Active Screening for Recurrent Diseases: A Reinforcement Learning Approach

Han-Ching Ou, Haipeng Chen, Shahin Jabbari and Milind Tambe
Harvard University
Cambridge, Massachusetts
{hou@g.,hpchen@seas.,jabbari@seas.,milind_tambe@}harvard.edu

ABSTRACT

Active screening is a common approach in controlling the spread of recurring infectious diseases such as tuberculosis and influenza. In this approach, health workers periodically select a subset of population for screening. However, given the limited number of health workers, only a small subset of the population can be visited in any given time period. Given the recurrent nature of the disease and rapid spreading, the goal is to minimize the number of infections over a long time horizon. Active screening can be formalized as a sequential combinatorial optimization over the network of people and their connections. The main computational challenges in this formalization arise from i) the combinatorial nature of the problem, ii) the need of sequential planning and iii) the uncertainties in the infectiousness states of the population.

Previous works on active screening fail to scale to large time horizon while fully considering the future effect of current interventions. In this paper, we propose a novel reinforcement learning (RL) approach based on Deep Q-Networks (DQN), with several innovative adaptations that are designed to address the above challenges. First, we use graph convolutional networks (GCNs) to represent the Q-function that exploit the node correlations of the underlying contact network. Second, to avoid solving a combinatorial optimization problem in each time period, we decompose the node set selection as a sub-sequence of decisions, and further design a two-level RL framework that solves the problem in a hierarchical way. Finally, to speed-up the slow convergence of RL which arises from reward sparseness, we incorporate ideas from curriculum learning into our hierarchical RL approach. We evaluate our RL algorithm on several real-world networks. Results show that our RL algorithm can scale up to 10 times the problem size of state-of-the-art (the variant that considers the effect of future interventions but un-scalable) in terms of planning time horizon. Meanwhile, it outperforms state-of-the-art (the variant that scales up but does not consider the effect of future interventions) by up to 33% in solution quality.

KEYWORDS

Reinforcement Learning, Contact Network, Public healthcare

ACM Reference Format:
Han-Ching Ou, Haipeng Chen, Shahin Jabbari and Milind Tambe. 2021. Active Screening for Recurrent Diseases: A Reinforcement Learning Approach. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 12 pages.

Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.

1 INTRODUCTION

Active screening (or contact tracing) aims at selecting a subset of nodes in a social network for screening, so as to prevent the spread of transmissive diseases. It is a system used by health workers globally to stop infectious diseases, including influenza, sexually transmitted diseases, and tuberculosis. When an individual is tested positive, they are marked as infected. The health worker, or contact tracer, records who else have been exposed and marks them as contacts or potentially infected individuals. The potentially infected individuals might not voluntarily seek treatment and testing. In case any of such individuals are infected, they can spread the disease further. Active screening aims to target these individuals and slow down the spread of the disease [13, 53].

There are many challenges in implementing active screening [34]. First, not every individual in a social network can be screened due to limited resources (in this case contact tracers or amount of tests available). Therefore, for each screening round, we need to optimally select a subset of nodes which is a *combinatorial optimization* problem. Second, the health states of the many individuals in the network are unknown (sans people who get tested actively or passively). Finally, the above challenges are significantly amplified when *sequential planning* is involved as we need to account for the future effects of current screening actions.

Most of the computer science literature on infectious disease focuses on conditions leading to disease eradication, which often relies on the assumptions such as complete knowledge of individual health states [13, 14]. Unfortunately, such knowledge of health states is not always available, making these approaches difficult to apply in many real-world scenarios. Recently Ou et al. [34] address the challenges of uncertain states and limited budget. However, their approaches either do not scale with the planning time horizon or fail to fully account for future actions. See Section 2 for more details and Section 5 for comparisons with our approach.

Due to the superior performance of RL approaches in solving long term planning problems [33, 45, 46], in this paper we propose a novel RL approach that builds upon a powerful variant of RL called DQN [32]. We first formulate the multi-round active screening problem as a Markov Decision Process (MDP), where the state is a vector representing the probability of each node in the network being infected, and the action is to select which subset of nodes to actively screen. Due to the extremely high-dimensional state and action spaces, vanilla DQN algorithms cannot be directly applied to solve our problem efficiently. We therefore design several innovative adaptations over vanilla DQN that fully exploit the problem structure of multi-round active screening. First, we show that the node features in the underlying contact network are inter-correlated. To efficiently capture the intrinsic correlations between different

nodes, we use GCNs as the function approximator to represent the Q-function. Second, because in each time period we need to select a subset of nodes to actively screen, this leaves vanilla DQN unscalable as it needs to solve a combinatorial optimization problem in the action selection procedure. To avoid this we decompose the node set selection problem in each time period as a sub-sequence of decisions, and then design a novel two-level RL framework that solves the problem in a hierarchical manner. It has two types of agents. The primary agent works at the main sequence level and interacts with the environment, while multiple secondary agents work at the sub-sequence level and are responsible for generating actions sequentially within each time period. Last, we find that the reward signals for the secondary agents are sparse. To speed up the slow convergence of secondary agents' policies that arises from the sparseness of rewards, we incorporate ideas from curriculum learning into our algorithm. Intuitively, the algorithm warm-starts at the beginning of training with a simpler task, which has limited action choice and true state information. As the training goes on, the algorithm gradually increases task difficulty by providing uncertain state information and more action choice until the problem becomes the same as the original active screening problem.

Our main contributions are summarized as follows. (i) We are the first to formulate the multi-round active screening problem for recurrent diseases as a Markov Decision Process (MDP). (ii) To solve the formulated MDP, we propose a novel solution algorithm on the basis of DQN, with several innovative adaptations that fully exploit the problem structure of the formulated MDP. (iii) We conduct extensive experiments on various real-world networks with distinct network properties to evaluate the effectiveness of our proposed approach. The empirical results show that our approach can scale up to 10 times the problem size of state-of-the-art (the variant that considers the effect of future interventions but is not scalable) in terms of planning time horizon. Meanwhile, it outperforms state-of-the-art (the variant that scales up but does not consider the effect of future interventions) by up to 33% in terms of the total number of healthy people. The robustness analysis shows that it works better than baselines even with network structure uncertainty. Interestingly, the policy analysis results show that compared with the baselines, our approach does not rely on node structural importance (e.g., degree and betweenness), and thus is fairer in the sense that it tends to spread the screening across different nodes.

2 RELATED WORK

Due to the close relationship between the spread of disease and the structure of contact networks, network epidemiology models have gained much more attention compared to homogeneous models [15, 30]. Numerous papers have studied the properties of these graph-based models. For example, epidemic threshold of the recurrent disease is a particular vibrant sub-discipline for recurrent disease [6, 38, 39, 56]. In these studies, the steady-state of the whole population is analyzed, including sufficient conditions for disease eradication. A wide range of assumptions are used in these models, spanning from scale-free static graphs to random or even dynamic networks. However, none of them consider human interventions.

Vaccination problem is another topic of great importance in the network epidemiology. For non-recurrent disease, when a vaccine exists, strategies should be developed to allocate the vaccine in the most efficient way. Due to the economic cost of vaccination and possible side effects, one may want to keep the number of vaccinated individuals small. These methods assume the intervention offers permanent immunity and thus focus on a one-shot action instead of multiple rounds of actions [2, 18, 41, 42, 49, 54, 59].

For recurrent diseases, temporary quarantines, tracking or treatment are often considered as alternatives in the absence of permanent cures like vaccines. Furthermore, these interventions are required to be performed over several rounds given the recurrent nature of the disease. The multi-round intervention for network epidemiology is first studied from a theoretical prospective [10, 11, 44]. They prove that the disease could be eradicated for a certain budget threshold in each round under the assumption that the infectiousness states of all the individuals in the network are perfectly observable. In reality, such a perfect observation is often not available. Hoffmann and Caramanis [17] analyze the impact of state uncertainty, indicating even a small amount of uncertainty on state will have a large impact on the eradication time and the budget needed. Ou et al. [34] study the multi-round intervention with unknown states in the context of active screening or contact tracing. They prove the NP-hardness of the problem and provide a gradient based algorithm with two variants. The first variant takes future actions into account but scales poorly with the time horizon (number of rounds) and graph size. The second variant, which does not have the scalability issues of the first variant, does not account for future actions. However, for rapidly spreading diseases, planning for a long-term time horizon is an essential and important task. This is even more important for diseases that are hard to eliminate [31]. Since Ou et al. [34] is the closest work to us, we compare our results with them in details in Section 5.

RL has attracted a lot of interest from researchers in the machine learning and artificial intelligence communities [33, 45, 46]. It is an experiment-driven and mathematical framework that trains an agent through trial and error [22, 50, 51]. With the rise of deep learning, researchers further overcome the computational limitations of traditional RL by utilizing the representation power of deep neural networks [1, 32], such as recurrent neural networks (RNNs) [58], convolutional neural networks (CNNs) [28] and graph convolutional neural networks (GCNs) [27]. The early attempt of our work can be found in [35] where standard reinforcement learning is applied yet unable to surpass Ou et al. [34]. In this work, a novel hierarchical reinforcement learning framework that utilize curriculum learning is proposed. Such approach significantly enhance the performance not only compared to [35] but also the previous state of the art Ou et al. [34].

Active screening in each time period is a combinatorial optimization problem, an important branch of optimization with numerous practical applications [16]. They are usually NP-hard and thus polynomial-time heuristic algorithms for finding approximate solutions are often used in practice [21]. Recent advances use deep neural networks to learn heuristics for *one time period* graph combinatorial problems [4, 23, 26]. They have shown promising results by combining RL with different graph embedding methods [7, 27]. For multi-round problems, Song et al. [48] use a weight sharing technique that addresses problems with relatively small selection

budget. To the best of our knowledge, no previous work in this thread of study has been applied to tackle active screening.

3 PROBLEM FORMULATION

In this work, we focus on a sequential decision making problem with a large time horizon, where in each round (time step) we aim to optimally select which individuals in a social network to actively screen, so that the expected number of un-infected individuals over the time horizon is maximized.

3.1 Problem Background

The environment we consider is based on the well-known network (Susceptible-Infected-Susceptible) SIS model. SIS models capture the dynamics of recurrent diseases, where permanent immunity is not possible. We adopt a discrete-time SIS model for modeling the disease dynamics propagating on a given graph[1], where each node represents an individual, and each edge indicates the link between individuals in which disease can spread.

Disease model: Given a graph $G = (V, E)$, each node $v \in V$ in the discrete-time SIS model can be in either of the two states, susceptible (S) or infected (I). In each time step (or round) t, similar to the independent cascade model [25], each node in I state may infect its neighboring nodes that are in S state with a fixed probability β. Each infected node may also be cured and become susceptible again with a fixed probability γ. This represents the probability of a node going for a passive screening, which means that the case is discovered while the individual voluntarily visits a health facility. We denote the true infectiousness state of a given node v at time t as $\mathbf{x}_t^v$. This true state is assumed to be unknown and the contact network is assumed to be known to the health workers.[2]

Intervention model: We wish to automatically train an active screening agent to learn a policy that controls the spread of disease. The intervention we consider involves multi-rounds of active screening over a time horizon of T days. In each time step $t = 0, \ldots, T$ we have a budget of k health workers that can visit and actively screen a set of nodes denoted as $\mathbf{a}_t \subseteq V$. In real world, there are some people that visit health facilities voluntarily without the active intervention from the health workers. To model such observations, we assume the nodes which turn from I to S state are observed by the health workers at the start of each round. We denote such set as $\mathbf{o}_t \subseteq V$. After knowing this information, the agent will select the set of nodes $\mathbf{a}_t$ to screen under the budget constraint k. After being screened, the infected patients recover back to the susceptible state, while individuals in the susceptible state remain susceptible. The process repeats until the given time horizon ends. The goal is to maximize the accumulated number of un-infected patients across time, which can be written as $R = \Sigma_{t=0}^{T} \Sigma_{v \in V} \mathbb{1}_{\mathbf{x}_t^v = S}$ with $\mathbb{1}$ being the indicator function. The multi-round active screening is essentially a sequential decision making problem with combinatorial actions in each time step.

3.2 Markov Decision Process Formulation

We start by formulating the active screening problem as a MDP.

[1]We use network and graph interchangeably to denote social networks.

[2]We will also show empirically in Sec. 5 how our algorithm works when the contact network is unknown.

States: The hidden state of our problem is the combinatorial health state of each individual node which is partially observable. To represent the observation uncertainty in the current state, we follow Ou et al. [34] by defining a *belief state* b_v for every node v, which can be interpreted as the approximate probabilities of each node being infected. See Appendix A for the details.

Actions: Given the current state, the agent can choose any subset of nodes $C \subseteq V$, $|C| \leq k$ to screen. The size of the action space is $\binom{V \setminus \mathbf{o}_t}{k}$ at each round, where $\mathbf{o}_t$ is the set of nodes that are passively screened (so there is no advantage in actively screening them).

Rewards: The objective of the active screening problem is to maximize the accumulated number of susceptible nodes. It is natural to consider the step wise reward signal as number of susceptible nodes after the active screening, denoted as $r_t = \Sigma_{v \in V} \mathbb{1}_{\mathbf{x}_t^v = S}$. Since every infected individual has a fixed probability γ of being observed by the agent, the step wise reward can be easily estimated.

Transitions: In belief of the health states, screened or observed nodes ($v \in \mathbf{o}_t \cup \mathbf{a}_t$) are updated by their ground truth values and the remaining nodes are updated by inferring their posterior probabilities. The key to state transition is the update of belief state, which is defined following [34]. We refer to Appendix A for the detailed description of belief state update. We also refer to Appendix B for a summary of notations.

4 METHODOLOGY

Despite a well defined MDP, it is extremely challenging to solve it due to various reasons. One of the main challenges is that both state space and action space we are facing are high-dimensional. For the state space, even when the true state is available, there are a total of $2^{|V|}$ possible states. When uncertainty is involved, the state values are continuous and therefore the number of states is infinitely large. Furthermore, the states of individual nodes are not independent from each other, but are correlated due to potential contacts from the network. For the action space, in each time period we need to choose a combination (subset) of nodes from the entire network. For a reasonably large network, this combinatorial action space grows exponentially with respect to the screening budget k, and becomes intractable as k typically scales as the graph size grows. In the following we show how these challenges are handled in our approach. A summary of notations related to the algorithm is included in Appendix B.

4.1 Basics of DQN

Due to the superior performance of RL algorithms in solving large scale MDPs [33, 45, 46], we adopt RL as the basis of our solution. More specifically, the backbone of our approach is a hierarchical RL algorithm based on DQN. In this sub-section, we first introduce the basics of RL and DQN, following which we then describe several ideas that further adapt DQN to our formulated problem. We need to emphasize that we do not claim novelty in each of the adaptions, but instead the novelty lies in the innovative way of combining the ideas into solving the particular problem of interest.

RL is a learning framework where agents learn to perform actions in an environment so as to maximize a certain objective. The two underlying components of RL are the environment, which is defined as the MDP in this paper, and the agent, which represents the

learning algorithm. At each time step t, the agent takes an *action* based on its *policy* $\pi(\mathbf{a}_t|\mathbf{s}_t)$, where $\mathbf{s}_t$ and $\mathbf{a}_t$ are respectively the state and action of the MDP defined above. The agent then interacts with the environment with the selected action and the environment returns a reward r_t for that action as well as the state $\mathbf{s}_{t+1}$ of the next time step. Q-learning [57] is a value-based RL approach that is based on the notion of Q-function (i.e., state-action value function). The Q-function measures the expectation of accumulated rewards of an action $\mathbf{a}_t$ given state $\mathbf{s}_t$. In the training phase of Q-learning, the policy usually exploits the action with the highest Q-value with a high probability $1-\varepsilon$, and explores random actions with a small probability ε. The Q-function is typically estimated using the Bellman equation: $Q^{i+1}(\mathbf{s}_t, \mathbf{a}_t) = r_t + \alpha \max_{\mathbf{a}_{t+1}} Q^i(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})$, where i indicates the training iteration and α is the discount factor. DQN [32] improves Q-learning by representing it using deep neural networks, together with other techniques like experience replay over a number of episodes ($\mathcal{E}$), which basically stores the historical training trajectories in a "replay buffer" and updates the Q-function by minimizing the loss function $(y - Q(s,a))^2$ with batch data from the replay buffer using gradient descent algorithms. Here y is the "target" which is estimated using the above Bellman equation (a technique usually called Temporal Difference learning), and $Q(s,a)$ is directly obtained by feeding s and a to the Q-function.

4.2 GCN-based Function Approximator

With the deep neural networks based function approximator, DQN addresses the exponentially large and continuous state space in our formulated MDP. However, one shortcoming of such a function approximator is that it does not capture the intrinsic correlations between node features. Intuitively, the infectious statuses of linked nodes in a social network are inter-dependent.

Graph convolutional neural networks (GCNs) [27] embed the graph structure itself into its network directly, and thus have superior performance on graph type inputs. Each layer of GCN is given by $\mathbf{z}^{l+1} = \sigma(\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}\mathbf{z}^l\mathbf{W})$, in which $\mathbf{z}^l$ is the input of l-th layer, $\mathbf{D}$ is the diagonal node-degree matrix that normalizes the adjacency matrix $\mathbf{A}$, $\mathbf{W}$ is the trainable weight matrix and $\sigma(\cdot)$ is the activation function. The convolutional layers in GCNs can facilitate the nature of message passing and automatically aggregate the information from neighboring nodes. Such message passing is similar to the infection spread in our epidemic model. The advantage of using GCNs is that we do not need the hand-crafted graph features to represent the information about the graph structure such as the node degrees or eigenvalue of adjacency matrix [29]. Inspired by recent advances that combine the power of RL and GCNs-based deep function approximators [24, 26, 40], we use GCNs in this paper to represent the Q-function.

Our adaptation of the GCNs takes the belief state as input. The action and observation can be naturally encoded in the belief state as we can update the corresponding elements in the belief state vector to their true state. Thus we do not need to encode them as additional features.We thus combine the observation with the graph structure that is represented as the adjacency matrix $\mathbf{A}$ to our state representation in a very structured way. The GCNs learn the underlying graph embedding and automatically form the representation and output the Q-value estimation for our RL agent.

4.3 Sequence of Sequence Framework

In addition to the challenges that arise from the high-dimensional and graph-structured state space, as described previously, another challenge is the combinatorial action space in each time step of screening. To address this challenge, we propose a hierarchical RL approach by re-formulating each time step in the original MDP as a sub-sequence of decisions by itself. We call this framework sequence of sequence (SOS). We refer to the original multi-rounds of active screening as *time* sequence. Correspondingly, we refer to the sub-sequence problem of selecting k nodes in each round as the *budget* sequence. In each of the budget sequence, we are solving a separate sequential decision making problem with a final reward R_t, a finite time horizon of k, and an action space $V \setminus \mathbf{o}_t$ whose size is equal to the network size.

The SOS framework allows for a tractable action space which can be used by an RL algorithm. However, two additional issues arise in such conversion of the action space. First, although the states and actions are well represented by GCNs, the algorithm does not take into account the remaining budget at the budget sequence. In fact, the policies should be very different when there is plenty of budget left versus little budget left. Intuitively, this is because the actions taken when there is more budget left should consider more about its future effect, while actions taken when there is less budget left tend to be more myopic. Therefore, a single-agent framework in the budget sequence, which treats all states equally for different remaining budget, usually does not work well. Second, by introducing the SOS framework, only the final step in the k budget-steps for the budget sequence gets a reward signal. The sparseness of reward is known to slow the convergence of RL [19]. Therefore distributing the reward R_t to each action in the budget sequence for proper reward signaling is a non-trivial task.

4.4 Primary and Secondary Agents

Inspired by hierarchical RL [8, 9, 37, 52], we propose a multi-agent RL approach with a two-level structure that manages the time and budget sequences in a hierarchical way. The overall flow of the two-level structure is depicted in Fig. 1. The idea is to capture the remaining budget information by having k secondary agents, where each secondary agent maintains a policy for a different remaining budget value.[3] The primary agent, as shown in Algorithm 1, manages the time period reward signal in a given time step of the time sequence. The reward signal acts as the secondary agents' total reward in the budget sequence and is distributed over the k secondary agents. In the following, we use superscripts I and II to distinguish concepts that correspond to the primary and secondary agents.

Primary Agent In the time sequence, the primary agent works as follows. First, it receives the combinatorial action and the secondary agents' memories that store the trajectories from the secondary agents (line 6). Passing this action to the environment, it receives the observation and reward (line 7). It then handles the new state representation of the next time step (lines 9 and 10). Finally, it updates both primary and secondary agents' memories ($\mathcal{M}^I$ and $\mathcal{M}^{II}$) (lines 11 and 12). Each element of $\mathcal{M}^I$ and $\mathcal{M}^{II}$ is basically

[3] An alternative is to train a single agent and encode the remaining budget information directly as part of the state. However, we show via ablation study in Appendix C that this leads to sub-optimal solution quality.

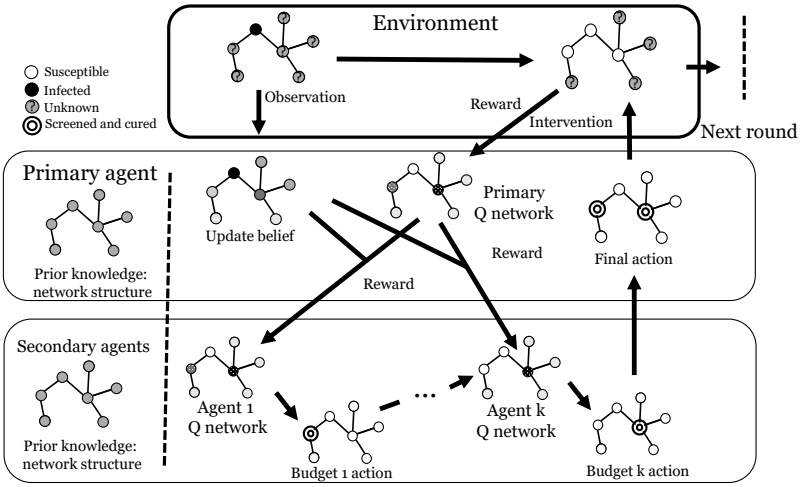


Figure 1: The overall flow of one round decision making in a budget sequence using our two-level RL structure. The top row depicts the environment, the second row is the workflow of the primary agent, and the third row is the workflow of the k secondary agents. The primary agent starts by observing the initial state of the environment. It then updates the belief of the state. The belief state is passed down to the first secondary agent, which then evaluates the value of each feasible node action using its own Q network and decides which node to select. The successive secondary agents work sequentially until the k-th secondary agent, i.e., when budget k is spent. The selected actions are collected to get the final action set $\mathbf{a}_t^I$ which is uploaded to the primary agent. The primary agent then interacts with the environment using this action and gets reward signals.

a tuple of state, action, next state and rewards that will be used to train the primary and secondary agents' Q-functions. Note that the state of the secondary agents is slightly different from that of the primary agent, which will be explained in the next paragraph. These memories are used to update the GCN-based Q-functions by minimizing the loss function $(y - Q(s,a))^2$ through gradient descent (lines 16 and 17), where y is the target. At each time step t, the target of each agent is given by:

$$y^I = r_t^I + \alpha Q^I(\mathbf{s}_{t+1}^I, \mathbf{a}_{t+1}^I), \tag{1}$$

$$y_i^{II} = r_i^{II} + \alpha Q_{i+1}^{II}(\mathbf{s}_i^{II}, a_{i+1}^{II}) \text{ for } i = 1, \ldots, k-1, \tag{2}$$

$$y_k^{II} = r_k^{II} + \alpha Q^I(\mathbf{s}_{t+1}^I, a_{t+1}^I). \tag{3}$$

Note that in lines 5 and 8, the belief state and reward are obtained using the idea of curriculum learning that mitigates the reward sparseness issue for the secondary agents. We will elaborate this idea in the next subsection.

Secondary Agents As for the budget sequence, instead of training a single agent and performing a batch selection of nodes, we train k secondary agents to handle each budget sequentially. As described above, the purpose of doing so is to differentiate secondary agents who know there is plenty of budget left and those who know there is little budget left, so they could learn different policies. The state $\mathbf{s}_i^{II}$ of a secondary agent i is obtained by encoding the primary agent's action (i.e., the set of nodes selected so far) taken upon the state of the previous budget step. The action a_i^{II} for each secondary agent i is to choose one node to add to the primary agent's action set $\mathbf{a}^I$. $\mathbf{a}^I$ is initialized as an empty set (in line 1) and will be updated by appending actions from each secondary agent. In the for loop that represents a budget sequence (from line 3 to line 8), each

Algorithm 1 Primary Agent

1: **for** *episodes* $= 1, ..., \mathcal{E}$ **do**
2: Initialize and acquire initial belief ($\mathbf{b}_0$) and observation ($\mathbf{o}_0$)
3: $\mathbf{s}_0^I \leftarrow$ *Graph Embeding*$(\mathbf{A}, \mathbf{b}_0)$
4: **for** $t \in 0, ..., T$ **do**
5: $\tilde{\mathbf{b}}_t \leftarrow$ *Curriculum Belief Transform*$(\tau, \mathbf{b}_t)$
6: $\mathbf{a}_t^I, m^{II} \leftarrow$ *Secondary Agent*$(\tilde{\mathbf{b}}_t, Q^I)$
7: $\mathbf{o}_{t+1}, r_t^I \leftarrow$ *Environment*$(\mathbf{a}_t^I)$
8: $\tilde{r}_t^I \leftarrow$ *Curriculum Reward Transform*(τ, r_t^I)
9: $\mathbf{b}_{t+1} \leftarrow$ *Belief Update*$(\mathbf{b}_t, \mathbf{o}_t, \mathbf{a}_t^I)$
10: $\mathbf{s}_{t+1}^I \leftarrow$ *Graph Embeding*$(\mathbf{A}, \mathbf{b}_{t+1})$
11: $\mathcal{M}^I \leftarrow \mathcal{M}^I \cup \{(\mathbf{s}_t^I, \mathbf{a}_t^I, \tilde{r}_t^I, \mathbf{s}_{t+1}^I)\}$
12: $\mathcal{M}^{II} \leftarrow \mathcal{M}^{II} \cup m^{II}$
13: **end for**
14: Decrease τ
15: **end for**
16: *Fit* Q^I *with regressor net using* $\mathcal{M}^I$
17: *Fit* $Q_0^{II} ... Q_{k-1}^{II}$ *with regressor nets using corresponding* $\mathcal{M}^{II}$

secondary agent i will select the action a_i^{II} that maximizes its Q-function $Q_i^{II}(s_i^{II}, a_i^{II})$ and add it to the primary agent's action set $\mathbf{a}^I$ (lines 4 and 5). After that, it receives the reward in line 6, which is obtained from the primary agent (as a proxy) in a temporal difference learning manner. Next, the secondary agent will encode the primary agent's action $\mathbf{a}^I$ (i.e., the set of nodes selected so far) into its current state $\mathbf{s}_i^{II}$, which is used as next state $\mathbf{s}_{i+1}^{II}$ and pass this information to the next secondary agent (line 7). Finally, it stores the above information as memory, so it can be used later to

update the Q-functions in Equations (1)- (3). For extremely large graphs, we reduce the memory and computation time by assigning fewer than k secondary agents, where each secondary agent is responsible for a portion of the budget instead. For example, for a budget of 20, if we assign 10 secondary agents, each secondary agent needs to select $20/10 = 2$ nodes at a time.

Algorithm 2 Secondary Agents

1: $\mathbf{a}^I, m_1^{II} ... m_k^{II} \leftarrow \emptyset$
2: $\mathbf{s}_0^{II} \leftarrow Encoding(\mathbf{s}^I, \mathbf{a}^I)$
3: **for** $i \in 1, ..., k$ **do**
4: $\quad a_i^{II} \leftarrow \arg\max Q_i^{II}(s_i^{II}, a_i^{II})$
5: $\quad \mathbf{a}^I \leftarrow \mathbf{a}^I \cup a_i^{II}$
6: $\quad \mathbf{r}_i^{II} \leftarrow Q^I(\mathbf{s}^I, \mathbf{a}^I) - Q^I(\mathbf{s}^I, \mathbf{a}^I \setminus a_i^{II})$
7: $\quad \mathbf{s}_{i+1}^{II} \leftarrow Encoding(\mathbf{s}^I, \mathbf{a}^I)$
8: $\quad m_i^{II} \leftarrow m_i^{II} \cup \{(\mathbf{s}_i^{II}, a_i^{II}, \mathbf{r}_i^{II}, \mathbf{s}_{i+1}^{II})\}$
9: **end for**
10: **return** $\mathbf{a}^I, m^{II}$

4.5 Curriculum Learning

As discussed earlier, a major issue with the two-level framework is the sparseness of rewards for the secondary agents. Inspired by curriculum learning [5], we address this by incrementally increasing the complexity of the learning tasks for the secondary agents. This is called *Curriculum Transformation* in lines 5 and 8 in Algorithm 1 and is described as the following equations:

$$\tilde{\mathbf{b}}_t = \tau \mathbf{x}_t + (1-\tau)\mathbf{b}_t, \tag{4}$$

$$\tilde{r}_t^I = \tau \bar{r}_t + (1-\tau) r_t^I, \tag{5}$$

where $\tilde{\mathbf{b}}_t$ and $\tilde{\mathbf{r}}_t^I$ are respectively the belief state and reward of the primary agent (used to update the target values for both the primary and secondary agents) after curriculum transformation. τ is an auxiliary coefficient that gradually decreases from 1 to 0 in the first few epochs of training. It adjusts task difficulties from a relatively easier problem ($\tau = 1$) to the original problem ($\tau = 0$). At the early stage of training ($\tau = 1$), we warm up the learning by feeding the algorithm with the true state information $\mathbf{x}_t$ (Eq. (4)). Moreover (Eq. (5)), we set the reward to be $\bar{r}_t = \sum_{v \in \mathbf{a}_t^I} \mathbb{1}_{b_t^v = S}$, which means the total number of infected nodes in the action set $\mathbf{a}_t^I$, instead of in the total number of susceptible nodes S. In this way, the algorithm learns to greedily cure nodes that are infected. As the training continues, decreasing the auxiliary coefficient τ takes two effects. First, it gradually removes the true state information. It is worth noting that the true state is only used in warming up the training, and is not used during testing. Second, it shifts the reward from being constrained in the set of infected nodes to the true reward, and explores potentially more optimal actions outside the set of infected nodes. When τ is 0, the belief and reward become identical to the original problem ($\tilde{\mathbf{b}}_t = \mathbf{b}_t$ and $\tilde{r}_t^I = r_t^I$).

5 EXPERIMENTS

Datasets We evaluate the effectiveness of our proposed approach on different real-world contact networks. These datasets are open-sourced and are also used in Ou et al. [34] which is the closest related work to ours. The nodes in each of these networks represent human individuals, the edges represent different forms of contact. We next provide a brief description for each of the datasets. (i) **Hospital** [55]: A contact network collected by wearable devices in a university hospital. A link is built if close range interactions with long enough duration is detected. (ii) **India** [3]: A network collected from one of the villages in India by households surveying. The edges represent the real world social contact. (iii) **Face-to-face** [20]: A network describing face-to-face behavior during the exhibition "INFECTIOUS: STAY AWAY" in 2009 at the Science Gallery in Dublin. The network simulates the close contact of individuals that influenza might spread through. (iv) **Flu** [43]: A network of close proximity interactions in an American high school sampled in an ordinary day. The network is collected using wireless sensor network technology. An edge is built for close physical proximity. (v) **Irvine** [36]: A network collected from an online student community in UC Irvine to analyze spread of information or rumour using epidemic models. The edges represent the communication over online messages. These networks display diversity as captured by parameters such as network size $|V|$, spectral radius $1/\lambda_A^*$, average degree d, average shortest path length ρ_L and assortativity ρ_D as depicted in Table 2.

Table 1: Properties of the contact network datasets.

Network	$\lvert V\rvert$	$\frac{1}{\lambda_A^*}$	d	ρ_L	ρ_D
Hospital [55]	75	0.027	30.37	1.60	-0.18
India [3]	202	0.095	6.85	3.11	0.02
Face-to-face [20]	410	0.042	13.49	3.63	0.23
Flu [43]	788	0.003	300.23	1.62	0.05
Irvine [36]	1899	0.021	14.57	3.06	-0.18

Experimental setting In all experiments, we fix the passive screening rate γ to 0.05. Due the the high diversity of the networks, it is difficult to find one fixed transmission rate β that is suitable for every network. A fixed transmission rate is either too high for the dense networks so that the whole network will become infected no matter what policy is deployed, or too low for the sparse networks so that the disease will be eradicated without any intervention. We thus adjust the transmission rate according to the density of the network. Specifically, we adjust β based on the spectral radius $1/\lambda_A^*$, also known as the epidemic threshold, where λ_A^* is the largest eigenvalue of the network adjacency matrix. If there are no additional interventions, the disease will not be eradicated eventually if and only if $\beta > \gamma/\lambda_A^*$ [56]. We set $\beta = 10\gamma/\lambda_A^*$, which is 10 times the value corresponding to the epidemic threshold. We further set $k = 0.1|V|$ for the active screening budget in each time period. Finally, we set the total time horizon to $T = 100$. All the results presented are averaged over 30 trials. For our RL approach, we set the future discount factor α to 0.98, exploration probability in the epsilon greedy approach is 0.1 and the learning rate is 0.005. We trained the RL agents for 100 episodes for 100 iterations of refits.

Table 2: Average improvement of different algorithms. *Full-REMEDY* does not scale to $T = 100$. Thus its results are not included. The numbers in the brackets are improvements over the best alternative *Fast-REMEDY*.

Network	Reduction in infections compared with no intervention				
	Eigenvalue	Max-Degree	Random	Fast-REMEDY	**RL**
Hospital	1019±99	1015±131	2344±136	3837±340	**4196±416 (9.4%)**
India	2668±258	3115±292	6388±259	10033±518	**11270±501 (12.3%)**
Face-to-face	4225±211	4705±219	8948±173	9919±499	**13283±301 (33.9%)**
Flu	7706±190	7725±203	9636±163	10298±460	**11743±503 (14.0%)**
Irvine	48490±298	49163±378	42277±270	53159±673	**65128±781 (22.5%)**

The memory capacity of the relay buffer is set to 5000 tuples for each agent. We used the sigmoid activation function. There are four layers of sizes 8,16,8 and 32. For all graphs except the largest one, the training finishes within 3 hours on a laptop with 6 cores, 2.60 GHz intel CPU, and 16 GB RAM. For the largest graph, *Irvine* network, it takes about one day to finish on the same laptop and is significantly shortened after using an HPC.[4]

Baselines We simply call our approach RL. The baselines we are testing against are (i) *Eigenvalue* (greedily choosing nodes that decrease the largest eigenvalue of the remaining sub-graph after removal until the budget is exhausted), (ii) *MaxDegree* (choosing k nodes with the largest degrees), (iii) *Random* (randomly selecting nodes), (iv) *Full-REMEDY* (the algorithm in [34] that is un-scalable to large time horizons) and (v) *Fast-REMEDY* (the scalable version of *Full-REMEDY* that does not account for the future effect of actions).

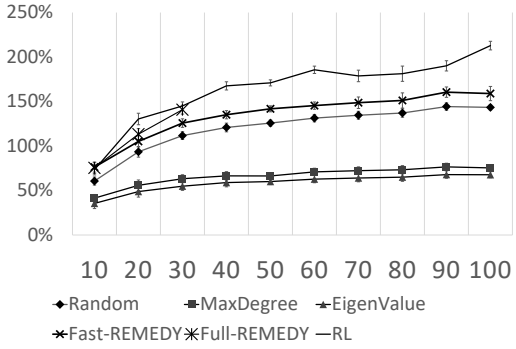


Figure 2: The performance of each algorithm for different time horizons in the Face-to-face network. The x-axis is the time horizon and the y-axis is the improvement of solution quality over no intervention.

5.1 Solution Quality

Table 2 shows the increase in average reward compared with no intervention. We can see that our approach outperforms the state-of-art (i.e., *Fast-REMEDY*) by a margin of 9 – 33%. There are a few observations worth noting. First, we are implementing the same baselines on the same datasets with significantly larger time horizon compared with Ou et al. [34]. The ranking of these baselines is consistent with Ou et al. [34] that are run over a shorter time horizon of 10. Furthermore, results on *Hospital* and *Flu* show a more significant difference as we adjust the transmission rate according to the graph density. Second, *Eigenvalue* and *MaxDegree* baselines perform similarly to each other when we increase the time horizon where as in Ou et al. [34], *MaxDegree* clearly outperforms *Eigenvalue* when the time horizon is short. This is expected as the *Eigenvalue* baseline is aimed for long term disease eradication by increasing the epidemic threshold and thus preforms better in the long term. Finally, in *Face-to-face* and *Irvine* networks, our approach performs significantly better compared with the best baseline *Fast-REMEDY*. Interestingly, these are also the networks where *Full-REMEDY* outperforms *Fast-REMEDY* in Ou et al. [34]. In these networks, the algorithms can benefit more by looking ahead compared with other networks.

[4]Codes and Data can be found in https://bit.ly/32wtsEk

5.2 Scalability Against Time Horizon

To study how well our approach scales against planning time horizon compared with baselines, we conduct experiments on various time horizons ranging from 10 – 100. Figure 2 shows the performance on the face-to-face network for each algorithm on the y-axis when varying the time horizon on the x-axis. *Full-REMEDY* does not scale over time horizons longer than 30 even on a high performance computer and our approach is in par or better with its performance in these short time horizons. We pick face-to-face network as an example and similar trends can be observed for all the networks in Appendix D. Particularly, in the largest network Irvine, our approach scales 10 times of that in *Full-REMEDY*, meanwhile with better solution quality compared with *Fast-REMEDY*.

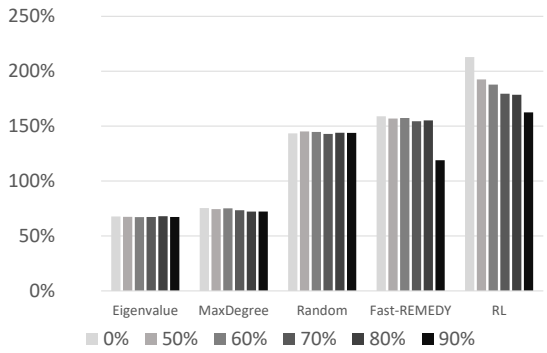


Figure 3: The performance of each baseline for different edge removal fractions. The x-axis indicates the improvement over no intervention.

5.3 Robustness Against Structure Uncertainty

Although we assume perfect knowledge of graph structure (sans the infectious state of the individuals), one of the main obstacles in implementing active screening in practice is the lack of this perfect knowledge. To evaluate how robust different methods are against structural properties, we design and run the methods on two models for structural uncertainty. In the first one, we assume a constant fraction of the edges are unobserved where this fraction is a parameter. In the second one, we assume all the edges adjacent to

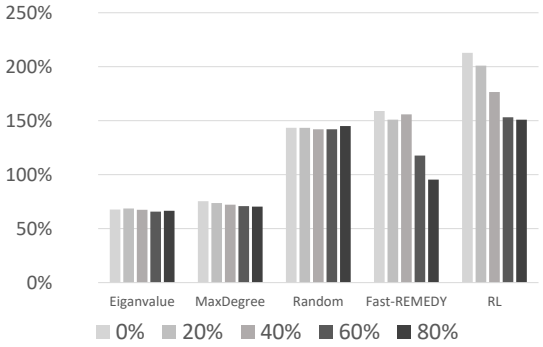


Figure 4: The performance of each baseline for different node removal fractions. The x-axis indicates the improvement over no intervention.

a constant fraction of nodes in the graph are unobserved. In both of these models, we train our RL policy on the observed network and measure the performance of the learned policy on the actual network. We call these models edge and node removal, respectively.

We point out that in the first model some of the properties of the original graph like the nodes with maximum degree are preserved [12]. In the second model, many of the properties of the original network like centrality are likely to be changed [47].

Although our approach is the only learning algorithm that can benefit from different training sub-graphs, to make fair comparison, we train our RL policy on a single sub-graph. This is corresponding to the real world scenario where partial contact information is missing without being noticed. The results are summarized in Figures 3 and 4, respectively. Although the performance of our approach decays as the uncertainty increases, it still outperforms all the other baselines. Again, we show the result of face-to-face network as an example due to space limit. The results of the other networks have similar trends and can be found in Appendix D.

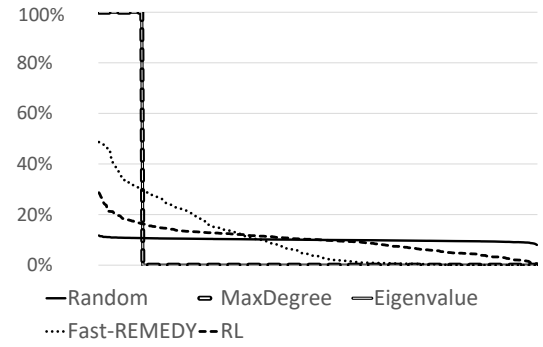
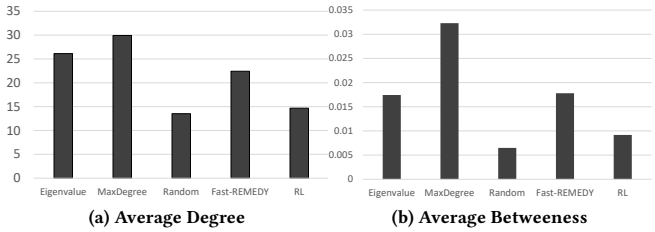


Figure 5: The node picking frequency distribution for each algorithm, sorted from high to low.

5.4 Policy Analysis

To gain insight on the patterns of how different approaches select nodes, we study the frequency in which each of the nodes is selected. The results are summarized in Figure 5 for the Face-to-face network. Similarly, results for other networks are in Appendix D. Each point in x-axis represents a node, sorted by the frequency of being picked by the corresponding algorithm. The y-axis represents the frequency a certain node is picked by the algorithm. We sort all the algorithm's node picking frequency in order to show their distribution. In this figure, *Random* is the fairest algorithm as it picks each node with equal frequency, whereas *MaxDegree* and *Eigenvalue* always pick the same set of nodes as we have a static network. *Fast-REMEDY* selects the most frequently picked nodes half of the times while almost never picks 40% of the nodes. Our RL approach does not pick the structurally important nodes as often as *Fast-REMEDY* does, which is shown in figure 6. It is less structure dependent and tends to select a larger variety of nodes. By taking future actions into account, it depends more on the observation information and has a surprising side effect that outputs a fairer policy that gives more nodes chances to be screened.



(a) Average Degree **(b) Average Betweeness**

Figure 6: Average degree & betweeness of the nodes picked.

6 CONCLUSION

We make the first attempt at addressing the multi-round active screening problem using reinforcement learning. We formulate the problem as a MDP with high dimensional state and action spaces, which cannot be efficiently solved using classical RL algorithms like DQN. We then design several innovative adaptations to vanilla DQN, including GCN-based value function approximator that exploits the correlations of nodes, a primary-secondary agents framework that decomposes the combinatorial action selection in each time period into a sub-sequence of node selection, and a curriculum learning component that addresses the sparseness of reward for the secondary agents. Empirical results show that in terms of solution quality, our approach outperforms the state-of-the-art *Fast-REMEDY* by a margin of 9% – 33%, and works better than baselines even with network structure uncertainty. In the largest network we experimented which 1899 nodes, our approach is able to scale up to a planning horizon 10 times that in state-of-the-art approach *Full-REMEDY*. Interestingly, policy analysis results show that compared with most baselines (except for *Random*), our approach is fairer in the sense that it tends to spread the screening across different nodes. For future work, we plan to incorporate uncertainty on the graph structure in training our RL algorithms and further improve the robustness of our approach.

ACKNOWLEDGMENTS

Chen and Jabbari were supported by the Center for Research on Computation and Society. This work was supported by the Army Research Office (MURI W911NF1810208). This research was also supported by NSF CCF-1522054.

REFERENCES

[1] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine* 34, 6 (2017), 26–38.

[2] F. G. Ball, E. S. Knock, and P. D. O'Neill. 2015. Stochastic epidemic models featuring contact tracing with delays. *Math biosciences* 266 (2015), 23–35.

[3] A. Banerjee, A. G. Chandrasekhar, E. Duflo, and M. O. Jackson. 2013. The diffusion of microfinance. *Science* 341, 6144 (2013), 1236498.

[4] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. 2016. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940* (2016).

[5] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*. 41–48.

[6] Marian Boguná, Claudio Castellano, and Romualdo Pastor-Satorras. 2013. Nature of the epidemic threshold for the susceptible-infected-susceptible dynamics in networks. *Physical review letters* 111, 6 (2013), 068701.

[7] Hanjun Dai, Bo Dai, and Le Song. 2016. Discriminative embeddings of latent variable models for structured data. In *ICML*. 2702–2711.

[8] Peter Dayan and Geoffrey E Hinton. 1993. Feudal reinforcement learning. In *NeurIPS*. 271–278.

[9] Thomas G Dietterich. 2000. Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of AI research* 13 (2000), 227–303.

[10] Kimon Drakopoulos, Asuman Ozdaglar, and John N Tsitsiklis. 2014. An efficient curing policy for epidemics on graphs. *IEEE TNSE* 1, 2 (2014), 67–75.

[11] Kimon Drakopoulos, Asuman Ozdaglar, and John N Tsitsiklis. 2016. When is a network epidemic hard to eliminate? *Mathematics of Operations Research* (2016).

[12] Thomas DuBois, Stephen Eubank, and Aravind Srinivasan. 2012. The effect of random edge removal on network degree sequence. *Electronic journal of combinatorics* 19, 1 (2012).

[13] Ken TD Eames and Matt J Keeling. 2003. Contact tracing and disease control. *Proceedings of the Royal Society of London. Series B: Biological Sciences* 270, 1533 (2003), 2565–2571.

[14] Luca Ferretti, Chris Wymant, Michelle Kendall, Lele Zhao, Anel Nurtay, Lucie Abeler-Dörner, Michael Parker, David Bonsall, and Christophe Fraser. 2020. Quantifying SARS-CoV-2 transmission suggests epidemic control with digital contact tracing. *Science* 368, 6491 (2020).

[15] Johan Giesecke. 2017. *Modern infectious disease epidemiology*. CRC Press.

[16] Alexander K Hartmann and Martin Weigt. 2005. *Phase transitions in combinatorial optimization problems*. Vol. 67. Wiley Online Library.

[17] Jessica Hoffmann and Constantine Caramanis. 2018. The Cost of Uncertainty in Curing Epidemics. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 2 (2018), 31.

[18] Petter Holme. 2004. Efficient local strategies for vaccination and network attack. *EPL (Europhysics Letters)* 68, 6 (2004), 908.

[19] Alex Irpan. 2018. Deep Reinforcement Learning Doesn't Work Yet. https://www.alexirpan.com/2018/02/14/rl-hard.html.

[20] L. Isella, J. Stehlé, A. Barrat, C. Cattuto, J.-F. Pinton, and W. Van den Broeck. 2011. What's in a crowd? Analysis of face-to-face behavioral networks. *J. Theor. Biol.* (2011), 166.

[21] David S Johnson. 1974. Approximation algorithms for combinatorial problems. *Journal of computer and system sciences* 9, 3 (1974), 256–278.

[22] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4 (1996), 237–285.

[23] Harshavardhan Kamarthi, Priyesh Vijayan, Bryan Wilder, Balaraman Ravindran, and Milind Tambe. 2019. Learning policies for Social network discovery with Reinforcement learning. *arXiv preprint arXiv:1907.11625* (2019).

[24] Harshavardhan Kamarthi, Priyesh Vijayan, Bryan Wilder, Balaraman Ravindran, and Milind Tambe. 2020. Influence Maximization in Unknown Social Networks: Learning Policies for Effective Graph Sampling.. In *AAMAS*. 575–583.

[25] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the 9th ACM SIGKDD*. 137–146.

[26] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. 2017. Learning combinatorial optimization algorithms over graphs. In *NeurIPS*. 6348–6358.

[27] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR-17*. Toulon.

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NeurIPS*. 1097–1105.

[29] Cong Li, Huijuan Wang, and Piet Van Mieghem. 2012. Degree and principal eigenvectors in complex networks. In *ICRN*. Springer, 149–160.

[30] Fredrik Liljeros, Christofer R Edling, and Luis A Nunes Amaral. 2003. Sexual networks: implications for the transmission of sexually transmitted infections. *Microbes and infection* 5, 2 (2003), 189–196.

[31] Dermot Maher, Chris Dye, Katherine Floyd, Andrea Pantoja, Knut Lonnroth, Alasdair Reid, Eva Nathanson, Thad Pennas, Uli Fruth, Jane Cunningham, et al. 2007. Planning to improve global health: the next decade of tuberculosis control. *Bulletin of the World Health Organization* 85 (2007), 341–347.

[32] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602* (2013).

[33] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *nature* 518, 7540 (2015), 529–533.

[34] Han-Ching Ou, Arunesh Sinha, Sze-Chuan Suen, Andrew Perrault, Alpan Raval, and Milind Tambe. 2020. Who and When to Screen: Multi-Round Active Screening for Network Recurrent Infectious Diseases Under Uncertainty. In *AAMAS*. 992–1000.

[35] Han Ching Ou, Kai Wang, Finale Doshi-Velez, and Milind Tambe. [n.d.]. Active Screening on Recurrent Diseases Contact Networks with Uncertainty: a Reinforcement Learning Approach. ([n. d.]).

[36] P. Panzarasa, T. Opsahl, and K. M. Carley. 2009. Patterns and dynamics of users' behavior and interaction: Network analysis of an online community. *J. Assoc. Inf. Sci. Technol.* (2009), 911.

[37] Ronald Parr and Stuart J Russell. 1998. Reinforcement learning with hierarchies of machines. In *NeurIPS*. 1043–1049.

[38] Roni Parshani, Shai Carmi, and Shlomo Havlin. 2010. Epidemic threshold for the susceptible-infectious-susceptible model on random networks. *Physical review letters* 104, 25 (2010), 258701.

[39] Romualdo Pastor-Satorras and Alessandro Vespignani. 2001. Epidemic spreading in scale-free networks. *Physical review letters* 86, 14 (2001), 3200.

[40] Wei Qiu, Haipeng Chen, and Bo An. 2019. Dynamic Electronic Toll Collection via Multi-Agent Deep Reinforcement Learning with Edge-Based Graph Convolutional Networks.. In *IJCAI*. 4568–4574.

[41] Yizhi Ren, Mengjin Jiang, Ye Yao, Ting Wu, Zhen Wang, Mengkun Li, and Kim-Kwang Raymond Choo. 2018. Node Immunization in Networks with Uncertainty. In *(TrustCom/BigDataSE)*. IEEE, 1392–1397.

[42] S. Saha, A. Adiga, B. A. Prakash, and A. K. S. Vullikanti. 2015. Approximation algorithms for reducing the spectral radius to control epidemic spread. In *Proceedings of the 2015 SIAM ICDM*. SIAM, 568–576.

[43] M. Salathé, M. Kazandjieva, J. W. Lee, P. Levis, M. W. Feldman, and J. H. Jones. 2010. A high-resolution human contact network for infectious disease transmission. *Proceedings of the National Academy of Sciences* 107, 51 (2010), 22020–22025.

[44] K. Scaman, A. Kalogeratos, and N. Vayatis. 2016. Suppressing epidemics in networks using priority planning. *IEEE TNSE* (2016).

[45] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. *nature* 529, 7587 (2016), 484–489.

[46] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *nature* 550, 7676 (2017), 354–359.

[47] Jeffrey A Smith and James Moody. 2013. Structural effects of network sampling coverage I: Nodes missing at random. *Social networks* 35, 4 (2013), 652–668.

[48] Hyungseok Song, Hyeryung Jang, Hai H Tran, Se-eun Yoon, Kyunghwan Son, Donggyu Yun, Hyoju Chung, and Yung Yi. 2019. Solving continual combinatorial selection via deep reinforcement learning. *arXiv preprint arXiv:1909.03638* (2019).

[49] C. Sun and Y.-H. Hsieh. 2010. Global analysis of an SEIR model with varying population size and vaccination. *Applied Mathematical Modelling* 34, 10 (2010), 2685–2697.

[50] Richard S Sutton et al. 1998. *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge.

[51] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[52] Richard S Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence* 112, 1-2 (1999), 181–211.

[53] David Taylor-Robinson. 1994. Chlamydia trachomatis and sexually transmitted disease.

[54] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. 2012. Gelling, and melting, large graphs by edge manipulation. In *Proceedings of the 21st ACM international CIKM*. ACM, 245–254.

[55] P. Vanhems et al. 2013. Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. *PloS one* 8 (2013), 73970.

[56] Y. Wang, C. Chakrabarti, D.and Wang, and C. Faloutsos. 2003. Epidemic spreading in real networks: An eigenvalue viewpoint. In *SRDS*. IEEE, 25–34.

[57] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3-4 (1992), 279–292.

[58] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329* (2014).

[59] Yao Zhang and B Aditya Prakash. 2015. Data-aware vaccine allocation over large networks. *ACM TKDD* 10, 2 (2015), 1–32.

A BELIEF UPDATE

Our belief update procedure follows that of [34]. Let b_t^v denote the probability of node v being I at time step t. Before any observation and action happens, we initialize $b_0^v = 0.5$. Suppose the belief of the probabilities in previous time step *before observation* is b_{t-1}^v. When the observation is received, the belief of probabilities of the previous time step is then:

$$\hat{b}_{t-1}^v = \begin{cases} 1, & v \in \mathbf{o}_t \\ \frac{(1-\gamma)b_{t-1}^v}{(1-b_{t-1}^v)+(1-\gamma)b_{t-1}^v}, & otherwise \end{cases}$$

Based on these probabilities, the probability of v being I in the current time step is given by:

$$b_t^v = \begin{cases} 0, & v \in \mathbf{o}_t, \\ (1 - \prod_{u \in \delta(v)}(1 - \beta \hat{b}_{t-1}^u))(1 - \hat{b}_{t-1}^v) + \hat{b}_{t-1}^v, & otherwise \end{cases} \tag{6}$$

in which $\delta(v)$ denotes neighbor of node v. Such probability is embed into the GCN of both primary agent and secondary agent 0, encoded as $\mathbf{s}_0^{II} = \mathbf{s}^I$. Based on the output of its Q function, each worker i will select its action a_i^{II} and add it to the master action set $\mathbf{a}^I$. After updating the current belief base on the selected action ($b_t^v = 0$) for $v \in \mathbf{a}^I$, b_t^v will be passed to and encoded by the GCN of next worker as $\mathbf{s}_{i+1}^{II}$. This process repeats until the last worker agent is reached. The final b_t^v is then used in the belief update of next time step.

B TABLE OF NOTATIONS

Table 3 shows a summary of the notations used in this paper. Note that for simplicity, we do not distinguish between the primary and secondary agents for concepts related to the algorithm.

Table 3: Notations for major concepts.

Symbol	Description
G	contact network
S	susceptible state
I	infectious state
β	transmission probability
γ	cure probability
t	time step
T	time horizon
k	screening budget for each time step
$\mathbf{x}_t$	true state at time t (not available in testing)
$\mathbf{a}_t$	set of nodes actively screened (action at time t)
$\mathbf{o}_t$	set of self-report nodes (observation at time t)
$\mathbf{b}_t$	Belief state at time t
$\mathbf{s}_t$	state representation for Q function
r_t	step wise reward
Q	Q function
α	future discount factor
τ	auxiliary coefficient for curriculum learning
$\bar{r}_t$	Initial step wise reward for curriculum learning

C ABLATION STUDY

To show the effectiveness of our two-level RL framework and the curriculum learning component, we conduct an ablation study on a sample network Face-to-face. Fig. 7 shows the ablation study results on the Face-to-face network. We evaluate 4 settings: (i) Single agent without curriculum learning (CL); (ii) Two-level (k agents) RL without CL; (iii) Single agent with CL and (iv) Full (i.e., two-level RL with CL). Note that in (i) and (iii), the remaining budget is directly hard-coded as part of the state information to the GCN. By comparing (i) with (iii) or comparing (ii) with (iv), we can see that curriculum learning is critical in improving the solution quality. On the other hand, by comparing (i) with (ii) or comparing (iii) with (iv), we can see that the two-level primary and secondary agents framework, which trains a different secondary agent policy, is also improving the solution quality by a large margin. On the contrary, hard-coding the remaining budget into the state leads to sub-optimal solution quality.

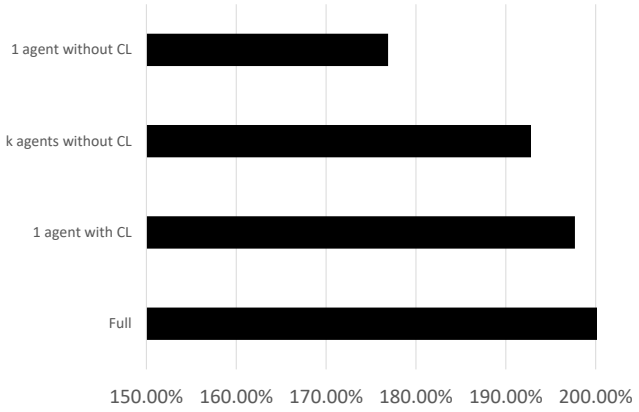
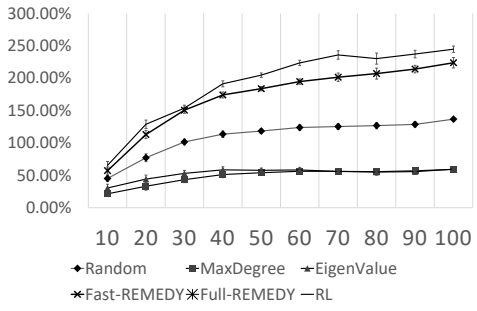


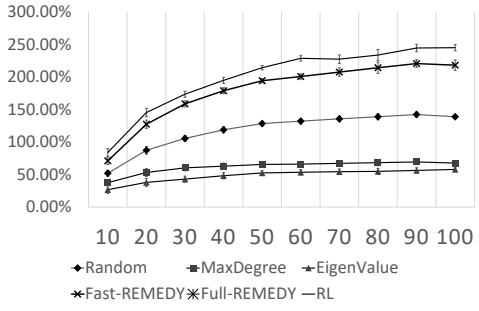
Figure 7: Ablation study run on the Face-to-face network. The x-axis is the percentage of improvement over no-intervention. The y-axis denotes different variants of our approach.

D EXPERIMENTAL RESULTS ON OTHER NETWORKS

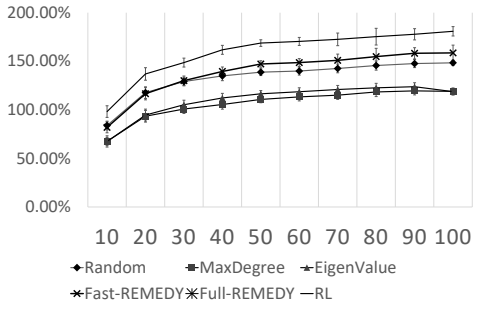
Figure 8 to 11 show subtracted result experiment results of *Hospital*, *India*, *Flu* and *Irvine* network. Similar trends can be observed as in the main text.



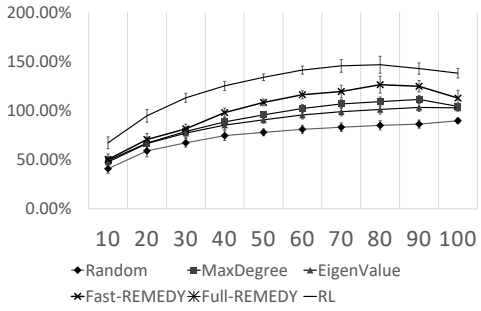
(a) Hospital



(b) India

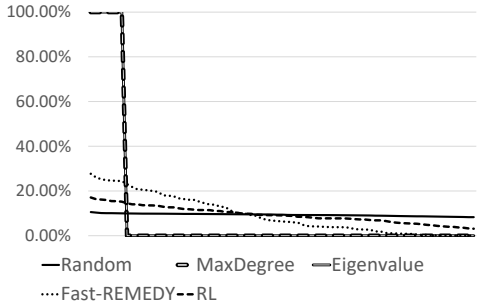


(c) Flu

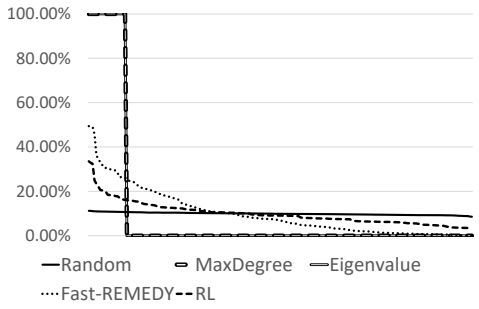


(d) Irvine

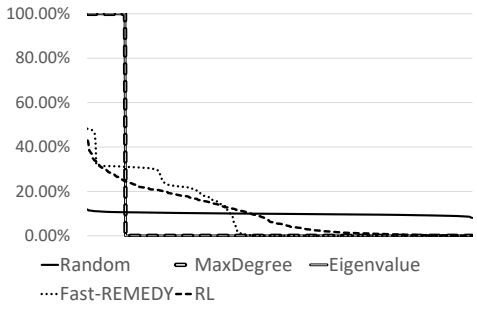
Figure 8: Performance under node information removal.



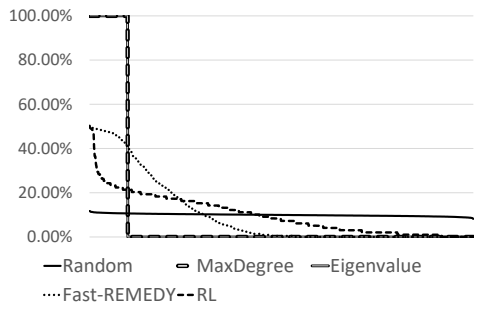
(a) Hospital



(b) India

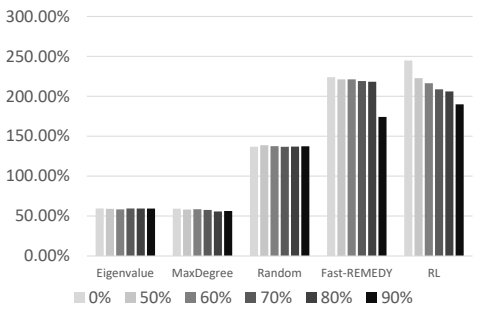


(c) Flu

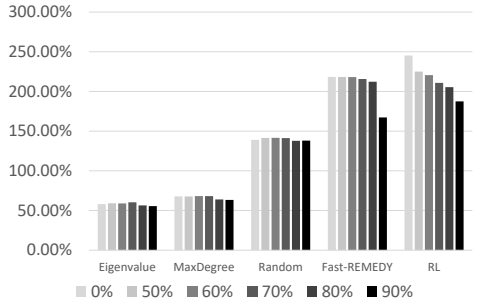


(d) Flu

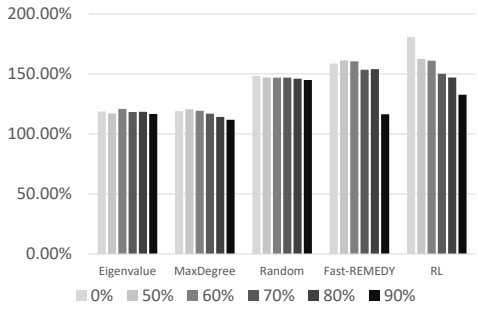
Figure 9: Node picking frequency.



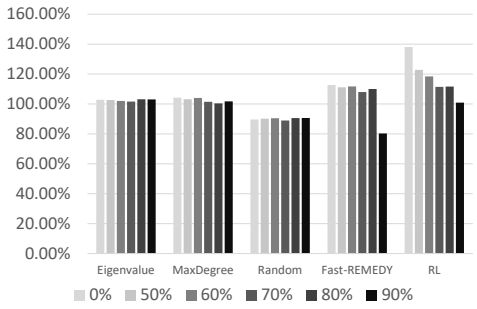
(a) Hospital



(b) India

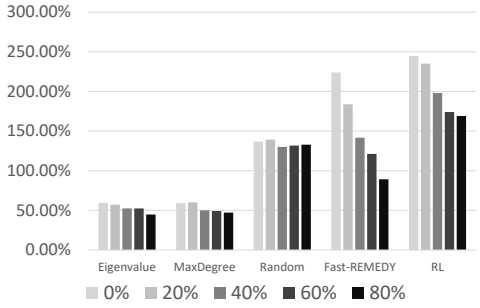


(c) Flu

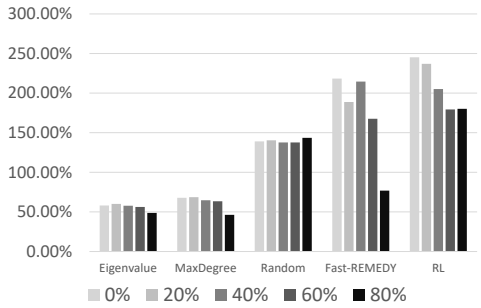


(d) Irvine

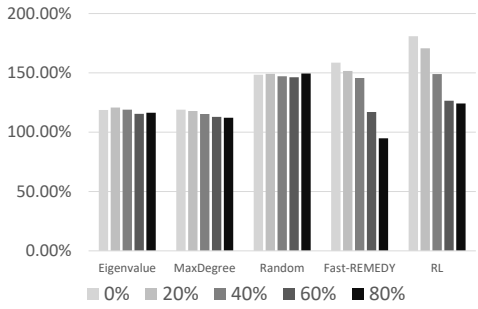
Figure 10: Performance under edge information removal.



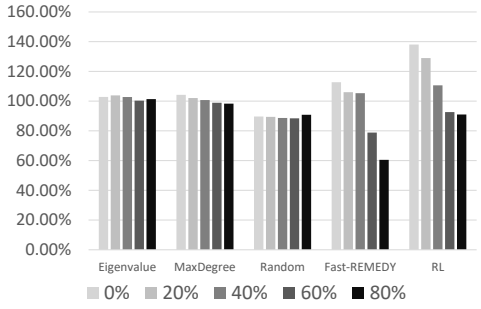
(a) Hospital



(b) India



(c) Flu



(d) Irvine

Figure 11: Performance under node information removal.