Lim *et al. Computational Social Networks* (2015) 2:8
DOI 10.1186/s40649-015-0015-6

Computational Social Networks
a SpringerOpen Journal

RESEARCH Open Access

Classifying latent infection states in complex networks

Yeon-sup Lim[1*], Bruno Ribeiro[2] and Don Towsley[1]

*Correspondence:
ylim@cs.umass.edu
[1] School of Computer Science, University of Massachusetts Amherst, 140 Governors Drive, 01003 Amherst, USA
Full list of author information is available at the end of the article

Abstract

Algorithms for identifying the infection states of nodes in a network are crucial for understanding and containing epidemics and cascades. Often, however, only the infection states of a small set of nodes are known. Moreover, the length of time that each node has been infected is also unknown. This missing data – infection states of most nodes and the infection times of the observed infected nodes – poses a challenge to the study of real-world epidemics/cascades.

In this work, we develop techniques to identify the latent infected nodes in the presence of missing infection time-and-state data. Based on likely epidemic paths predicted by a simple susceptible-infected epidemic model, we propose a measure (*infection betweenness centrality*) for uncovering unknown infection states. We evaluate infection state classifiers based on Naive Bayes, Naive Bayes with kernel density estimation, and decision trees, in combination with features including infection betweenness centrality and different centrality measures. Our experimental results show that among the set of features that include degree and different centrality measures, infection betweenness centrality is the most effective feature for identifying latent infected nodes.

Keywords: Epidemics; Information cascades; Information diffusion

Introduction

Networks are underlying mediums for the spread of epidemics such as diseases, rumors, and computer viruses. Determining the infection states of network nodes is the first step to taking corrective or preventive action to stop or slow the spread of an epidemic. Unfortunately, the infection states of network nodes are often unknown; for example: in the spread of computer malware (say, a contaminated email attachment) in a large organization, a network IT specialist will likely only inspect the computers of users that open trouble tickets; a similar problem occurs with the spread of rumors over online social networks. Hence, the problem of effectively identifying the infection states of unobserved nodes given a set of observed nodes is of central importance in the study of infection cascades.

Our research question is: Given a set of nodes with known infection states and the network topology, can we correctly uncover the unknown infection states of the remaining nodes? In this work, we consider a network where an epidemic starts from a single source. Each node appears in one of two states: (i) susceptible, capable of being infected, (ii) infected, able to spread the epidemic further. We also assume that the infection states

of a subset of nodes are known and the full network structure (adjacency matrix) is available.

Let G be an undirected graph with node set V and adjacency matrix $\mathbf{A} = [A_{ij}]$. Suppose that an epidemic starts at a single node (denoted 'source') and propagates to neighbors in G. Let $X_i(t) \in \{0, 1\}$ denote the state of node $i \in V$ at time $t \geq 0$ where $X_i(t) = 0$ means node i is susceptible and $X_i(t) = 1$ that it is infected. Assume that an infected node contaminates neighbors at rate λ. Then,

$$X_i(t) : 0 \rightarrow 1 \quad \text{at rate } \lambda \sum_{j \in V} X_j(t) A_{ji}.$$

Assume that at some arbitrary time there are l nodes with observed infection state $L = \{(1, X_1), \ldots, (l, X_l))\}$. The states of the remaining $u = |V| - l$ nodes are unknown with l typically much smaller than u. Given the set of observed nodes L, our goal is to correctly assign an infection state X_i to node $i = l + 1, \ldots, l + u$.

Our contributions are as follows:

- We introduce a measure for estimating the states of unobserved nodes, denoted *infection betweenness centrality*. We evaluate classifiers based on Naive Bayes, Naive Bayes with kernel density estimation [1], and decision trees [2], in combination with features including infection betweenness centrality and different centrality measures on inferring unknown states. Our experiments show that the inclusion of infection betweenness centrality significantly contributes to the quality of infection state classification.
- We investigate the impact of parameters, such as the degree distribution, on the estimation performance of the classifiers. Our experiments show that infection state classification becomes more accurate as the degree distribution of network becomes less skewed.

The remainder of the paper is organized as follows: 'Measuring infection state' section proposes infection betweenness centrality. In 'Infection state estimation' section, we introduce infection state classifiers using infection betweenness centrality and different centrality measures. 'Experimental results' section represents the experimental results about the performance of classifiers with infection betweenness centrality. 'Related work' section reviews the related literature. Finally, 'Conclusion' section presents our conclusions and future work.

Measuring infection state

Propagation properties

Under the assumption that an epidemic propagates from a single source to neighboring nodes following the susceptible-infected (SI) model [3], we identify the following properties.

Let S_o denote the set of observed susceptible nodes and I_o the set of observed infected nodes.

- **Property 1**: If removing all nodes in S_o from the network disconnects the network, then one of the disconnected components contains all of the infected nodes
- **Property 2**: Let $S \in V$ be a cut set that divides I_o into multiple components, then at least one node in S is infected

To prove the correctness of the properties, we state and prove the following theorems:

Theorem 1. *In the SI model, all infected nodes reside in one connected component.*

Proof. In the SI model, a node can be infected only by its infected neighbors. Therefore, all infected nodes are connected to the source node through at least one path that consists of infected nodes, i.e., at least one path (via the source node) exists between every pair of infected nodes. This means that all infected nodes are in one connected component. □

Theorem 2. *In the SI model, removing all susceptible nodes yields a single component consisting only of infected nodes.*

Proof. As shown in the proof of Theorem 1, every pair of infected nodes is connected by a path consisting only of infected nodes. Removing all susceptible nodes in the network cannot disconnect such a path between the pair of infected nodes. Thus, two infected nodes are always connected to each other even after removing some or all susceptible nodes. Since every pair of infected nodes is connected after susceptible nodes are removed, by definition they belong to a single connected component. Consequently, Property 1 is proved by Theorem 1 and Theorem 2. □

Using Theorem 1, we prove Property 2 as follows:

Proof. of Property 2 Based on the Theorem 1, the nodes in I_o reside in one connected component. Let v_1 and v_2 denote nodes in I_o that belong to different components disconnected by cut set S. Since v_1 and v_2 are in one connected component before the connected component is divided by the nodes in S, a path that consists of infected nodes exists between v_1 and v_2. By the definition of cut set, any path between v_1 and v_2 includes at least one node in S. That is, at least one node in S is infected. □

As an example, consider the topology shown in Figure 1. Removal of node 5, which is observed and susceptible, divides the graph into two components, $\{1, 2, 3, 4\}$ and $\{6, 7\}$. Only component $\{1, 2, 3, 4\}$ contains infected nodes (Property 1): we can determine that nodes 6 and 7 are susceptible (deterministic susceptible nodes). Observed infected nodes $\{1, 4\}$ reside in two components when nodes 2 and 3 are removed, which have not been observed. Because the removal of nodes 2 and 3 places infected nodes 1 and 4 in distinct

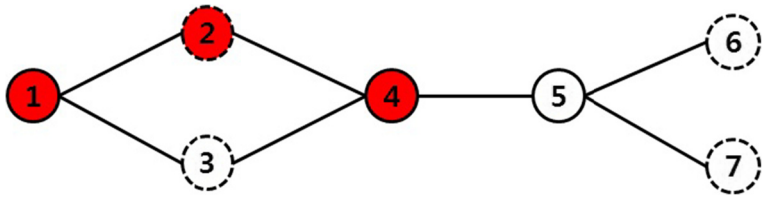

Figure 1 Example topology: Red nodes and white nodes represent infected and susceptible nodes, respectively. Dotted circles (nodes 2, 3, 6, and 7) show nodes with unknown infection state, and full circles (nodes 1, 4, and 5) show nodes with known infection state.

components, node 2 and/or node 3 must be infected (Property 2). Using Property 1, we can reduce the number of nodes whose states are unknown by ignoring nodes in components that can be isolated by healthy nodes. In the rest of this paper, we focus only on the nodes whose states remain unknown after applying Property 1. Although Property 2 does not provide a direct way to determine unknown infection states, it points to the importance of particular nodes in possibly infecting known infected nodes.

These properties hold under the assumption of the SI model and the existence of only one source of the infection. If multiple sources exist, infected nodes can be present in multiple connected components; thus, a partially observed set of infected nodes is not guaranteed to reside in one connected component. If an epidemic propagates according to the susceptible-infected-recovered (SIR) model [3] where a node can recover and cannot infect neighbors after the recovery, the properties hold by treating recovered nodes as infected nodes. To this end, we need to know the history of infected state changes. Since we seek to solve the problem without any temporal information, we focus on the SI model in the rest of this paper. In future work, we will investigate propagation properties that can be utilized for classifying latent infection states under more extensive assumptions such as multiple sources and different epidemic models.

Infection betweenness centrality

Let G' be a subgraph constructed by removing all nodes that are susceptible according to Property 1. The number of paths of length $r \geq 0$ between a pair of nodes (i, j) in G', $N_{ij}^{(r)}$, is

$$N_{ij}^{(r)} = \left(\mathbf{A}^r\right)_{ij},$$

where $\mathbf{A}$ is the adjacency matrix of G'.

Suppose that each path of length r is given a weight $\alpha > 0$; then

$$N_{ij} = \sum_{r=0}^{\infty} \alpha^r N_{ij}^{(r)} = \sum_{r=0}^{\infty} \left(\alpha^r \mathbf{A}^r\right)_{ij}.$$

is the weighted sum of paths from i to j. We can write this expression in matrix notation

$$\mathbf{N} = \sum_{r=0}^{\infty} \alpha^r \mathbf{A}^r = (\mathbf{I} - \alpha\mathbf{A})^{-1}.$$

Let $N_u(i, j)$ denote the weighted sum of paths from node i to j through node u. Given $G'' = G' - \{u\}$, we calculate $N_u(i, j)$ by subtracting the weighted sum of paths from i to j in G'' from the sum in G'; however, constructing G'' and performing the inverse operation for $\mathbf{N}$ of each G'' requires additional computation. Therefore, we resort to a simple approximation $N_u(i, j) \approx \mathbf{N}_{iu} \times \mathbf{N}_{uj}$. By summing over all possible nodes $u \in V$, we define

$$\mathbf{M}_{ij} = \sum_{u \in V} N_u(i, j) = \sum_{u \in V} \mathbf{N}_{iu}\mathbf{N}_{uj} = (\mathbf{N}^2)_{ij} \qquad \forall (i, j) \in E.$$

We define the infection betweenness of node u between two infected nodes i and j as:

$$B_u(i, j) = \frac{N_u(i, j)}{\mathbf{M}_{ij}},$$

which is the fraction of the weighted sum of paths from i to j through u over the total weighted sum of paths from i to j; thus, node u is more likely to be infected by node i or j as $B_u(i, j)$ increases. We assume that $B_u(i, j)$ is the probability that node u is contaminated by

node i or j and define *infection betweenness centrality* of node u given the set of observed infected nodes as:

$$P(u) = 1 - \prod_{i,j \in I_o,\ i \neq j} \left(1 - B_u(i,j)\right), \tag{1}$$

where I_o is the set of observed infected nodes.

Infection state estimation

Now, we introduce several classifiers that use infection betweenness centrality and other node features.

Node features

As features for building classifiers, we consider five node characteristics that are available using information regarding the network topology and the observed nodes in addition to infection betweenness centrality, P defined in Equation 1. The first five features are the following: degree normalized by the maximum degree in the network D, observed infected neighbor ratio R (the fraction of infected neighbors in observed neighbors), betweenness centrality $C^{(b)}$, closeness centrality $C^{(c)}$, and eigenvector centrality $C^{(e)}$ [3].

Algorithms for building classifiers

We choose four algorithms for building infection state classifiers, which are often used for a classification problem such as traffic classification [4-6], as follows:

Naive-Bayes (NB, NBK). Under the assumption that there are no correlations between features given the class (infection state) variable, **NB** derives a conditional probability for the relationships between the attribute values and the class. To this end, **NB** must estimate the distribution of feature values. However, continuous valued features can have a large range and it is hard to derive the unbiased distribution from the observed frequency distribution. In order to address this problem, **NB** assumes that values of each feature follows a particular distribution such as a Gaussian distribution [1].

We evaluate the performance of **NB** to classify the states of unobserved nodes as well as Naive Bayes using kernel density estimation (**NBK**); kernel density estimation uses multiple (Gaussian) distributions, and is generally more effective than using a single (Gaussian) distribution [1].

C4.5 Decision tree (C4.5) constructs a decision tree model in which each internal node represents a test on features, each branch an outcome of the test, and each leaf node a class label [2]. In order to use a decision tree for classification, a given tuple (whose class we want to predict), corresponding to node features, walks through the decision tree from the root to a leaf. The label of the leaf node is the classification result.

We also consider a classifier that combines the predictions of multiple classifiers. The simplest way to combine predictions of various classifiers is to take a vote by averaging their estimates. The problem with voting is that it is not clear which classifier is to be trusted. To overcome this limitation, **stacked generalization**, or **stacking** for short, uses a *metalearner* that replaces the voting procedure. After all of the other classifiers are trained using the training set, **stacking** again trains the metalearner as a classifier for a final decision using all predictions of the other classifiers as additional features; the metalearner itself is a trained classifier, such as **NB** and **C4.5**, which is used to discover which

classifiers are most reliable [7]. In 'Stacking' section, we examine **stacking** that combines the predictions of **NB**, **NBK**, and **C4.5**.

Experimental results

To test our approach, we use datasets from several real-world networks. We classify the datasets into three categories – biological, collaboration, and device networks: YEAST (biological), GRQC, HEPTH (collaboration), POWER, and OREGON (device) as described in Table 1. We run batches of simulations for each network while varying the fraction of observed nodes from 5% to 25%. In each run, we simulate an SI cascade starting at a randomly selected seed node with infection rate $\lambda = 0.5$ until 10% of nodes are infected. The parameter α of infection betweenness centrality is set to 0.01 to guarantee that it is less than the reciprocal of the largest eigenvalue of the adjacency matrix of the reduced graph (the condition that α must satisfy for the sum **N** in the equation of infection betweenness to converge). If a network has multiple connected components as does YEAST, we assume that an epidemic starts at a node in the largest connected component. In order to evaluate accuracy, we use three metrics: precision, recall, and F-measure [7].

- Precision: the fraction of correctly classified nodes in nodes whose state is classified as infected.
- Recall: the fraction of nodes whose state is classified as infected out of all infected nodes.
- F-measure: a measure to consider both precision and recall in a single metric by taking their harmonic mean $\left(\frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}}\right)$.

Infection state determined by propagation property

Our first set of experiments investigates how many nodes can be handled by using Property 1. We denote the nodes whose infection states can be fully determined using Property 1 as *deterministic* nodes. Figure 2 presents the average proportion of deterministic nodes in the set of unobserved nodes according to the fraction of observed nodes.

We observe that in all networks, the average proportion of deterministic nodes increases as more nodes are observed. This is because by removing more observed susceptible nodes from the network, we are more likely to divide the network into multiple

Table 1 Topologies[a]

Topology	Type	*n*	*m*	*c*	σ	*s*	*d*[b]	Description
YEAST	Bio.	1,870	2,277	0.0672	3.1374	6.5044	19	Yeast Protein Interaction Network [16]
GRQC	Collab.	5,242	28,980	0.5296	7.9179	3.8317	17	Collaboration networks from ArXiv general relativity and quantum cosmology [17]
HEPTH	Collab.	9,877	51,971	0.4714	6.1864	3.0213	18	Collaboration networks from ArXiv high energy physics [17]
POWER	Device	4,941	6,594	0.0801	1.7913	2.1898	46	Topology of the Western States Power Grid of the United States [18]
OREGON	Device	11,174	23,409	0.2964	33.0948	46.4017	10	Topology of Autonomous Systems (AS) peering information inferred from Oregon route-views between 31 March 2001 and 26 May 2001 [19]

[a] *n*, *m*, *c*, σ, *s*, and *d* are the number of nodes, the number of edges, clustering coefficient, standard deviation of degree distribution, skewness of degree distribution [20], and diameter of network, respectively.
[b] *d* is calculated with the largest connected component if a network has multiple connected components.

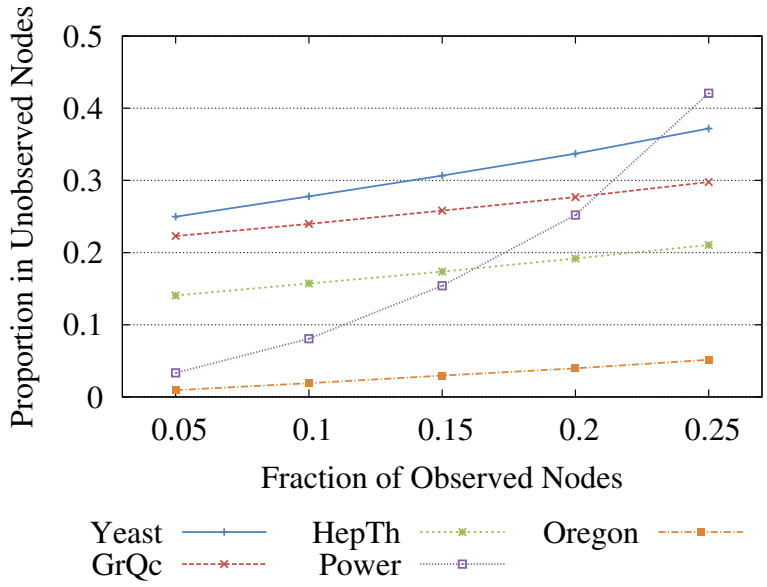

Figure 2 Proportion of deterministic nodes in unobserved nodes according to the fraction of observed nodes.

disconnected components, all but one of which contains no infected node. In POWER, the proportion increases more steeply than in other networks as more nodes are observed. This means that POWER is more easily disconnected than other networks as more nodes are removed. In contrast, OREGON yields a smaller fraction of deterministic nodes compared with other networks. Note that OREGON has exceptionally high degree skewness, that is, there exist nodes that have a significantly larger degree than other nodes in OREGON. Since the existence of such nodes makes it hard to disconnect OREGON, a lower proportion of unobserved nodes can be identified by Property 1. Recall that OREGON is a snapshot of the Internet and is a representative example of a scale-free network that follows a power-law degree distribution. Such a scale-free network is known to be resilient to random attacks [8], that is, it is not likely to be disconnected even though randomly selected nodes are removed. By regarding the removal of observed susceptible nodes from the network as random attacks, we can expect that scale-free networks have a small proportion of deterministic nodes by Property 1.

Incorporating infection betweenness centrality into classifiers

In this subsection, we evaluate the classifiers, described in Section 'Algorithms for building classifiers', using infection betweenness centrality and other node features. To apply those algorithms to experiments, we use the WEKA software suite [9], often used to perform various experiments with machine learning algorithms. For each topology, we collect the features of unobserved nodes from 30 simulations and then aggregate the collected feature instances into a training set. We run another 70 simulations to use as testing sets.

Predictive features

To examine which features provide meaningful information for identifying latent infected nodes, we investigate the performance of the classifiers with each feature when we create

cascades that infect approximately 10% of the nodes in the network and then reveal the infection states of 15% of the nodes. Revealed nodes are randomly selected to fairly compare the predictive power of each feature by obtaining uniformly distributed feature observation. Figure 3 shows the average F-measure of **NB** and **C4.5** with each feature for all the networks. The best feature will have an F-measure close to one (darker squares). We observe that infection betweenness centrality (P) produces the darkest column showing it to be the best predictive feature in both the **NB** and **C4.5** classifiers over nearly all networks. In the case of **C4.5**, observed infected neighbor ratio (R) yields a similar performance to infection betweenness centrality. We also see that normalized degree and closeness centrality (D and $C^{(c)}$) are also meaningful features in several networks although not as good as infection betweenness centrality. However, except for infection betweenness centrality, the effectiveness of other features differs significantly depending on the network and the classifier.

Effect of infection betweenness centrality

Figure 4a shows the F-measure of each classifier using all features except for infection betweenness centrality. In all of the considered networks, the classifiers yield F-measure of less than 0.5. We observe that the best classifier differs according to the network, but there is no significant difference between the classifiers for each network. Note the significant

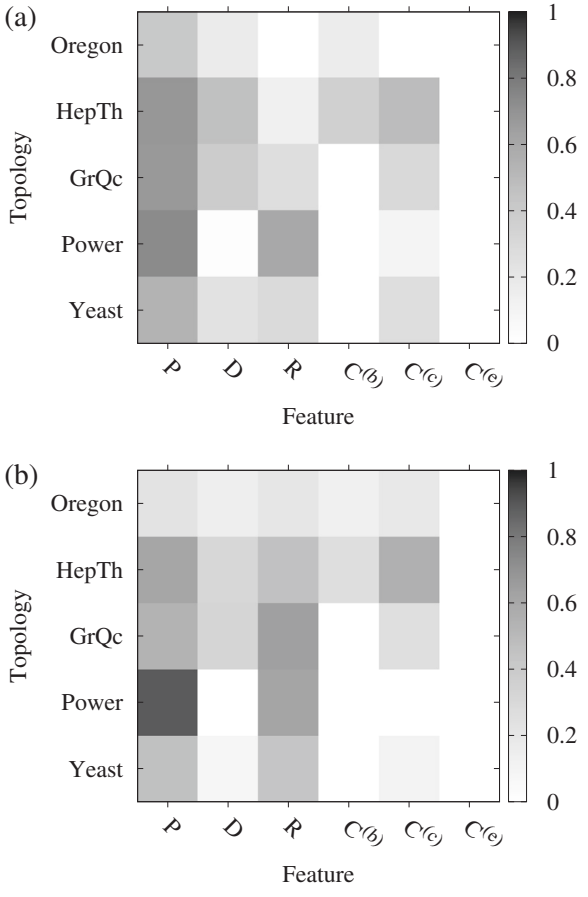

Figure 3 Predictive power of each feature: **(a)** NB, **(b)** C4.5.

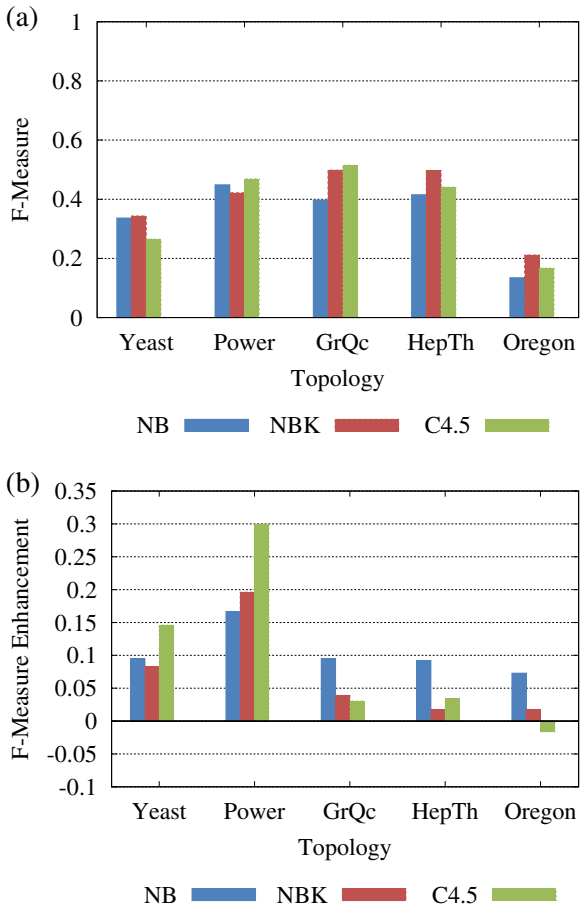

Figure 4 Performance of classifiers: **(a)** F-measure when using all features except for *P*, **(b)** F-measure enhancement by including *P*.

low performance of the classifiers in OREGON: the F-measure of even the best classifier, **NBK**, is around 0.2. This is because the predictive power of each feature is low in OREGON than in the other networks as shown in Figure 3. Next, we compare the classifiers using all of the features to those excluding infection betweenness centrality in order to check whether infection betweenness centrality can improve the performance of the classifiers.

Figure 4b shows the F-measure of each classifier using all six features minus the F-measure of the same classifiers using five features (which excludes infection betweenness centrality). Including infection betweenness centrality improves the performance of all classifiers on all networks with the exception of **C4.5** when applied to OREGON. This shows that classification can be improved by combining infection betweenness centrality with the other node features. In particular, the inclusion of infection betweenness centrality in classification enhances considerably the performance of the classifiers for YEAST and POWER, e.g., using all features increases the F-measure of **C4.5** applied to YEAST and POWER by approximately 0.15 and 0.3, respectively. In the case of **NB**, adding infection betweenness centrality enhances performance by almost the same amount (around 0.3) regardless of the network. This is because the predictive power of infection betweenness centrality for **NB** is similar across the networks as shown in Figure 3a. Note

that the inclusion of infection betweenness centrality in **C4.5** increases the F-measure except when applied to OREGON. Even with OREGON, the inclusion of infection betweenness centrality in **C4.5** yields a negligible decrease in F-measure. We observe then that for **C4.5**, infection betweenness centrality is by far the most important feature as adding infection betweenness centrality to the feature set in all other cases increases classification accuracy.

Prediction v.s. fraction of observed nodes

Figure 5 compares the average precision and recall of each classifier as a function of the fraction of observed nodes. As before, we consider a case where 10% of nodes are infected. Here, we also compare our classifiers against random classification (random), which tosses a biased coin and with probability 0.1 (0.1 is the fraction of infected nodes) to declare a node to be infected. As shown in Figure 5, our classifiers outperform random both in precision and recall. Also, the precision and recall of our classifiers increase with the fraction of observed nodes; as expected, increasing the fraction of observed nodes

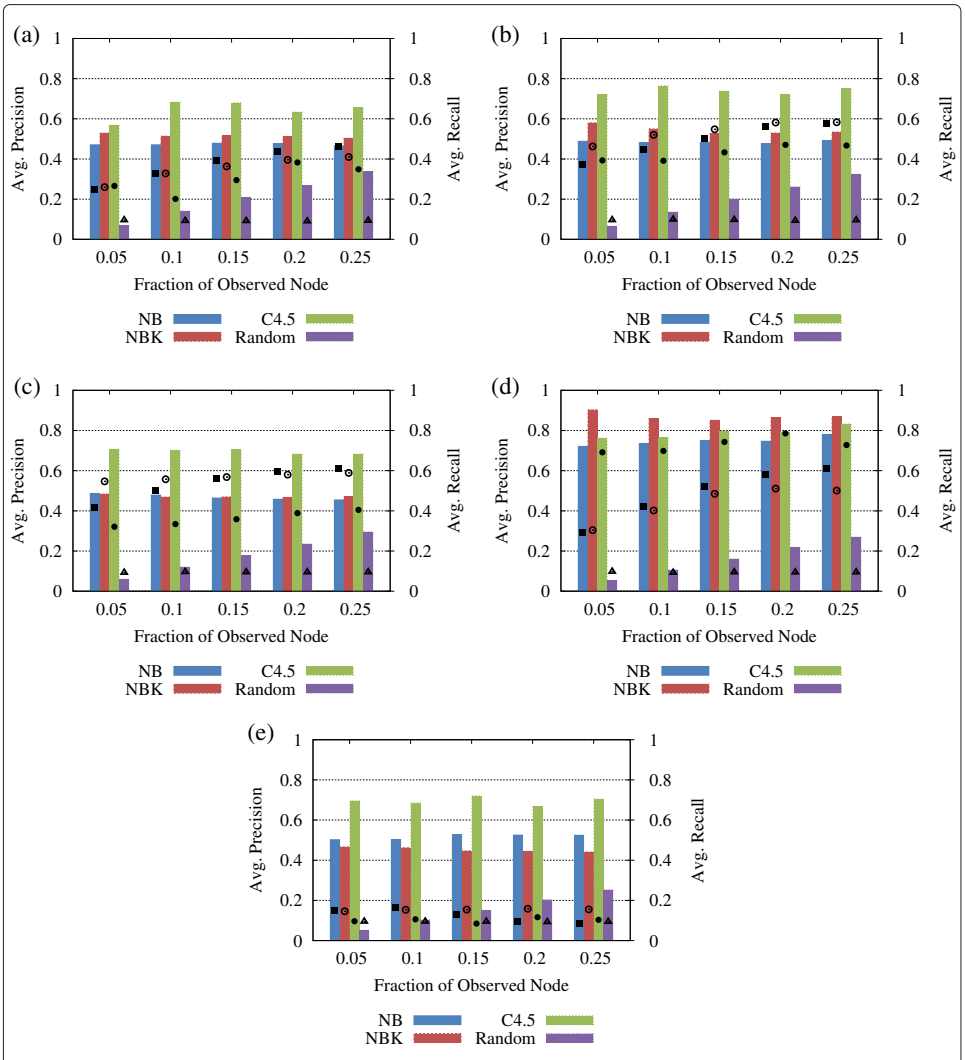

Figure 5 Accuracy for varying fraction of nodes with observed state. **(a)** YEAST, **(b)** GRQC, **(c)** HEPTH, **(d)** POWER, **(e)** OREGON (Bar: precision, dot: recall).

provides more information about the infection states of the unobserved nodes. **C4.5** exhibits the best precision over all classifiers on almost all networks: the only exception is POWER, where **NBK** yields slightly better precision than **C4.5**. Comparing the precisions of each network, we observe that our classifiers show the best precision for POWER followed by GRQC, HEPTH, YEAST, and OREGON; POWER is almost planar, likely making the classification task easier. In the next section, we also explore how network characteristics affect the performance of our classifiers.

Figure 5 shows that **NBK** yields the best recall performance over all networks except POWER. Note that the precision of **NBK** is lower than that of **C4.5** except for POWER. This means that **NBK** is more likely to classify unknown node states as infected, resulting in the higher recall, but those classifications are not as accurate as those made by **C4.5**. All classifiers yield better recall performance when applied to POWER than the other networks. Also, OREGON remains the most difficult network within which to correctly identify the infected nodes. Even though all classifiers yield relatively high precisions (greater than 0.5) in OREGON, their recall performance in OREGON is less than 0.2, which is similar to that of random guessing. That is, in OREGON, our classifiers make correct decisions when they classify unknown states to infected, but many infected nodes are classified as susceptible.

Impact of network characteristics

We now investigate the impact of network characteristics on the performance of our classifiers (using all six features). To this end, we investigate the Pearson correlation coefficient between the ranks according to the F-measure performance of the classifiers and network characteristics for each network; for instance, **NBK** yields the worst performance when applied to OREGON; furthermore, OREGON has the largest degree skewness. Table 2 presents the Pearson's correlation coefficient [10] between the ranks of network characteristics and F-measure performance.

Table 2 shows that the performance of the classifiers is strongly negatively correlated with degree skewness and the standard deviation of degree. As degree skewness and the standard deviation of degree decrease, the classifiers become more accurate. Interestingly, there is little correlation between clustering coefficient and classification performance even though an epidemic is more likely to propagate to nodes in a same cluster.

Combining decisions from multiple classifiers

We now investigate whether infection state estimation can be improved by combining decisions from multiple classifiers. To check the performance of the combined classifier, **stacking**, we compare its F-measure to that of **C4.5** using all features, which typically yields the best performance in our experiments. To find the best metalearner for infection state estimation, we investigate the performance of **stacking** with different metalearners

Table 2 Correlation coefficient between ranks according to F-measure performance and network characteristics

Characteristic	Correlation NB	NBK and C4.5
Clustering coefficient	0.1	0.2
Standard deviation of degree	−0.7	−0.6
Degree skewness	−1.0	−0.9

(**C4.5**, **NB**, or **NBK**). Similar to previous experiments, we utilize the **stacking** algorithm implemented in the WEKA [9].

Figure 6 presents the measured F-measures of **C4.5** and **stacking** with different metalearners. As shown in Figure 6, **C4.5** yields worse performance as a metalearner than **NB** and **NBK** except for POWER even though it performs best as a single classifier: for our considered networks, NBK is the best metalearner to combine multiple decisions from the classifiers with **stacking**.

We also observe that **stacking** improves performance for the networks where no classifier works well whereas it exhibits worse accuracy in networks where one classifier notably outperforms others. For example, recall that in terms of F-measure, **C4.5** significantly outperforms **NB** and **NBK** in POWER. In POWER, **stacking** with all metalearners fails to achieve performance enhancement while in YEAST and OREGON **stacking** with **NB** or **NBK** improves F-measure performance. Notice that the networks in which **stacking** yields performance improvement, such as YEAST and OREGON, have a comparatively high standard deviation in degree distribution and also large degree skewness, which result in worse performance of a single classifier as shown in Table 2. This shows that a classifier for estimating latent infection state can be adaptively chosen according to observed network characteristics, e.g. applying **stacking** if the degree skewness is larger than some threshold and otherwise using **C4.5**. Devising a strategy to select an

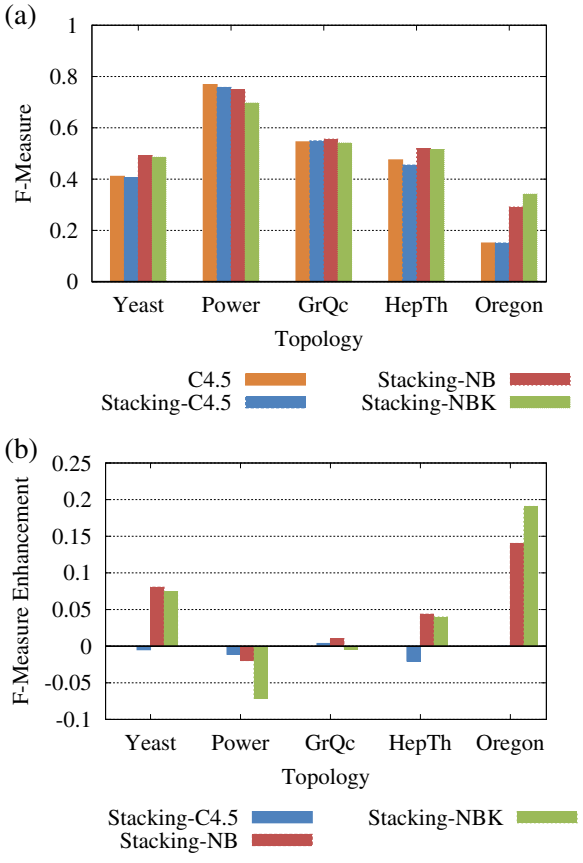

Figure 6 Comparison between C4.5 and stacked generalization: **(a)** F-measure, **(b)** F-measure enhancement by applying stacking.

appropriate classifier based on extensive experiments using more networks is a part of our future work.

Related work

Several methods to detect the presence of network worms and rumor spreading nodes have been proposed in the literature. However, there has been little rigorous work done on inferring the infection state from incomplete data obtained at a relatively few observed nodes without the aid of infection timestamps.

Shah and Zaman [11] studied the problem of finding the source of a computer virus in a network. They focused on how to find the source among the set of infected nodes that are observed, which is different from our goal. Based on their metric called *rumor centrality*, they constructed a machine-learning estimator that finds the source exactly or within a few hops in networks. They also analyzed the asymptotic behavior of their virus source estimator for regular trees and geometric trees.

Sadikov et al. [12] present an estimation method of network properties, such as the number of weakly connected components, given a sampled network. By formulating a simple k-tree model and approximating it to the original network, their method can estimate the properties of original networks; they showed that their method can accurately estimate properties of the original network even when 90% of nodes are not sampled.

Closely related to our work is that of Gomez et al. [13], who develop an algorithm for inferring the topology of the network over which a diffusion propagates. Given the observed times when nodes become infected, they determine paths through which the diffusion most likely took, i.e., a directed graph where a contagion passed through. In contrast, our work tries to identify the infection state of each unobserved node given a limited number of nodes with known infection state and no infection timestamps.

Zou et al. [14] developed an early detection system that can detect the presence of a worm in the Internet by using Kalman filter. The proposed detection approach monitors traffic data at ingress/egree point of a local network. Even with biased monitored data, it can accurately predict the overall vulnerable population size and estimate how many hosts are actually infected in the global Internet system. However, their goal is not to exactly identify the infected nodes in networks.

Sawaya et al. [15] proposed a flow-based attacker detection method focusing on the characteristics of attackers that send flows to both the object TCP port and generally closed TCP port in the global network. Thus, we need to inspect the flows from each node to identify whether it is from an attacker.

Conclusion

In this paper, we studied the problem of identifying infected nodes in a network without individually inspecting all nodes in the network. Based on the well known SI model, we reduce the problem space by utilizing the propagation properties in the model. We examined how network characteristics affect the effectiveness of propagation properties on reducing the problem space. Then, we defined the *infection betweenness centrality* for identifying the latent infection states of nodes. Our empirical results show that the classifiers using the infection betweenness centrality along with other network-wide features

outperform random guessing and the same classifiers without it. We analyzed the impact of the amount of missing data as well as the impact of network characteristics on the effectiveness of the classifiers. Our experimental study also shows that the infection state estimation can be improved by combining multiple classifiers in networks with high degree skewness such as OREGON. Devising a strategy to select an appropriate classifier based on more extensive experiments is a part of our future work.

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
This paper is an extended version of a paper which has been published at Simplex'14. All authors contributed to the conception and design of this study. All authors read and approved the final manuscript.

Acknowledgements
This work was supported by the NSF grant CNS-1065133, ARL Cooperative Agreement W911NF-09-2-0053, and ARO under MURI W911NF-08-1-0233.

Author details
[1]School of Computer Science, University of Massachusetts Amherst, 140 Governors Drive, 01003 Amherst, USA.
[2]Carnegie Mellon University, 15213 Pittsburg, USA.

Received: 4 November 2014 Accepted: 19 March 2015
Published online: 21 April 2015

References

1. John, GH, Langley, P: Estimating continuous distributions in Bayesian classifiers. In: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. UAI'95, pp. 338–345. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, (1995)
2. Quinlan, JR: Improved use of continuous attributes in c4.5. J. Artif. Intelligence Research. **4**, 77–90 (1996)
3. Newman, M: Networks: An Introduction. Oxford University Press, Inc., New York, NY, USA (2010)
4. Moore, AW, Zuev, D: Internet traffic classification using bayesian analysis techniques. In: Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems. SIGMETRICS '05, pp. 50–60. ACM, New York, NY, USA, (2005)
5. Kim, H, Claffy, K, Fomenkov, M, Barman, D, Faloutsos, M, Lee, K: Internet traffic classification demystified: Myths, caveats, and the best practices. In: Proceedings of the 2008 ACM CoNEXT Conference. CoNEXT '08, pp. 11–11112. ACM, New York, NY, USA, (2008)
6. Lim, Y-S, Kim, H-C, Jeong, J, Kim, C-K, Kwon, TT, Choi, Y: Internet traffic classification demystified: on the sources of the discriminative power. In: Proceedings of the 6th International COnference. Co-NEXT '10, pp. 9–1912. ACM, New York, NY, USA, (2010)
7. Witten, IH, Frank, E: Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2005)
8. Albert, R, Jeong, H, Barabasi, A-L: Error and attack tolerance of complex networks. Nature. **406**(6794), 378–382 (2000)
9. Hall, M, Frank, E, Holmes, G, Pfahringer, B, Reutemann, P, Witten, IH: The WEKA data mining software: an update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)
10. Wikipedia: Spearman's rank correlation coefficient - Wikipedia, The Free Encyclopedia. [Online: http://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient. Accessed 13-Jan-2014]
11. Shah, D, Zaman, T: Detecting sources of computer viruses in networks: theory and experiment. In: Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems. SIGMETRICS '10, pp. 203–214. ACM, New York, NY, USA, (2010)
12. Sadikov, E, Medina, M, Leskovec, J, Garcia-Molina, H: Correcting for missing data in information cascades. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining. WSDM '11, pp. 55–64. ACM, New York, NY, USA, (2011)
13. Gomez Rodriguez, M, Leskovec, J, Krause, A: Inferring networks of diffusion and influence. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '10, pp. 1019–1028. ACM, New York, NY, USA, (2010)
14. Zou, CC, Gong, W, Towsley, D, Gao, L: The monitoring and early detection of internet worms. IEEE/ACM Trans. Networking. **13**(5), 961–974 (2005)
15. Sawaya, Y, Kubota, A, Miyake, Y: Detection of attackers in services using anomalous host behavior based on traffic flow statistics. In: Applications and the internet (SAINT), 2011 IEEE/IPSJ 11th international symposium on, pp. 353–359, (2011). doi:10.1109/SAINT.2011.68
16. Jeong, H, Mason, SP, Barabási, AL, Oltvai, ZN: Lethality and centrality in protein networks. Nature. **411**(6833), 41–42 (2001)
17. Leskovec, J, Kleinberg, J, Faloutsos, C: Graph evolution: densification and shrinking diameters. ACM Trans. Knowl. Discov. Data. **1**(1) (2007). doi:10.1145/1217299.1217301. http://doi.acm.org/10.1145/1217299.1217301
18. Watts, DJ, Strogatz, SH: Collective dynamics of 'small-world' networks. Nature. **393**(6684), 440–442 (1998)

19. Leskovec, J, Kleinberg, J, Faloutsos, C: Graphs over time: densification laws, shrinking diameters and possible explanations. In: Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery in data mining, pp. 177–187. ACM, Chicago, Illinois, USA, (2005). doi:10.1145/1081870.1081893. http://doi.acm.org/10.1145/1081870.1081893
20. Wikipedia: Skewness - Wikipedia, The Free Encyclopedia. [Online: http://en.wikipedia.org/wiki/Skewness. Accessed 13-Jan-2014]