Applied Network Science

RESEARCH Open Access

Effective and scalable methods for graph protection strategies against epidemics on dynamic networks

Arie Wahyu Wijayanto* and Tsuyoshi Murata

*Correspondence:
ariewahyu@net.c.titech.ac.jp
Department of Computer Science, Tokyo Institute of Technology, Tokyo, Japan

Abstract

Dynamic networks are networks with temporal relationship features which evolve over time by the inclusion and deletion of nodes and edges. Suppressing the epidemic spreading in such networks is quite challenging. The problem of protecting a limited number of nodes to restrain the spreading of malicious attacks or dangerous rumor in the networks is called graph protection problem. However, most of existing strategies only consider to protect at once regardless the evolving network structure and incoming attacks over time, i.e., these strategies either pre-protect important nodes before the epidemic starts or post-allocate the protection while the attacks have already spread over the network. In this paper, we introduce multiple-turns protection strategies, which divide the size of protection budget into several turns and protect nodes according to the currently observed temporal snapshot of dynamic networks. We construct a minimum vertex cover of the input network efficiently using reinforcement learning approach. To capture the state of the input network, a feature-based representation of each node is constructed using a graph embedding technique. Experimental evaluations show that our proposed methods, namely ReProtect and ReProtect-p effectively restrain epidemic propagation in synthetic and real-world network datasets. By protecting about 15% of nodes, our methods can obtain up to 84% of surviving nodes and outperform other baseline methods on two popular epidemic models: SIS and SIR.

Keywords: Graph protection, Node immunization, Dynamic networks, Reinforcement learning

Introduction

With the rising popularity of massive-scale online social networks such as Facebook, Instagram, Twitter, etc., people are more connected and can share information with each other (Zhuang et al. 2013). These platforms play a vital role in the dissemination of positive information such as new ideas, innovations, and hot topics. However, they may also become channels for the spreading of malicious rumors, misinformation, or even dangerous virus and malware. The rumor spreading can severely threaten public safety and financial stability. For instance, some people may post on social networks a rumor about an upcoming big earthquake. It will cause chaos among society and hence may hinder the normal public order. In this situation, it is necessary to find individuals, if their account deactivated or removed from the network, would block further rumor spreading. The

problem is known as *graph protection problem* where the goal is to protect a number of nodes to restrain the epidemic propagation by maximizing the ratio of surviving nodes in a network (Wijayanto and Murata 2017; 2018b). In this problem, *the protection budget* constraints the number of nodes we are allowed to protect.

Real-world social networks and collaboration networks have highly dynamic structures, and they evolve rapidly over time (Zhan et al. 2017; Wang et al. 2016). The inherently dynamic nature of the network leads to dynamic network representations. Dynamic networks are defined as networks which evolve over time by the addition and removal of nodes and edges (Bakker et al. 2018; Moore et al. 2006; Zhuang et al. 2013). Dynamic networks have temporal relationship features which specify the number of connection among nodes that are active at a certain time.

Restraining the epidemic spreading in dynamic networks is obviously more challenging than in static networks because of the temporal changing of the network structure. However, most of the existing work failed to address the incoming rumor or virus attacks during the temporal transition in dynamic networks. The existing strategies either preemptively protect critical nodes prior to epidemic attacks, behaving as prevention efforts (Prakash et al. 2010; Chen et al. 2016; Wijayanto and Murata 2017), or post-emptively allocate the protection while the epidemics have already propagated over the network, simulating as delayed reactions (Zhan et al. 2017; Zhang et al. 2017; Zhang and Prakash 2015; Song et al. 2015). In this paper, we introduce a multiple-turns graph protection strategy by dividing the protection budget into several turns and protecting nodes based on the currently observed temporal structure of networks.

On the other hand, most of the current works of graph protection strategy mostly fall into one of the following drawbacks: (1) protecting only particular areas of the networks, as demonstrated by centrality-based methods (Prakash et al. 2010; Buono and Braunstein 2015; Zhao et al. 2014) (2) scalability issue, as demonstrated by dominator tree-based methods (Zhang and Prakash 2014; Zhan et al. 2017; Zhang and Prakash 2015) (3) lack of convergence guarantee in large size networks, as shown by eigendecomposition-based methods (Tong et al. 2010; Chen et al. 2016; Wijayanto and Murata 2017; Prakash et al. 2010). We propose the construction of minimum vertex cover to determine the protected nodes in an efficient and scalable method. The *minimum vertex cover* (MVC) is the set of nodes which cover all edges of networks in a minimum possible size of nodes. As we explain later, MVC serves as the protection threshold of the network (see "Proposed methods" section for our detailed explanation).

In recent years, reinforcement learning (RL) approaches have obtained many state-of-the-art results in solving various complex problems (Mnih et al. 2015; Riedmiller 2005). RL allows autonomous agents to learn to improve their performance with experience. In this work, we utilize RL approach to construct MVC from the currently observed network snapshot. Specifically, we propose n-step fitted Q-Learning to obtain the MVC solution of input network by leveraging the neural network as a function approximator. Neural network architecture allows us to efficiently accelerate the training and execution of our proposed methods in mini-batch processing and multiple graphics processing units to deal with large size networks. In order to handle the different size and structure of each temporal snapshot of dynamic networks, each node is represented in a fixed-length feature vector using a graph embedding technique.

Extensive evaluations in both synthetic and real-world network datasets show that our proposal effectively restrains epidemic spreading. In Email network dataset, by protecting about 15% of nodes, our methods can achieve up to 84% of surviving nodes and outperform other baseline methods. Comprehensive evaluations under two most popular epidemic models, i.e., SIS and SIR, confirms the effectiveness and scalability of our methods.

The novelty of our methods arises primarily from the application of more stochasticity and learning ability to graph protection problem, specifically on dynamic networks. In large-scale social networks, the changing of relationship structure and rumor spreading patterns may come and arise on a regular basis. Therefore, there is an opportunity to learn the current condition into a model using reinforcement learning. By learning the given temporal structure of observed networks and existing epidemics, this will provide a new incentive to predict future protection from previously learned actions in the same dynamic networks.

This paper extends our preliminary idea in (Wijayanto and Murata 2018a). In addition to the contents in (Wijayanto and Murata 2018a), this paper includes the following: detailed explanation of the proposed methods; evaluation on synthetic networks, as well as more real-world network datasets; review of the relevant related work; discussion of scalability and computational complexity; evaluation of parameter sensitivity; addition of stronger baselines methods such as Betweenness, GraphShield and NetShield+; and evaluation on SIR epidemic model.

The remainder of this paper is organized in the following manner. We formalized the problem and definition in "Problem formulation" section. The review of recent most related studies is presented in "Related work" section. Our proposed methods, namely ***ReProtect*** and ***ReProtect-p*** are described in "Proposed methods" section. The result of experimental simulations are provided in "Evaluation" section. Finally, concluding remarks of our work is provided in "Conclusion" section.

Problem formulation

In this section, we formalize the definitions and problems used throughout this paper. We summarize the symbols and notations in Table 1.

Definition 1. ***Protecting*** a node means removing all of its corresponding edges. The number of nodes we are allowed to protect is constrained by the ***protection budget*** ($k \in \mathbb{Z}_{>0}$). At time t, a node in a network can belong to any of the following states: *susceptible* and *infected*. ***Attacking*** a node means initially infect the node in a network. Figure 1 shows the example of protection and attack in a static network.

Definition 2. ***Graph Protection Problem***

Let $G = (V, E)$ be an undirected connected graph with set of nodes V and set of edges E. Let θ be the surviving ratio of nodes that remain uninfected at the end of epidemics.

Given an input graph G, SIS or SIR epidemic model, and a protection budget k, the goal is to find a set of nodes $S \in V$ such that θ is maximized, subject to the size of S is equal to constraint budget k. The protection is performed by removing all edges connected to the set of nodes S in graph G to get a new graph $G^{(S)}$.

Definition 3. ***Dynamic Network***

Let $\{1, \cdots, T\}$ be a finite set of discrete time steps. Let $V^D = \{1, \cdots, n\}$ be a set of nodes which appear within time $\{1, \cdots, T\}$. Let $G_t = (V_t, E_t)$ be a graph representing the

Table 1 Summary of Symbols and Notations

Notation	Definition and description
$G^D = (V^D, E^D)$	dynamic network G^D with the node set V^D and the edge set E^D
G_t	snapshot of dynamic network G^D at time t
k	protection budget, i.e., the number of nodes in graph G^D that can be protected
S	set of k nodes selected for protection
N	number of nodes in graph G^D
M	number of edges in graph G^D
β	infection rate
δ	recovery rate
θ	ratio of surviving nodes in graph G^D at the end of epidemics
$w(u, v)$	edge weight between node u and v
V_c	vertex cover of graph G_t
V_c^*	minimum vertex cover of graph G_t
$\mathbb{S}$	current partial solution, set of selected V_c^* nodes of graph G_t
d	size of embedding vector dimension
$h(\mathbb{S})$	feature-based representation of $\mathbb{S}$ in d-dimensional vector
B	batch samples of training
$\mathbb{M}$	experience replay memory of n-step fitted Q-Learning
ψ_i	set of neural network parameters (weights) of respective embedding variable i
Ψ	collection of neural network's set of parameters (weights) $\Psi = \{\psi_i\}_{i=1}^{7}$

snapshot of the network at time t. $V_t \subseteq V^D$ is a subset of nodes V^D observed at time t. (t, u, v) represents an edge from vertex $u \in V_t$ to $v \in V_t$ at time t.

A dynamic network $G^D = (V^D, E^D)$ is a series $\langle G_1, \cdots, G_T \rangle$ of static networks where each $G_t = (V_t, E_t)$ is a snapshot of nodes and their edges at time t such that $V^D = \bigcup_t V_t$. For the sake of consistency, the time during which the nodes are observed is assumed as finite. Following the definition by (Habiba et al. 2010) and (Bakker et al. 2018), the temporal length of G is assumed to be divided into discrete steps $\{1, \cdots, T\}$. The non-trivial problem of appropriate time discretization is beyond the scope of our work.

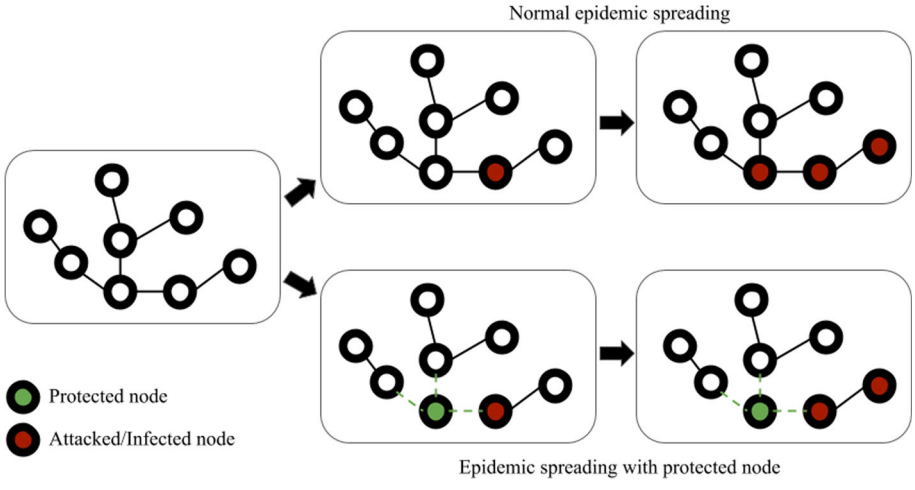

Fig. 1 Example of a static network. Green colored node indicates the node is protected. Dashed green colored edges indicate the edges are removed or inactivated because of protection. Initial red colored node on network indicates the node is attacked. Other red colored nodes indicate the nodes are infected

Definition 4. ***SIS Epidemic Model***

Susceptible-Infected-Susceptible (SIS) is an epidemic epidemic model which define that each node in graph G with N number of nodes would be in one of the following two states: *susceptible* and *infected*. Let $\mathcal{S}(t)$ be the number of susceptible nodes, and let $\mathcal{I}(t)$ be the number of infected nodes at time t. At each timestamp t, susceptible nodes can be infected by their infected neighbors with infection rate β. Also, each infected node can get recovered to susceptible state with recovery rate δ. In the homogeneous case of well-mixed populations, this model can be formalized as non-linear differential equations:

$$\frac{ds}{dt} = -\beta is, \frac{di}{dt} = \beta is - \delta i, \tag{1}$$

being $s(t) = \mathcal{S}(t)/N$ and $i(t) = \mathcal{I}(t)/N$ the respective proportions of states at time t. A continuous-time epidemic process under constant infection rate β and recovery rate δ on any network can be described by Markov theory. Following the definition of SIS epidemic in network by Pastor-Satorras (2015), the individual-based mean-field (IBMF) and degree-based mean-field (DBMF) approach can be used to analytically simulate the SIS model.

Definition 5. ***SIR Epidemic Model***

In Susceptible-Infected-Recovered (SIR) model, each node in graph G belong to any of the *susceptible*, *infected*, or *recovered* state. Each of recovered node is resistant of any infection. Let $\mathcal{R}(t)$ be the number of recovered nodes. Following the definition by Kermack and McKendrick (1927), for the homogeneous case of well-mixed populations, this model is formalized as:

$$\frac{ds}{dt} = -\beta is, \frac{di}{dt} = \beta is - \delta i, \frac{dr}{dt} = \delta i, \tag{2}$$

being $s(t) = \mathcal{S}(t)/N$, $i(t) = \mathcal{I}(t)/N$, and $r(t) = \mathcal{R}(t)/N$ the respective proportions of states at time t. In addition to IBMF and DBMF approach, following the definition of SIR epidemic in network by Pastor-Satorras (2015), we can analytically describe the SIR model using generating function approach, where the probability that a link exists is related to the probability of transmission of the disease from an infected node to a connected susceptible one.

Definition 5. ***Multiple-turns Graph Protection Problem on Dynamic Networks***

Let $G^D = \left(V^D, E^D\right)$ be an undirected dynamic graph as an input, with a series of a known sample $\langle G_1, \cdots, G_T \rangle$ of snapshots where each $G_t = (V_t, E_t)$ represent a static network at time t. Let k be the protection budget, which $k < |V^D|$ and allocated into several turns according to the number of observed snapshots of G^D. Protection budget for snapshot G_t at time t is denoted by k_t such that $k = \sum_{t=1}^{T} k_t$.

Let us denote S, a set of k protected nodes from graph G^D and $S = \sum_{t=1}^{T} S_t$ where S_t denote a subset of k_t protected nodes of snapshot graph G_t at time t. Protection means removing corresponding edges of the set of nodes S_t in graph G_t to get a new graph $G_t^{(S)}$. Under random attack strategies, l nodes are randomly attacked (i.e., initialized as infected nodes) from graph G^D such that $l = \sum_{t=1}^{T} l_t$ at each turn in time t. We define θ as the ratio of surviving nodes of graph G^D.

Given an input graph G^D, SIS or SIR epidemic model, and a protection budget k, the goal is to find S such that θ is maximized, subject to the size of S is equal to constraint budget k, i.e., calculating the following combinatorial optimization:

$$\begin{aligned} S^* &= \arg\max_{S \in V} \theta \\ \text{s.t.} |S| &= k, k = \sum_{t=1}^{T} k_t \end{aligned} \tag{3}$$

Related work

In this section, we review the relevant existing studies related to our work. We first review the fundamental work of epidemic modeling on dynamic networks, then we discuss some related work on graph protection strategy and its application in dynamic networks. Finally, some problems related to graph protection on dynamic networks are presented.

Fundamental work of epidemic modeling on dynamic networks

The properties of dynamic networks are essentially different from those in static networks. (Braha and Bar-Yam 2006; 2009) found that the overlap of the centrality in dynamic networks and that in the aggregated (static) network is quite low. They also demonstrated that the static topology is unable to capture the dynamic properties of social networks. Hill and Braha (2010) propose a reinforced random walk approach to explain dynamic centrality phenomena and qualitatively reproduce the characteristic features of real-world networks. Those studies (Braha and Bar-Yam 2006; 2009; Hill and Braha 2010) provide an important foundation of dynamic network properties.

Holme presents a systematic review of dynamic networks and discusses methods for topological and temporal structure analysis (Holme and Saramäki 2012; Holme 2015). More specifically, Pastor-Satorras et al. (2015) discuss a fundamental review of epidemic model on dynamic networks, which also recently emphasized by Enright and Kao (2018).

Graph protection strategy and its application in dynamic networks

The study of graph protection strategies has mostly been introduced by assuming the static topologies of network structure. Pastor-Satorras and Vespignani investigated the effect of random uniform and targeted high-degree immunization of individuals on homogeneous complex networks and scale-free networks (Pastor-Satorras and Vespignani 2002). Chen et al. proposed NetShield (Tong et al. 2010) and NetShield+ (Chen et al. 2016) which use the properties of matrix perturbation to find a set of nodes in static networks to be pre-emptively protected (Tong et al. 2010). Zhang and Prakash (2014; 2015) developed DAVA and DAVA-fast, two post-emptive polynomial-time heuristics methods which merge all infected nodes into a supernode by building a weighted dominator tree of input network. NIIP (Song et al. 2015) extracts a maximum directed acyclic graph from a static network then implements a Monte Carlo simulation to approximate the distribution of k over each time point t given the probability of a functional node getting infected. Wang et al. investigated a rumor blocking in static networks by considering dynamic Ising propagation model which consists of the individual tendency and global popularity of the

rumor Wang et al. (2016; 2017). Under the constraint of user experience utility, they proposed DRIMUX method to protect a set of nodes in t time interval to limit the spreading of rumor.

In dynamic networks, Prakash et al. proposed greedy algorithms, called NLDS, as pre-emptive protection of the dynamic networks (Prakash et al. 2010). The methods are composed on different variants which select protected nodes based on the highest degree centrality, acquaintance (random neighbor) or the largest eigenvalue of the adjacency matrix. Liu & Gao investigated a different task of influence blocking in dynamic email networks (Liu and Gao 2011). They introduced an adaptive Autonomy-Oriented Computing which actively propagates the vaccination patches to counter a virus-embedded email spreading. VAILDN is introduced by Zhan et al. (2017) as a post-emptive scheme protection. By merging all infected nodes into one supernode and building a weighted dominator tree of modified input network, VAILDN determines the protected nodes based on each sub-tree benefit comparison.

Table 2 shows the comparison of our proposed method to the relevant existing work on graph protection strategy. To summarize, none of the existing works investigated the suppressing the epidemic spreading by multiple-turns graph protection strategies on dynamic networks.

Problems related to graph protection on dynamic networks

There are some problems related to our work. Epidemic containment using link deactivation (Bishop and Shames 2011; Van Mieghem et al. 2011; Matamalas et al. 2018), aims to deactivate a set of links (instead of nodes) to contain epidemic spreading in the networks. Van Mieghem proposes a link removal approach to decrease the spectral radius of graph during epidemic spreading (Van Mieghem et al. 2011). Bishop discusses a mechanism for reducing the speed of disease propagation (Bishop and Shames 2011). Matamalas introduces an epidemic controlling approach based on the deactivation of most important

Table 2 Comparison of the proposed method to related existing work

Method name	Input network	Protection scheme	Node selection mechanism
Degree (Tong et al. 2010)	Static network	Pre-Emptive	Degree centrality
Betweenness (Tong et al. 2010)	Static network	Pre-Emptive	Betweenness centrality
DAVA (Zhang and Prakash 2014)	Static network	Post-Emptive	Dominator tree-based
NIIP (Song et al. 2015)	Static network	Post-Emptive	Dominator tree-based
NetShield+ (Chen et al. 2016)	Static network	Pre-Emptive	Eigendecomposition-based
GraphShield (Wijayanto and Murata 2017)	Static network	Pre-Emptive	Eigendecomposition-based
TIM (Buono and Braunstein 2015)	Multiplex static network	Pre-Emptive	Degree centrality
MultiplexShield (Wijayanto and Murata 2018b)	Multiplex static network	Pre-Emptive	Eigendecomposition-based
VAILDN (Zhan et al. 2017)	Dynamic network	Post-Emptive	Dominator tree-based
NLDS Degree (Prakash et al. 2010)	Dynamic network	Pre-Emptive	Degree centrality
NLDS EigenValue (Prakash et al. 2010)	Dynamic network	Pre-Emptive	Eigendecomposition-based
Proposed method	Dynamic network	Multiple-turns	Degree-ordered MVC

links transmitting the disease (Matamalas et al. 2018). These studies are different from our focus as they are focusing on link selection instead of node selection. Additionally, in the real-world social networks, nodes represent users while links/edges represent friendship connections among users. For a network administrator, such as in Facebook or Twitter, it is more reasonable to temporarily deactivate a certain user in the case of rumor spreading than to deactivate part of the users' friendship relations. While in human contact networks, it is more plausible to immunize an important person than to restrict a combination of several peer-to-peer interactions.

Network dismantling (Braunstein et al. 2016; Ren et al. 2018) is another problem related to our work. It is the problem of determining a minimum set of nodes in which removal breaks the network structure into subcritical connected components at minimum cost. Braunstein et al. (2016) provides insightful finding that the dismantling problem is an intrinsically collective problem and that optimal dismantling sets cannot be viewed as a collection of individually well-performing nodes. Ren et al. (2018) proposed a method based on the spectral properties of a node-weighted Laplacian operator to solve the problem.

Influence maximization problem on dynamic networks is also related to our work. While in the influence maximization we aim to maximize the influence spreading (information diffusion) (Tong et al. 2017; Murata and Koga 2018), the graph protection tries to restrain and contain any of those spreading process. Tong et al. (2017) introduced a greedy adaptive seeding strategy as an efficient heuristic for maximizing influence in dynamic social networks. Murata and Koga (2018) proposed three new methods for solving the problem which are the extensions of the methods for static networks.

Proposed methods

In this section, we propose new methods for multiple-turns graph protection problem in dynamic networks, namely ***ReProtect*** and ***ReProtect-p***. To restrain the spreading of epidemic in dynamic networks, we divide the protection budget wisely into several turns. The protected nodes are selected in each turn according to the currently observed temporal snapshot of dynamic network. Using the multiple-turns protection, we aim to address the changing of network structure and incoming rumor or virus attacks during the temporal transition in dynamic networks.

Figure 2 illustrates our proposed method in each turn, which takes a temporal snapshot of dynamic networks at time t as an input and determines the set of protected nodes. In each given turn, we determine the most *critical* set of nodes of the input network. A node

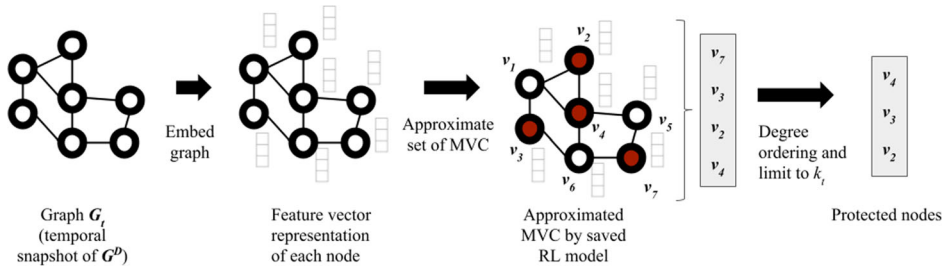

Fig. 2 Schematic illustration of the proposed method. Given a temporal snapshot of a dynamic network at time t, our proposed method selects a set of protected nodes

is considered as a *critical node* if it is assumed that protecting such node contribute to block large-scale epidemic spreading (Chen et al. 2016; Wang et al. 2016, 2017).

The main idea of our method can be described in the following key points:

1. Minimum vertex cover (MVC)

At first, we aim to find the set of the most critical nodes in the input network. Many previous studies suggest that a certain critical node criterion is best for a certain type of network structure. For instance, degree centrality is most suitable for dense and highly centralized network (Lawyer 2015; Chen et al. 2016), while betweenness centrality and connectivity are well fit for clustered networks with the existence of graph bridges (Italiano et al. 2012; Khan et al. 2015; Lawyer 2015).

We propose to consider a minimum vertex cover (MVC) as a criterion to determine set of critical nodes from networks. Given a graph $G = (V, E)$, a *vertex cover* is a subset of the nodes $V_c \subseteq V$ such that every edge of G is connected to V_c. Hence, this set of nodes V_c in graph G cover every edge in G. A *minimum vertex cover* is a vertex cover with the smallest possible number of nodes. Every graph trivially has a vertex cover where $V_c = V$. Figure 3a shows the vertex cover, and Fig. 3b shows the minimum vertex cover for the same graphs. The complexity of vertex cover problem is NP-Complete, and that of the minimum vertex cover problem is NP-Hard.

As shown in Fig. 2, our input is a static network G_t, the observed snapshot of dynamic network at time t. We aim to completely cover all the connections in G_t, which are represented by edges, by the smallest possible size of nodes. The size definition of MVC is intuitively aligned with the limited size of the protection budget in graph protection problem. Following the definition of graph protection problem, we can show the role of MVC as the protection threshold in a network.

Theorem 1 (Protection Threshold) *The protection threshold is the minimum required size of S to disconnect graph G such that no propagation may occur among nodes. Given an undirected connected graph $G = (V, E)$, a minimum vertex cover of G is also a protection threshold of G.*

Proof A vertex cover V_c of G is a subset of the nodes $V_c \subseteq V$ such that $(u, v) \in E \Rightarrow u \in V_c \vee v \in V_c$. A minimum vertex cover V_c^* is a V_c with the smallest size as follows:

$$V_c^* = \arg\min_{V_c} |V_c| \tag{4}$$

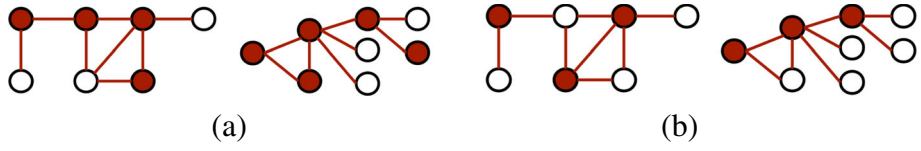

Fig. 3 Vertex cover and minimum vertex cover shown on the same underlying graphs. Red colored node indicates the node in the cover because all edges are covered. **a** Vertex cover (not minimum). **b** Minimum vertex cover Figure 3:

Since all edges in graph G is covered by V_c^*:

$$(u, v) \in E \Rightarrow u \in V_c^* \vee v \in V_c^*, \tag{5}$$

then by removing all corresponding edges in G connected to V_c^* we get $G^{(V_c^*)} = \left(V_c^*, E^{(V_c^*)}\right)$. Thus, $G^{(V_c^*)}$ has no edge, i.e., $E^{(V_c^*)} = \{\}$, $\left|E^{(V_c^*)}\right| = 0$.

According to Definition 1 and 2, protecting the set S of nodes in G is removing all edges of G connected to S. This is a minimax function of minimizing the size S to get the maximum edges in G covered as follows:

$$S^* = \arg\min_{S} |\arg\max_{E^{(S)}} |E^{(S)}|| \tag{6}$$

Consequently, by protecting minimum vertex cover V_c^*, i.e., $S = V_c^*$, then $G^{(S)}$ has no edge. Hence, a minimum vertex cover V_c^* of G is also a protection threshold of G. □

2. Top-*k* highest degree MVC

Let us recall that MVC is a set of nodes without any requirement of ordering. Intuitively, given k budget, selecting any k nodes from V_c^* may result in a different set of nodes. Additionally, not all of the node in MVC should have the same priority to be protected within a limited budget. We consider that the more connected a node v to its neighbors in G, the more critical node v to be protected. Hence, after obtaining MVC nodes from the input network, we reorder MVC nodes using their degree value within the input network.

Suppose that at time t we are given an input temporal snapshot graph G_t and protection budget k_t. Under the constraint of limited protection budget (k_t), we select top-k_t MVC nodes based on their degree value within graph G_t.

3. Reinforcement learning as solution approximation

Despite the protection threshold guarantee of MVC, finding the MVC nodes of graphs is NP-Hard (Hartman and Weigt 2006). We consider a reinforcement learning (RL) approach to approximate the solution. RL approach aims to obtain an optimal solution by maximizing the cumulative rewards without given any pre-defined deterministic policies (Mnih et al. 2015; Khalil et al. 2017). Such advantage enables us to exploit the known best policy while also consider exploring unknown policies to obtain an optimal solution.

More specifically, we leverage the n-step fitted Q-Learning (Khalil et al. 2017) to obtain MVC approximation with an efficient training process and scalable implementation. Hence, our proposed methods take the advantage of n-step Q-Learning (Sutton and Barto 1998) and fitted Q-iteration (Riedmiller 2005).

We let the n-step fitted Q-Learning iteratively learn to construct a vertex cover (V_c) solution of the input network. We define the RL environment as follows:

- State ($\mathbb{S}$): set of currently selected V_c nodes from input graph
- Action ($\mathbb{A}$): add new node v to vertex cover set $\mathbb{S}$
- Reward ($\mathbb{R}$): -1, as our goal is to get the minimum size of vertex cover, we set a penalty for adding a new node into V_c set.
- Termination criteria: all edges are covered

To quantify how good is taking an action $a \in \mathbb{A}$ given a state $s \in \mathbb{S}$, in Q-Learning, we have the Q-Function (Watkins 1989). Q-Function evaluates the pair of state and action

and maps it into a single value, called Q-Value, using the following Bellman optimality equation:

$$Q(s,a) = r + \lambda(\max(Q(s',a'))) \tag{7}$$

where $s \in \mathbb{S}$ is a given state, $a \in \mathbb{A}$ is the current action, r is the current reward, λ is the discount factor of the future rewards, $s' \in \mathbb{S}$ is the next state, and $a' \in \mathbb{A}$ is the next action. The calculation of Q-Function is performed and updated iteratively for each possible pair of state and action. The result of all Q-Value is stored in a table, called Q-Table. The best action for a given state is indicated by the highest Q-Value.

To obtain the maximum expected cumulative reward achievable from a given pair of state and action, we can compute the optimal Q-Function, denoted as Q^*, using the following equation:

$$Q^*(s,a) = \max \mathbb{E}[\sum_{t \geq 0} \lambda^t r_t | s_0 = s, a_0 = a] \tag{8}$$

where s_0 and a_0 are the initial state and action respectively, t indicates a step which consists of: observe a state, perform an action, retrieve a reward, and observe the next state.

As the number of all possible pair of state and action can be very large, calculating the Q-Value in Q-Table is not efficient. Especially, if we are handling a large-size input network, using Q-Table is computationally infeasible and resource-consuming. A non-linear function approximator can be used to estimate the optimal Q-Function in Eq. (8) such that:

$$Q(s,a,\Psi) \approx Q^*(s,a) \tag{9}$$

where Ψ is the function parameters (weights) of our non-linear function approximator $Q(s,a,\Psi)$. A neural network or a kernel function can be used as the non-linear function approximator of Q-Function (Sutton and Barto 1998).

Recent studies show that neural networks or convolutional neural networks achieve state-of-the-art results as function approximators (Mnih et al. 2015; Sutton and Barto 1998). The neural network architecture also speed up learning in finite problems, due to the fact that it can generalize from earlier experiences to previously unseen states (Mnih et al. 2015). In this paper, we propose a convolutional neural network as the function approximator of optimal Q-Function. Recall that in Q-Function, our input is a given state and action to obtain Q-Value as output. The state is the given input graph with currently selected V_c nodes. The actions are the possible nodes to be included into current V_c. In convolutional neural network architecture, our input should represents both of those state and action. Hence, we need a same fixed-length feature representation of the graph and each of its node. Therefore, in our construction of minimum vertex cover, our function approximator in Eq. (9) will be denoted as:

$$\hat{Q}(h(\mathbb{S}), v, \Psi) \tag{10}$$

where $h(\mathbb{S})$ and v represent the fixed-length feature representation of the state $\mathbb{S}$ and an action of adding node v using the neural network set of weights Ψ.

4. Graph embeddings as feature-based representations

We leverage an efficient and scalable graph embedding technique, called Structure2Vec (Dai et al. 2016; Khalil et al. 2017), to embed the input graph and each of its node. This graph embedding technique computes a d-dimensional feature embedding μ_v for each node $v \in V$, given the current partial solution $\mathbb{S}$.

Given a temporal snapshot graph G_t, we embed each node v by constructing a d-dimensional embedding μ_v. All of $\mu_v^{(0)}$ entries are initialized as zero, and for every $v \in V$ we update it iteratively in T iterations as follows:

$$\mu_v^{(t+1)} = \text{ReLU}\left(\psi_1 x_v + \psi_2 \sum_{u \in N(v)} \mu_u^{(t)} + \psi_3 \sum_{u \in N(v)} \text{ReLU}(\psi_4 w(u,v))\right), \tag{11}$$

with x_v is node v own tag, whether being already selected or not. Selected node will be given tag = 1, otherwise 0. $N(v)$ is the set of neighbors of node v in graph G_t. $\sum_{u \in N(v)} \mu_u^{(t)}$ is the feature of node v neighbors. $w(u,v)$ is the neighbors' edge weight, to consider the weighted connection in weighted graph. ψ_1, ψ_2, ψ_3, and ψ_4 are the function parameters (weights) which specified as $\psi_1 \in \mathbb{R}^d$, $\psi_2 \in \mathbb{R}^{dxd}$, $\psi_3 \in \mathbb{R}^{dxd}$, and $\psi_4 \in \mathbb{R}^d$. ReLU is the rectifier linear unit activation function applied elementwise to input where $\text{ReLU}(x) = x$ if $x > 0$ and 0 otherwise.

Here we will explain how to get the function $\hat{Q}(h(\mathbb{S}_t), v; \Psi)$ in Eq. (10). Once the embedding μ_v for each node $v \in V$ is calculated using Eq. (11) after T iteration, we get $\mu_v^{(T)}$. The pooled embedding of the entire graph G_t is then given by

$$\sum_{u \in V} \mu_u^{(T)} \tag{12}$$

Then we can use it to estimate the optimal Q-Function in Eq. (10) as follows:

$$\hat{Q}(h(\mathbb{S}), v; \Psi) = \psi_5^{\top} \text{ReLU}\left(\text{concat}\left(\psi_6 \sum_{u \in V} \mu_u^{(T)}, \psi_7 \mu_v^{(T)}\right)\right), \tag{13}$$

being $\sum_{u \in V} \mu_u^{(T)}$ is the pooled embedding of the entire graph. ψ_5, ψ_6, and ψ_7 are the neural network parameters (weights) which specified as $\psi_5 \in \mathbb{R}^{2d}$, $\psi_6 \in \mathbb{R}^{dxd}$, and $\psi_7 \in \mathbb{R}^{dxd}$.

To this end, we show that the pooled embedding of the entire graph is used as a surrogate to represent the state. And the embedding of each node is used as a surrogate to represent the action. The function $\hat{Q}(h(\mathbb{S}), v)$ is depend on the collection of seven parameters $\Psi = \{\psi_i\}_{i=1}^7$ which are learned during the training phase and will be evaluated during the evaluation phase. Figure 4 shows the architecture illustration of neural networks used in this paper.

a. Training Phase

Algorithm 1 illustrates our proposed training phase. In each training iteration, our method returns the neural network's set of parameters Ψ which successfully get V_c from graph G. In line 5, we specify how to select a new node by balancing exploration and exploitation. In this case, the exploration means selecting a random nodes with probability ϵ. The exploitation means we aim to get the maximum expected cummulative rewards, i.e. by selecting a node which maximizes the function $\hat{Q}(h(\mathbb{S}_t), v; \Psi)$. $h(\mathbb{S}_t)$ is the embedding of state $\mathbb{S}$ at step t. The exploration probability ϵ is set to decrease from 1.0 to 0.05

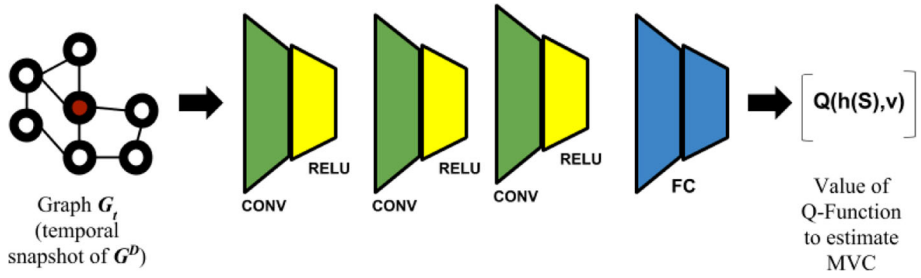

Fig. 4 Architecture illustration of the neural network used in our work. Green colored shape represents the convolutional layer. Yellow colored shape represents ReLU activation function. Blue colored shape represents the fully connected layer

Algorithm 1: n-step Fitted Q-Learning for Approximating the Minimum Vertex Cover Solution

Input: adjacency list of graph G

Output: neural network's set parameters (weights) Ψ

1 Initialize experience replay memory $\mathbb{M}$ to capacity N

2 **for** *episode* $e = 1, \cdots, E$ **do**

3 Initialize the state to empty $\mathbb{S}_1 = \{\}$

4 **for** *step* $t = 1, \cdots, T$ **do**

5 $$v_t = \begin{cases} \text{random node} v \in \bar{\mathbb{S}}_t, & \text{with probability} \epsilon \\ \arg\max_{v \in \bar{\mathbb{S}}_t} \hat{Q}(h(\mathbb{S}_t), v; \Psi), & \text{otherwise} \end{cases}$$

6 Add v_t to partial solution: $\mathbb{S}_{t+1} = (\mathbb{S}_t, v_t)$

7 **if** $t \geq n$ **then**

8 Add tuple $(\mathbb{S}_{t-n}, v_{t-n}, \mathbb{R}_{t-n,t}, \mathbb{S}_t)$ to $\mathbb{M}$

9 Sample random batch from $B \overset{iid.}{\sim} \mathbb{M}$

10 Update Ψ by Stochastic Gradient Descent over loss function $(y - \hat{Q}(h(\mathbb{S}_t), v_t; \Psi))^2$ for B

11 **end**

12 **end**

13 **return** Ψ

linear to the iteration step. To efficiently train the neural network, we perform batch processing as described in line 9.

The loss function which learned to minimize is as follows:

$$\left(y - \hat{Q}(h(\mathbb{S}_t), v_t; \Psi)\right)^2 \tag{14}$$

being $y = \sum_{i=0}^{n-1} r(S_{t+i}, v_{t+i}) + \lambda \max'_v \hat{Q}\left(h(\mathbb{S}_{t+n}), v'; \Psi\right)$. n is the number of step updates.

b. Evaluation Phase

Algorithm 2 illustrates the evaluation phase of our proposed method. To get the best-trained neural network's set of parameters (weights) Ψ^*, we evaluate the training result against a set of given graph G^D available snapshots. We use this neural network set of parameters in the testing simulation of the graph protection.

c. Testing Phase

Algorithm 2: MVC Evaluation

Input: snapshots of graph G^D, number of training iteration $iter_t$
Output: neural network set of best-trained parameters (weights) Ψ^*

Initialize $\Psi^* = 0$ and $|V_c| = 0$
for *training* $i = 1, \cdots, iter_t$ **do**
    Initialize a neural network with set of parameters/weights Ψ_i
    Get V_c of each snapshot of G^D using Ψ_i
    if $|V_c|_i < |V_c|$ **then**
        $\Psi^* = \Psi_i$
end
return Ψ^*

Algorithm 3: ***ReProtect***

Input: G_t current snapshot of graph G^D at time t, and k_t current budget at time t
Output: a set S of k_t nodes

Initialize S to be empty, $S = \{\}$
Embed each node in G_t into d-dimensional feature vector using Eqs. (11) and (12)
Obtain the set of minimum vertex cover nodes V_c^* (unordered) using the best-trained neural network (with set of parameter Ψ^*)
Reorder the set of minimum vertex cover nodes V_c^* based on their degree values within graph G_t
Get S from top k_t nodes in V_c^*
return S

Algorithm 3 shows the testing phase of multiple-turns graph protection strategy on dynamic networks. We are given an input snapshot of graph G_t and budget k_t. Each node in G_t is embedded into d-dimensional feature vector. The size of d is equal to the embedding size during training in Algorithm 1. The minimum vertex cover of G_t is then constructed using the best-trained neural network's set of parameters Ψ^* resulted from Algorithm 2. Finally, we get a set S of top-k_t degree-ordered MVC nodes to be protected from the current temporal snapshot of graph G_t.

We also propose ***ReProtect-p*** method, a variant of ***ReProtect***, which trained on the perturbed version of each available snapshot of dynamic networks. The perturbation is performed by removing edges probabilistically from the snapshot graph. Specifically, for each edge, we generate a random number. If the edge weight is smaller than the generated random number, the edge will be removed. We introduce this variant to provide more variety to the training data and avoid possible overfitting issue.

Computational complexity analysis

Based on Algorithm 3, we present the analysis of computational complexity of our proposed ***ReProtect*** method. The cost of step 1 to initialize empty set S is constant. The step

2 and 3 are to construct an approximated MVC set of graph G_t which has the complexity of $O(p \cdot M)$ based on the analysis by Dai et al. (2016); Khalil et al. (2017). p is the constant number of node testing steps, equals to the number of nodes divided by the number step updates in Q-Learning. M is the number of edges. In n-step Q-Learning, we update the value of each action based on the rewards of taking the sequence of n actions consecutively. n is called as the number of step updates. Suppose that the number of nodes in graph G_t is 500 and the number of step updates is 5, then p is a constant number equals to 100. One can see that p ranges from 1 to the number of nodes in graph G_t.

Getting the ordered MVC nodes in step 4 has an average $O(N \cdot logN)$ using QuickSort, where N is the number of nodes in G_t. Therefore, the total computational complexity of our ***ReProtect*** method is $O(p \cdot M + N \cdot logN)$. The difference of ***ReProtect*** and ***ReProtect-p*** is only on training process. Similarly, we can infer that the computational complexity of ***ReProtect-p*** method is also $O(p \cdot M + N \cdot logN)$.

Evaluation

In this section, we provide experimental evaluations of our proposed methods. The goal of this evaluation is to answer the following questions:

1. *(Effectiveness)* How effective are the proposed methods in restraining epidemic spreading in both synthetic and real-world dynamic networks? We define the measurement of effectiveness using the surviving ratio (θ) of nodes in dynamic network G^D at the end of epidemics.
2. *(Scalability)* Are the proposed methods scalable with respect to the changing of graph size (in terms of the number of nodes) and different protection budget size (k)?
3. *(Sensitivity Analysis)* How is the effectiveness of our proposed methods in the different values of epidemic parameters, such as the infection rate (β) and recovery rate (δ)?

Dataset

We evaluate our proposed methods on various real-world dynamic network datasets, which summarized in Table 3.

- *Dutch College* dataset is a directed network of friendship ratings among 32 university freshmen (Van de Bunt et al. 1999). Each student was asked to rate the others at seven different time points.
- *Hospital* dataset contains the temporal network of human contacts between patients and health-care workers in a hospital ward in Lyon, France (Vanhems et al. 2013). Data was collected in December 2010.
- *Hypertext 2009* dataset is the network of contacts of the attendees of the ACM Hypertext 2009 conference (Stehlé et al. 2011). In the network, a node represents a conference visitor, and an edge represents a face-to-face contact.
- *PrimarySchool* dataset contains the temporal network of contacts between teachers and children used in the study of BMC Infectious Diseases 2014 (Gemmetto et al. 2014; Stehlé et al. 2011).
- *Highschool 2013* dataset contains the temporal network of contacts between students in a high school in Marseilles, France (Mastrandrea et al. 2015). The data was collected in December 2011 and November 2012.

Table 3 Statistics of dynamic network datasets

Dataset	#nodes	#edges	timespan	#snapshots	Average degree of network snapshots	Average degree of the aggregated network
Dutch college	32	3,062	21 days	7	{1.40; 18.07; 21.31; 21.81; 21.50; 24.06; 22.38}	26.38
Hospital	75	32,424	1 day	5	{8.33; 19.35; 18.45; 16.88; 13.87}	30.37
Hypertext 2009	113	20,818	12 h	6	{12.58; 12.04; 15.77; 10.19; 13.03; 10.53}	38.87
PrimarySchool	242	125,773	1 h	18	{6.87; 15.77; 17.39; 16.67; 21.15; 15.00; 8.74; 17.79; 6.89; {10.56; 15.47; 14.37; 18.08; 24.36; 14.72; 9.88; 20.87; 9.72}	68.74
Highschool 2013	329	188,508	1 day	5	{14.37; 16.60; 14.26; 14.66; 13.88}	35.58
Infectious	410	17,298	1 h	8	{6.15; 8.22; 11.25; 10.57; 12.48; 7.82; 6.49; 5.61}	13.49
Email	986	332,334	30 days	19	{9.33; 10.18; 8.65; 9.78; 9.80; 10.22; 9.82; 11.40; 10.35; 7.26; {9.70; 11.37; 11.17; 6.10; 10.75; 11.40; 11.99; 8.48; 4.06}	32.58

Table 4 Ratio of surviving nodes (θ) on synthetic network

Epidemic model	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
SIS	0.2020	0.4344	0.7673	0.7552	0.3216	0.7502	0.7676	0.8049	**0.8252**
SIR	0.2021	0.4350	0.7672	0.7559	0.3220	0.7503	0.7676	0.8035	**0.8237**

- *Infectious* dataset is the network of face-to-face people behavior during the Dublin Science Gallery 2009 exhibition (Isella et al. 2011).
- *Email* dataset was obtained from the email communication between institution members (the core) from a large European research institution (Paranjape et al. 2017). A directed edge (u, v, t) means that person u sent an e-mail to person v at time t in the network.

Comparison methods

Recall that to the best of our knowledge, there is no previous work has been proposed to handle the multiple-turns graph protection problem on dynamic networks. Here, we investigate the performance comparison of the following methods:

- *None*: simulates the condition without any protection.
- *GreedyMVC*: approximates the set of MVC nodes of the input graph by greedily selects the uncovered edge with the maximum sum of degrees of its endpoints (Khalil et al. 2017). Then protects k nodes from this unordered MVC set.
- *Degree* (Prakash et al. 2010): protects k highest degree nodes of the current snapshot of the dynamic network. This method represents the concept of NLDS-Degree by Prakash et al. (2010).
- *Betweenness*: protects k nodes with the highest betweenness centrality of the current snapshot of the dynamic network.
- *NetShield+* (Tong et al. 2010; Chen et al. 2016; Prakash et al. 2010): aims to protect a set of k nodes considering the largest eigenvalue of adjacency matrix. This methods represents the stronger variant of eigendecomposition-based methods by Chen et al. (2016) and NLDS-EigenValue by Prakash et al. (2010).
- *GraphShield* (Wijayanto and Murata 2017): protects k nodes by taking into account the role of graph connectivity and degree centrality.
- *ApproxDegree*: simulates the 2-approximation algorithm to get the MVC nodes (Chakrabarti; Hartman and Weigt 2006). We add the degree-ordering nodes to this method for protecting the top-k highest degree of MVC nodes.
- ***ReProtect*** and ***ReProtect-p***: are our proposed methods.

Experimental setting

In the training phase, we use the embedding dimension size 64, batch size 64, embedding iteration 5 as suggested in Structure2Vec[1] (Dai et al. 2016). The setting of n-step is set to 5 and learning rate as 0.0001 and number of training iteration as 100,000. These three settings are commonly applied in n-step Q-Learning (Sutton and Barto 1998). In the evaluation phase, we consider the number of evaluation iteration as 100.

For a fair comparison, unless specified otherwise, all of the methods are simulated under the same setting as follows: infection rate $\beta = 0.8$, recovery rate $\delta = 0.2$, and the initial number of attacked nodes (l) equals to the protection budget (k). We simulate $l = k, l_t = k_t$, and $k_1 = k_2 = \cdots = k_t$. Random attack evaluation is employed in all experiments. The setting applies for evaluation in both SIS and SIR epidemic model.

All results presented in this section are the average of multiple simulations. Unless specified otherwise, we take the average from 100 simulations for each result. The initial condition is all nodes susceptible except the attacked ones, which are infected.

We let the epidemic spreading arrive at the stationary state before changing to the next snapshot of the network for SIS model. While for SIR model, we count the ratio of surviving nodes at the highest outbreak point, right before the final regime of epidemics as suggested by Pastor-Satorras et al. (2015). For continuity, in SIR model, we restart the epidemic spreading in the new snapshot after the final regime of epidemic spreading in the previous snapshot. Gillespie algorithm (Kiss et al. 2017) is used to simulate the epidemic spreading on networks. Additionally, we follow the time discretization method of dynamic network by Zhuang et al. (2013).

Finally, all of the experiments are performed on the same machine, Ubuntu 16.06 LTS PC with an Intel(R) Core(TM) i9-7900X CPU @ 3.30GHz CPU and NVIDIA GTX 1080 Ti SLI GPU.

Evaluation of effectiveness on synthetic network

We evaluate the performance of all comparison methods on a synthetic network generated using Dynamic Attributed Network with Community Structure Generator[2] (DANCER) (Largeron et al. 2017). Due to the simplicity setting of graph protection problem, we only consider the temporal network structure of the generated network and ignore their attribute and community assignment provided by DANCER. We generate a dynamic network with 100 nodes and ten temporal snapshots[3].

Table 4 shows the average result of 100 simulations under the constraint of protection budget $k = 0.25N$, being N is the number of nodes in the input graph. Both of our proposed methods obtain a higher ratio of surviving nodes than other competitors. ***ReProtect-p*** achieves the highest protection effectiveness.

If we vary the number of given budget k, both of our proposed methods outperform the other baseline methods as shown in Fig. 5. When the given protection budget k is too small, ***ReProtect*** and ***ReProtect-p*** exhibit competitive performances with other methods, but with an increasing k, they easily outperform other baseline methods, such as Degree, GraphShield, and Betweenness. On the other hand, ***ReProtect*** can also outperform other competing methods, even though it needs a bigger protection budget to obtain the similar performance of ***ReProtect-p***. An introduction of more data variety using graph perturbation into training process helps our proposed method to get a better result, as in ***ReProtect-p***.

Evaluation of effectiveness on real-world networks

On real-world networks, we compare the performance of all comparison methods on seven different datasets and two different epidemic models, i.e., SIS and SIR model. Table 5 and 6 show the result of surviving nodes ratio on SIS and SIR epidemic model respectively. The results are averaged from 100 simulations under the constraint of protection budget $k = 0.15N$, with N is the number of nodes in the input graph. Both of our proposed methods consistently reach the highest ratio of surviving nodes. Additionally, in most cases, the proposed method with more training data variety using the perturbed graph, namely ***ReProtect-p*** achieves a better result than the regular training as in ***ReProtect***. Tables 7 and 8 present the standard deviation of the surviving nodes ratio.

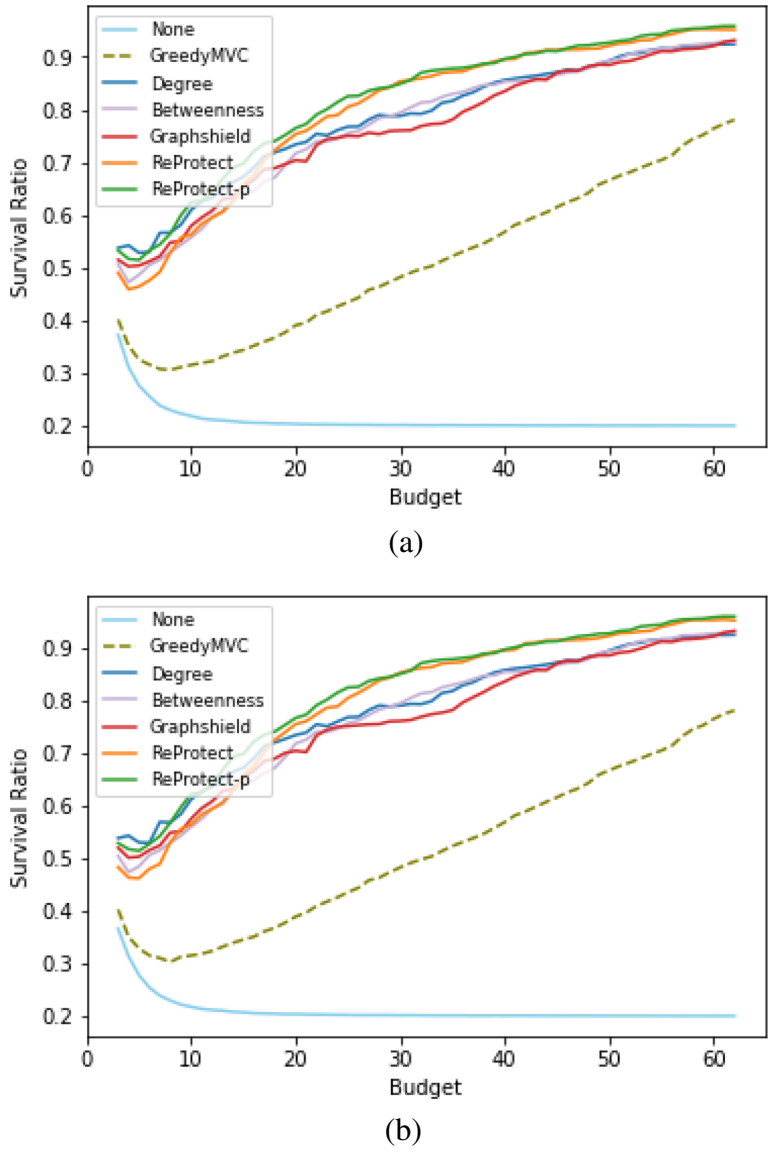

Fig. 5 Effectiveness evaluation on synthetic network. Both of our proposed methods (green and orange colored lines) outperform the other competitors. Higher is better. **a** SIS Epidemic Model. **b** SIR Epidemic Model

To evaluate the performance comparison in different protection budget k, we vary the given k as shown in Figs. 6 and 7. Both of our proposed methods are able to outperform other competitors align with the increase given budget in all datasets, while constantly maintain competitive performance in a very small size of k. The consistency of better performance shown by our methods in many different numbers of available protection budget indicates the reliability as protection strategies.

We consider that the reinforcement learning is more suitable for graph protection on dynamic networks due to at least two major reasons. *First*, reinforcement learning approach using convolutional neural network as function approximator gives us a potential benefit to learn from previously solved MVC of network snapshot. By learning the given temporal structure of observed networks, this provides an incentive to predict

Table 5 Ratio of surviving nodes (θ) on real-world networks (SIS Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	**ReProtect**	**ReProtect-p**
DutchCollege	0.2167	0.4065	0.6311	0.6334	0.4368	0.6430	0.6610	**0.6658**	0.6568
Hospital	0.2031	0.4425	0.6725	0.6565	0.2956	0.6696	0.6718	0.7052	**0.7075**
Hypertext2009	0.2063	0.3819	0.6197	0.6142	0.2724	0.6234	0.6257	0.6377	**0.6449**
PrimarySchool	0.2280	0.3999	0.8382	0.8154	0.3153	0.8385	0.8355	0.8371	**0.8409**
Highschool2013	0.2044	0.3403	0.5351	0.5339	0.2285	0.5413	0.5399	**0.5681**	0.5561
Infectious	0.2318	0.5155	0.7793	0.7369	0.3154	0.7836	0.7887	0.7894	**0.8003**
Email	0.2310	0.4579	0.8261	0.7969	0.2481	0.8228	**0.8836**	0.8335	0.8469

Table 6 Ratio of surviving nodes (θ) on real-world networks (SIR Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
DutchCollege	0.2123	0.4067	0.6292	0.6348	0.4355	0.6492	0.6600	**0.6623**	0.6576
Hospital	0.2033	0.4432	0.6745	0.6558	0.2953	0.6733	0.6724	**0.7069**	0.7034
Hypertext2009	0.2066	0.3802	0.6208	0.6120	0.2729	0.6262	0.6254	0.6374	**0.6442**
PrimarySchool	0.2275	0.3995	0.8386	0.8154	0.3158	0.8391	0.8353	0.8365	**0.8416**
Highschool2013	0.2044	0.3406	0.5366	0.5335	0.2286	0.5411	0.5398	**0.5681**	0.5564
Infectious	0.2300	0.5167	0.7784	0.7364	0.3144	0.7860	0.7882	0.7915	**0.8003**
Email	0.2313	0.4582	0.8277	0.7970	0.2476	0.8241	**0.8837**	0.8341	0.8471

Table 7 Standard deviation of the surviving nodes ratio (θ) (SIS Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
Synthetic	0.0015	0.0032	0.0087	0.0071	0.0024	0.0079	0.0094	0.0087	0.0085
DutchCollege	0.0148	0.0119	0.0230	0.0241	0.0294	0.0267	0.0284	0.0226	0.0231
Hospital	0.0024	0.0042	0.0155	0.0111	0.0044	0.0176	0.0164	0.0169	0.0167
Hypertext2009	0.0035	0.0056	0.0127	0.0090	0.0039	0.0128	0.0131	0.0115	0.0127
PrimarySchool	0.0047	0.0052	0.0044	0.0045	0.0051	0.0051	0.0050	0.0044	0.0046
HighSchool2013	0.0015	0.0023	0.0067	0.0043	0.0016	0.0057	0.0076	0.0070	0.0071
Infectious	0.0068	0.0074	0.0109	0.0092	0.0070	0.0108	0.0125	0.0111	0.0109
Email	0.0014	0.0019	0.0023	0.0020	0.0014	0.0022	0.0023	0.0022	0.0022

Table 8 Standard deviation of the surviving nodes ratio (θ) (SIR Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
Synthetic	0.0014	0.0032	0.0084	0.0074	0.0024	0.0080	0.0097	0.0085	0.0086
DutchCollege	0.0147	0.0121	0.0228	0.0235	0.0289	0.0262	0.0300	0.0240	0.0244
Hospital	0.0024	0.0044	0.0154	0.0108	0.0045	0.0160	0.0161	0.0165	0.0157
Hypertext2009	0.0037	0.0055	0.0122	0.0093	0.0038	0.0126	0.0132	0.0118	0.0118
PrimarySchool	0.0047	0.0053	0.0046	0.0044	0.0050	0.0051	0.0053	0.0044	0.0044
Highschool2013	0.0016	0.0022	0.0067	0.0045	0.0016	0.0056	0.0076	0.0073	0.0072
Infectious	0.0069	0.0074	0.0113	0.0094	0.0070	0.0107	0.0125	0.0105	0.0113
Email	0.0015	0.0019	0.0022	0.0020	0.0015	0.0022	0.0024	0.0021	0.0022

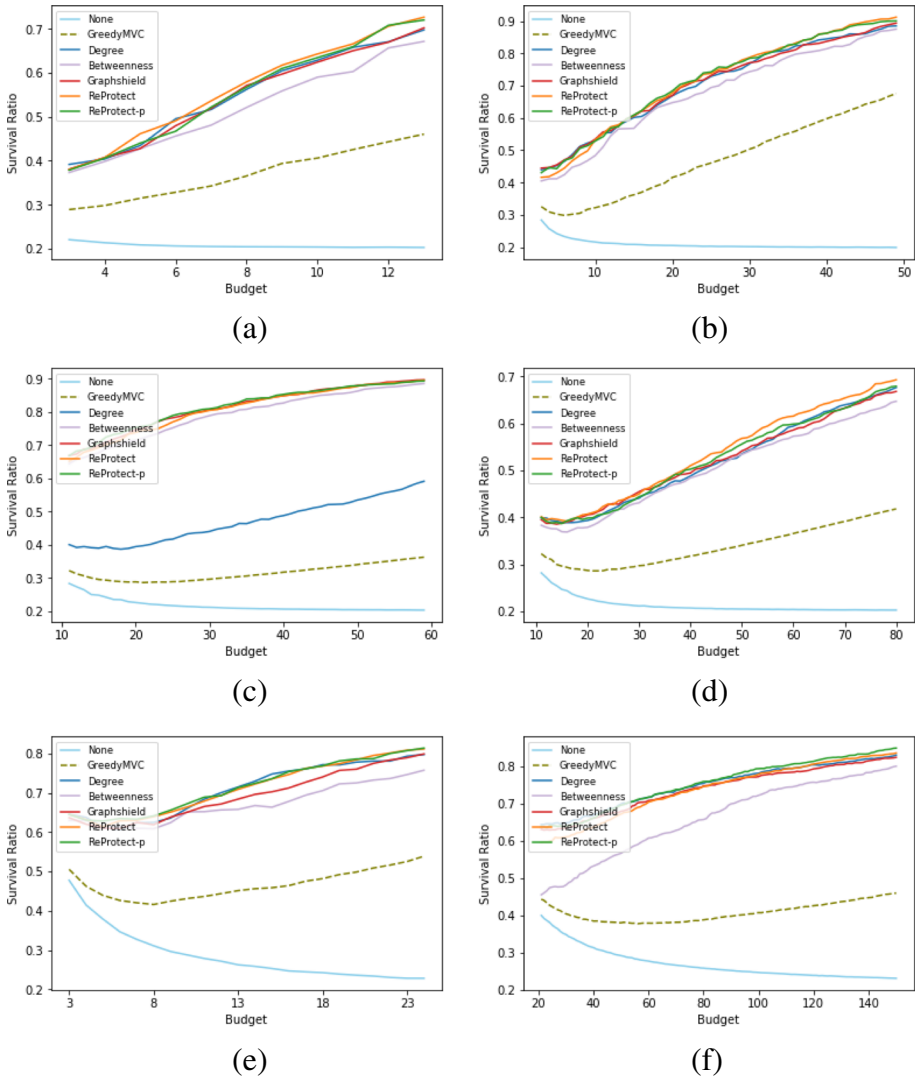

Fig. 6 Effectiveness evaluation on SIS epidemic model. Both of our proposed methods (green and orange colored lines) outperform the other competitors. Higher is better. **a** Hospital. **b** Hypertext 2009. **c** PrimarySchool. **d** Highschool 2013. **e** Infectious. **f** Email Figure

future protection from previously learned actions in the same dynamic networks. The benefit of learning could not be obtained by traditional MVC approximation algorithms. *Second*, the nature of convolutional neural networks (CNN) provide us not only scalability in handling the large size networks which may contain up to billion nodes, but also easily parallelizable in multiple CPUs and GPUs. Here, we leverage our approach on top of recent advances in deep learning technology. Traditional MVC approximation algorithms are not specifically designed for this computationally expensive task.

Evaluation of effectiveness on the aggregate networks

According to the observations of Braha and Bar-Yam (2006; 2009), the snapshots static networks are quite different from the aggregate network itself. *The aggregate network* is the network obtained by ignoring time and aggregating all of the temporal edges in the

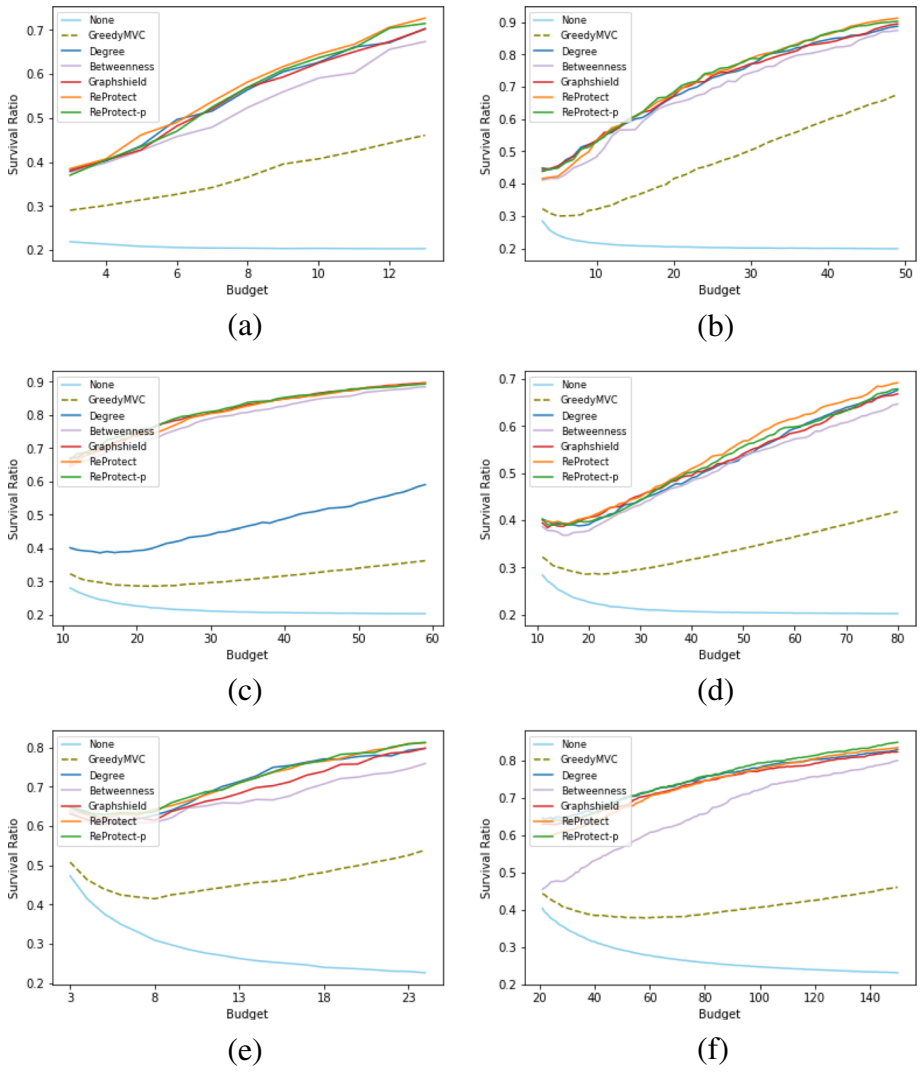

Fig. 7 Effectiveness evaluation on SIR epidemic model. Both of our proposed methods (green and orange colored lines) outperform the other competitors. Higher is better. **a** Hospital. **b** Hypertext 2009. **c** PrimarySchool. **d** Highschool 2013. **e** Infectious. **f** Email Figure

dynamic network (Braha and Bar-Yam 2006; 2009). An interesting question arises, how does the multiple-turns time-based protection strategies analyzed in our proposed methods compare with the protection strategies when implemented on the aggregate network? In this subsection, we report the effectiveness evaluation on the aggregate networks of the same synthetic and real-world datasets.

Tables 9 and 10 show the result of surviving nodes ratio (θ) on SIS and SIR epidemic model respectively. The results are averaged from 100 simulations under the constraint of protection budget $k = 0.15N$, with N is the number of nodes in the input graph. Tables 11 and 12 present the standard deviation of the surviving nodes ratio. Compared with the results in Tables 5 and 6, we found that the multiple-turns time-based strategies are beneficial and more effective than the aggregate-based strategies. The aggregate-based strategies are the protection strategies applied on the aggregate networks

Table 9 Ratio of surviving nodes (θ) on the aggregate networks (SIS Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
Synthetic	0.2000	0.4018	0.4037	0.4000	0.2200	0.4038	0.4037	0.4038	0.4039
DutchCollege	0.1875	0.3125	0.3126	0.3125	0.2739	0.3126	0.3126	0.3126	0.3125
Hospital	0.2000	0.3333	0.3334	0.3333	0.2133	0.3334	0.3334	0.3334	0.3333
Hypertext2009	0.2035	0.3262	0.3263	0.3198	0.2135	0.3262	0.3260	0.3260	0.3262
PrimarySchool	0.1983	0.3223	0.3223	0.3223	0.2066	0.3223	0.3223	0.3223	0.3223
Highschool2013	0.1989	0.3213	0.3213	0.3217	0.2050	0.3212	0.3212	0.3213	0.3213
Infectious	0.2791	0.3374	0.3574	0.3341	0.2874	0.3386	0.3596	0.3588	0.3616
Email	0.2150	0.3618	0.3984	0.3552	0.2168	0.3954	0.3985	0.3982	0.3980

Table 10 Ratio of surviving nodes (θ) on the aggregate networks (SIR Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
Synthetic	0.2000	0.4016	0.4036	0.4000	0.2200	0.4037	0.4039	0.4040	0.4036
DutchCollege	0.1875	0.3125	0.3125	0.3125	0.2739	0.3125	0.3126	0.3125	0.3125
Hospital	0.2000	0.3333	0.3334	0.3333	0.2133	0.3334	0.3334	0.3334	0.3334
Hypertext2009	0.2035	0.3262	0.3261	0.3198	0.2137	0.3262	0.3263	0.3263	0.3262
PrimarySchool	0.1983	0.3223	0.3223	0.3223	0.2066	0.3223	0.3223	0.3223	0.3223
HighSchool2013	0.1989	0.3212	0.3213	0.3217	0.2050	0.3213	0.3213	0.3213	0.3213
Infectious	0.2790	0.3388	0.3593	0.3333	0.2869	0.3397	0.3616	0.3626	0.3592
Email	0.2153	0.3618	0.3981	0.3553	0.2168	0.3952	0.3984	0.3983	0.3980

Table 11 Standard deviation of the surviving nodes ratio (θ) on the aggregate networks (SIS Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
Synthetic	0.0005	0.0042	0.0057	0.0004	0.0000	0.0057	0.0056	0.0058	0.0056
DutchCollege	0.0000	0.0000	0.0010	0.0000	0.0132	0.0010	0.0014	0.0000	0.0010
Hospital	0.0000	0.0000	0.0006	0.0000	0.0000	0.0006	0.0007	0.0009	0.0006
Hypertext2009	0.0000	0.0032	0.0032	0.0031	0.0032	0.0032	0.0031	0.0033	0.0030
PrimarySchool	0.0000	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
HighSchool2013	0.0005	0.0007	0.0007	0.0012	0.0005	0.0008	0.0007	0.0009	0.0006
Infectious	0.0256	0.0252	0.0338	0.0273	0.0264	0.0277	0.0333	0.0315	0.0334
Email	0.0032	0.0040	0.0049	0.0035	0.0034	0.0049	0.0049	0.0050	0.0050

Table 12 Standard deviation of the surviving nodes ratio (θ) on the aggregate networks (SIR Epidemic Model)

Dataset	None	GreedyMVC	Degree	Betweenness	NetShield+	GraphShield	ApproxDegree	ReProtect	ReProtect-p
Synthetic	0.0007	0.0043	0.0055	0.0006	0.0000	0.0056	0.0057	0.0060	0.0055
DutchCollege	0.0000	0.0000	0.0014	0.0000	0.0132	0.0014	0.0000	0.0000	0.0000
Hospital	0.0000	0.0000	0.0007	0.0000	0.0000	0.0013	0.0008	0.0010	0.0013
Hypertext2009	0.0000	0.0033	0.0033	0.0032	0.0031	0.0032	0.0032	0.0031	0.0032
PrimarySchool	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
HighSchool2013	0.0005	0.0007	0.0007	0.0012	0.0005	0.0007	0.0007	0.0008	0.0006
Infectious	0.0248	0.0273	0.0327	0.0280	0.0241	0.0297	0.0337	0.0338	0.0350
Email	0.0033	0.0042	0.0050	0.0035	0.0034	0.0048	0.0048	0.0048	0.0049

under the assumption that the time-aggregated networks are accessible and known a priori. We observe that the time-aggregation of all edges make the network denser thus require more nodes to be protected. The average degree of nodes in each snapshot of the network compared with that of in the aggregated network is shown in Table 3. In the aggregated network, the average degree of nodes is higher than in each network snapshot.

Evaluation of scalability

Let us recall our second evaluation goal, which aims to measure how scalable is the proposed method with respect to the changing of graph size and different k budget size. In this subsection, we report the result of scalability evaluation by investigating the computational running time of our proposed methods. Different values of k were used to evaluate the scalability in different scale of protection set.

To perform simulation by changing the number of nodes, we generate synthetic dynamic networks using Dynamic Attributed Network with Community Structure Generator (DANCER) (Largeron et al. 2017). We only consider the temporal network structure of the generated network and ignore their attribute and community assignment provided by DANCER. We generate dynamic networks with 10 temporal snapshots and the number of nodes is changed from $N = \{100; 200; 300; 500; 1000; 1500; 2000\}$. The budget size is changed from $\{10; 20; 30; 40; 50\}$.

From Fig. 8, it can be inferred that our methods scale almost linearly with respect to the number of nodes. Hence, the proposed methods are scalable with respect to the changing

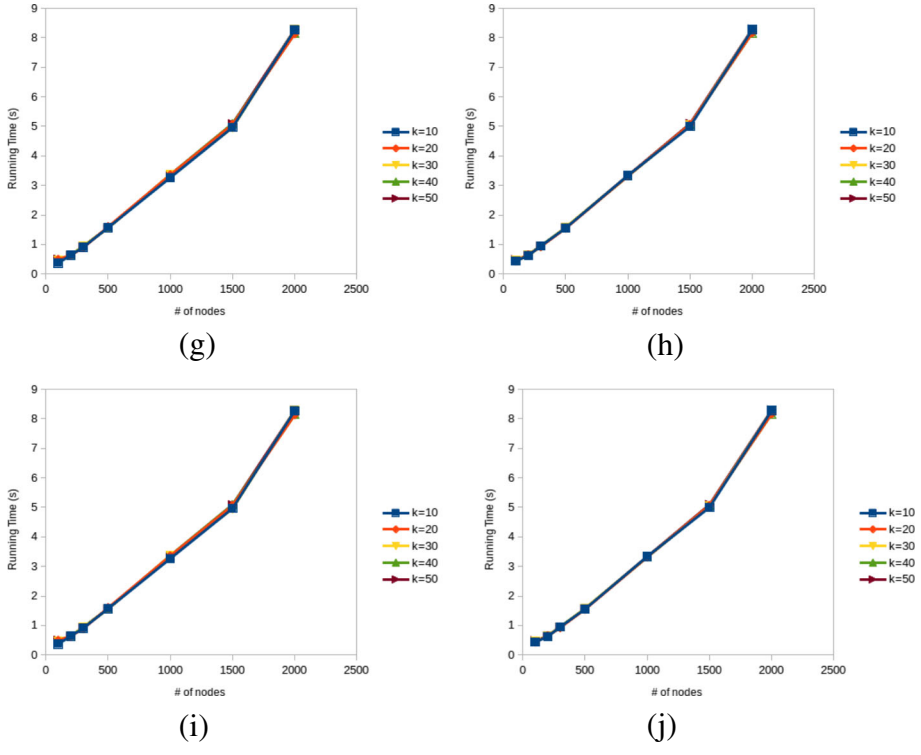

Fig. 8 Scalability evaluation of the proposed methods. **g** ***ReProtect*** on SIS Model. **h** ***ReProtect*** on SIR Model. **i** ***ReProtect-p*** on SIS Model. **j** ***ReProtect-p*** on SIR Model

of graph size, which means they are applicable for large size networks. Running our methods on graph with 2000 nodes takes less than 9 seconds. Further paralelization of neural network design can also be applied to speed up the running time.

Evaluation of sensitivity to epidemic parameters

In SIS and SIR model, epidemic parameters consist of the infection rate (β) and the recovery rate (δ). To analyze the sensitivity of our proposed methods, the effectiveness comparison with different epidemic parameters are shown in this subsection.

Prakash et al. (2011) demonstrated using empirical simulations that the ratio of infection rate over recovery rate $\left(\frac{\beta}{\delta}\right)$ takes the role as constant dependent of epidemic threshold in various epidemic model including SIS and SIR. Epidemic threshold is an intrinsic property of a network. When the strength of the virus is greater than the epidemic threshold, then the epidemic would breakout (Prakash et al. 2011). The ratio of $\frac{\beta}{\delta}$ is commonly called as the *epidemic propagation rate* (Wijayanto and Murata 2018b; Prakash et al. 2011).

We perfom simulations to confirm the effectiveness of our proposed methods using the same network dataset in **Effectiveness on Synthetic Network** subsection under three scenarios:

(1) Comparison of survival ratio θ when the epidemic propagation rate $\left(\frac{\beta}{\delta}\right)$ changes
(2) Comparison of survival ratio θ when the infection rate (β) changes
(3) Comparison of survival ratio θ when the recovery rate (δ) changes

For a fair analysis and comparison, simulations are performed under a fixed protection budget(k).

Comparison of survival ratio θ when the epidemic propagation rate $\left(\frac{\beta}{\delta}\right)$ changes

We change the ratio of $\frac{\beta}{\delta}$ from $\left\{\frac{0.9}{0.1}; \frac{0.8}{0.2}; \frac{0.7}{0.3}; \frac{0.6}{0.4}; \frac{0.5}{0.5}; \frac{0.4}{0.6}; \frac{0.3}{0.7}; \frac{0.2}{0.8}; \frac{0.1}{0.9}\right\}$. Figure 9 shows the comparison of survival ratio θ of all methods in SIS and SIR epidemic model. The results are averaged from 100 simulations under the fixed protection budget $k = 0.25N$, with N is the number of nodes of the input network. In all of these conditions, both of our proposed methods obtain higher survival ratio θ than other competitors.

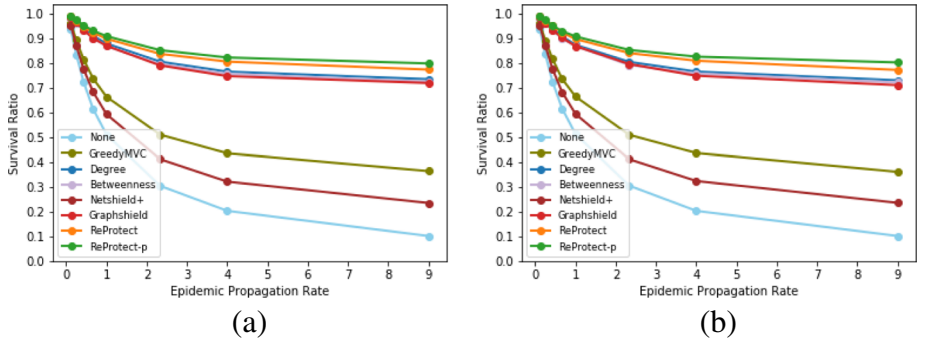

Fig. 9 Evaluation of sensitivity to the epidemic propagation rate (the ratio of $\frac{\beta}{\delta}$). Higher is better. **a** SIS Model. **b** SIR Model

Comparison of survival ratio θ when the infection rate (β) changes

We change the infection rate (β) from $\{0.9; 0.8; 0.7; 0.6; 0.5; 0.4; 0.3; 0.2; 0.1\}$ with fixed recovery rate (δ). Figure 10 shows the comparison of survival ratio θ of all methods. The results are presented from the average of 100 simulations with a fixed protection budget $k = 0.25N$, where N is the number of nodes of the input network. Both of our proposed methods could achieve highest survival ratio θ regardless the value of infection rate and epidemic models.

Comparison of survival ratio θ when the recovery rate (δ) changes

We investigate the comparison of survival ratio θ by changing the recovery rate (δ) from $\{0.9; 0.8; 0.7; 0.6; 0.5; 0.4; 0.3; 0.2; 0.1\}$ with fixed infection rate (β) and the protection budget $k = 0.25N$. N is the number of nodes of the input network. As shown in Fig. 11, in SIS and SIR epidemic model, both of ***ReProtect*** and ***ReProtect-p*** methods obtain higher survival ratio θ than other competitors. The results are averaged from 100 simulations.

Conclusion

In this paper, we addressed the multiple-turns graph protection problem to restrain epidemic spreading on dynamic networks. The protection budget is divided into several turns and selects protected nodes based on the presently observed temporal snapshot of dynamic networks. By proving the role of minimum vertex cover (MVC) as the protection threshold of the network, we choose to protect the highest degree of MVC nodes at the size of each allocated protection budget. We introduce methods utilizing the n-step fitted Q-Learning to efficiently learn the MVC construction from the input graph under reinforcement learning approach. Graph embedding technique is incorporated as a feature-based representation of the input network states. We demonstrate the effectiveness and scalability of our methods, namely ***ReProtect*** and ***ReProtect-p***. Extensive evaluations on synthetic and real-world network datasets show that our proposed methods outperform other baseline methods while maintaining the scalability. Further investigation of two different epidemic model simulation, i.e., SIS and SIR model, also confirm the effectiveness and scalability of our methods.

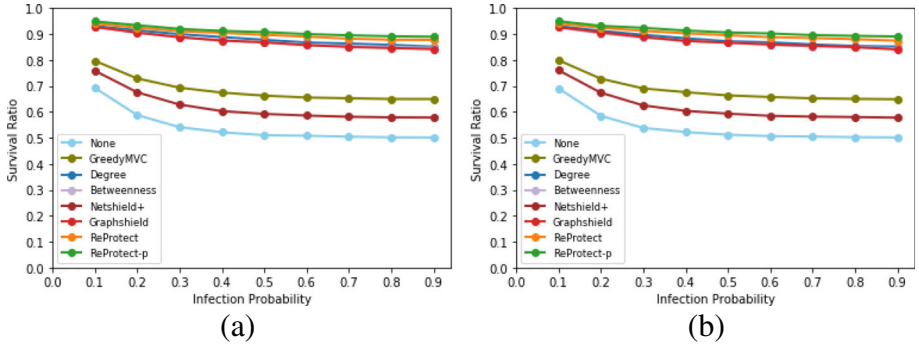

Fig. 10 Evaluation of sensitivity to the infection rate (β). Higher is better. **a** SIS Model. **b** SIR Model

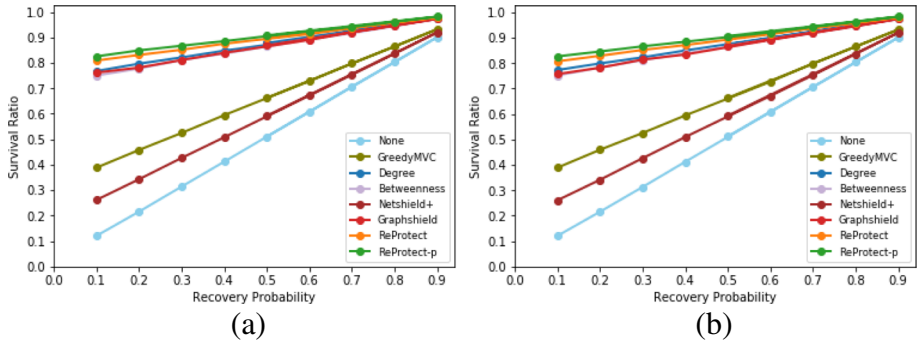

Fig. 11 Evaluation of sensitivity to the recovery rate (δ). Higher is better. **a** SIS Model. **b** SIR Model

The strategy of handling graph protection problem against non-trivial targeted attacks in dynamic networks is left for our future work. Extending our methods into a multi-agent policy gradient reinforcement learning to achieve better training efficiency will also be our next consideration.

Abbreviations

CNN: Convolutional neural network; DANCER: Dynamic attributed network with community structure generator; DAVA: Data-aware vaccine allocation; DBMF: Degree-based mean-field; DRIMUX: Dynamic rumor influence minimization with user experience; IBMF: Individual-based mean-field; MVC: Minimum vertex cover; NIIP: Node immunization over infectious period; NLDS: Non-Linear dynamical system; ReLU: Rectifier linear unit; RL: Reinforcement learning; SIS: Susceptible-Infected-Susceptible epidemic model; SIR: Susceptible-Infected-Recovered epidemic model; TIM: Targeted immunization method; VAILDN: Vaccination allocation in large dynamic networks

Acknowledgements

We thank the anonymous reviewers for their valuable suggestions and comments.

Funding

This work was supported by JSPS Grant-in-Aid for Scientific Research(B) (Grant Number 17H01785) and JST CREST (Grant Number JPMJCR1687). A.W.W. is suported by Indonesia Endowment Fund for Education (LPDP) for the educational scholarship.

Availability of data and materials

All of the datasets used in this paper are publicly accessible online in their respective references.
Hospital, Hypertext 2009, PrimarySchool, Highschool 2013, and Infectious dataset can be obtained from SocioPatterns (http://www.sociopatterns.org/datasets).
Dutch College dataset is available at Konect (http://konect.uni-koblenz.de/networks).
Email dataset can be found at SNAP (http://snap.stanford.edu/data).
The source code of our proposed methods will be accessible by request upon acceptance.

Authors' contributions

All authors contribute to the writing of the paper. All authors read and approved the final manuscript.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Received: 27 October 2018 Accepted: 12 March 2019
Published online: 18 April 2019

References

Bakker C, Halappanavar M, Visweswara Sathanur A (2018) Dynamic graphs, community detection, and riemannian geometry. Appl Netw Sci 3(1):3. https://doi.org/10.1007/s41109-018-0059-2

Bishop AN, Shames I (2011) Link operations for slowing the spread of disease in complex networks. EPL (Europhys Lett) 95(1):18005
Braha D, Bar-Yam Y (2009) Time-Dependent Complex Networks: Dynamic Centrality, Dynamic Motifs, and Cycles of Social Interactions(Gross T, Sayama H, eds.). Springer, Berlin. https://doi.org/10.1007/978-3-642-01284-6_3
Braha D, Bar-Yam Y (2006) From centrality to temporary fame: Dynamic centrality in complex networks. Complexity 12:59–63
Braunstein A, Dall'Asta L, Semerjian G, Zdeborová L (2016) Network dismantling. Proc Natl Acad Sci 113(44):12368–12373. https://doi.org/10.1073/pnas.1605083113
Buono C, Braunstein LA (2015) Immunization strategy for epidemic spreading on multilayer networks. EPL (Europhys Lett) 109(2):26001
Chakrabarti A Approximation Algorithms: Vertex Cover (Computer Science 105 - Winter 2005). http://tandy.cs.illinois.edu/dartmouth-cs-approx.pdf Accessed 27 Dec 2017
Chen C, Tong H, Prakash BA, Tsourakakis CE, Eliassi-Rad T, Faloutsos C, Chau DH (2016) Node immunization on large graphs: Theory and algorithms. IEEE Trans Knowl Data Eng 28(1):113–126
Dai H, Dai B, Song L (2016) Discriminative embeddings of latent variable models for structured data. In: Balcan MF, Weinberger KQ (eds). Proceedings of The 33rd International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 48. PMLR, New York. pp 2702–2711
Enright J, Kao RR (2018) Epidemics on dynamic networks. Epidemics 24:88–97. https://doi.org/10.1016/j.epidem.2018.04.003
Gemmetto V, Barrat A, Cattuto C (2014) Mitigation of infectious disease at school: targeted class closure vs school closure. BMC Infect Dis 14(1):695. https://doi.org/10.1186/PREACCEPT-6851518521414365
Habiba, Yu Y, Berger-Wolf TY, Saia J (2010) Finding spread blockers in dynamic networks. In: Giles L, Smith M, Yen J, Zhang H (eds). Advances in Social Network Mining and Analysis. Springer, Berlin. pp 55–76
Hartman AK, Weigt M (2006) Phase Transitions in Combinatorial Optimization Problems: Basics, Algorithms and Statistical Mechanics. Wiley-VCH Verlag, Weinheim
Hill SA, Braha D (2010) Dynamic model of time-dependent complex networks. Phys Rev E 82:046105
Holme P (2015) Modern temporal network theory: a colloquium. Eur Phys J B 88(9). https://doi.org/10.1140/epjb/e2015-60657-4
Holme P, Saramäki J (2012) Temporal networks. Phys Rep 519(3):97–125. https://doi.org/10.1016/j.physrep.2012.03.001. Temporal Networks
Isella L, Stehlé J, Barrat A, Cattuto C, Pinton J, Van den Broeck W (2011) What's in a crowd? analysis of face-to-face behavioral networks. J Theor Biol 271(1):166–180. https://doi.org/10.1016/j.jtbi.2010.11.033
Italiano GF, Laura L, Santaroni F (2012) Finding strong bridges and strong articulation points in linear time. Theor Comput Sci 447:74–84. https://doi.org/10.1016/j.tcs.2011.11.011. Combinational Algorithms and Applications (COCOA 2010)
Kermack WO, McKendrick AG (1927) A contribution to the mathematical theory of epidemics. Proc R Soc Lond A Math Phys Eng Sci 115(772):700–721. https://doi.org/10.1098/rspa.1927.0118
Khalil E, Dai H, Zhang Y, Dilkina B, Song L (2017) Learning combinatorial optimization algorithms over graphs. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, Garnett R (eds). Advances in Neural Information Processing Systems 30. Curran Associates, Inc., New York. pp 6339–6349
Khan SA, Bölöni L, Turgut D (2015) Bridge protection algorithms – a technique for fault-tolerance in sensor networks. Ad Hoc Netw 24:186–199. https://doi.org/10.1016/j.adhoc.2014.08.016
Kiss IZ, Miller JC, Simon PL (2017) Mathematics of Epidemics on Networks: From Exact to Approximate Models. Springer, New York
Largeron C, Mougel PN, Benyahia O, Zaïane OR (2017) Dancer: dynamic attributed networks with community structure generation. Knowl Inf Syst 53(1):109–151. https://doi.org/10.1007/s10115-017-1028-2
Lawyer G (2015) Understanding the influence of all nodes in a network. Sci Rep 5(8665):1–9. http://dx.doi.org/10.1038/srep08665
Liu J, Gao C (2011) Adaptive immunization in dynamic networks. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 6804 LNAI:673–683. https://doi.org/10.1007/978-3-642-21916-0_71
Mastrandrea R, Fournet J, Barrat A (2015) Contact patterns in a high school: A comparison between data collected using wearable sensors, contact diaries and friendship surveys. PLoS ONE 10:1–26. https://doi.org/10.1371/journal.pone.0136497
Matamalas JT, Arenas A, Gómez S (2018) Effective approach to epidemic containment using link equations in complex networks. Sci Adv 4(12). https://doi.org/10.1126/sciadv.aau4212
Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, Bellemare MG, Graves A, Riedmiller M, Fidjeland AK, Ostrovski G, Petersen S, Beattie C, Sadik A, Antonoglou I, King H, Kumaran D, Wierstra D, Legg S, Hassabis D (2015) Human-level control through deep reinforcement learning. Nature 518:529–533. https://doi.org/10.1038/nature14236
Moore C, Ghoshal G, Newman MEJ (2006) Exact solutions for models of evolving networks with addition and deletion of nodes. Phys Rev E 74:036121
Murata T, Koga H (2018) Extended methods for influence maximization in dynamic networks. Comput Soc Netw 5(8):1–21. https://doi.org/10.1186/s40649-018-0056-8
Paranjape A, Benson AR, Leskovec J (2017) Motifs in temporal networks. In: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. WSDM '17. ACM, New York. pp 601–610. https://doi.org/10.1145/3018661.3018731
Pastor-Satorras R, Castellano C, Van Mieghem P, Vespignani A (2015) Epidemic processes in complex networks. Rev Mod Phys 87:925–979. https://doi.org/10.1103/RevModPhys.87.925
Pastor-Satorras R, Vespignani A (2002) Immunization of complex networks. Phys Rev E 65:036104. https://doi.org/10.1103/PhysRevE.65.036104
Prakash BA, Chakrabarti D, Faloutsos M, Valler N, Faloutsos C (2011) Threshold conditions for arbitrary cascade models on arbitrary networks. In: 2011 IEEE 11th International Conference on Data Mining. IEEE, New York. pp 537–546. https://doi.org/10.1109/ICDM.2011.145

Prakash BA, Tong H, Valler N, Faloutsos M, Faloutsos C (2010) Virus propagation on time-varying networks: Theory and immunization algorithms. In: Balcázar JL, Bonchi F, Gionis A, Sebag M (eds). Machine Learning and Knowledge Discovery in Databases. Springer, Berlin. pp 99–114
Ren X, Gleinig N, Helbing D, Antulov-Fantulin N (2018) Generalized network dismantling. CoRR abs/1801.01357. 1801.01357
Riedmiller M (2005) Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method. In: Proceedings of the 16th European Conference on Machine Learning. ECML'05. Springer, Berlin. pp 317–328
Song C, Hsu W, Lee ML (2015) Node immunization over infectious period. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. CIKM '15. ACM, New York. pp 831–840
Stehlé J, Voirin N, Barrat A, Cattuto C, Isella L, Pinton J, Quaggiotto M, Van den Broeck W, Régis C, Lina B, Vanhems P (2011) High-resolution measurements of face-to-face contact patterns in a primary school. PLoS ONE 6(8):23176. https://doi.org/10.1371/journal.pone.0023176
Sutton RS, Barto AG (1998) Reinforcement Learning: An Introduction. MIT Press, Cambridge
Tong H, Prakash BA, Tsourakakis C, Eliassi-Rad T, Faloutsos C, Chau DH (2010) On the vulnerability of large graphs. In: 2010 IEEE International Conference on Data Mining. IEEE, New York. pp 1091–1096. https://doi.org/10.1109/ICDM.2010.54
Tong G, Wu W, Tang S, Du D (2017) Adaptive influence maximization in dynamic social networks. IEEE/ACM Trans Netw 25(1):112–125. https://doi.org/10.1109/TNET.2016.2563397
Van de Bunt GG, Van Duijn MAJ, Snijders TAB (1999) Friendship networks through time: An actor-oriented dynamic statistical network model. Comput Math Org Theory 5(2):167–192
Van Mieghem P, Stevanović D, Kuipers F, Li C, van de Bovenkamp R, Liu D, Wang H (2011) Decreasing the spectral radius of a graph by link removals. Phys Rev E 84:016101. http://dx.doi.org/10.1103/PhysRevE.84.016101
Vanhems P, Barrat A, Cattuto C, Pinton JF, Khanafer N, Regis C, Kim B-a, Comte B, Voirin N (2013) Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. PLoS ONE 8(9):73970. https://doi.org/10.1371/journal.pone.0073970
Wang B, Chen G, Fu L, Song L, Wang X, Liu X (2016) Drimux: Dynamic rumor influence minimization with user experience in social networks. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16). pp 791–797
Wang B, Chen G, Fu L, Song L, Wang X (2017) Drimux : Dynamic rumor influence minimization with user experience in social networks. IEEE Trans Knowl Data Eng 29(10):2168–2181
Watkins CJCH (1989) Learning from Delayed Rewards. Cambridge University, Cambridge
Wijayanto AW, Murata T (2017) Flow-aware vertex protection strategy on large social networks. In: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017. ASONAM '17. ACM, New York. pp 58–63. https://doi.org/10.1145/3110025.3110033
Wijayanto AW, Murata T (2018) Learning adaptive graph protection strategy on dynamic networks via reinforcement learning. In: 2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI). WI 2018. IEEE, New York. pp 534–539. https://doi.org/10.1109/WI.2018.00-41
Wijayanto AW, Murata T (2018) Pre-emptive spectral graph protection strategies on multiplex social networks. Appl Netw Sci 3(1):5. https://doi.org/10.1007/s41109-018-0061-8
Zhan J, Rafalski T, Stashkevich G, Verenich E (2017) Vaccination allocation in large dynamic networks. J Big Data 4(1):2. https://doi.org/10.1186/s40537-016-0061-4
Zhang Y, Prakash BA (2014) Dava: Distributing vaccines over networks under prior information. In: Proceedings of the 2014 SIAM International Conference on Data Mining. SIAM, Philadelphia. pp 46–54. https://doi.org/10.1137/1.9781611973440.6. https://epubs.siam.org/doi/abs/10.1137/1.9781611973440.6
Zhang Y, Prakash BA (2015) Data-aware vaccine allocation over large networks. ACM Trans Knowl Discov Data 10(2):20–12032
Zhang Y, Ramanathan A, Vullikanti A, Pullum L, Prakash BA (2017) Data-driven immunization. In: 2017 IEEE International Conference on Data Mining (ICDM). IEEE, New York Vol. 00. pp 615–624. https://doi.org/10.1109/ICDM.2017.71
Zhao D, Wang L, Li S, Wang Z, Wang L, Gao B (2014) Immunization of epidemics in multiplex networks. PLoS ONE 9(11):1–5. https://doi.org/10.1371/journal.pone.0112018
Zhuang H, Sun Y, Tang J, Zhang J, Sun X (2013) Influence maximization in dynamic social networks. In: 2013 IEEE 13th International Conference on Data Mining. IEEE, New York. pp 1313–1318. https://doi.org/10.1109/ICDM.2013.145