**Optimal Control Policy on COVID-19: An Empirical Study on Lockdown and Travel Restriction Measures using Reinforcement Learning**

Kailiang Liu
Shenzhen Middle School, Nigang West Road No.1068, LuoHu District, Shenzhen, Guangdong Province, China; 1535001171@qq.com

**ABSTRACT:** Starting at the end of January 2020, the global outbreak of COVID-19 has changed the lifestyles of many people due to its rapid person-to-person transmission. To reduce the spread, countries have taken numerous approaches, among which testing, vaccination, sanitization, lockdowns, and construction of quarantine centres are commonly implemented. Nonetheless, all these measures are expensive in terms of resources and have an effect on controlling the spread of the virus, quality of life, resource consumption and economic growth. In this essay, a novel intelligent method based on reinforcement learning (RL) is developed to provide recommendations on the optimal level of control measures, such as travel restrictions and lockdown policies, with the purpose to prohibit the further spread of the pandemic. By specifying continuous action space and defining customized reward functions, we provide a new learning framework to study such types of policy control problems, along with discussions on their theoretical motivations. Experiments practiced with actual COVID-19 data demonstrates that the suggested deep reinforcement learning algorithm based on the Deep Deterministic Policy Gradient (DDPG) model performs better than alternative RL algorithms as well as actual control measures.

**KEYWORDS:** Reinforcement Learning; COVID-19 optimal control policy; Continuous action space; Deep deterministic policy gradient.

**Introduction**

Human civilization has witnessed several pandemics in the past, including plagues, leprosy, smallpox, tuberculosis, AIDS, cholera, and malaria, see Rajaei A *et al* (2019).¹ The historic records of pandemics suggest an implicit pattern that the frequency of disease outbreak increases as communication between civilizations grows, and it can therefore be reasoned that at this point of time when globalization is accelerating at an unprecedented rate, such contingencies are more likely to occur in the foreseeable future, as suggested in a WHO report in 2016.² Thus, it is quite imperative to review the lessons learned out of our experiences with the current COVID-19 global pandemic in order to build a resilient society with people prepared to combat the social, health, and economic impacts of pandemics. Preparedness is a key factor in mitigating pandemics. It encompasses inculcating awareness about the outbreak and fostering response strategies to avoid loss of life and socioeconomic havoc. While the emergence of a harmful microorganism with pandemic potential may be unpredictable, pandemics can be prevented. Preparedness includes technological readiness to identify pathogen identity, fostering drug discovery, and developing reliable theoretical models for prediction, analysis, and control of pandemics.

The COVID-19 pandemic has led to numerous difficult situations. For now, there hasn't been any 100% reliable immune vaccine to protect humans from infection. Some popular approaches, which are designed to control the spread of the infection in the current circumstances, are to enhance the frequency of testing as well as to broaden the coverage of vaccination in society. However, having all individuals tested and vaccinated is a time- and labour-consuming task. Besides being constrained by the available resources, governments also struggle in balancing the trade-offs between disease control and economic recovery. Another common measure is stricter sanitization in crowded locations, so as to disinfect public areas and thus control the spread of infection. However, this is also a resource expensive measure. It would be impossible to put in place sanitization at every corner and street of the country. On the other hand, lockdown approaches and quarantine plans are measures that rely comparatively less on extra consumption of medical, physical, or human resources. However, under current economic situations, it is unlikely to suspend various ongoing economic activities of the country and rely on the government to support and take care of its people, providing necessities for the mildly infectious people and keep available mission-critical ventilators for patients in severe conditions. Obviously, it is not possible to allow open and mask-free social gatherings, as the chance of spread is very high. There may exist other techniques that are not currently explored or discovered. Despite the ongoing techniques, including testing, sanitization and social distancing/lock down/quarantine, there still needs to be an optimal level for each parameter in order to protect people from the virus and support routine activities while minimizing impacts caused on quality of life and economy. This paper focuses on tackling this problem using a quantitative, model-based approach. Specifically, we will build a reinforcement learning agent to formulate an
optimal policy, which will recommend the appropriate degree of regional lockdown and travel restriction respectively.

- **Literature Review**

In this section, we will discuss the use of reinforcement learning algorithms in recommending COVID-19 related policies and introduce related research on this front. Reinforcement learning is a recent and popular research field, with the potential to make a resounding and laying impact on mankind's history. It is a robust framework for optimizing specific tasks autonomously and has attracted great attention in both research and industry. The creation of intelligent machines using reinforcement learning will likely drive the understanding of human intelligence to places we have never been before. Intelligence, under this circumstance, is addressed as the ability to conform and apply knowledge acquired through past experiences. Arguably, if we can understand how to reach the optimal decisions for each and every problem, we likely understand the algorithm that recommends the optimal decision outcomes. An optimal decision may not necessarily be equal to maximizing immediate returns at hand; it may well represent the capability of trading not only immediate rewards for long-term goals, but also possibly a good, certain future. A good example would be joining a startup where a lot of factors are still uncertain instead of staying in a stable position. Goals that require a much longer time to materialize and have uncertain long-term value are usually the most difficult to achieve. Different from supervised and unsupervised learning, reinforcement learning is self-renewative by continuously interacting with the environment and resetting its actions based on reward signals. It has been confirmed that it can deliver a record high performance in games such as GO or DOTA, and it can be of more practical significance when combined with deep neural networks.

It is a nontrivial task to determine the proper level of policy measure for each country, largely due to the different demographic, social and economic structure in different countries and areas. Due to the nature of the disease, it is challenging to trace the source of an infection case, thus adding more difficulty and uncertainty in policymaking. Therefore, the government is faced with partial knowledge and incomplete picture about the status of the disease, when trying to identify the optimal restrictive policy. Such knowledge will only grow through a long-term study of the virus and its clinical characteristics. In addition, there is often a lag between infection date and reporting date, depending on the recording process in each region. Insufficient medical equipment, testing kits and vaccination provisioning have led to almost 3 million deaths globally as of mid-April 2021, which is also likely conservative due to disparate, and even purposefully underestimated reporting and information sharing. This pandemic has led to disruptions in almost all aspects of lives and businesses in society.

Public health measures, including regional lockdowns and travel restrictions, have been analysed in the past to study their impact on controlling the spread of infectious disease. A typical research direction is the simulation of the spread of epidemics, which is a flexible framework to adjust to different disease structures and local intervention policies. This study has shown utility in simulating the spread of diseases and estimating the resulted impact. As for further targeted study on specific restrictive measure, Chinazzi et al. (2019) reviewed the effect of travel restrictions on the spread of COVID-19 and found that it is an effective measure in reducing the number of imported cases to other countries up to 77%, while only playing a limited role in controlling domestic spread. On this front, pre-emptive lockdown measures seem more promising in terms of slowing down and even reducing local transmission, as shown by Tian H et al. (2020). When only a limited proportion of the total population is infected, the lockdown measures could play an important role in reducing the number of death cases, as shown by the simulations by Khadilkar H et al. (2020). In their work, a reinforcement learning agent was constructed to learn an optimal policy that recommends quantitative lockdown measures, in consideration of health and economic impact. These interventions have proven to be effective at an aggregate level, but a more customized approach is still needed for individual countries and areas, with their unique demographic and socioeconomic characteristics being significant consequences of COVID-19 pandemic.

The complex dynamics among multiple factors caused by COVID-19 requires a robust and data-driven approach in order to formulate optimal preventive measures. To this end, Kwak GH et al. (2021) proposed the use of deep learning approaches to recommend global public health strategies for COVID-19 pandemic. In their work, Duelling Double Deep Q-Network (D3QN) was used to discover optimal lockdown and travel restriction policies for individual countries and regions. However, the recommended actions are in a discrete space with a fixed number of values, which are still quite limited in applying to all countries and regions. A continuous action space is thus recommended, so as to allow for a more flexible and targeted policy recommendation. In our paper, we extend the work of Kwak GH et al. (2021) and design a continuous action space for the reinforcement learning setting, where the agent is allowed to give an optimal action for lockdown and travel restriction, both varying within a fixed range and taking a numeric value, instead of a limited discrete space. In addition, we also deploy a more systematically advanced reinforcement learning algorithm named Deep Deterministic Policy Gradient (DDPG), whose superior performance is shown via experiments.

It is also worth noting that other research has studied COVID-19 pandemic using deep reinforcement learning from other angles. Regina P et al. (2021) proposed incorporating healthcare system parameters besides disease related characteristics to train a control model based on its transmission dynamics. Mahdavi M et al. (2021) explored COVID-19 related mortality risks using common machine learning algorithms such as Support Vector Machine and found that a small set of non-invasive features are predictive, indicating the potential use of these features in predicting intervention subject and location. Bednarski BP et al. (2021)
analysed the use of reinforcement learning and deep learning models in optimizing the redistribution of medical equipment, thus instructing humanity to become better prepared for similar public health crises such as COVID-19. To the best of our concerns, none of the existing literature has studied the use of reinforcement learning in recommending optimal policies on lockdown and travel restriction with a continuous action space. Our major contribution is summarized as follows.

- We raise a new reinforcement learning framework based on continuous state and action representation space. This is more flexible than the discrete space used in Kwak GH et al (2021), since a continuous action space allows for more flexible and individualized policy recommendation on regional lockdown and travel restrictions.
- We introduce a new formulation of the optimal policy recommendation problem, based on a customized definition of the learning environment, including components on action and state space, as well as reward function. As we will show later, the customized environment is a flexible design which can be further tailored for more targeted and cost-sensitive learning.
- We proposed the use of the advanced DDPG algorithm in training of reinforcement learning agent and corroborated its superior performance by running experiments and comparing the results with multiple baselines. In addition, we also provide proof on the policy gradient theorem, a key component in DDPG architecture.

Methods

Basics of reinforcement learning:

Deep reinforcement learning is a genre of algorithms within the field of machine learning, aiming to solve artificial intelligence problems. It works by creating computer programs, or agents, to solve problems that normally demand intelligence. Compared with other types of machine learning algorithms, its uniqueness lies in its learning framework. Specifically, it learns through trial and error by interacting with the external environment and collecting feedback, including state and reward signals. This means that there is no labelled data, or an explicit correct answer to work with, which is different from supervised learning. The feedback from the environment could be simultaneously sequential, evaluative, and sampled from an original distribution. Powered by deep neural networks, an agent could be trained to approximate the true reward distribution using non-linear function approximation, with the ultimate goal of maximizing the long-term returns.

The goal of reinforcement learning is to train a decision-making agent that could achieve its target (maximizing cumulative rewards) despite existing uncertainties in its environment. At each time stamp \( t \), an agent has a combination of action \( a_t \) and state \( s_t \) along with reward \( r_t \) for each step. In a typical interaction process, the action \( a_t \) is sent to the environment, which moves to the next state and provides reward \( r_t \) to the agent. In this way, a reinforcement learning agent tries to maximize the cumulative rewards received after a certain policy, an entire series of actions, is performed. One typical example that involves evaluative feedback in deep reinforcement learning is a Pong-playing agent; the agent lives in a Pong emulator and plays the game many times, learns by taking different actions and observing the corresponding effects. The agent will successfully play Pong with superior performance after training for many iterations.

Deep reinforcement learning focuses on creating computer programs, or agents, that can tackle complex and sequential decision-making problems under uncertainties. Notably this is a topic of interest in many fields, such as control theory and operations research. The trained agent will act as a decision maker, which will provide suggestions on what actions to take given a specific situation, or state from the environment. The environment is everything apart from the agent, over which the agent has no control. It is represented by a set of variables that characterize and make up the environment. The set of variables and all the different values that they can take jointly form the state space. One state is an instantiation, or a sample, from the whole state space. It is obvious that the agent may not necessarily have access to the full state of the environment. For states that are accessible, i.e., observations, are used to train the agent. This is also referred to as partially observable state space. At each state, the agent will have a set of actions to choose from, by mapping an input state to an output action. The agent influences the environment by taking a specific action. As a result, the environment may change its feedback, including state and reward, in response to the agent’s action. The transition from one state to another, based on a particular action from the agent, is referred to as the transition function, and is usually unknown to the agent. Depending on the type of algorithm, the transition function may be learned to better approximate the dynamics of the environment. Besides, mapping a specific set of state and action to a reward is referred to as the reward function. The set of transition and reward functions jointly form the model of the environment, which again is not available in most learning problems. The figure below (Figure 1) depicts the typical interaction cycle between the agent and the environment.

![Figure 1: Reinforcement learning interaction cycle.](image-url)

The reward signals from the environment can be simultaneously sequential, evaluative, and sampled. Therefore, to obtain higher rewards, the agent needs to perform long-term thinking, balance exploitation and exploration, and finally generalize well the future scenarios. It typically follows a three-step process: first the agent interacts with the environment by taking an action, obtaining the next state and corresponding reward, then the agent evaluates...
agent improves its actions by changing its internal algorithmic representations of the world and all actions, expecting higher rewards in the next interaction cycle. The mapping from state to action is called policy, the mapping from state to a scalar value is called state value function, and the mapping function from a state action pair to a scalar value is called action value. Depending on specific requirements, the agent may choose to learn either an optimal policy or an optimal value. As we will introduce in the following sections, the type of algorithm used in our approach employs both policy- and value-based methods, which has been shown to deliver superior performance compared with policy or value-based methods alone. We will also specify what constitutes the environment and the agent in our COVID-19 simulation in later sections. In a nutshell, the optimal policy is automatically learned based on our proposed algorithm, which works as a function of environmental state, in particular disease parameters, such as number of deaths and infections, along with population characteristics such as density.

To facilitate our discussions in the following, we provide a table containing the mathematical notations used in this paper (Table 1).

Table 1: Mathematical notations used in this paper.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s \in S$</td>
<td>States</td>
</tr>
<tr>
<td>$a \in A$</td>
<td>Actions</td>
</tr>
<tr>
<td>$r \in R$</td>
<td>Rewards</td>
</tr>
<tr>
<td>$s, a, R_t$</td>
<td>States, actions, and rewards at time step $t$ of an interaction trajectory</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Discounting factors used to penalize the future rewards, where $\gamma \in [0, 1]$</td>
</tr>
<tr>
<td>$G_t$</td>
<td>Long-term returns, as a sum of immediate and future discounted rewards, $G_t = \sum_{i=t}^{\infty} \gamma^i R_{t+i}$</td>
</tr>
<tr>
<td>$P(s', r</td>
<td>s, a)$</td>
</tr>
<tr>
<td>$\pi(a</td>
<td>s)$</td>
</tr>
<tr>
<td>$V(s)$</td>
<td>State value function measuring the expected long-term returns or value of being in state $s$</td>
</tr>
<tr>
<td>$V^*(s)$</td>
<td>The value of state $s$ following policy $\pi$, where $V^*(s) = E_{\pi}[G_t</td>
</tr>
<tr>
<td>$Q(s, a)$</td>
<td>Action value function measuring the expected long-term returns or value of being in state $s$ and action $a$</td>
</tr>
<tr>
<td>$Q^*(s, a)$</td>
<td>The value of policy $\pi$ in state $s$, following policy $\pi$, where $Q^*(s, a) = E_{\pi}[G_t</td>
</tr>
</tbody>
</table>
| $A^*(s, a)$ | Advantage function $A(s, a) = Q(s, a) - V(s)$, which has lower variance by extracting the baseline state value from the action value performance, by approximating action value function using $Q(s, a; w)$, where $w$ denotes the weight of a neural network. Following a states-in-values-out architecture, the network could even yield a high-performance implementation that outputs the value for all actions at once, given a specific state $s$.

The objective function given by the action value network is as follows.

$$L(w) = E_{\pi,w}[\left( q(s, a; w) - Q(s, a; w) \right)^2]$$

Where the objective is to minimize the loss with respect to the optimal action value function $q^*(s, a)$ for the current state $s$ and action $a$, using the current estimate $Q(s, a; w)$ parameterized by $w$. The optimal action value $q^*(s, a)$ is defined as follows.

$$q^*(s, a) = \max_a E[Q_t(s_t, a_t)|s_t = s, A_t = a_t, \forall s \in S, \forall a \in A(s)]$$

This essentially means the optimal action value function is the policy that renders the maximum expected returns from each and every action in each and every state. In the meantime, $q^*(s, a)$ itself is not available in advance, it is often estimated using another target network, serving as a proxy target value from the target network for current estimation. In deep Q-networks, we is represented using an online Q-learning target, defined as follows.

$$y = r_{t+1} + \gamma \max_a Q(s_{t+1}, a; w)$$

Which is a combination of the actual experienced rewards and the estimated action values of the next state. Plugging in $y_t$ from the target networks gives the following.

$$L(w) = E_{\pi,w}[(r + \gamma \max_a Q(s', a'; w') - Q(s, a; w))^2]$$

Where $y_t$ represents the parameters of the online Q-learning network, and $w$ denotes the parameters for the target network, which is usually updated every few iterations. By differentiating through this equation to update the network weight using gradient descent algorithm, we have:

$$\nabla_i L(w) = E_{\pi,w}[c(r + \gamma \max_a Q(s', a'; w') - Q(s, a; w)) \nabla_i Q(s, a; w)]$$

It is worth noting that the gradient does not involve the target, thus it only goes through the predicted value, namely $Q(s_t, a_t; w)$. Policy Networks:

Since reinforcement learning problems can have both continuous state and action spaces, which can be either high dimensional with discrete values or low dimensional with continuous values, it is important to use function approximation that generalizes unseen states or actions, making the algorithm more efficient. In deep Q-networks, as proposed in the seminal work by Minh et al. (2015)¹⁴, was used to generalize past experiences to new situations with superior performance, by approximating action value function using $Q(s, a; w)$, where $w$ denotes the weight of a neural network. Following a states-in-values-out architecture, the network could even yield a high-performance implementation that outputs the value for all actions at once, given a specific state $s$.

The objective function given by the action value network is as follows.

$$L(w) = E_{\pi,w}[\left( q(s, a; w) - Q(s, a; w) \right)^2]$$

Where the objective is to minimize the loss with respect to the optimal action value function $q^*(s, a)$ for the current state $s$ and action $a$, using the current estimate $Q(s, a; w)$ parameterized by $w$. The optimal action value $q^*(s, a)$ is defined as follows.

$$q^*(s, a) = \max_a E[Q_t(s_t, a_t)|s_t = s, A_t = a_t, \forall s \in S, \forall a \in A(s)]$$

This essentially means the optimal action value function is the policy that renders the maximum expected returns from each and every action in each and every state. In the meantime, $q^*(s, a)$ itself is not available in advance, it is often estimated using another target network, serving as a proxy target value from the target network for current estimation. In deep Q-networks, this is represented using an online Q-learning target, defined as follows.

$$y = r_{t+1} + \gamma \max_a Q(s_{t+1}, a; w)$$

Which is a combination of the actual experienced rewards and the estimated action values of the next state. Plugging in $y_t$ from the target networks gives the following.

$$L(w) = E_{\pi,w}[(r + \gamma \max_a Q(s', a'; w') - Q(s, a; w))^2]$$

Where $y_t$ represents the parameters of the online Q-learning network, and $w$ denotes the parameters for the target network, which is usually updated every few iterations. By differentiating through this equation to update the network weight using gradient descent algorithm, we have:

$$\nabla_i L(w) = E_{\pi,w}[c(r + \gamma \max_a Q(s', a'; w') - Q(s, a; w)) \nabla_i Q(s, a; w)]$$

It is worth noting that the gradient does not involve the target, thus it only goes through the predicted value, namely $Q(s_t, a_t; w)$. Policy Networks:

Since reinforcement learning problems can have both continuous state and action spaces, which can be either high dimensional with discrete values or low dimensional with continuous values

$$l(\theta) = \sum_i d_i(s_i, a_i) - \sum_i d_i(s_i) \sum_a a_i(s_i, a_i)$$

Where $d_i(\cdot)$ denotes the on-policy stationary state distribution of the Markov chain for $\pi$ under policy $\pi$. In other words, $d_i(\cdot) = \lim_{t \to \infty} P(s_{t+1} = s_i|s_t = s, \pi)$. The final stationary probability that $s_{t+1}$ when starting from initial state $s_t$ and following policy $\pi_{\theta}$ for a total $t$ steps. The parameters $\theta$ of the policy network can then be sequentially optimized using gradient ascent algorithm, based on the gradient $\nabla_{\theta} J(\theta)$.

It is challenging to compute the gradient $\nabla_{\theta} J(\theta)$ due to its dependence on both action selection $\pi_{\theta}(a|s)$ and the stationary distribution $d_i(\cdot)$. Based on the work of Sutton & Barto (2017),¹⁵ the derivative $\nabla_{\theta} J(\theta)$ can be transformed to a form that does not involve the derivative of state distribution $d_i(\cdot)$. We build on the work of Sutton & Barto (2017)¹⁵ and provide
the following theorem to characterize the policy gradient used in our algorithm.

**Theorem 1:**

The gradient of \( J(\theta) \) does not depend on \( d_\pi(s) \) and only depends on \( \pi_\theta \), through the following:

\[
\nabla J(\theta) = \sum_{s,a} \nabla_{\theta} Q(s,a) \pi_a(s) = \sum_{s,a} \nabla_{\theta} \mathbb{E}_{s_t \sim \pi} [Q(s,a) | s_t = s] \pi_a(s)
\]

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
\nabla J(\theta) = \sum_{s,a} \nabla_{\theta} Q(s,a) \pi_a(s) = \sum_{s,a} \nabla_{\theta} \mathbb{E}_{s_t \sim \pi} [Q(s,a) | s_t = s] \pi_a(s)
\]

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).

**Proof of Theorem 1:**

We first analyse the derivative of state value function \( V_\pi(s) \).

\[
P_t(s \rightarrow s',k) = \sum_{a} \pi_t(a \mid s) P_t(s \rightarrow s',k \mid a)
\]

Until now, we have a nice recursive definition where the future state value function \( V_\pi(s') \) can be further expanded using a similar rule. From an interaction perspective with the environment, we have the following sequence of visitation, where \( P_t(s \rightarrow s',k) \) denotes the probability of transitioning from state \( s \) to state \( s' \) at step \( k \), following the policy \( \pi_t \).
consists of disease related features such as the number of confirmed, recovered, deceased cases per day, as well as region specific features such as GDP, population, and life expectancy, etc. Additional features are also derived, including the relative ratio and cumulative statistics of these features. Since the training environment for this problem is not provided in any of the existing open-source libraries, we created our own customized environment, with a reasonably defined reward function to support optimal decision making of the agent. Our framework, along with its multiple parameters for reward definition, is designed to be flexible enough to support future extensions. In the following sections, we will introduce the details on reward function definition, following illustrations of the experiment results and discussions.

**Reward definition:**

A proper reward function should be defined in order to guide the agent in the search of the optimal policy, in our case controlling the real pandemic situation by tuning the intensity of lockdown and travel restriction on a daily basis. Intuitively, a rise in the number of deaths or infections should be discouraged, thus resulting in a negative reward under such circumstances. On the other hand, when some actions have succeeded in controlling the number of infection and death cases under a certain level, then they should be encouraged, and thus the corresponding rewards should be positive. The reward used in our environment consists of two components: death and recovery, each of which taking a different reward or penalty structure. The total reward per day is then derived by summing up both components. The following table illustrates the overall sign direction for the rewards definition logic, based on death severity. The same logic applies to recovery, which will take an opposite sign.

**Table 2:** Example of designing the reward sign direction for death severity.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Explanation</th>
<th>Reward sign</th>
</tr>
</thead>
<tbody>
<tr>
<td>Getting worse</td>
<td>More cases than yesterday with any actions taken</td>
<td>Negative</td>
</tr>
<tr>
<td>Stable</td>
<td>No additional death cases than yesterday due to certain actions taken</td>
<td>Positive</td>
</tr>
<tr>
<td>Getting better</td>
<td>Reduced number of deaths due to certain actions taken</td>
<td>Positive</td>
</tr>
</tbody>
</table>

Based on this overarching rule, we designed the reward function $r_t$ as follows. Note that customized weighting between these two components is possible, with the flexibility of incorporating additional perspectives such as number of infections.

$$
r_t^{\text{death}} = \begin{cases} 
- c_0 & \text{if death}_t > \text{death}_{t-1} \text{ and death}_t \neq 0 \\
= c_0 & \text{if death}_t = \text{death}_{t-1} \text{ and death}_t \neq 0 \text{ and lockdown } > 0 \text{ and travel restriction } > 0 \\
= c_0 & \text{if death}_t < \text{death}_{t-1} \text{ and death}_t \neq 0 
\end{cases}
$$

$$
r_t^{\text{recovery}} = \begin{cases} 
c_0 & \text{if recovery}_t = \text{recovery}_{t-1} \text{ and recovery}_t \neq 0 \text{ and lockdown } > 0 \\
= -c_0 & \text{if recovery}_t < \text{recovery}_{t-1} \text{ and recovery}_t \neq 0
\end{cases}
$$

**Action space:**

In this paper, we propose to use continuous action space to determine the intensity level for lockdowns and travel restrictions. The proposed values for these two action outputs vary within a fixed range, thus this is a more flexible and customized recommendation engine compared with discrete action space. From an algorithmic perspective, since a regular deep Q-learning network is unable to deal with continuous action space due to the curse of dimensionality, the DDPG algorithm used in our model could, by design, handle the continuous output space very well.

**State space:**

Since disease and region related features are time series in nature, it is important to expose the trend of these metrics as part of the observations in a state. For example, if we only provide the number of infections for the current day, the agent will have no indication of whether such a metric is improving or worsening. By incorporating prior information from previous days, we allow the agent to infer serial dependence structure from the multivariate time series data, thus aiding the long-term decision on whether to ramp up or ease certain control measures. In our experiment, we provided past 10 days of feature values as a single state input. In other words, the agent is exposed to a snapshot of previous 10 days of feature input values per day, on a rolling basis.

**Results**

By training a reinforcement learning agent using actual data and corresponding simulations via experience replay, the algorithm will return a policy that provides theoretically reward-maximizing parameters on the control policies under given states. Yet, in order to test its validity, the reward outputs from the algorithm-based policy should then be compared with the real reward outputs under the same state. Once most theoretical rewards exceed their corresponding original ones, it can be concluded that our agent managed to discover an optimal policy for pandemic control.

We first calculated the rewards based on actual lockdown and travel restriction and plotted the distribution of rewards via a frequency plot shown below. Figure 2 suggests that most of the countries and regions are taking insufficient lockdown or travel restriction measures in the fight against COVID-19. However, there are some countries showing good performance in controlling the spread of the disease, which corresponds with our actual observations.

As a comparison, we first implemented the Actor Critic algorithm using Kronecker-Factored Trust Region (ACKTR) method, as proposed by Wu et al. (2017).¹⁶ This serves as a baseline model for comparison with our DDPG based model. We trained the agent using ACKTR algorithm for a total of 5000 epochs, and then used the trained agent to interact with the environment and generate a list of rewards, which were then added up to form a final cumulative reward. We repeated this experiment 100 times, and then compared it with the actual observed rewards by taking the difference between

\[ c_0 \quad \text{if } \text{recovery}_t = \text{recovery}_{t-1} \text{ and } \text{recovery}_t \neq 0 \text{ and lockdown } > 0 \text{ and travel restriction } > 0 \]

\[ -c_0 \quad \text{if } \text{recovery}_t < \text{recovery}_{t-1} \text{ and } \text{recovery}_t \neq 0 \]
them. Figure 3 suggests that the agent-based recommendation system can generate policies that are better than existing ones for most of the time. The superior policies also surpass the existing policies by a huge margin in terms of the cumulative rewards.

![Figure 2: Reward distribution based on actual lockdown and travel restriction policies.](image)

**Figure 2:** Reward distribution based on actual lockdown and travel restriction policies. The “Frequency” axis stands for the number of policies that with a certain interval of reward. The figure suggests that most rewards are negative indicating the largely insufficient actions taken by most countries and regions.

![Figure 3: Distribution of reward differences based on lockdown and travel restriction policies recommended by ACKTR model and those from actual control measures.](image)

**Figure 3:** Distribution of reward differences based on lockdown and travel restriction policies recommended by ACKTR model and those from actual control measures. The figure suggests that the agent-based recommendation system is able to generate policies that are better than existing ones for most of the time. The superior policies also surpass the existing policies by a huge margin in terms of the cumulative rewards.

We also plotted the distribution of cumulative rewards side by side. Figure 4 also suggests that the trained agent using ACKTR can better recommend optimal policies that maximize the long-term returns.

![Figure 4: Side by side comparison of density plot on cumulative rewards.](image)

**Figure 4:** Side by side comparison of density plot on cumulative rewards. The “Data” axis stands for corresponding reward value calculated with each given policy. The figure suggests that the trained agent using ACKTR is able to better recommend optimal policies that maximize the long-term returns.

Now let us visit the experiment results using DDPG model. Under the same overall setting, we set the discounting factor to 0.99, maximum number of episodes to 100, maximum mean reward per 100 steps to 10000, and the learning rate for both policy and value networks to 0.0001. We set the replay buffer size to 10000, with a batch size of 32. For the soft update parameter \( \tau \), we adjusted it to 0.005. Figure 5 contains the moving average rewards for both training and evaluation, suggesting that the DDPG algorithm is efficient and effective in capturing the long-term systematic patterns in the data.

![Figure 5: Moving average rewards for both training and evaluation using DDPG algorithm.](image)

**Figure 5:** Moving average rewards for both training and evaluation using DDPG algorithm. The evaluation performance quickly stabilizes to around 6000 in early episodes and slowly improves afterwards, suggesting that the DDPG algorithm is efficient and effective.

Besides, we also compared its empirical performance with actual lockdown and travel restriction measures. By taking the difference between the cumulative rewards using DDPG and that from actual control measures, we have the following Figure 6, indicating a uniformly positive reward difference and thus superior performance.

![Figure 6: Distribution of reward differences based on lockdown and travel restriction policies recommended by DDPG model and those from actual control measures.](image)

**Figure 6:** Distribution of reward differences based on lockdown and travel restriction policies recommended by DDPG model and those from actual control measures. The figure suggests that DDPG model is able to generate uniformly superior control policies compared with actual lockdown and travel restriction measures.

**Discussion**

Coming up with an optimal control policy that balances the immediate evolution of the situation and long-term impact is
a challenging task. One needs to consider not only the current factors such as infection status, but also the pre-existing trends of this metric, so as to properly adjust and optimally determine the next control policy. By proposing an automated framework to learn from the simulated environment based on actual data, it is thus our hope that the trained agent serves as a decision support engine to assist policymaking of local authorities, in this case the level of lockdown and travel restriction. Plus, the recommended policies should be flexible enough to cater to individual characteristics of different countries and regions.

Normally, the level of lockdown and travel restriction in a region is graded in a discrete classification. For instance, level 0 lockdown could mean no control measures at all, and level 2 of lockdown could suggest an entire shutdown of transportation routine. Nonetheless, it is unlikely for the government to exactly borrow and copy the standards of lockdown classification from other countries or regions. One major contribution of our work is to specify the actual levels of lockdown to be expressed as continuous (in this case, a floating number that lies on [0, 2], similarly for travel restriction measures). With the support of the DDPG algorithm, such outputs could more appropriately cater to practical implementations.

In addition, defining a proper reward function is of great importance to the successful training of a reinforcement learning agent. The reward serves as a weak signal. Compared with supervised learning, instead of directly indicating the correct answer to the current learning problem at hand, a reward signal simply indicates the immediate feedback from the environment. Due to the lack of transition dynamics of the environment, simply maximizing the immediate rewards may lead to early and sub-optimal convergence. Therefore, the reward definition should reflect the attributes of the environment which the agent is expected to learn from and maximize, at the right time, via a long planning horizon. Inappropriately defined reward functions may confuse the learning algorithm when trying to identify the right direction to improve, or in some cases, mislead the agent into leading something not in the right direction. In our paper, the customized reward function is defined in a way that it encourages early control of the disease and penalizes bad or even worse situations due to the lack of proper control actions. We believe such a definition meets the overarching objective of disease control.

Nonetheless, it should be noted that in real implementations, both the control measures and the spread of the pandemic lead to economic loss in society, which is not taken into consideration as part of our reward function definition. The potential downstream impact is that the optimal policy suggested by our agent will be too costly to practice, as it is more beneficial to choose the safe and conservative side of control, as early as possible. Incorporating the economic impact, along with the delayed influence on economic growth as a whole, would be an interesting future research direction.

### Conclusion

In this paper, we study the problem of formulating an optimal policy for regional control measures on the spread of COVID-19, and propose the use of DDPG algorithm, a powerful reinforcement learning model, to tackle this challenge. We analyse the structure of the optimal control problem by detailed discussions on the definition of three integral components of reinforcement learning: action space, state space, and reward function. To demonstrate the effect of the DDPG model, we provide theoretical motivations on its value and policy networks, with arguments on its superior performance compared with alternative models. In addition, we perform experiments using actual COVID-19 data and show that our proposed model provides better empirical results in terms of total cumulative rewards, when compared with both actual control measures and those from an alternatively trained agent. By this process, our agent shows better results in terms of controlling the spread of the pandemic. These results suggest that DDPG is a promising model for similar types of control problem.

### Acknowledgements

I would like to thank Professor Wu Yu for guiding me through the research process and teaching me how to write a research paper.

### References


**Author**

Kailiang Liu is currently studying in Shenzhen Middle School, and is inclined to apply with the direction of mathematics and computer science in future college study.