Using Reinforcement Learning for Multi-Objective Cluster-Level Optimization of Non-Pharmaceutical Interventions for Infectious Disease

Xueqiao Peng peng.969@osu.edu
Jiaqi Xu xu.4015@osu.edu
Xi Chen chen.10183@osu.edu
Dinh Song An Nguyen nguyen.2687@osu.edu
Andrew Perrault perrault.17@osu.edu
The Ohio State University, Columbus, OH, USA

Abstract

In the early stages of an infectious disease crisis, non-pharmaceutical interventions (NPIs) such as quarantines and testing can play an important role. Optimizing the delivery of NPIs is challenging as they can impose substantial direct costs (e.g., test costs) and human impacts (e.g., quarantine of uninfected individuals) and can be especially difficult to target for infections that may spread pre- or asymptomatically. In addition, superspreading, a common characteristic of many infectious diseases, induces informational dependencies across a cluster (group of individuals exposed by the same seed case). We formulate NPI optimization as a partially observable Markov decision process (POMDP), which we aim to solve with reinforcement learning (RL). We find RL provides a promising technical foundation that even modern approaches struggle. We propose a novel RL approach that leverages a supervised learning decoder as well as permutation invariant, fixed-size observation representations. Through extensive experimentation and evaluation, we show that our optimized policy can outperform all benchmarks by up to 27%. Additionally, we show that the policies discovered by RL can be distilled into decision trees to simplify deployment while still achieving strong performance. We publicly release our code and RL environments at: https://github.com/XueqiaoPeng/Covid-RLSL

Keywords: reinforcement learning, machine learning, contact tracing, public health

1. Introduction

The COVID-19 pandemic has highlighted the crucial role of non-pharmaceutical interventions (NPIs) in effectively managing the spread of infectious diseases. Implementation of NPIs requires careful consideration of multiple objectives, including prevention of viral transmission and reduction of costs associated with quarantine measures. Contact tracing has been widely adopted and extensively studied in infectious disease crises, particularly the context of COVID-19 (Keeling et al., 2020; Wang et al., 2022; Lai et al., 2020; Kerr et al., 2021).

Nevertheless, optimizing NPIs within a cluster remains a computationally challenging problem in many settings. First, the action space is naturally combinatorially large because an action must be selected for each contact. Second, the problem is inherently multi-objective as interventions have costs associated with them. For example, sensing actions, such as testing, can provide valuable information, but require resources to deploy, and quarantining has human impacts. Third, inferring the probability that an individual is infectious can be difficult for infections that can be transmissible without symptoms. Finally, the constraints of deployment make it desirable that NPI policies can be executed without the need for computation.

In this work, we aim to develop a generic approach for cluster-level optimization of NPIs based on reinforcement learning (RL) (Sutton and Barto, 2018). We find that modern RL approaches fail to outperform naive baselines such as quarantining all contacts or quarantining symptomatic contacts. We augment RL in several ways. First, we observe that the high-

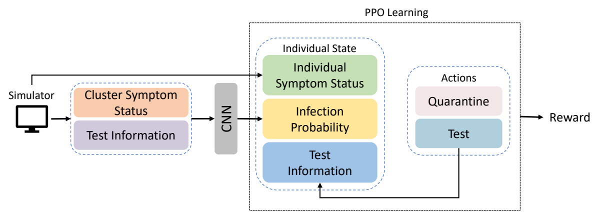


Figure 1: We combine a infection probability decoder that uses supervised learning with a reinforcement learning-based policy.

dimensional state and action spaces exhibit substantial permutation *equivariance*. For example, the order of the contacts does not matter—permuting the contacts and actions should produce the same result. This observation and the combinatorial action space motivate the development of an *egocentric* fixed-size state for each contact. Second, we find that learning to predict the probability of infectiousness via RL training is inefficient, and that this quantity has an important structural role in optimal policies in many settings. Thus, we develop a supervised learning module for the infectiousness inference task that leverages convolutional neural networks (Goodfellow et al., 2016), viewing the cluster state as if it were an image.

We summarize our approach in Fig. 1. Our vision is that, in an infectious disease crisis, an agent-based model simulating the infection would be developed based on observations and expert estimation (see, e.g., McAndrew et al. (2022)). This model could be used to evaluate and optimize policies, e.g., using the methods of this paper, and could be refined using contact tracing data from the field. We thus develop a minimally complex agent-based model for SARS-CoV-2 using published research from the early stages of the pandemic and use it as a testbed.

This paper makes the following contributions:

- We propose a novel RL approach for finding optimal contact tracing policies. Our approach combines RL with supervised learning and a permutation invariant, egocentric, state representation. The resulting agent can be trained and deployed simultaneously across all cluster sizes.

- To motivate the use of a supervised learning decoder, we show the existence of a simple, yet optimal, *threshold* policy for contact tracing in the setting where no sensing actions are available.

- We develop a simple branching process-based model for SARS-CoV-2 and compare our policies with baselines. We show that we achieve better rewards across a range of objective parameters, even when distilled into decision trees that can be widely distributed.

Related work We identify two main thrusts of work that optimize contact tracing and NPIs: network and branching process. Network models represent connections between individuals as edges in a (possibly dynamic) contact graph (Meirom et al., 2021; Kompella et al., 2020; Chen et al., 2023; Ou et al., 2020, 2021). These approaches can leverage network structure in their decisions, but make the strong assumption that the entire contact network is known at each time step. The closest existing approach to ours is RLGN (Meirom et al., 2021), which formulates the problem as a sequential decision-making task within a temporal graph process. In contrast, we take a cluster-based, tree-structured view of contagion (Kretzschmar et al., 2021; Meister and Kleinberg, 2023), but add agent-based temporal elements. This approach has the advantage of aligning more closely with the information available to decision makers in many practical settings and requires less detailed information to construct.

2. Problem Description

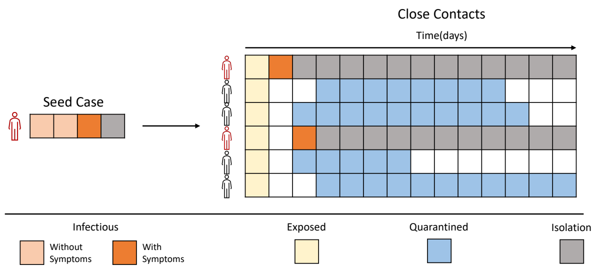


Figure 2: Cluster-based view of intervention planning.

We aim to create NPIs that operate on the cluster level. Fig. 2 shows a motivating example, taken from the cluster-level agent-based simulator we construct for SARS-CoV-2. A *seed case* exposes six *contacts* on the same day. Contacts 1 and 4 eventually become infected and show symptoms on day 2 and day 3, respectively. Contacts 2, 3, 5, and 6 never become

Table 1: Parameters of the SARS-CoV-2 cluster infection trajectory generator and test action model.

Parameter	Assumed value	Details and references
Incubation time	Log-normal: Log mean 1.57 days and log std 0.65 days	Mean: 5.94 days. Bi et al. (2020)
Duration of infectious period	7 days—2 days before and 5 days after onset if symptomatic	Bi et al. (2020)
Probability of infection	0.03	Perrault et al. (2020)
Probability that an infected individual shows symptoms	0.8	Buitrago-Garcia et al. (2020)
Probability of symptoms without infectiousness	0.01 per day	Hinch et al. (2020)
Probability of an asymptomatic infection	0.2	Buitrago-Garcia et al. (2020)
Probability seed case is highly transmissive	0.109	Perrault et al. (2020)
Infectiousness multiplier for highly transmissive individuals	24.4	Perrault et al. (2020)

Test parameters	TP = 0.71, FN = 0.01
	FP = 0.29, TN = 0.99
		Caulley et al. (2021)

Delays	Time to begin tracing a seed case = 3 days Test reporting delay = 1 day	Assumed—realism.
Cluster Size	Sample from uniform distribution on [2,40]	Assumed—we would like to find policies that perform well across cluster sizes.

infected. In this example, we must make binary action choice for each contact on each day: quarantine or not. The goal of the NPI policy is to identify and quarantine (isolate) contacts that are infected and not quarantine uninfected contacts, but the infectious state is not directly observable. The optimal policy depends on trade-offs between different objectives: failing to isolate infected contacts, quarantining uninfected contacts, and direct policy costs (e.g., of tests). Formally, we define the objective we aim to maximize as:

$$(-S_1 - \alpha_2 \times S_2 - \alpha_3 \times S_3)/N, \tag{1}$$

where

- S_1 is the count of *transmission days* where an infected individual is not isolated,
- S_2 is the count of days where a quarantined individual is not infected, and α_2 (which we assume is in $[0, 1]$) is the weight for this term,
- S_3 is the sum of the action costs (e.g., test cost) and α_3 is the weight for this term, and
- N, which is the number of contacts, normalizing the objectives to a score per contact.

In summary, the objective function seeks to minimize the number of transmission days, minimize the number of days of non-effective quarantine, and minimize the cost associated with actions. Intuitively, $1/\alpha_2$ is the number of quarantine days of an uninfectious contact we are willing to accept in exchange for one additional day of isolation of an infectious contact.

We remark that the objective value for any NPI policy can be evaluated for a cluster as long as we have an "infection trajectory" for each contact, a record of if and when they become infectious and if and when they exhibit symptoms (which is needed if the policy execution depends on symptom status). This is because these infection timing events are unaffected by the NPI actions we consider.

Formally, we define an *infection trajectory* for contact $n \in N$ as the infectiousness state $i_n^{(t)} \in I \in \{0, 1\}$ and the symptom observation $o_n^{(t)} \in \{0, 1\}$, the true infectiousness of contact and observable symptom state, respectively, of contact n on day t, for all $t \in [T]$ (where $[T] = \{1, 2, \dots, T\}$). We assume that each of these is binary for simplicity and that t is measured in days, but these are not requirements (e.g., $i_n^{(t)}$ could be a continuous viral load and S_1 could then represent risk-adjusted transmission days). We define an *cluster infection trajectory* as an infection trajectory for each contact in a cluster.

We require either a generator for cluster infection trajectories or a large library of them that we can sample from during training. As an example, we construct a generator for early SARS-CoV-2 using the parameters and sources shown in Tab. 1 (more details on our generator can be found in Appendix A). Trajectories run from $t = 1$ to $t = 30$, and $t = 3$ is the first time actions are allowed to be taken (modeling a contact tracing delay). Many of the required components of such a generator are distributions that are often estimated in the early stages of an outbreak. Components that are not known can be filled in conservatively or as a belief distribution (e.g., by aggregating expert opinion).

We allow for any set of NPI actions as long they can be simulated on any infection trajectory and their impact on S_1, S_2 and S_3 is defined. For example, a quarantine action, when applied to contact n on day t, causes S_1 to be not incremented if $i_n^{(t)} = 1$, and increments S_2 if $i_n^{(t)} = 0$. A more complex quarantine action may have a failure rate (an individual may not quarantine if directed), incur an additional financial cost (which would be added to S_3), or may include sensing component (see below). An action with a sensing component reveals information about the contact's infectiousness state $i_n^{(t)}$ according to some distribution, e.g., a test with a binary outcome according to the confusion matrix of test parameters in Table 1. More complex actions can combine sensing and quarantine, e.g., test and quarantine only if positive.

Our simulated environment has four actions: null action (S_3 cost of 0, no effect), quarantine (S_3 cost of 0), test (S_3 cost of 1, draw outcome according to Table 1), and test and quarantine only if results are positive (draw outcome according to Table 1, S_3 cost of 1).

3. Approach

The optimization problem from the previous section can be formulated as a partially observable Markov decision process (POMDP). However, solving this POMDP directly is intractable, even with modern RL techniques. Some hope arrives from the result that, under a simplified model that contains only quarantine actions, the POMDP can be solved optimally if the probability that an individual is infectious can be estimated—but this is itself a challenging problem due to the high dimensional observation space. Motivated by this observation, we formulate our solution approach: we use a convolutional neural network (CNN) to estimate the probability of infectiousness for each individual in a cluster, and this output, along with an egocentric state representation for each contact, serves as the state for the RL agent.

3.1. POMDP Formulation

We define a POMDP (Kaelbling et al., 1998) as $\langle S, A, R, P, \Omega, O, \gamma, S_0 \rangle$, where S and A are the state and action spaces, respectively, $R : S \times A \to \mathbb{R}$ is the reward function, $P : S \times A \to \Delta S$ is the transition function, Ω is the observation state, $O : S \times A \to \Delta\Omega$ is the observation probabilities, $\gamma \in [0, 1]$ is the discount factor, and $S_0 : \Delta S$ is the distribution of initial states.

We describe how to interpret the control problem of the previous section as a POMDP. The cluster infection trajectory and the current time t are contained in the state. The only aspect of the state that changes when an action is taken is the time t. As this is a POMDP, the state is not observable by the agent directly—instead, the agent has to rely on action-dependent observations. The observation emitted contains all of the information that is always available regardless of action (the time t, the symptom $o_n^{(t)}$ for each contact). Additionally, if an action with a sensing component is taken, it contains the sensing return (e.g., positive or negative, PCR cycle count). The action set is combinatorially structured—we select one action for each contact in the cluster. If we have N contacts, we have an action space of size $|A_p|^N$, where $|A_p|$ is the number of actions avaiable for each contact. The reward can be calculated for any policy from Obj. 1 for any cluster infection trajectory.

In principle, solving this POMDP results in the optimal control policy. In practice, solving it exactly is not possible due to the high computational complexity of the best-known algorithms. Nonetheless, we show that the POMDP has useful structure that can be exploited by RL methods.

3.2. Optimal Policy Without Sensing Actions

We consider a simplified POMDP where only quarantine actions are available. We show that, if the posterior probability of infection can be calculated exactly (i.e., the probability of infection of each contact given all observations so far), the optimal policy has a *threshold-type* form: if and only if the posterior probability of infection is above a threshold, we quarantine. We show this initially for a costless quarantine action with 100% efficiency as this is what we use in experiments (Thm. 1). We then generalize the result to any menu of non-sensing actions because the expected reward of each action can be exactly calculated given the posterior probability of infection (Thm. 2).

Let $o_{[N]}^{[t]} = \{o_n^{(t')} : 0 \leq t', n \in [N]\}$ represent all symptom observations for a cluster up to day t. Let $p_n^{(t)} = P\left(i_n^{(t)} = 1 \mid o_{[N]}^{[t]}\right)$ represent the posterior probability that contact n is infected given the symptom observations so far.

Theorem 1 *With a costless quarantine action that is always successful and a null action, the optimal policy is to quarantine only if* $p_n^{(t)} > \frac{\alpha_2}{1+\alpha_2}$.

Proof Because we have access to the exact posterior probability of infection, we can calculate the expected objective value for each action exactly:

$$\mathbb{E}[r] = \begin{cases} -\alpha_2 \cdot (1 - p_n^{(t)}) & \text{if quarantined.} \\ -p_n^{(t)} & \text{if not quarantined.} \end{cases} \tag{2}$$

We can then show that if $p_n^{(t)} > \frac{\alpha_2}{1+\alpha_2}$, the quarantine action has higher expected objective value. ■

We can use the above proof technique to derive the optimal policy for any menu of non-sensing actions. A useful generalization is when the quarantine action has a cost and a failure rate.

Theorem 2 *With a quarantine action with success rate* $0 \leq \beta \leq 1$ *and cost of 1, and a null action, the optimal policy is to quarantine only if* $p_n^{(t)} > \frac{\alpha_2 \cdot \beta + \alpha_3}{(1+\alpha_2) \cdot \beta}$.

These results highlight the importance of the posterior probability of infection. Next, we dedicate our attention to producing useful estimates of $p_n^{(t)}$.

3.3. Supervised Learning Decoder

The generator for or library of cluster infection trajectories provides us with a large number of $(o_{[N]}^{[t]}, i_{[N]}^{(t)})$ pairs (where $i_{[N]}^{(t)}$ is the infectiousness state for all contacts in a cluster). A natural question is whether we can produce useful estimates of $p_n^{(t)}$ from $o_{[N]}^{[t]}$ using a supervised learning approach. While it is possible for RL to produce strong policies without explicitly computing $p_n^{(t)}$, it is inefficiently positioned to do so because the information about $i_n^{(t)}$ must be inferred from the reward signal.

A key question for applying supervised learning is how to represent the observation space $o_{[N]}^{[t]}$. We have two desiderata. First, we would like the representation size to not vary with cluster size. We can also achieve this property in the RL agent, resulting in an agent that simultaneously be deployed across all cluster sizes, which makes both training and deployment simpler. Second, there is an advantage to using a representation that inherently accounts for the permutation equivariance that arises due to the ordering of individuals, i.e., if we permute the order of individuals in an observation, our supervised learning model would ideally predict $i_{[N]}^{(t)}$, but with the same permutation applied.

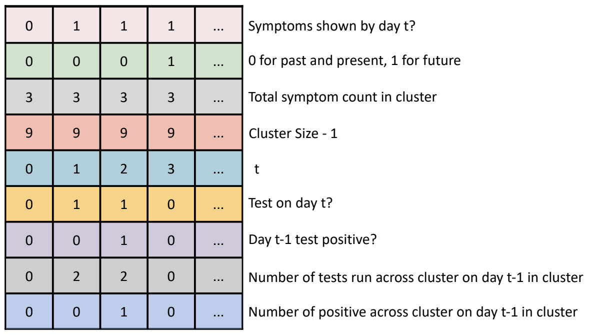
After testing several representations that satisfy these properties, we arrive at the $9 \times T$ matrix shown in Fig. 3 (recall T is the trajectory length). This is an egocentric representation of the observation—it is from the perspective of a particular contact and contains all information gathered so far. We train the supervised learning model f to produce output of dimension $[0,1]^T$, i.e., given $o_{[N]}^{[t]}$ for some $t \leq T$, predict $p_n^{(t')}$ for all $t' \in [T]$.

We show that this representation can achieve an AUC of 0.95 for the SARS-CoV-2 cluster infection trajectory generator if an appropriate architecture is selected. We experiment with a variety of supervised learning model architectures in Tab. 2 and find that convolutional neural networks (CNNs) are generally most effective. In single-layer CNN architectures, we find that larger 2D convolutions tend to achieve higher AUC, and that a single convolution layer followed by a linear layer performs just as well as deeper architectures—this setup of a (5, 2) 2D convolution followed by a linear layer is what we use in the experiments below.[1]

3.4. Reinforcement Learning

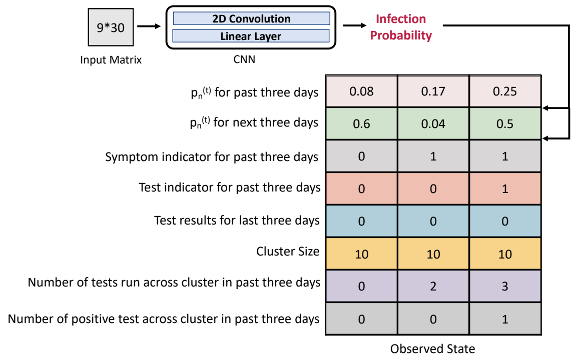
To make RL effective, we will develop a compact state representation that includes supervised learning outputs. As with supervised learning, we want the RL state representation to have the same size for all clusters and to naturally encode permutation invariance. In doing so, we can also reduce the action space size from combinatorial by factorizing across the contacts, i.e., training a single policy which is applied separately to each contact—this is without loss of performance in the setting without sensing actions if $p_n^{(t)}$

1. These experiments were performed on an earlier representation, which only had five rows. In the following sections, we use (9, 2) 2D convolution followed by a linear layer.



0	1	1	1	...	Symptoms shown by day t?
0	0	0	1	...	0 for past and present, 1 for future
3	3	3	3	...	Total symptom count in cluster
9	9	9	9	...	Cluster Size - 1
0	1	2	3	...	t
0	1	1	0	...	Test on day t?
0	0	1	0	...	Day t-1 test positive?
0	2	2	0	...	Number of tests run across cluster on day t-1 in cluster
0	0	1	0	...	Number of positive across cluster on day t-1 in cluster

Figure 3: The observation representation used for supervised learning, shown on a cluster of size 10 after observing the outcome of t=2.



$p_n^{(t)}$ for past three days	0.08	0.17	0.25
$p_n^{(t)}$ for next three days	0.6	0.04	0.5
Symptom indicator for past three days	0	1	1
Test indicator for past three days	0	0	1
Test results for last three days	0	0	0
Cluster Size	10	10	10
Number of tests run across cluster in past three days	0	2	3
Number of positive test across cluster in past three days	0	0	1

Figure 4: The supervised learning (CNN) output is used as input to the RL state which prioritizes immediately relevant information.

Table 2: We find that two-layer architectures using a 2D convolution followed by a linear layer achieve performance on par with larger models.

		Cluster size = 4	8	16	32
1 Layer	Conv1d (5,2)	0.798	0.807	0.823	0.830
	Conv1d (5,3)	0.814	0.830	0.835	0.839
	Conv2d (5,2)	0.800	0.814	0.827	0.830
	Conv2d (5,3)	0.832	0.820	0.838	0.840
	Conv2d (5,4)	0.858	0.849	0.843	0.859
	Conv2d (5,5)	0.864	0.895	0.893	0.893
2 Layer	Conv1d (5,2) Conv1d (1,2)	0.824	0.830	0.833	0.840
	Conv2d (5,3) Conv2d (1,3)	0.883	0.903	0.898	0.897
	Conv2d(5,2) Linear Layer	0.955	0.960	0.947	0.961
	Conv2d (5,3) Linear Layer	0.951	0.960	0.940	0.964
3 Layer	Conv1d (5,3) Conv1d (1,3) Linear Layer	0.958	0.957	0.950	0.961
4 Layer	Conv1d (4,3) Conv1d (2,3) Conv1d (1,3) Linear Layer	0.958	0.958	0.953	0.965
xgboost		0.763	0.732	0.804	0.770

is correct. The representation we use is a 9×3 matrix shown in Fig. 4. As with the supervised learning representation, it is egocentric and time-specific.

The following training procedure for the RL policy and supervised learning decoder is used. First, a fixed but stochastic *seed policy* generates 200 cluster infection trajectories and sensing actions, which are used to train the supervised learning decoder. These trajectories, along with the decoder outputs, are then used to train the RL policy. If performance is sufficient, terminate. If it is not (which happens when the current RL policy produces actions with a distribution that is too different from the seed policy), use the current RL policy to select sensing actions, continue training the decoder using these new actions, and then retrain the RL policy with the new supervised learning outputs. This process can be repeated any number of times, or the RL policy and the decoder can be trained in parallel.

For RL training, we use Proximal Policy Optimization (PPO) (Schulman et al., 2017). The RL policy is a multi-layer perceptron with two layers and 128 hidden units (a standard architecture for PPO). In experiments, for each of six different policy initializations (seed), train for 80000 environment interactions, and pick the best based on 100 evaluation runs. All training is performed on an Intel Xeon E5-2680 v4 with 28 cores and 128 GB of RAM (Center, 1987), and a single RL training run takes 3 hours on average.

Note that the optimal RL policy depends on α_2 and α_3 and thus a different policy must be trained for each different setting. However, the decoder depends on (α_2, α_3) only indirectly due to the sensing actions it sees in training. Thus, the same cluster infection trajectories and decoder can be reused across multiple RL training runs.

4. Experiments

We compare different control policies in our SARS-CoV-2 cluster infection trajectory generator to evaluate our policy search procedure. For α_2, we use three values of $0.05, 0.1$ and 0.2. For α_3, we use values of $0.01, 0.1$, and 0.2.

Table 3: Obj. 1 multiplied by 100 (higher is better). RLSL finds the best policy in all settings except $\alpha_2 = 0.05$ and $\alpha_3 = 0.2$, where RLSL, Threshold, Symptom-Based Quarantine and Always Quarantine are essentially tied—testing appears to provide no benefit here. The largest gaps between RLSL and others occur when α_2 is large and α_3 is small.

	$\alpha_2 = 0.05$ $\alpha_3 = 0.01$	$\alpha_2 = 0.05$ $\alpha_3 = 0.1$	$\alpha_2 = 0.05$ $\alpha_3 = 0.2$	$\alpha_2 = 0.1$ $\alpha_3 = 0.01$	$\alpha_2 = 0.1$ $\alpha_3 = 0.1$	$\alpha_2 = 0.1$ $\alpha_3 = 0.2$	$\alpha_2 = 0.2$ $\alpha_3 = 0.01$	$\alpha_2 = 0.2$ $\alpha_3 = 0.1$	$\alpha_2 = 0.2$ $\alpha_3 = 0.2$
RLSL	$\mathbf{-88.92 \pm 1.68}$	$\mathbf{-105.20 \pm 1.03}$	-112.78 ± 2.23	$\mathbf{-94.58 \pm 2.34}$	$\mathbf{-109.52 \pm 2.54}$	$\mathbf{-116.37 \pm 3.17}$	$\mathbf{-102.21 \pm 2.39}$	$\mathbf{-124.88 \pm 2.82}$	$\mathbf{-133.37 \pm 3.35}$
Threshold	-107.36 ± 7.67	-107.36 ± 7.67	$\mathbf{-107.36 \pm 7.67}$	-130.50 ± 4.76	-130.50 ± 4.76	-130.50 ± 4.76	-157.03 ± 4.37	-157.03 ± 4.37	-157.03 ± 4.37
Symptom-Based Quarantine	-113.58 ± 2.85	-113.58 ± 2.85	-113.58 ± 2.85	-134.32 ± 6.88	-134.32 ± 6.88	-134.32 ± 6.88	-158.42 ± 10.24	-158.42 ± 10.24	-158.42 ± 10.24
14-Day Quarantine	-141.50 ± 13.80	-141.50 ± 13.80	-141.50 ± 13.80	-175.50 ± 13.34	-175.50 ± 13.34	-175.50 ± 13.34	-276.50 ± 11.70	-276.50 ± 11.70	-276.50 ± 11.70
CDC 12/20	-203.67 ± 5.56	-213.40 ± 6.54	-228.80 ± 5.28	-235.37 ± 5.36	-246.30 ± 5.04	-277.23 ± 5.88	-310.77 ± 4.18	-330.67 ± 5.70	-344.10 ± 6.56
Always Quarantine	-110.50 ± 2.95	-110.50 ± 2.95	-110.50 ± 2.95	-215.44 ± 2.66	-215.44 ± 2.66	-215.44 ± 2.66	-425.91 ± 1.76	-425.91 ± 1.76	-425.91 ± 1.76
No Quarantine	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01	-249.33 ± 7.01

Table 4: S_1, S_2 and S_3 per contact across different cluster sizes (lower is better and - indicates 0), where RLSL and Threshold are set to $\alpha_2 = \alpha_3 = 0.1$. RLSL tests slightly more than CDC 12/20 (1.55 vs. 0.918 tests per contact) to dramatically decrease S_1 and S_2.

	S_1	S_2	S_3
RLSL	0.459 ± 0.017	4.195 ± 0.148	1.553 ± 0.035
RLSL ($N = 4$)	0.391 ± 0.017	3.907 ± 0.013	1.430 ± 0.007
RLSL ($N = 8$)	0.419 ± 0.014	4.600 ± 0.032	1.689 ± 0.010
RLSL ($N = 16$)	0.524 ± 0.071	3.821 ± 0.049	2.017 ± 0.056
RLSL ($N = 32$)	0.541 ± 0.043	4.031 ± 0.029	2.043 ± 0.019
Threshold	1.198 ± 0.040	1.751 ± 0.013	-
Threshold ($N = 4$)	0.973 ± 0.059	1.976 ± 0.057	-
Threshold ($N = 8$)	1.009 ± 0.044	1.659 ± 0.036	-
Threshold ($N = 16$)	1.321 ± 0.048	1.617 ± 0.018	-
Threshold ($N = 32$)	1.438 ± 0.056	1.762 ± 0.028	-
Symptom-Based Quarantine	1.413 ± 0.036	0.228 ± 0.005	-
14-Day Quarantine	0.753 ± 0.007	10.274 ± 0.040	-
CDC 12/20	1.273 ± 0.068	7.334 ± 0.084	0.918 ± 0.004
Always Quarantine	-	21.788 ± 0.085	-
No Quarantine	2.481 ± 0.046	-	-

4.1. Comparison Policies

The policies introduced by this paper are: **Threshold** is the threshold-type policy suggested in Sec. 3.2 (which does not use sensing actions); and **RLSL**, our primary contribution, combining RL with a supervised learning decoder.

We compare to several benchmark policies. **Symptom-Based Quarantine** quarantines if an individual exhibits symptoms on the day before the observed day and otherwise does not. **14-day Quarantine** quarantines individuals from the initial day they exhibit symptoms until either 14 days have passed or until they no longer exhibit symptoms, whichever is later. **CDC 12/20** (Appendix D) is a complex policy based on late 2020 (CDC) guidelines (CDC, 2020). It quarantines symptomatic contacts for 10 days. Asymptomatic contacts are tested on day 5 and released on day 8 if the test is negative and they have no symptoms. If the test is positive, they are quarantined for 14 days since the exposure. **Always Quarantine** always performs the quarantine action. **No Quarantine** always performs the null action.

Our experimental results report the average objective value and standard error taken over 30 random clusters.

4.2. Analysis

We first show the performance among the all the policies in Tab. 3. We find that RLSL is able to find the strongest policy in all settings except $\alpha_2 = 0.05$ and $\alpha_3 = 0.2$, where RLSL, Threshold, Symptom-Based Quarantine and Always Quarantine are all competitive (with perhaps an edge to Threshold). Threshold is the second strongest performer in all other settings. RLSL can achieve large improvements over the benchmarks of up to 35%. We see improvements across all settings, but they are largest when α_2 is large and α_3 is small, i.e., where tests can be leveraged and the decision to quarantine or not is challenging.

The best benchmark policy is Symptom-Based Quarantine except when $\alpha_2 = 0.05$, where Always Quarantine is slightly better. Symptom-Based Quarantine is often competitive with Threshold, despite the presence of extensive asymptomatic and presymptomatic transmission, as well as symptoms without infection, in the generator.

We report objective values broken out by component and by cluster size as measured per contact, where $\alpha_2 = \alpha_3 = 0.1$ is used to train RLSL and set the parameters (Tab. 4). Here we can intuitively grasp the effects of the different policies. 14-

Table 5: RLSL and Threshold always achieve dramatically higher objective values than RL Only, which has no supervised learning component. For $\alpha_2 = 0.05$ and $\alpha_3 \in \{0.1, 0.2\}$, RLSL with no sensing action scores slightly better than standard RLSL. In many settings, we are able to find decision tree policies that perform similarly to the RLSL or Threshold policies, which are much more complex.

	$\alpha_2 = 0.05$	$\alpha_2 = 0.05$	$\alpha_2 = 0.05$	$\alpha_2 = 0.1$	$\alpha_2 = 0.1$	$\alpha_2 = 0.1$	$\alpha_2 = 0.2$	$\alpha_2 = 0.2$	$\alpha_2 = 0.2$
	$\alpha_3 = 0.01$	$\alpha_3 = 0.1$	$\alpha_3 = 0.2$	$\alpha_3 = 0.01$	$\alpha_3 = 0.1$	$\alpha_3 = 0.2$	$\alpha_3 = 0.01$	$\alpha_3 = 0.1$	$\alpha_3 = 0.2$
RLSL	$\mathbf{-88.92 \pm 1.68}$	-105.20 ± 1.03	-112.78 ± 2.23	-94.58 ± 2.34	$\mathbf{-109.52 \pm 2.54}$	$\mathbf{-116.37 \pm 3.17}$	$\mathbf{-102.21 \pm 2.39}$	$\mathbf{-124.88 \pm 2.82}$	$\mathbf{-133.37 \pm 3.35}$
RLSL (always test)	-94.47 ± 0.97	-291.50 ± 3.97	-518.10 ± 3.38	-96.48 ± 3.14	-292.30 ± 2.86	-531.90 ± 4.48	-107.90 ± 4.23	-320.80 ± 4.70	-531.50 ± 3.97
RLSL (never test)	-98.67 ± 2.33	$\mathbf{-98.67 \pm 2.33}$	$\mathbf{-98.67 \pm 2.33}$	-128.25 ± 2.50	-128.25 ± 2.50	-128.25 ± 2.50	-148.41 ± 6.44	-148.41 ± 6.44	-148.41 ± 6.44
RL Only	-150.50 ± 5.19	-211.80 ± 7.81	-228.40 ± 6.84	-178.90 ± 7.63	-244.80 ± 8.06	-320.90 ± 9.70	-202.90 ± 13.32	-294.20 ± 13.34	-333.20 ± 7.55
Threshold	-107.36 ± 7.67	-107.36 ± 7.67	-107.36 ± 7.67	-130.50 ± 4.76	-130.50 ± 4.76	-130.50 ± 4.76	-157.03 ± 4.37	-157.03 ± 4.37	-157.03 ± 4.37
Decision Tree	-103.15 ± 4.20	-104.96 ± 3.12	$\mathbf{-97.46 \pm 3.66}$	$\mathbf{-91.10 \pm 1.30}$	-121.90 ± 2.66	-143.25 ± 3.80	-127.43 ± 2.52	-131.53 ± 2.89	-161.23 ± 3.03

Day Quarantine, CDC 12/20, and Always Quarantine quarantine widely, resulting in $S_2 \approx 10.3, 7.3, 21.8$ days of quarantine without infection per contact (respectively) and achieving $S_1 \approx 0.75, 1.27, 0.0$ as a result. Symptom-based quarantine takes a different approach, preventing only 57% of transmission days, but incurring minimal costs to do so. The best point in this trade-off space, which we visualize in Appendix C, depends on α_2 and α_3. RLSL uses about 50% more tests than CDC 12/20, but reduces S_1 to about 40% lower than 14-day quarantine with 60% less S_2 cost. Threshold is simply more efficient than non-testing competitors at the trade-off between S_1 and S_2 by allowing S_1 to be larger to vastly reduce S_2.

In an ablation study (Tab. 5), we gain a more detailed view into the operation of the RLSL policy. We see that the introduction of the SL outputs to the RL state results in vastly improved performance in all tested scenarios compared to RL Only, which uses the state representation of Fig. 4 without the first two rows. RL Only performs worse than Symptom-Based Quarantine in all settings. RLSL (never test) and the decision tree policy (described below) sometimes outperform RLSL, indicating that the training procedure could still be improved.

Interpretable Policy In contrast to the benchmarks, both RLSL and Threshold require neural network outputs, i.e., computation, to run. We experiment with a procedure to convert RLSL's policy into a decision tree that can be distributed on paper. We use nine interpretable features: days since exposure, days since positive test, days since symptom, yesterday's test result (0 if no test), whether tested yesterday, the number of symptomatic contacts in the cluster today, the number of positive tests in the cluster so far, and the cluster size. Using these features, we train a decision tree to predict RLSL's action. We consider five types of actions: (1) quarantine and no test, (2) quarantine and test, (3) no quarantine and test, (4) no quarantine and no test, (5) test and, if positive, quarantine; otherwise, no quarantine. The results are shown in the last row of Table 5. In three cases, the decision tree policy is at least as strong as policy produced by RLSL. We show two of the decision tree policies and provide a detailed description of the training procedure in Appendix E. We believe that these policies can be further improved (see Discussion).

5. Discussion and Conclusion

This work aims to develop a generic multi-objective optimization approach for cluster-level optimization of NPIs. We formulate this problem as a POMDP that we solve with RL, leveraging a supervised learning decoder, a permutation equivariant state representation, and a factorized action space. We demonstrate the potential of our approach—in a simple agent-based model of SARS-CoV-2, we can achieve substantially higher objective values than baseline policies. Our optimized policy can outperform all benchmarks by up to 27%. Moreover, the developed policies exhibit applicability across various cluster sizes and can be trained on consumer hardware, the fact that these policies can be implemented on consumer-grade hardware enhances their practicality and scalability, making them accessible for broader real-world application. In addition, our approach has shown promise in formulating strong and interpretable policies across multiple settings. This aspect is particularly important as it contributes to the transparency and understandability of the policies, which are crucial factors in public health interventions.

Our agent-based model represents a classic probabilistic framework for simulating disease dynamics, which can be applied to other epidemics. It is built using key disease parameters derived from various sources during a crisis, incorporating the inherent uncertainties in these estimates. In the early stages of a crisis, we emphasize the importance of focusing on superspreading dynamics, given their significant impact on the effectiveness of interventions, as demonstrated in our findings. Utilizing this model, we can create an environment based on a branching process, which is then optimized using the approach outlined in this paper.

Our approach combines RL and SL techniques. RL is a powerful optimization technique, but it has some drawbacks. One significant limitation is its inherent difficulty in exploit problem structure. In this setting, the underlying POMDP has substantial structure in the belief state that can be exploited to greatly simplify the learning task. We extract this information using a combination of manual insight and brute force supervised learning. It is an open question as to whether RL techniques can learn to discover such structure through experimentation. Another challenge with RL is its well-known instability during the training phase (Dulac-Arnold et al., 2021). We attempt to reduce this instability through using multiple initializations, but we still see evidence of it in the $\alpha_2 = 0.05$, $\alpha \geq 0.1$ settings, where reducing the action space produces higher objective values. Despite these challenges, we believe the advantages of using RL, especially in terms of its capability to provide high-quality solutions for complex optimization problems, outweigh its limitations.

While no existing work uses the same modeling framework or policy search space as ours, in some cases, we can compare our results. The model of Perrault et al. (2020) is most similar to ours, and the risk-based quarantine (RBQ) policies they evaluate can be compared to Threshold and RLSL, but due to different assumptions, the amount of reduction in transmission they achieve relative to the status quo is much less. This is because they assume that individuals self-isolate even in the absence of an intervention and that some individuals drop out of quarantine. Threshold can be viewed as a policy that generalizes the RBQ approach they suggest, in that Threshold generates an infinite family of optimal risk-based families for different risk tolerance levels. However, Threshold's policies are less interpretable than RBQ.

Kucharski et al. (2020) provides another point to compare the effectiveness of contact tracing. In their setting, combining self-isolation, household quarantine, and comprehensive manual contact tracing of all contacts resulted in a 64% reduction in disease transmission, which is equivalent to S_1. In our setting, we find a reduction of 69.95% for two-week quarantine, suggesting that interventions have a comparable impact in our (much simpler) model for COVID-19 transmission.

The primary emphasis of this paper was producing strong cluster-level policies, but we believe this setting may produce challenges in interpretable RL. Most existing work on interpretable RL is focused on the problem of producing an inherently interpretable policy (e.g., a decision tree) that operates on a fixed feature space (see, e.g., Milani et al. (2022)). In this setting, we believe it is very challenging to train any RL policy from raw features. From our experience, the current method of learning interpretable decision-tree policies, while obtaining great performance, assumes an unconstrained decision-tree space, resulting in less interpretable models. For some settings, the performance of the decision tree is poor and unstable when the depth of the tree is small. However, we find that some strong interpretable policies that operate on an interpretable feature set do exist and further study of how to produce them may be worthwhile.

Looking ahead, an area for further investigation is the development of more effective policies for the coordination of limited resources across clusters and across time, especially during the early stages of an outbreak. We assume that the objective weights α are given. In practice, it is non-trivial to determine these weights. α_2 depends on myriad factors, such as outbreak stage (can containment be achieved?), public willingness, and may vary per cluster (e.g., a cluster in a nursing home may have a lower α_2 because of the high risk to contacts of contacts). α_3 combines considerations of availability (tests, vaccines) and cost (labor) and may be split into two weights, and both of which may vary substantially with time.

References

Qifang Bi, Yongsheng Wu, Shujiang Mei, Chenfei Ye, Xuan Zou, Zhen Zhang, Xiaojian Liu, Lan Wei, Shaun A Truelove, Tong Zhang, et al. Epidemiology and transmission of COVID-19 in 391 cases and 1286 of their close contacts in Shenzhen, China: a retrospective cohort study. *The Lancet infectious diseases*, 20(8):911–919, 2020.

Diana Buitrago-Garcia, Dianne Egli-Gany, Michel J Counotte, Stefanie Hossmann, Hira Imeri, Aziz Mert Ipekci, Georgia Salanti, and Nicola Low. Occurrence and transmission potential of asymptomatic and presymptomatic SARS-CoV-2 infections: A living systematic review and meta-analysis. *PLoS medicine*, 17(9):e1003346, 2020.

Lisa Caulley, Martin Corsten, Libni Eapen, Jonathan Whelan, Jonathan B Angel, Kym Antonation, Nathalie Bastien, Guillaume Poliquin, and Stephanie Johnson-Obaseki. Salivary detection of COVID-19. *Annals of internal medicine*, 174(1): 131–133, 2021.

CDC. COVID-19: When to quarantine — CDC. https://web.archive.org/web/20201231011236/https://www.cdc.gov/coronavirus/2019-ncov/if-you-are-sick/quarantine.html, 2020.

Ohio Supercomputer Center. Ohio supercomputer center, 1987. URL http://osc.edu/ark:/19495/f5s1ph73.

Xingran Chen, Hesam Nikpey, Jungyeol Kim, Saswati Sarkar, and Shirin Saeedi-Bidokhti. Containing a spread through sequential learning: to exploit or to explore? *arXiv preprint arXiv:2303.00141*, 2023.

Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

Robert Hinch, Will Probert, Anel Nurtay, Michelle Kendall, Chris Wymant, Matthew Hall, Katrina Lythgoe, Ana Bulas Cruz, Lele Zhao, Andrea Stewart, et al. Effective configurations of a digital contact tracing app: a report to NHSX. *Retrieved July*, 23:2020, 2020.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Matt J Keeling, T Deirdre Hollingsworth, and Jonathan M Read. Efficacy of contact tracing for the containment of the 2019 novel coronavirus (COVID-19). *J Epidemiol Community Health*, 74 (10):861–866, 2020.

Cliff C Kerr, Robyn M Stuart, Dina Mistry, Romesh G Abeysuriya, Katherine Rosenfeld, Gregory R Hart, Rafael C Núñez, Jamie A Cohen, Prashanth Selvaraj, Brittany Hagedorn, et al. Covasim: an agent-based model of COVID-19 dynamics and interventions. *PLOS Computational Biology*, 17(7):e1009149, 2021.

V Kompella, R Capobianco, S Jong, J Browne, S Fox, L Meyers, P Wurman, and P Stone. Reinforcement learning for optimization of COVID-19 mitigation policies. In *2020 AAAI Fall Symposium on AI for Social Good, AI4SG 2020*, 2020.

Mirjam E Kretzschmar, Ganna Rozhnova, and Michiel Van Boven. Isolation and contact tracing can tip the scale to containment of COVID-19 in populations with social distancing. *Frontiers in Physics*, page 677, 2021.

Adam J Kucharski, Petra Klepac, Andrew JK Conlan, Stephen M Kissler, Maria L Tang, Hannah Fry, Julia R Gog, W John Edmunds, Jon C Emery, Graham Medley, et al. Effectiveness of isolation, testing, contact tracing, and physical distancing on reducing transmission of sars-cov-2 in different settings: a mathematical modelling study. *The Lancet Infectious Diseases*, 20(10):1151–1160, 2020.

Shengjie Lai, Nick W Ruktanonchai, Liangcai Zhou, Olivia Prosper, Wei Luo, Jessica R Floyd, Amy Wesolowski, Mauricio Santillana, Chi Zhang, Xiangjun Du, et al. Effect of non-pharmaceutical interventions to contain COVID-19 in China. *nature*, 585(7825):410–413, 2020.

Thomas McAndrew, Allison Codi, Juan Cambeiro, Tamay Besiroglu, David Braun, Eva Chen, Luis Enrique Urtubey De Cèsaris, and Damon Luk. Chimeric forecasting: combining probabilistic predictions from computational models and human judgment. *BMC Infectious Diseases*, 22(1):833, 2022.

Eli Meirom, Haggai Maron, Shie Mannor, and Gal Chechik. Controlling graph dynamics with reinforcement learning and graph neural networks. In *International Conference on Machine Learning*, pages 7565–7577. PMLR, 2021.

Michela Meister and Jon Kleinberg. Optimizing the order of actions in a model of contact tracing. *PNAS Nexus*, 2023.

Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. A survey of explainable reinforcement learning. *arXiv preprint arXiv:2202.08434*, 2022.

Han-Ching Ou, Arunesh Sinha, Sze-Chuan Suen, Andrew Perrault, Alpan Raval, and Milind Tambe. Who and when to screen: Multi-round active screening for network recurrent infectious diseases under uncertainty. In *Proceedings of 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Auckland New Zealand, 2020 May 9-13*. ACM, 2020.

Han-Ching Ou, Haipeng Chen, Shahin Jabbari, and Milind Tambe. Active screening for recurrent diseases: A reinforcement learning approach. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 992–1000, 2021.

Andrew Perrault, Marie Charpignon, Jonathan Gruber, Milind Tambe, and Maimuna S Majumder. Designing efficient contact tracing through risk-based quarantining. *medRxiv*, pages 2020–11, 2020.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Xutong Wang, Zhanwei Du, Emily James, Spencer J Fox, Michael Lachmann, Lauren Ancel Meyers, and Darlene Bhavnani. The effectiveness of COVID-19 testing and contact tracing in a US city. *Proceedings of the National Academy of Sciences*, 119(34): e2200652119, 2022.

Appendix A. Details of Cluster Infection Trajectory Generator

We simulate the environment as follows:

```
Algorithm 1 Infection Simulation

  1. Generate seed case.
     CASE = GENERATECASE()

  2. Generate close contacts.
     CONTACTS = GENERATECONTACTS(CASE)
```

A.1. Generate Seed Case

The first step of the simulation is creating the seed case for the cluster, an individual who is confirmed as COVID-19 positive. These individuals are presumed to have experienced symptoms, followed by self-isolation, and subsequent confirmation of their positive test results. Prior to isolating themselves, each seed case is assumed to have come into contact with close contacts, potentially exposing them to the SARS-CoV-2 virus. We consider the circumstance that seed case is highly transmissive—with some probability, i.e., it is much more likely to transmit to its contacts. We use a probability of 0.109 and infectiousness multiplier of 24.4, values determined by Perrault et al. (2020) to produce transmission heterogeneity similar to that observed in SARS-CoV-2.

A.2. Generate Close Contact

We initiate the cluster size by sampling from a uniform distribution across $[2, 40]$, as we aim to produce a policy that is strong as possible for all cluster sizes. Subsequently, we must determine which contacts are exposed to the virus and whether they become infected. If the initial case is not highly contagious, the probability of infection among close contacts is 0.03. However, if the seed case is highly transmissive, the infectiousness rate increases by a factor of 24.4.

Once a contact is infected, there is an 0.8 probability of developing symptoms. Independently, there is a probability of 0.01 per day of displaying symptoms independently of infection status. We obtain the time from exposure to symptoms from a log-normal distribution with a log mean of 1.57 days and a log standard deviation of 0.65 days. Similarly, the time to exhibit symptoms is drawn from a log-normal distribution with a log mean of 2.70 days and a log standard deviation of 0.15 days, as described in Bi et al. (2020).

```
Algorithm 1: GENERATECONTACTS(CASE)
Input: Seed case: i
Output: Close Contact j and its symptom and
        status of infection
Initialize contacts[] from contact distribution
if i is highly transmissive then
    $p_{\text{infected}} = p_{\text{infected}} * 24.4$
end
for j in contacts[] do
    if Bernoulli($p_{\text{infected}}$) then
        j.infected = 1
        if Bernoulli($p_{\text{symptomatic}}$)=1 then
            j.symp_start = Lognormal(1.57, 0.65)
            j.symp_duration = Lognormal(2.70,
             0.15)
            j.inf_duration =Lognormal(6.67, 2)
            j.symp_day = [j.symp_start,
             j.symp_start+j.symp_duration]
            j.inf_day = [j.symp_start - 2,
             j.symp_start+j.inf_duration]
        else
            j.inf_start = Lognormal(1.57, 0.65)
            j.inf_duration =Lognormal(6.67, 2)
            j.inf_day = [j.inf_start - 2,
             j.symp_start+j.inf_duration]
        end
    symptomatic_not_infected_day
     =Binomial($p_{\text{symptomatic\_not\_infected}}$, $T$)
    for day in $T$ do
        if symptomatic_not_infected_day[day] = 1
         then
            j.sym_day[day] = 1
    end
end
return contacts
```

Appendix B. Experimental Settings

In this section, we will explain the detail of our experimental settings.

B.1. Supervised Learning Model

In the initial supervised learning model, we commence by running the simulator 200 times to generate the input data and corresponding labels. We perform Z-score normalization on the input data. The model is then trained using Adam optimizer with learning rate = 0.01 for a total of 200 epochs.

B.2. RL Model

Our process begins by training the RL model with the original Supervised Learning (SL) model for 50 epochs, gathering new trajectories in the process. Because our model considers that the actions taken by one agent during testing affect the features of other nearby contacts in the cluster, we subsequently retrain the SL model. Following this, we employ the updated SL model to train our RL model using the Proximal Policy Optimization (PPO) method (Schulman et al., 2017). In our experimental setup, we conduct RL training for a total of 200 epochs, with each epoch consisting of 800,000 steps.

Appendix C. Comparison Across α_2

We consider the $\alpha_2 \in [0, 0.7]$ and plot the value of Obj. 1 of all baselines in Fig. 5. For CDC 12/20 policy, we use the setting of $\alpha_3 = 0.01$. For each α_2, we train each policies for 100 epochs and test 30 times to get the average value. As observed, with the increase of α_2, the threshold policy gradually converges to -205. Symptom-based quarantine experiences a marginal reduction. Notably, when α_2 reaches 0.55, both No Quarantine and Symptom-based quarantine exhibit equivalent objectives. The most substantial decrease occurs in the Always Quarantine policy, closely followed by the 14-day quarantine policy. This is primarily due to the fact that in these two policies, the value of S_2 is significantly higher compared to other policies. CDC 12/20 policy's performance is slightly better than 14-day quarantine policy when α_2 approximately larger than 0.45.

Appendix D. CDC 12/20 Policy

We have translated the quarantine guidelines provided by the Centers for Disease Control and Prevention (CDC) as of December 10, 2020 CDC (2020). These guidelines recommend quarantining individuals who have had close contact with a confirmed COVID-19 case. There are two options available for ending the quarantine period.

- After day 10 without testing.

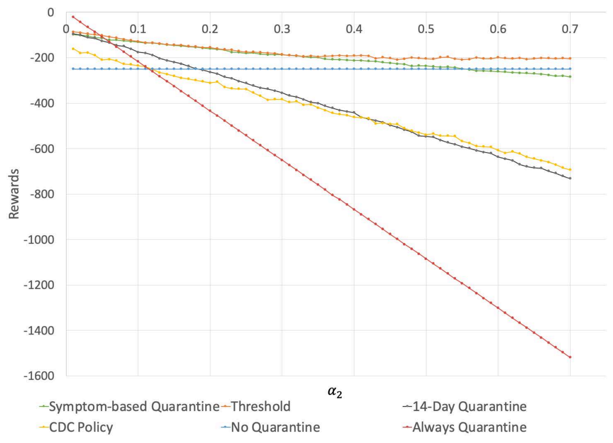


Figure 5: Comparison of benchmarks and Threshold across α_2 values.

- After day 7 after receiving a negative test result (test must occur on day 5 or later)

We condense this policy into a decision tree, as illustrated in Fig. 6.

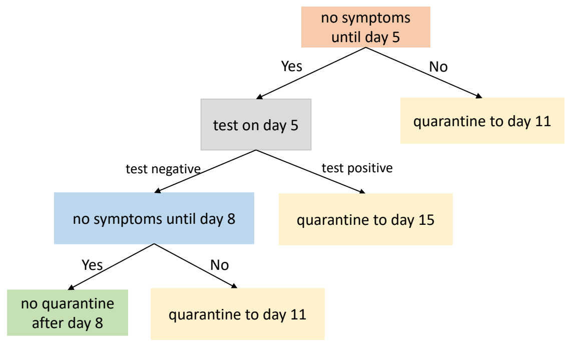


Figure 6: The decision tree for CDC 12/20 Policy.

Appendix E. Interpretable Policy

In this section, we present two decision trees distilled from our model, as depicted in Figs. 7 and 8. We first test RLSL policy for 100 epochs and collect the dataset, in which there are nine features and the action taken by agent as the label. The features are days since exposure, days since positive test, days since symptom, yesterday's test result (0 if no test), whether tested yesterday, the number of symptomatic contacts in the cluster today, the number of positive tests in the cluster so far, and the cluster size.We consider five types of actions: (1) quarantine and no test, (2) quarantine and test, (3) no quarantine and test, (4) no quarantine and no test, (5) test and, if positive, quarantine; otherwise, no quarantine. We use Gini inpurity to divide the decision trees. In our decision tree, left branch means yes and right branch means no. To get better performance, we try different depths for the decision tree. For $\alpha_2 = 0.05$ and $\alpha_2 = 0.1$, we set depth = 4, and for $\alpha_2 = 0.2$, the depth of the tree is 10. We also try 10 runs for each setting and select the best from among all the decision trees we generate.

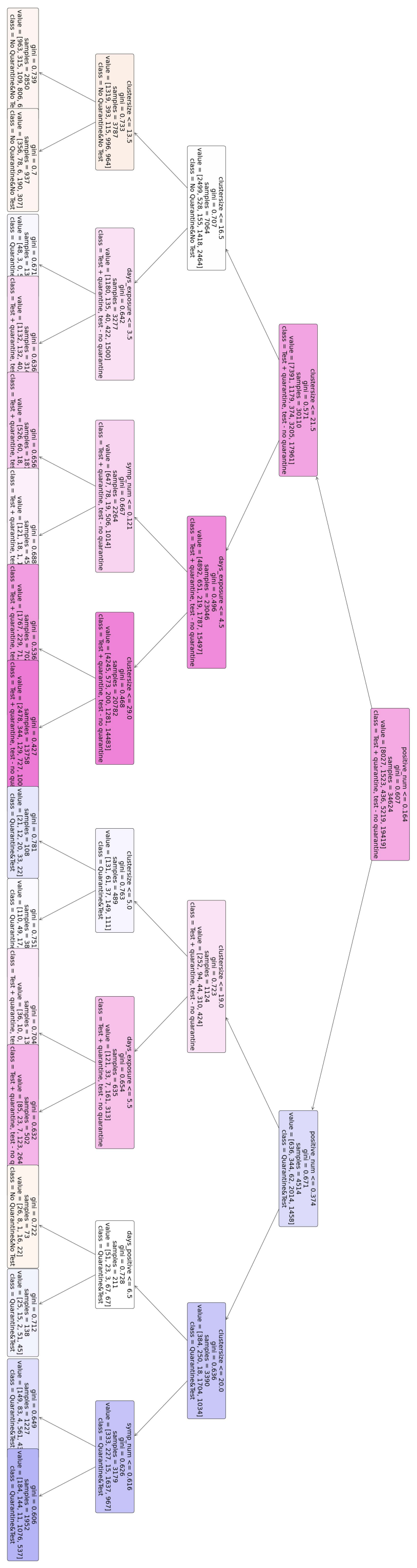


Figure 7: The decision tree for RLSL model with $\alpha_2 = 0.1$ and $\alpha_3 = 0.01$

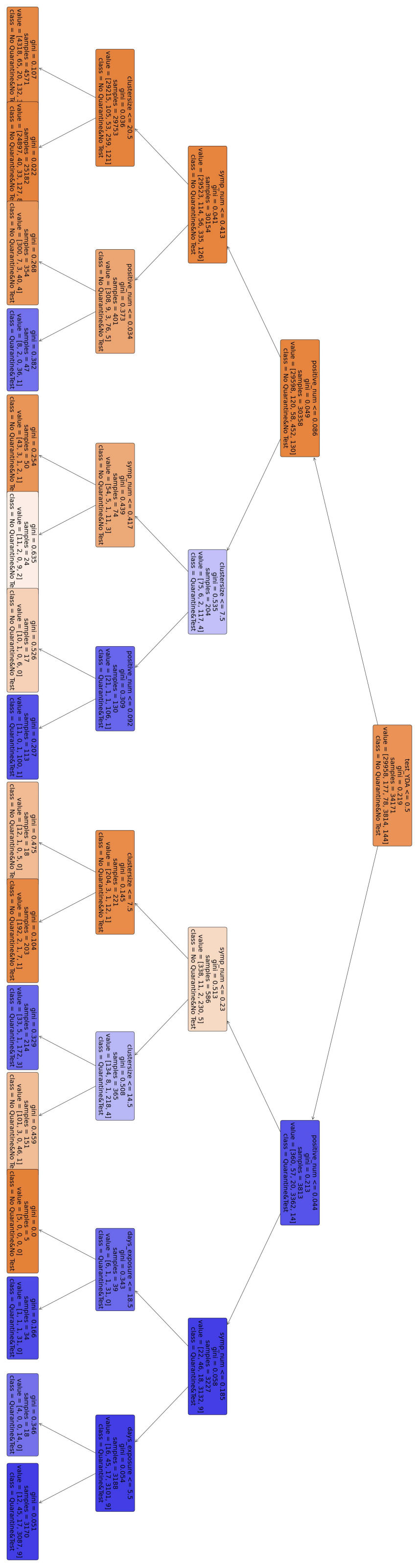


Figure 8: The decision tree for RLSL model with $\alpha_2 = 0.1$ and $\alpha_3 = 0.1$