structure of underlying network. Figure 5 shows the performance of two inference methods called DANI and Netrate with the Louvain community detection algorithm in terms of FNMI, F-measure and running time. We used a set of observed cascades as the input for both methods. In addition, DANI needs to know the number of edges that needs to be inferred, while Netrate outputs a subset of probable inferred links with weights more than a minimum threshold. For DANI, we performed the simulations with different number of links as inputs, as shown in Table 2. In addition, we checked different values of threshold and reported the best performance for Netrate.

The results show that when the order of given number of links for DANI is not close to the actual values, the performance of DANI degrades. Although Netrate is a state of



(a) FNMI versus μ (b) FNMI versus number of cascades

(c) Fmeasure versus μ (d) Fmeasure versus number of cascades

(e) Running Time versus μ (f) Running Time versus number of cascades

Fig. 5. Comparison of DANI with different input settings, and Netrate when a community detection is applied on their outputs against the proposed method (CoDi) for synthetic networks.

Table 2. Different Input Settings for DANI

Case name	Setting (number of input inferred links)
DANI-C1	200
DANI-C2	7,965 (actual number)
DANI-C3	99,000
DANI-C4	200,000

Note: We used various orders of values for the number of links, assuming there is no prior knowledge about the number of links for a network.

the art method for network inference, but when a community detection is applied on its output, the results are not accurate. Moreover, one of the major problems of applying a community detection algorithm over an inference method such as Netrate is the required high running time of the algorithm, as shown in Figure 5.

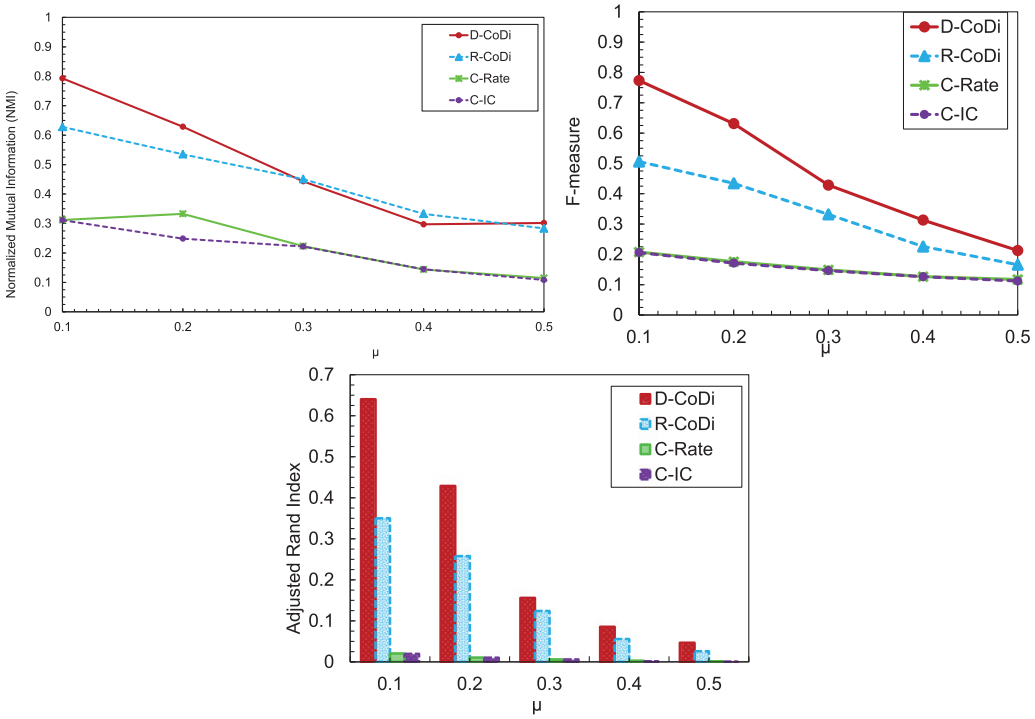


Fig. 6. The accuracy of community detection methods across different community structures for the LFR benchmark.

—Diffusion-based community detection methods

We measured the accuracy of detected communities against the ground truth communities over five different LFR networks. Dependency (accuracy and running time) to the number and length of cascades were the other measures that we considered in our experimental studies.

Accuracy of community detection: Figure 6 illustrates three metrics for different datasets (μs). Simultaneously achieving higher values for F-measure, NMI, and ARI corresponds to a better performance. On average, the proposed methods D-CoDi and R-CoDi gained 1.725 times and 1.313 times higher accuracy in the detection procedure over the previous works C-Rate and C-IC, respectively.

Number of communities: For the synthetic datasets, we know the ground truth communities. Hence, for C-IC and C-Rate methods, we used the true value for the number of communities as a prior knowledge. On the other hand, the two initializations of the proposed method detect the number of communities during their procedures. Therefore, the number of detected communities can not be used as a metric for comparing the proposed method with the previous works, in this case. Figure 7 demonstrates that the D-CoDi can detect more communities that are similar to the ground truth communities compared to R-CoDi.

Running time: The running time of different methods on synthetic datasets can be considered as a key performance metric to identify the scalability of these methods in

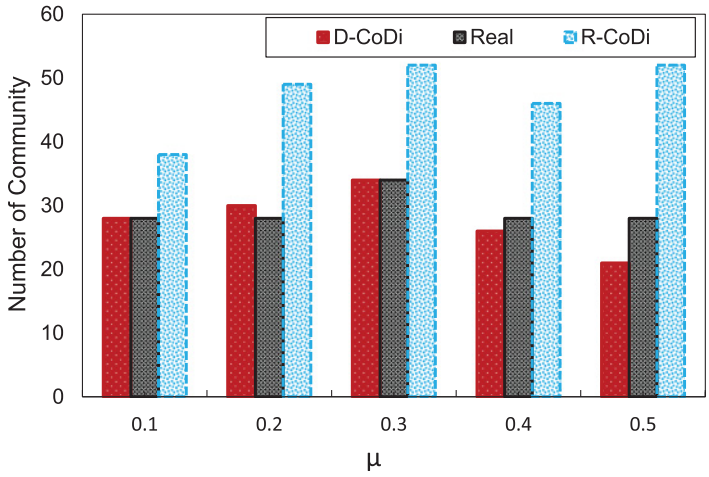


Fig. 7. Comparing the number of discovered communities for synthetic datasets by the two configuration of proposed method (D-CoDi and R-CoDi) against the actual LFR network communities.

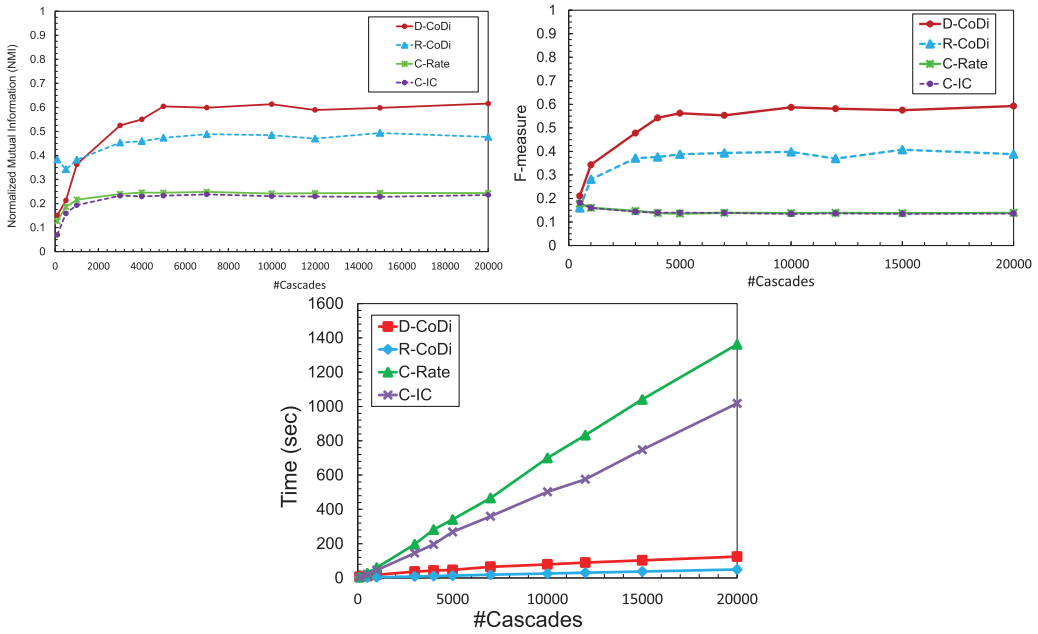


Fig. 8. The accuracy and running time of community detection methods versus the number of cascades over the synthetic network.

handling large datasets. The running time on the LFR benchmark for different number of cascades is shown in Figure 8. In general, the running time of two configurations of the proposed method is lower than the others. The lower running time of R-CoDi compared to D-CoDi is due to the lower complexity of the R-CoDi initialization step. The experimental results verify the capability of the proposed method in detection of communities with large volume of data while maintaining lower running time and higher accuracy,

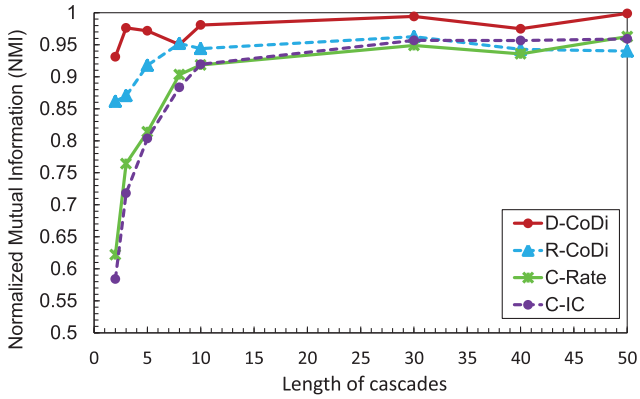


Fig. 9. The accuracy of community detection methods from diffusion information for different length of cascades over the network.

compared to the existing methods. Moreover, as the number of cascades increases, the running time of CoDi changes smoothly, while the running times of C-IC and C-Rate increase dramatically.

Cascade dependency: We studied the dependency of algorithms on the length and number of cascades as two important performance criteria in different methods. As Figure 8 shows, for the C-Rate and C-IC methods, increasing the number of cascades almost do not cause any significant changes in NMI and F-measure metrics. However, by increasing the number of cascades, CoDi can detect the communities more accurately.

To evaluate the dependency on the length of cascades, we examined the LFR benchmark with $\mu = 0.3$ for 12,000 different cascades. Figure 9 shows that the performance of CoDi does not depend on the length of cascades, while the performance of previous methods highly depend on the length of cascades, and they can only achieve acceptable accuracy in community detection for long cascades that pass the boundaries between the communities. However, in reality the cascades over social networks usually have low lengths [1, 31], and the observations from the Twitter dataset have shown that the spread of information over network takes about one or two steps in average [4].

In summary, the proposed method is independent of the length of propagating cascades, but it depends on the number of cascades, and at a fixed number of cascades it acts more accurately than the previous works. On the other hand, previous methods are heavily dependent on the length of cascades and increasing the number of cascades has no noticeable effect on their performance. Considering the diffusion behavior in real networks, the length of the spread for a contagion is not adjustable, while large amount of cascades can be distributed over the network to increase the available information. Therefore, the independency from the length of cascades can be considered as an advantage for the proposed method against the previous works.

—*Real networks*

—Network inference methods

As Figure 10 shows, D-CoDi outperforms the inference method by obtaining high modularity and low conductance when number of inferred links is far from actual values.

—Diffusion-based community detection methods

Accuracy and running time are the key performance metrics in comparing the algorithms on real networks.

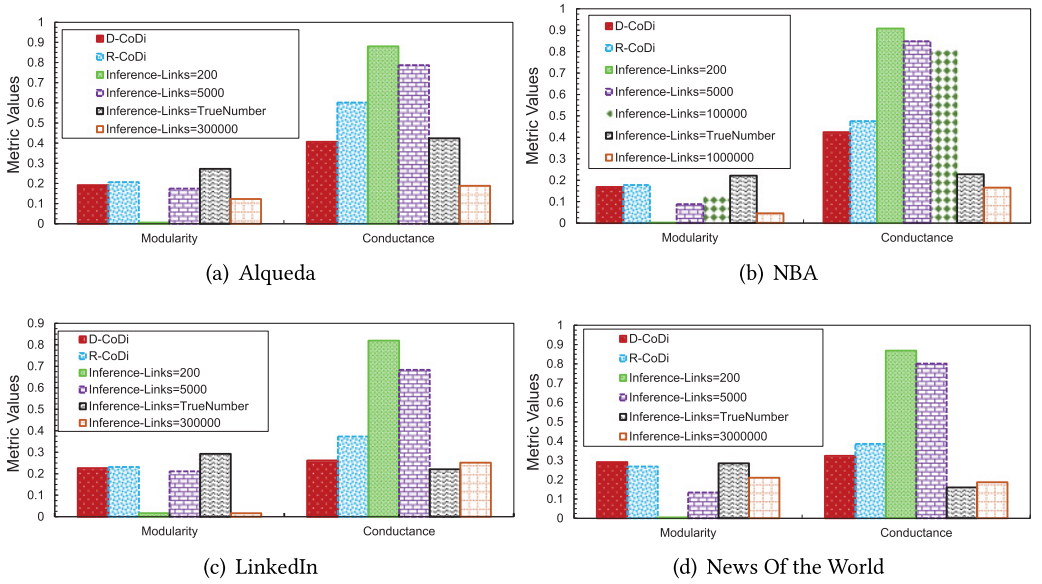


Fig. 10. Performance comparison of CoDi with the network inference method using different number of inferred links as input.

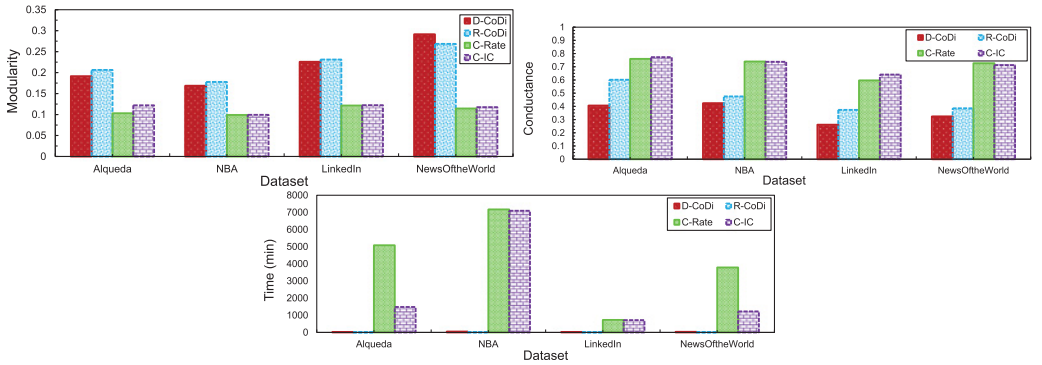


Fig. 11. Evaluation of modularity, conductance, and running time metrics over four different Memtracker datasets.

Cascade information and network connectivity: Since the blogosphere data does not provide any knowledge about the ground truth communities, the second category of metrics presented in Section 5.2 was used for evaluations. In this case, it is common to use the metrics such as modularity and conductance, simultaneously. Therefore, high modularity with low conductance indicates an appropriate performance for a community detection method.

The results of different methods for the blogs data is presented in Figure 11. Although both R-CoDi and D-CoDi achieve high modularity, the conductance for D-CoDi is lower than R-CoDi. The behavior of C-IC and C-Rare are nearly identical for both metrics. Therefore, the performance of these community detection methods can be sorted from best to worth as D-CoDi, R-CoDi, C-Rate, and C-IC on the four real Memtracker datasets

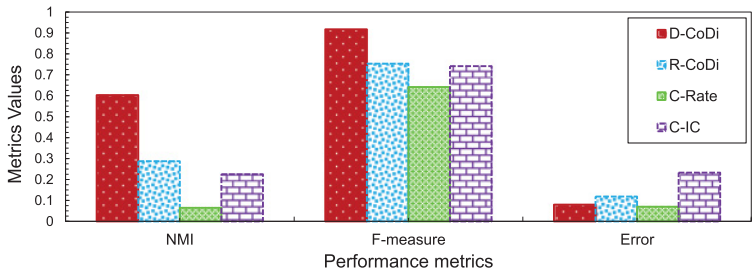


Fig. 12. Community detection metrics for discovering communities from real diffusion in Twitter when compared against the ground truth communities.

Table 3. Results of the Proposed Methods on the Large Twitter Dataset

Configuration	R-CoDi	D-CoDi
Time (hour)	2.2	16.17
# Community	46	199
Avg modularity	0.199	0.214
Avg conductance	0.734	0.60

of Figure 11. Moreover, the proposed methods have much less running time than the previous methods (C-Rate and C-IC).

Cascade information and ground truth communities: For the Twitter dataset, the actual communities of the network are known, and we would like to detect them by using the diffusion information. The running time of the competing methods in seconds are as follows: C-Rate: 669.1 seconds, C-IC: 342.0 seconds, D-CoDi: 38.59 seconds, and R-CoDi 13.35 seconds. Therefore, as shown in Figure 12, the proposed method (D-CoDi) can detect the communities with higher accuracy while maintaining low running time.

Large dataset with cascade information: The proposed methods were able to detect the communities of the large Twitter dataset in a reasonable time (2–16 hours), while the baseline methods C-Rate and C-IC did not produce any outputs even after three weeks of running. Therefore, the results of the baseline methods are not reported in Table 3. The indicated running times in the table verifies the scalability of the proposed methods. The results in Table 3 also indicates that high modularity with low conductance is the proper evidence for better performance of the proposed methods. Moreover, the results on synthetic networks with specific groups in Figure 7 indicates that the number of communities discovered by D-CoDi is closer to the actual communities compared to R-CoDi.

6 CONCLUSION

In this article, we proposed a method based on the CRF model for detecting community structures directly from the infection times of cascades information over the network without having any prior knowledge about the network structure. The proposed method does not suffer from the complexity problems that exist in structural learning and graphical model inference methods, and can detect the communities with high accuracy while maintaining high scalability. Moreover,

the good performance of this method on short cascades makes it a suitable method for detecting communities in real datasets. As the future works, we may consider supporting the overlapping communities by utilizing the proper membership distributions to the proposed algorithm, and considering the dynamic nature of the network to exploit dynamic communities or dynamic diffusion processes.

ACKNOWLEDGMENTS

We would like to thank Nicola Barbieri for the source code of his article. We also appreciate Mahsa Ghorbani of Advanced ICT Innovation Center, Sharif University of Technology, for her valuable comments.

REFERENCES

[1] Bruno D. Abrahao, Flavio Chierichetti, Robert Kleinberg, and Alessandro Panconesi. 2013. Trace complexity of network inference. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 491–499.

[2] Alessia Amelio and Clara Pizzuti. 2015. Is normalized mutual information a fair measure for comparing community detection methods? In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015 (ASONAM'15)*. ACM, New York, NY, 1584–1585.

[3] David Asley and Jon Kleinberg. 2010. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World*. Cambridge University Press, New York, NY.

[4] Eytan Bakshy, Jake M. Hofman, Winter A. Mason, and Duncan J. Watts. 2011. Everyone's an influencer: Quantifying influence on Twitter. In *Proceedings of the 4th International Conference on Web Search and Data Mining (WSDM'11)*. ACM, New York, NY, 65–74.

[5] Raquel A. Baños, Javier Borge-Holthoefer, and Yamir Moreno. 2013. The role of hidden influentials in the diffusion of online information cascades. *EPJ Data Science* 2, 1 (2013), 1–16.

[6] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2013. Cascade-based community detection. In *Proceedings of the 6th International Conference on Web Search and Data Mining (WSDM'13)*. ACM, New York, NY, 33–42.

[7] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2013. Influence-based network-oblivious community detection. In *Proceedings of the 13th International Conference on Data Mining (ICDM'13)*. IEEE, 955–960.

[8] Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. 2016. Efficient methods for influence-based network-oblivious community detection. *ACM Transactions on Intelligent Systems and Technology* 8, 2 (2016), 32:1–32:31.

[9] Julian Besag. 1974. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society: Series B (Methodological)* 36, 2 (1974), 192–236.

[10] Julian Besag. 1986. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society: Series B (Methodological)* 48, 3 (1986), 259–302.

[11] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. L. J. S. Mech. 2008. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* 2008, 10 (2008), P10008.

[12] Abdelhamid Salah Brahim, Bndicte Le Grand, and Matthieu Latapy. 2012. Diffusion cascades: Spreading phenomena in blog network communities. *Parallel Processing Letters* 22, 1 (2012).

[13] Yen-Liang Chen, Ching-Hao Chuang, and Yu-Ting Chiu. 2014. Community detection based on social interactions in a social network. *Journal of the Association for Information Science and Technology* 65, 3 (2014), 539–550.

[14] Sébastien Combéfis. 2007. *Viral marketing and Community Detection Algorithms*. Ph.D. Dissertation. Université Catholique de Louvain Faculté des sciences appliquées Département díngénierie informatique.

[15] Michael Conover, Jacob Ratkiewicz, Matthew Francisco, Bruno Gonçalves, Alessandro Flammini, and Filippo Menczer. 2011. Political polarization on twitter. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media (ICWSM'11)*. Barcelona, Spain.

[16] Himel Dev, Mohammed Eunus Ali, and Tanzima Hashem. 2014. User interaction based community detection in online social networks. In *Proceedings of the 19th International Conference on Database Systems for Advanced Applications (Lecture Notes in Computer Science)*, Vol. 8422. Springer, Bali, Indonesia, 296–310.

[17] Nan Du, Le Song, Ming Yuan, and Alex J. Smola. 2012. Learning networks of heterogeneous influence. In *Proceedings of Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2780–2788.

[18] Motahhare Eslami, Amirhossein Aleyasen, Roshanak Zilouchian Moghaddam, and Karrie G. Karahalios. 2014. Evaluation of automated friend grouping in online social networks. In *Proceedings of the Extended Abstracts on Human Factors in Computing Systems (CHI EA'14)*. ACM, New York, NY, 2119–2124.

[19] Santo Fortunato. 2010. Community detection in graphs. *Physics Reports* 486 (2010), 75–174.
[20] Manuel Gomez-Rodriguez, David Balduzzi, and Bernhard Schlkopf. 2011. Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*. Omnipress, Bellevue, Washington, 561–568.
[21] Manuel Gomez Rodriguez, Jure Leskovec, and Andreas Krause. 2010. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'10)*. ACM, New York, NY, 1019–1028.
[22] Adrien Guille, Hakim Hacid, Ccile Favre, and Djamel A. Zighed. 2013. Information diffusion in online social networks: A survey. *SIGMOD Record* 42, 2 (2013), 17–28.
[23] Eyke Hüllermeier and Maria Rifqi. 2009. A fuzzy variant of the rand index for comparing clustering structures. In *Proceedings of the IFSA/EUSFLAT Conference*. 1294–1298.
[24] Masahiro Kimura, Kazumasa Yamakawa, Kazumi Saito, and Hiroshi Motoda. 2008. Community analysis of influential nodes for information diffusion on a social network. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'08)*. Dept. of Electron. & Inf., Ryukoku Univ., Otsu, 1358–1363.
[25] Vincent Labatut. 2015. Generalized measures for the evaluation of community detection methods. In *International Journal of Social Network Mining* 2, 1 (2015), 44–63.
[26] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML'01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, 282–289.
[27] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. 2008. Benchmark graphs for testing community detection algorithms. *Physical Review E* 78, 4 (2008), 046110.
[28] Jure Leskovec. 2011. Web and blog datasets. *Stanford Network Analysis Project, 2011.* [Online]. Available: http://snap.stanford.edu/infopath/data.html.
[29] Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. 2008. Statistical properties of community structure in large social and information networks. In *Proceeding of the 17th International Conference on World Wide Web (WWW'08)*. ACM, New York, NY, 695–704.
[30] Jure Leskovec, Kevin J. Lang, and Michael Mahoney. 2010. Empirical comparison of algorithms for network community detection. In *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*. ACM, New York, NY, 631–640.
[31] Jure Leskovec, Ajit Singh, and Jon Kleinberg. 2006. Patterns of influence in a recommendation network. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. Springer, Berlin, 380–389.
[32] Yanhua Li, Wei Chen, Yajun Wang, and Zhi-Li Zhang. 2013. Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships. In *Proceedings of the 6th International Conference on Web Search and Data Mining (WSDM'13)*. ACM, Rome, Italy, 657–666.
[33] Zhifang Li, Yanqing Hu, Beishan Xu, Zengru Di, and Ying Fan. 2012. Detecting the optimal number of communities in complex networks. *Physica A: Statistical Mechanics and Its Applications* 391, 4 (2012), 1770–1776.
[34] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. 2008. *An Introduction to Information Retrieval*. Cambridge University Press.
[35] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK.
[36] Azadeh Nematzadeh, Emilio Ferrara, Alessandro Flammini, and Yong-Yeol Ahn. 2014. Optimal network modularity for information diffusion. *Physical Review Letters* 113, 8 (2014), 088701.
[37] Mark E. J. Newman. 2004. Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems* 38, 2 (2004), 321–330.
[38] M. E. Newman. 2006. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America* 103, 23 (June 2006), 8577–8582.
[39] M. E. J. Newman and M. Girvan. 2004. Finding and evaluating community structure in networks. *Physical Review* E 69, 026113 (2004).
[40] Huan-Kai Peng, Jiang Zhu, Dongzhen Piao, Rong Yan, and Ying Zhang. 2011. Retweet modeling using conditional random fields. In *Proceedings of the ICDM Workshops*, Myra Spiliopoulou, Haixun Wang, Diane J. Cook, Jian Pei, Wei Wang, Osmar R. Zaane, and Xindong Wu (Eds.). IEEE, 336–343.
[41] Michel Plantié and Michel Crampes. 2013. Survey on social community detection. *Computer Communications and Networks* (2013), 65–85.
[42] Gábor Rácz, Zoltán Pusztai, Balázs Kósa, and Attila Kiss. 2015. An improved community-based greedy algorithm for solving the influence maximization problem in social networks. *Annales Mathematicae et Informaticae* 44 (2015), 141–150.

[43] Maryam Ramezani, Hamid R. Rabiee, Maryam Tahani, and Arezoo Rajabi. 2017. DANI: A fast diffusion aware network inference algorithm. *arXiv preprint arXiv:1706.00941* (2017).

[44] Simone Romano, James Bailey, Xuan Vinh Nguyen, and Karin Verspoor. 2014. Standardized mutual information for clustering comparisons: One step further in adjustment for chance. In *Proceedings of the 31st International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*. 1143–1151.

[45] Shaghayegh Sahebi and William Cohen. 2011. Community-based recommendations: A solution to the cold start problem. In *Proceedings of the Workshop on Recommender Systems and the Social Web, RSWEB (RecSys'11)*. ACM, Chicago.

[46] Kazumi Saito, Ryohei Nakano, and Masahiro Kimura. 2008. Prediction of information diffusion probabilities for independent cascade model. In *Proceedings of KES (3)*, Ignac Lovrek, Robert J. Howlett, and Lakhmi C. Jain (Eds.), vol. 5179. Springer, 67–75.

[47] Zak Stone, Todd Zickler, and Trevor Darrell. 2008. Autotagging facebook: Social network context improves photo annotation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW'08)*. IEEE, 1–8.

[48] Charles Sutton and Andrew McCallum. 2011. An introduction to conditional random fields. *Foundations and Trends R in. Machine Learning* 4 (2011), 267–373.

[49] Yu Wang, Gao Cong, Guojie Song, and Kunqing Xie. 2010. Community-based greedy algorithm for mining top-K influential nodes in mobile social networks. In *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining (KDD'10)*. ACM, New York, NY, 1039–1048.

[50] Lilian Weng, Filippo Menczer, and Yong-Yeol Ahn. 2013. Virality prediction and community structure in social networks. *Scientific Reports* 3 (2013), 2522.

Received January 2016; revised February 2017; accepted June 2017