Active Screening on Recurrent Diseases Contact Networks with Uncertainty: a Reinforcement Learning Approach

Han Ching Ou[1], Kai Wang[1], Finale Doshi-Velez[1], and Milind Tambe[1]

Harvard University, Cambridge MA 02138, USA
{hou,kaiwang,final,milind_tambe}@g.harvard.edu

Abstract. Controlling recurrent infectious diseases is a vital yet complicated problem. A large portion of the controlling epidemic relies on patients visit clinics voluntarily. However, they may already transmit the disease to their contacts by the time they feel sick enough to visit the clinic, especially for conditions with a long incubation period. Therefore, active screening/case finding was deployed to provide a powerful yet expensive means to control disease spread in recent years. To make active screening success a given limit budget, one of the challenges that need to be addressed is that we do not know the exact state of each patient. Given the number of horizon and budget we have in each time step, we also need to plan our screening efficiently and screening the vital patients in time. Thus, we apply a reinforcement learning approach to solve active screening problems on the network SIS disease model. The first contribution of this work is that we identify three significant challenges in active screening problems: partially observable states, combinatorial action choice, high-dimensional state-action space. We further propose the corresponding solutions to overcome these challenges. Specifically, we resolve the issue of high-dimensional state-action space by encoding the actions and partially observable states into a lower dimension form, which is done by either manually, using domain expertise, or automatically, using the state of the art GCN approach. We show that our approach can scale up to large graphs and perform decently compared to other baselines of previous literature and current practice.

.

1 Introduction

Contagious diseases, such as influenza and sexually transmitted diseases (STDs) (e.g., gonorrhea and chlamydia) are critical public-health challenges that continue to threaten lives and impose significant economic burden on society. For example, the economic loss due to influenza in the USA alone is estimated to be \$11.2 billion in 2015 [14]. While low-cost treatment programs are available, individuals ignore symptoms and delay care, increasing transmission risk. As a result, health agencies engage in active screening or contact tracing efforts, where individuals are asked to undergo diagnostic tests and offered treatment if tests are positive [7,5]. However, active screening is expensive in developing countries. Even in USA, Braxton et al. [4] state that “In 2012, 52% of state and local STD programs experienced budget cuts. This amounts to reductions in clinic hours, contact tracing, and screening for common STDs." Efficiently identifying and intervening for infectious cases is therefore of vital importance.

However, in many settings, active screening/contact tracing is expensive and time consuming. There is a huge body of literature on spread and control of recurrent diseases (no permanent immunity). However, all these prior work assume perfect observation of who is infected and who is not. Also, most of these methods focus on eradication of disease, which is not possible if the screening resources are limited. Thus, important real world characteristics such as partial observation and limited resources have not been handled in any prior work.

2 Problem Statement

In this work, we aim to use reinforcement learning to decide which patients to actively screen in each time frame for a multi-round scenario with limited horizon. The environment we considered is based on the well-known SIS model [1,2]. An individual can either be in state S (a healthy individual *susceptible* to disease) or I (the individual is *infected*). SIS models capture the dynamics of recurrent diseases, where permanent immunity is not possible (e.g., TB, typhoid). We adopt a discrete time SIS model for modeling the disease dynamics propagating on a given graph $G = (V, E)$, where each node represents a single patient and each edge indicates the link between people which disease can spread. We assume the structure of the contact network G to be known yet the states of patients to be unknown. We can only observe the patients' current states while actively screening the patients. At each round, we have k resources that allow us to provide active screening to k patients. After being screened, the infected patients recover back to susceptible healthy patients, while the susceptible patients remain susceptible.

Given a contact network $G(V, E)$, infection spreads via the edges in the network. There are $|V|$ individuals, and we use $\delta(v)$ to denote neighbors of node v in the network. Each individual (node) v in the network at time t is in state $\mathbf{s}_v(t) \in \{S, I\}$. Let $\mathbf{t}_v(t)$ denote the state vector that represents the true state of node v at time t where S is represented as $[1, 0]^\top$ and I as $[0, 1]^\top$. Given the initial state, an infected node infects its healthy neighbors with rate α independently and recovers with probability c. The latter term represents the probability that the node may visit a doctor on its own initiative. The health state transition probabilities of a node is then given by $P\left[s_v(t+1) = \{S, I\}\right] = \mathbf{T}_v^N(t)\mathbf{t}_v(t)$ where

$$\mathbf{T}_v^N(t) = \begin{array}{c c} & \begin{array}{cc} S & I \end{array} \\ \begin{array}{c} S \\ I \end{array} & \begin{bmatrix} 1 - q_v & c \\ q_v & 1 - c \end{bmatrix} \end{array}, \tag{1}$$

Table 1. Notations

Notations	Definition
	Model
S	susceptible state
I	infected state
α	transmission rate
c	cure rate
t	time step number
T	terminal time step
k	budget for each time step
$\delta(v)$	set of v's neighbors
$\mathbf{s}_v(t)$	state of v at time t
$\mathbf{a}(t)$	set of nodes actively screened as action
$\mathbf{o}(t)$	set of nodes naturally cured as observation
$\mathbf{t_v}(t)$	true state vector of node v at time t
$\mathbf{T}_v^N(t)$	true state transition matrix for $V \setminus a(t)$
$\mathbf{T}_v^A(t)$	true state transition matrix for $a(t)$
	Algorithm
$b_v(t)$	marginal probability of v being in I state
$\mathbf{b}(t)$	belief vector of all nodes being in I state
$\bar{\mathbf{b}}(t)$	intermediate belief vector after knowing $o(t)$
$\mathbf{B}_v^N(t)$	transition matrix for $V \setminus a(t) \cup o(t)$
$\mathbf{B}_v^A(t)$	transition matrix for $a(t) \cup o(t)$

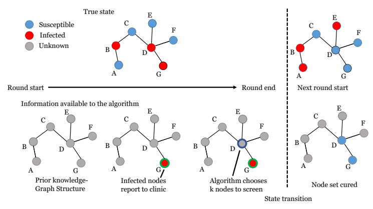
True state
Susceptible
Infected
Unknown
Round start
Round end
Next round start
Information available to the algorithm
Prior knowledge-Graph Structure
Infected nodes report to clinic
Algorithm chooses k nodes to screen
Node set cured
State transition

Fig. 1. The procedure of the ACTS problem.

and $q_v = 1 - (1-\alpha)^{|\{u \in \delta(v) \mid \mathbf{s_u}(t)=I\}|}$. The columns denote the state of v at time t and the rows denote the state at $t+1$. The transition probabilities follow the disease dynamics described earlier. In particular, q_v captures the exact probability that node v becomes infected from its infected neighbors $\{u \in \delta(v) \mid \mathbf{s_u}(t) = I\}$ and c captures the probability that I individuals recover without active screening.

Given such transition probabilities and an initial state, if no intervention happens, the network state evolves by flipping biased coins for each node to determine their next true state in each round. The process is repeated until the terminal step T is reached.

Motivated by active screening/contact tracing campaigns that have been practiced since the 1980s [5] and applied in various forms/diseases [4], we propose the Active Screening (ACTS) Problem. Given the SIS model in the previous section, an active screening agent seeks to determine the best node sets $\mathbf{a}(t) \subset V$ to actively screen and cure with a limited budget of $|\mathbf{a}(t)| \leq k$ at each round t. The agent does not know the ground truth health state of all individuals. The agent knows the network structure $G(V, E)$, the infection probability α, and recovery probability c. In addition, the agent observes the *naturally cured* node set $\mathbf{o}(t)$ at time t—because this set of patients come to the clinic voluntarily. Active screening starts after the agent acquires information about $\mathbf{o}(t)$. Let $\mathbf{a}(t)$ be the set of nodes that are actively screened at time t. A node $v \in \mathbf{a}(t)$ becomes cured at time $t+1$. Thus, the transition matrix for a node $v \in \mathbf{a}(t)$ is $P\left[\mathbf{s}_v(t+1) = \{S, I\}\right] = \mathbf{T}_v^A(t) t_v(t)$, where

$$\mathbf{T}_v^A(t) = \begin{array}{c|cc} & S & I \\ \hline S & 1 & 1 \\ I & 0 & 0 \end{array} \tag{2}$$

The action the agent takes at time t does not affect the transition matrix $\mathbf{T}_v^N(t)$ of the nodes not involved in active screening.

Fig. 1 illustrates an example of the problem procedure. The upper part of the figure shows how the true state of the network evolves and the lower part of the figure shows the information available to the algorithm. In this example, there are seven nodes A∼G. In each round, infected nodes (nodes B, D, and G in the example) flip a coin and report to the clinic with probability c. The algorithm acquires the information of the nodes that eventually report to the clinic and are about to be cured, which is $\{$G$\}$ this round. Based on this information, the algorithm will choose a set of nodes, say $\{$D$\}$, to actively screen. These two sets of nodes are guaranteed to be in S state in the next round. After that, the state of the network transitions and the next round starts.

It is worth noting that although both the nodes that voluntarily report to the clinic and the nodes that are actively screened are guaranteed to be in S state in the next round, their neighbors may still be infected by them in the current round. In the example, node E is infected by node D even though node D was actively screened. This allows us to simplify the state transitions because curing and spreading infection occur at the same time.

Our objective is to maximize the health quality of each individual at each round (in contrast to past work, which primarily focuses on the cost of eradicating the disease entirely). The objective of the ACTS problem is:

$$\min_{C_a(0),\dots,C_a(T)} \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t \sum_{v\in V} \mathbb{1}_{\mathbf{s}_v(t)=I}\right]. \tag{3}$$

where $0 \leq \gamma \leq 1$ is a future discount factor.

Problem Statement *(ACTS Problem) Given a contact network $G(V, E)$, the disease and active screening model, find an active screening policy such that the expectation of $\sum_{t=0}^{T} \sum_{v\in V} \mathbb{1}_{\mathbf{s}_v(t)=I}$ is minimized.*

3 Challenges

In this paper, we first identify three main challenges that we are facing in ACTS problem.

– (a) Partially observable states
– (b) Combinatorial maximization problem
– (c) High-dimensional state-action space

The first challenge corresponds to the standard challenge in partially observable Markov decision process (POMDP), where the state is unknown and only a partial state can be observed by the agent. There are some POMDP solvers but the common issue of these solvers is the scalability. POMDP solvers are extremely unscalabe, which are not suitable for our case as we have a large state and action space. The second issue comes along with large and combinatorial action choices. Any subset of patients with less than k elements is a feasible action, which can involve exponentially many actions and is hard to find the optimal action. The action selection problem can also be treated as a combinatorial problem, where our goal is to select the optimal k patients to provide active screening. In general, combinatorial maximization in active screening problem can be NP-hard to solve. Therefore, an efficient greedy algorithm is needed in our case, which will be discussed in the following section. The final challenge is the large state-action space. This corresponds to the issue of representation, where we require a compact representation of our state-action pair, while at the same time maintaining the information included in these state-action pairs. We will address these challenges in the following sections with our proposed solutions.

4 Partially Observable States

The first issue is that we do not have access to the hidden state, falling into the category of partially observable Markov decision process (POMDP), which prohibits the standard reinforcement learning methods to work here. There are also POMDP solvers that

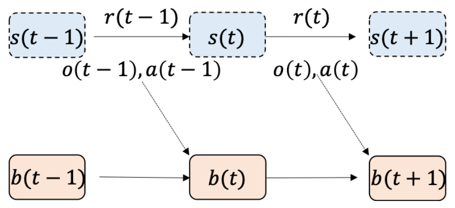

Fig. 2. Belief update based on action, observation, and previous state.

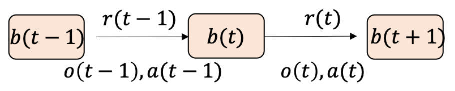

Fig. 3. MDP on the belief state.

can directly solve a POMDP with partially observable states. However, these POMDP solvers are generally extremely slow and non-scalable. In our setting, the scale of our problem is quite large. We have $|V|$ hidden states, where each patient corresponds to an entry of the hidden state. POMDP solvers do not apply to such high dimensional setting, which urges us to think of other method to resolve the issue of partially observable state.

Solution: The way we resolve this issue is to use the posterior belief $\mathbf{b}(t)$ estimate of the current state $\mathbf{s}(t)$ as our fake state, which might lose some information but hopefully is rich enough to represent the original state distribution. The posterior belief $\mathbf{b}(t)$ at time t is affected by the previous observation $\mathbf{o}(t-1)$, action $\mathbf{a}(t-1)$, and belief $\mathbf{b}(t-1)$. As shown in Figure 2, we can denote the belief update function as $\mathbf{b}(t) = g(\mathbf{b}(t-1), \mathbf{a}(t-1), \mathbf{o}(t-1))$. Now this update function is achieved by an approximate Bayesian approach, where people have been actively screened or observed in the clinic in the last round will be healthy in this round, and the people with no information can be updated by the conditional probability.

Once we get the posterior belief, we can use the belief as our new state, then run basic online reinforcement learning methods afterward. The new MDP structure is shown in Figure 3.

5 Combinatorial Maximization Problem

The second issue is that we cannot easily compute the optimal action $\max_{\mathbf{a}\in A} Q(\mathbf{b}, \mathbf{o}, \mathbf{a})$. Notice that since this is in the partially observable setting, so at the beginning of each round, an observation $\mathbf{o}$ will be given and can be used to determine the action. Given the exponential size of all feasible actions A, the maximization problem is a combinatorial optimization problem, which can be computationally expensive to solve. The way we try to resolve the issue is to employ different greedy algorithms. An incorrect estimate

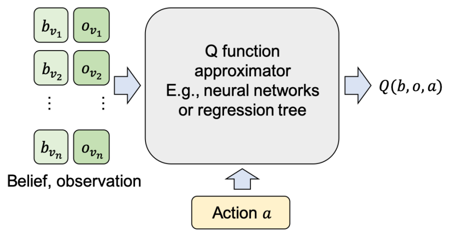

Fig. 4. Q function with belief, observation, and action as inputs. The function predicts a single scalar value as the Q value of the given belief, observation, and action.

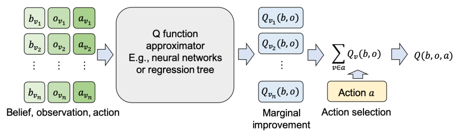
b_{v_1} o_{v_1} a_{v_1}
b_{v_2} o_{v_2} a_{v_2}
b_{v_n} o_{v_n} a_{v_n}
Belief, observation, action
Q function approximator E.g., neural networks or regression tree
$Q_{v_1}(b,o)$
$Q_{v_2}(b,o)$
$Q_{v_n}(b,o)$
Marginal improvement
$\sum_{v \in a} Q_v(b,o)$
Action a
Action selection
$Q(b,o,a)$

Fig. 5. Q function with belief and observation as inputs and with marginal improvement of selecting each patient as output. The predicted reward of a given action can be computed by taking summation of the marginal improvement of curing patients in the action.

of the maximum value Q value could lead to incorrect equilibrium of Bellman equation, which might break the optimality of the equilibrium of Bellman equation.

Solution: Our proposed solution here is to predict the improvement of each individual patient selection instead of an aggregated improvement of the entire set of patients. As shown in Figure 4, the standard Q function takes belief and observation (state), and action as inputs, and outputs a single scalar value as the predicted future reward after taking this action. Instead, as shown in Figure 5, our Q function outputs a scalar for each individual patient, representing the predicted marginal improvement if we actively screen this patient. By doing so, the optimization problem becomes much easier, where we can directly estimate the improvement of screening an additional patient, which only requires one additional access to the Q function. Nonetheless, this approach ignores the correlation between patients. It also assumes the reward of an entire action is separable into marginal improvement from each patients been screened. However, given the hardness of the original combinatorial maximization problem, we have to sacrifice some accuracy in exchange of the scalability. Using the Q function with marginal improvement outputs, we will show two heuristics in the following section.

5.1 Incremental Selection

The first heuristic method is to incrementally add patients into the screening set until we run out of the budget. Given the existing action, we can feed the action into the Q function to re-estimate the new marginal improvement of screening all the other patients again. This is aligned with the greedy algorithm in submodular maximization problems, which has a nice theoretical guarantee. The time complexity of this heuristic is linear in terms of the budget k, which can be a burden when the number of budget increases, especially in networks with large population. In our problem, for each iteration of new episode or each gradient update on a tuple of belief, observation, action, and next belief, we require to run the above heuristic to find an approximate optimal next action with time complexity $O(k)$. Although each iteration is not too expensive, in the reinforcement learning domain, iteratively training on this heuristic is still quite expensive.

5.2 One-shot Selection

Instead of iteratively updating the marginal improvement by the updated action, we can directly select the top k patients with the largest marginal improvement. This only requires one single evaluation with time complexity $O(1)$. It is also less accurate due to the one-shot selection. However, we can see a significant speedup while using this simplified heuristic.

A mixture of these two heuristics is to iteratively select a batch of patients and update the resulting predictions afterward. It can balance between the accuracy and the scalability, which is left as a future direction.

6 High-dimensional State-action Space

In Figure 5, in order to estimate the $Q(\mathbf{b}, \mathbf{o}, \mathbf{a})$ value, we need to feed the belief $\mathbf{b}$, observation $\mathbf{o}$, and action $\mathbf{a}$ into the Q function, which is a high-dimensional vector. From the function approximation perspective, in order to learn from high-dimensional data, the sample complexity could be much larger than of low-dimensional data. In the reinforcement learning domain, it corresponds to the need of a huge amount of data to collect and learn from. This is generally infeasible in the offline case and time consuming in the online case. Therefore, we would like to have a compact representation but at the same time maintaining the information containing in the input.

We first show a failed attempt and then show another two alternatives that we eventually use.

6.1 One-hot Encoding

One naive and straightforward encoding method is to use one-hot encoding to encode action and observation. This means we will have three vectors of length $|V|$, which are $\mathbf{b}, \mathbf{o}$ where $\mathbf{o}_v = 1$ if node v is observed and 0 and $\mathbf{a}$ where $\mathbf{a}_v = 1$ if node v is selected and 0 otherwise. Then we simply concatenate into one single vector with length $3|V|$ and feed into our regression model.

However, this encoding method has many drawbacks. First, the dimension is high. Second, it is hard for the regress to learn the close relationship of the i-th, $i + |V|$-th and $i + 2|V|$-th elements in the state vector, as they represents the same node. Finally, the encoding method did not encode the fact that the observation and action set of nodes ($\mathbf{o}$ and $\mathbf{a}$) are guaranteed to be cured in the next round and again relies the reinforcement learning to figure that out. Given there are a great amount of information to learn, this encoding method is not applicable due to its requirement of intensive amount of training to reason the relations between its high dimensional features.

Therefore, we propose the following two alternatives.

6.2 Nodewise Encoding

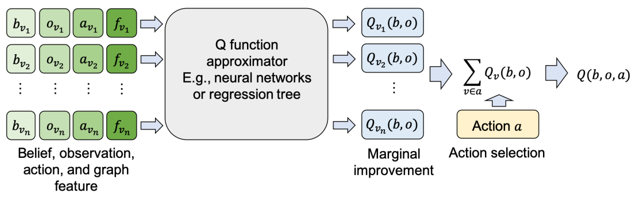
b_{v_1} o_{v_1} a_{v_1} f_{v_1}
b_{v_2} o_{v_2} a_{v_2} f_{v_2}
b_{v_n} o_{v_n} a_{v_n} f_{v_n}
Belief, observation, action, and graph feature
Q function approximator E.g., neural networks or regression tree
$Q_{v_1}(b,o)$
$Q_{v_2}(b,o)$
$Q_{v_n}(b,o)$
Marginal improvement
$\sum_{v\in a} Q_v(b,o)$
$Q(b,o,a)$
Action a
Action selection

Fig. 6. Instead of feeding all the features into a single Q function estimator, we can feed each node-dependent feature only into the Q function estimator to reduce the dimensionality.

Since feeding all the features at once is not working, an intuitive way is to consider each node independently as shown in Figure 6, where we feed the features related to node v to the Q function to get the predicted marginal improvement of actively screening node v. In order to take the graph structure into account, we can also include some local graph features as the features related to node v, e.g., the degree of node v, average path length starting from v etc. The belief, observation, action, and the graph features of node v are fed into the Q function and get a single scalar value, representing the prediction of node v. This can largely reduce the size of input by including only the features related to the corresponding nodes. However, we know that the disease can spread from nodes to their neighbors, which implies that we cannot simply ignore the neighbor nodes and their features. Although such abrupt reduction can efficiently reduce the dimension, it can lose a significant amount of information and graph structure at the same time.

6.3 Graph Convolutional Neural Networks

Another more structural ways to encode the features is to apply graph convolutional neural networks (GCNs) [11]. We can encode the features related to node v as the node features of v. The convolutional layers in GCNs can facilitate the nature of message passing and automatically aggregate the neighbor features as shown in Figure 7. By using GCNs, we do not need to hand-craft graph features to represent the graph structure. What we need to do is to make GCNs learn the underlying graph embedding and form the corresponding representation.

Figrue 8 demonstrates the flowchart of our GCN implementation. We replace the Q function estimator by a GCN, which takes node features and the graph as input and outputs a prediction for each node in the graph.

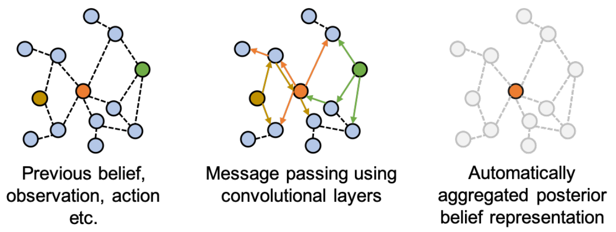
Previous belief, observation, action etc.
Message passing using convolutional layers
Automatically aggregated posterior belief representation

Fig. 7. Message passing in the convoluational layers can allow features like belief, observation, and action to propagate to the neighbor nodes.

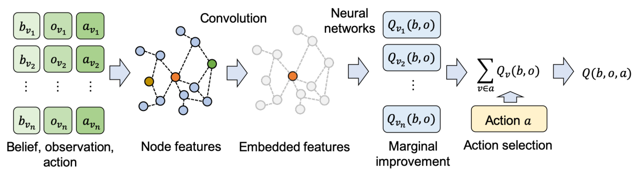

Fig. 8. Integrating GCN as our Q function can resolve the issue of high-dimensional input and also automatically extract the graph structure and form a compact feature embedding.

7 Experiments

In this project, we implement the two different encodings mentioned in Section 6. For the nodewise encoding, we follow the implementation of fitted Q learning [8] with extra trees regressor [9] as the Q function approximator. For the GCN encoding, we implement a deep Q network (DQN) [12] with GCN proposed by Morris et al. [13], where we maintain a fixed-size replay memory to continuously train the GCN. For both FQI and DQN implementation, we adopt the one-shot selection to speed up finding the optimal action and the maximization problem. We consider the online setting, where we can apply new policy and get data from the new policy.

7.1 Experiment Setup

We perform our experiments on real world contact networks that are publicly available. We set our finite horizon to be $T = 20$ which represents about 10 years of active screening with each period being six months [6]. We set the discount factor of 0.9. We assume $(\alpha, c) = (0.1, 0.1)$ and the network structure is known by surveys or estimations and set our budget k to be 10% of the population n in each period, thus the algorithm needs to scale well according to budget as graph size grows. Finally, we set the random policy probability to be $\epsilon = 0.05$ in our learning process.

We compare the following screening strategies with our reinforcement learning algorithm

(1a) RANDOM: Randomly select nodes for active screening.
(1b) MAXDEGREE: Successively choose nodes with the largest degree until the budget is reached.
(1c) EIGENVALUE: Greedily choose nodes that reduce the largest eigenvalue of $\mathbf{A}$ the most until the budget is reached.

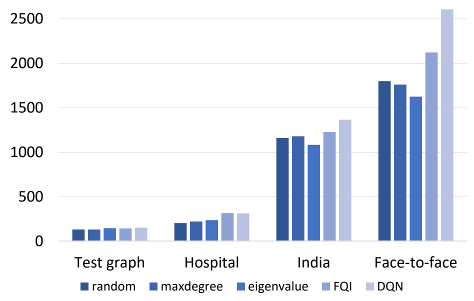

Fig. 9. Reward of different methods on different dataset over 10 times average.

Other then simulated graph, we evaluate the performance on the following realistic contact networks which is collected from diverse sources. The networks are carefully selected to have significant variation in size $|V|$, average degree d, average shortest path length ρ_L, assortativity ρ_D and epidemic threshold (spectral radius) $\frac{1}{\lambda_A^*}$ as table 2 shows.

(2a) **Test graph** we use the spatial preferential attachment model to generate a graph of 20 nodes. Such approach has the heavy-tailed degree distributions observed in many real networks.
(2b) **Hospital** [15]: A dense contact network collected in a university hospital to study the path of disease spread.
(2c) **India** [3]: A human contact network collected from a rural village in India where active screening with limited budget may take place.
(2d) **Face-to-face** [10]: A network describing face-to-face behavior during the exhibition INFECTIOUS: STAY AWAY in 2009 at the Science Gallery in Dublin, in which influenza might spread through close contact of individuals.

7.2 Experiment Result

Table 2. Properties of the contact network data sets.

Network	$\|V\|$	$\frac{1}{\lambda_A^*}$	d	ρ_L	ρ_D
Hospital [15]	75	0.027	15.19	1.60	-0.18
India [3]	202	0.095	3.43	3.11	0.02
Face-to-face [10]	410	0.042	6.74	3.63	0.23

In Figure 9, we can see that our implementation of FQI with simple nodewise feature embedding can outperform all the other baselines. Notice that the *maxdegree* baseline can also be treated as another simple nodewise feature embedding if we use the node degree as the only feature and with a simple Q function which directly returns the node degree as the predicted marginal improvement. This *maxdegree* baseline performs quite poorly as it ignores the belief, observation, and action, leading to a myopic action selection with poor performance. In our FQI implementation, we train a regression tree to select and extract important features to make the prediction, which allows us to deal with various features in a more elegant way.

Our second algorithm DQN with GCN feature embedding can further improve the solution quality. With the help of GCN, we do not need to hand-craft graph features of each node. GCN can automatically learn a good node features embedding and use them to make prediction. The DQN can also further improve the scalability and solution quality by maintaining a replay memory and continuously update the memory and train on it.

As presented, we can see that after encoding belief, observation, and the action, our approaches exceed all the baselines. By either manually extracting node related features or automatically maintaining the feature representation through GCN, FQI with regression tree and nodewise encoding (manually) and DQN with GCN (automatically) outperform other baselines.

Overall, as the graph size increases, we can also observe more improvement against the standard baselines. However, we are still facing the issues of scalability, where our implementations of FQI and DQN require a long training time compared to all other myopic baselines. The memory requirement of DQN is also another issue. We will leave the scalability issue as our future direction. Hopefully these reinforcement learning based approaches can be applied to larger networks and have a real impact to our society.

8 Conclusion

We implement reinforcement learning on an active screening model by encoding the high dimension action and state with large solution space into a low dimension, informative representation. This can be done either manually (FQI) or automatically (DQN). "Teaching" the regressor by efficiently encode information using domain knowledge helps improve the performance and reduce the amount of training needed. The future direction of this work is to scale our solution to even larger networks with millions of nodes and reach higher performance by overcoming the trade-off we made for scalability.

9 Appendix

9.1 Belief Update

First, we can encode $\mathbf{a}(t)$ into $\mathbf{b}(t)$ using our knowledge toward the model. From the model, we know that the node set in $\mathbf{o}(t)$ are resulted from infected nodes report with probability c. Using the Bayesian posterior probability, we can update the original belief vector to a intermediate belief vector $\bar{\mathbf{b}}(t)$ as have

$$\bar{b}_v(t) = \begin{cases} 1, & v \in o(t) \\ \frac{(1-c)b_v(t)}{(1-\mathbf{b}_v(t))+(1-c)\mathbf{b}_v(t)}, & otherwise \end{cases} \tag{4}$$

Next, we need to encode the action $a(t)$ and the fact that set $o(t) \cup a(t)$ are guaranteed to be cured in the next round. To accomplish this, we simply update the intermediate belief state $\bar{\mathbf{b}}(t)$ to the prediction of belief in the next round $\mathbf{b}(t+1)$ and use it as our state representation.

$$b_v(t+1) = \begin{cases} 0, & v \in o(t) \cup a(t), \\ p_v(1 - x_v) + x_v, & otherwise \end{cases} \tag{5}$$

where $p_v = 1 - \prod_{u \in \delta(v)}(1 - \alpha \bar{b}_u(t))$ and $\delta(v)$ represent the neighbor of v. This maintance and update our belief of the next time step $\mathbf{b}(t+1)$.

References

1. Anderson, R.M., May, R.M.: Infectious diseases of humans: dynamics and control. Oxford University Press (1992)
2. Bailey, N.T.: The mathematical theory of infectious diseases and its applications. Charles Griffin & Company Ltd (1975)
3. Banerjee, A., Chandrasekhar, A.G., Duflo, E., Jackson, M.O.: The diffusion of microfinance. Science **341** (2013)
4. Braxton, J., Davis, D.W., Emerson, B., Flagg, E.W., Grey, J., Grier, L., Harvey, A., Kidd, S., Kim, J., Kreisel, K., et al.: Sexually transmitted disease surveillance 2016. CDC (2017)
5. Cadman, D., Chambers, L., Feldman, W., Sackett, D.: Assessing the Effectiveness of Community Screening Programs. JAMA **251** (1984)
6. CDC: Tuberculosis: General information. MMWR. Recommendations and reports: Morbidity and mortality weekly report. (2011), `https://www.cdc.gov/tb/publications/factsheets/general/tb.pdf`
7. Eames, K.T., Keeling, M.J.: Contact tracing and disease control. Proc. R. Soc. Lond., B, Biol. Sci. **270** (2003)
8. Ernst, D., Geurts, P., Wehenkel, L.: Tree-based batch mode reinforcement learning. Journal of Machine Learning Research **6**(Apr), 503–556 (2005)
9. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63**(1), 3–42 (2006)
10. Isella, L., Stehlé, J., Barrat, A., Cattuto, C., Pinton, J.F., Van den Broeck, W.: What's in a crowd? analysis of face-to-face behavioral networks. J. Theor. Biol. **271** (2011)
11. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: ICLR-17. Toulon (2017)
12. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
13. Morris, C., Ritzert, M., Fey, M., Hamilton, W.L., Lenssen, J.E., Rattan, G., Grohe, M.: Weisfeiler and leman go neural: Higher-order graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33, pp. 4602–4609 (2019)
14. Putri, W.C., Muscatello, D.J., Stockwell, M.S., Newall, A.T.: Economic burden of seasonal influenza in the united states. Vaccine **36**(27) (2018)
15. Vanhems, P., et al.: Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. PloS one **8**, 73970 (2013)