Mini-Project 1 Report : Deep Q-Learning for Epidemic Mitigation

CS-456 : Artificial Neural Networks/Reinforcement Learning

Spring 2023 Semester

Project members:	***SCIPER:***
Théodore Maradan	315684
Albias Havolli	286826

Swiss Federal Institute of Technology Lausanne
Switzerland

June 4, 2023

Contents

List of Figures

List of Tables

Introduction

The aim of this project is to train an artificial agent using Deep-Q Learning to find a decision-making policy regarding the mitigation of an epidemic. Different action and observation spaces will be tested in order to optimize the performance of the agent.

1 Study the behavior of the model when epidemics are unmitigated

The first objective is to simulate an unmitigated epidemic which will serve as a benchmark for the different policies evaluated later in the report. The figure below shows the evolution of the different state variables globally and the observable variables, both globally and by city :

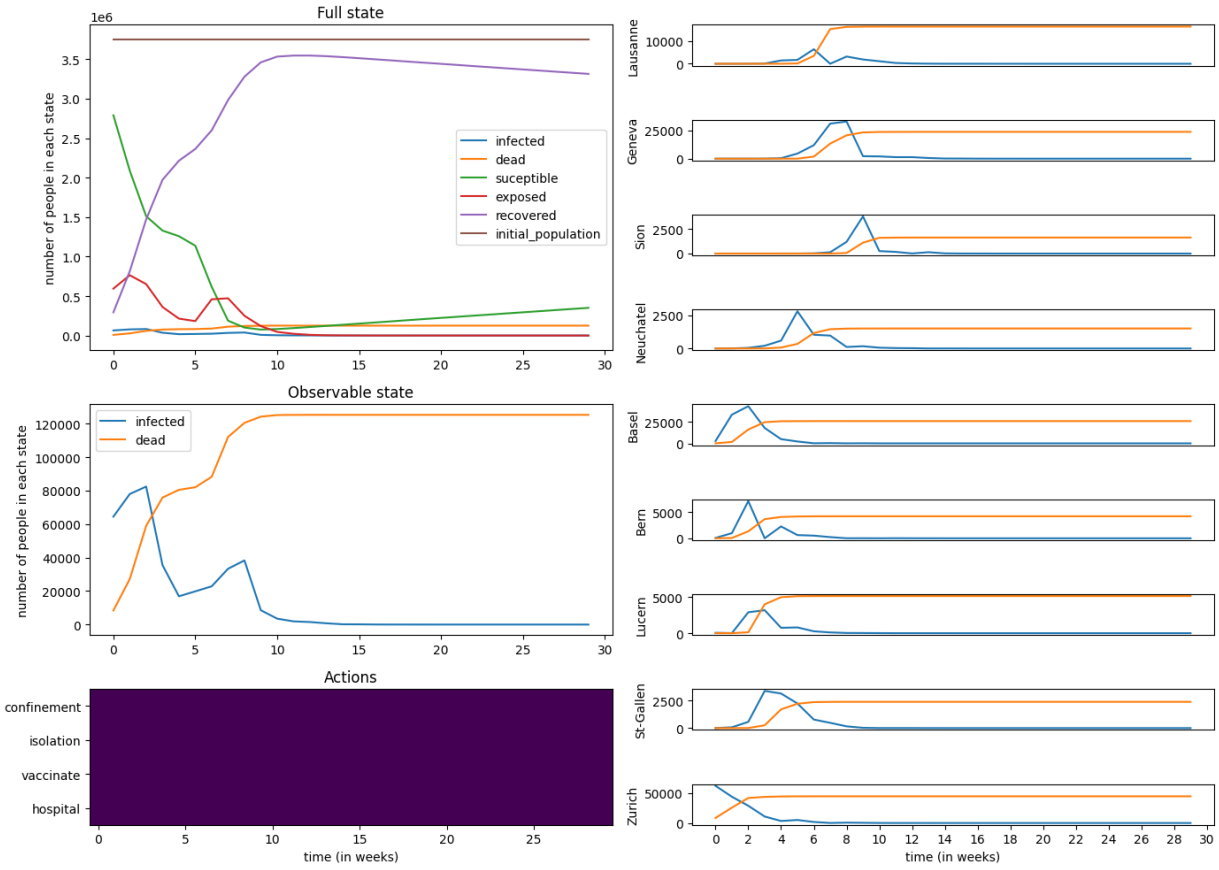

Figure 1.1: One episode simulation without epidemic mitigation

Please note that the plots were generated with a seed set to 2 and the horizontal axis is time (measured in weeks). The upper and middle left plots show the full and observable states respectively. The lower left plot shows that the epidemic was, indeed, not mitigated (no action taken). Finally, the right plots show the observable states for each city.

As can be seen, without epidemic mitigation, the number of deaths increases until reaching a maximum at week 10. On the other hand, the number of infected people tends to decrease and reaches 0 around week 10. At this point, most of the population is in the "recovered" state meaning that they are temporarily immune to the disease. However, as immunity is lost over time, a second wave of infections might occur in the future.

The effect of the magnitude of cross-contamination can be observed on the plots by cities. Indeed, the highest number of deaths can be found in Zürich and Basel which are two cities with high population and a high cross-contamination rate. This is confirmed by looking for example at Sion which has a low number of deaths due partly to its low population but also to its isolation from most of the other cities considered.

2 Professor Russo's policy

Let's now test the performance of the policy proposed by Professor Russo to mitigate the epidemic of MARVIN23.

2.1 Implement Pr. Russo's policy

Pr. Russo's policy is implemented as a python class and an episode is run with, again, the seed set to 2. The following figure shows the evolution of state and observable variables as well as the actions taken by the agent:

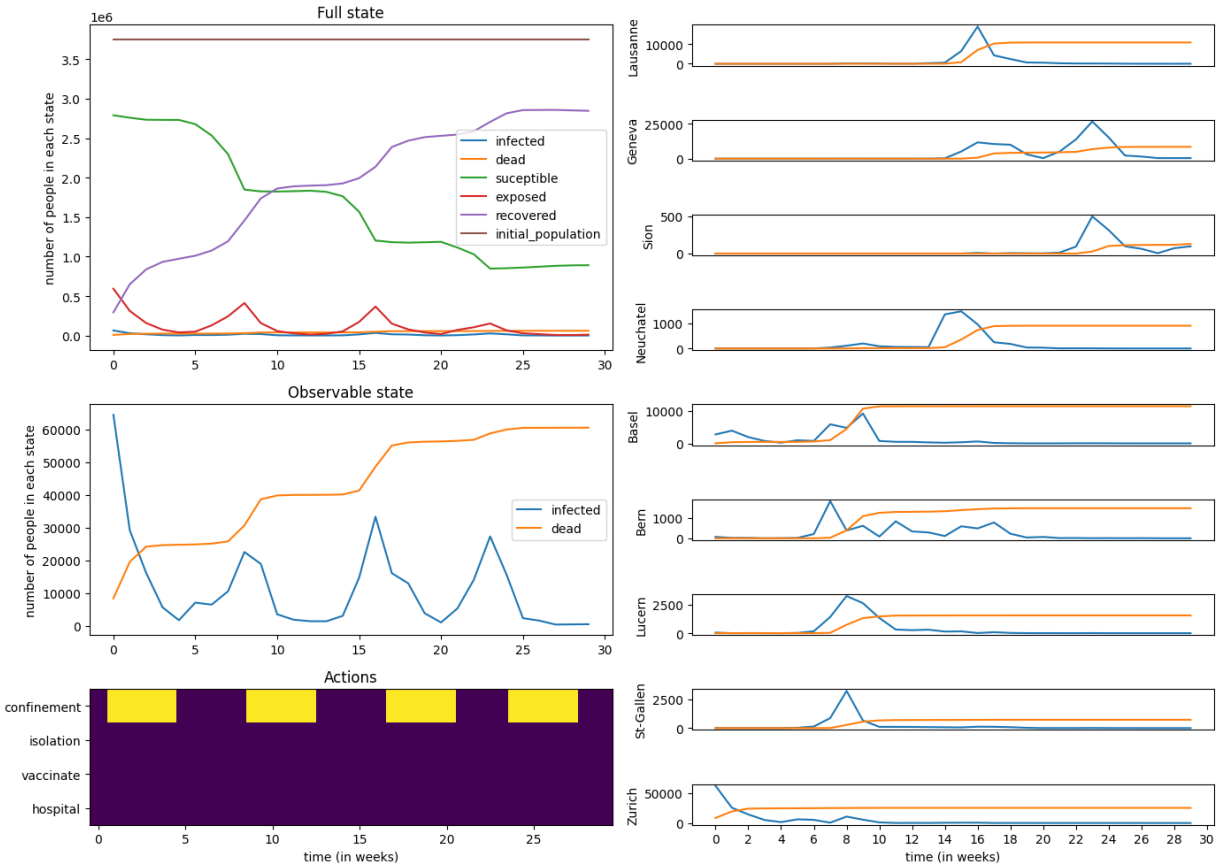

Figure 2.1: One episode simulation with Pr. Russo's policy

The plot of the observable state shows a clear difference with the unmitigated scenario. Indeed, the number of infections first goes down thanks to the confinement period but increases as soon as the confinement is finished. The number of deaths still tends to a maximum but is delayed compared to the unmitigated case (see Figure 1.1). The maximum number of deaths is decreased by approximately 50%.

2.2 Evaluate Pr. Russo's policy

Pr. Russo's policy π_{Russo} is evaluated by running 50 episodes using this policy to choose actions. The number of confined days, deaths, and the total reward per episode are summarized in the following histograms :

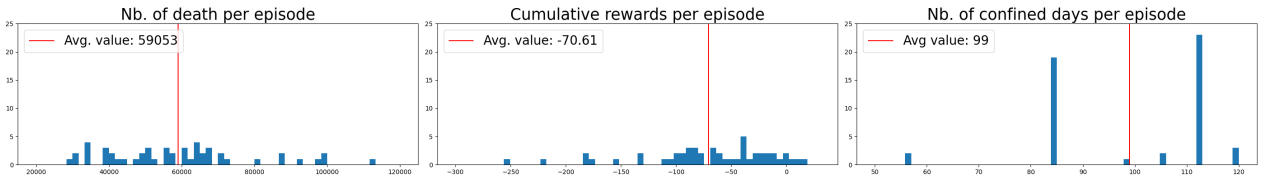

Figure 2.2: Histograms evaluating Pr. Russo's policy

3 Deep Q-learning with a binary action-space

Now, Deep Q-Learning with a binary action-space is evaluated as a possible improvement to Pr. Russo's policy.

3.1 Implementing Deep Q-Learning

Training and evaluation traces using Deep Q-Learning, a binary action-space, and a constant exploration are plotted below (see Figure 3.1). Those were generated with an ϵ-greedy algorithm. The training trace is plotted as a scatter plot for the three training processes. The evaluation trace is averaged over the three training processes.

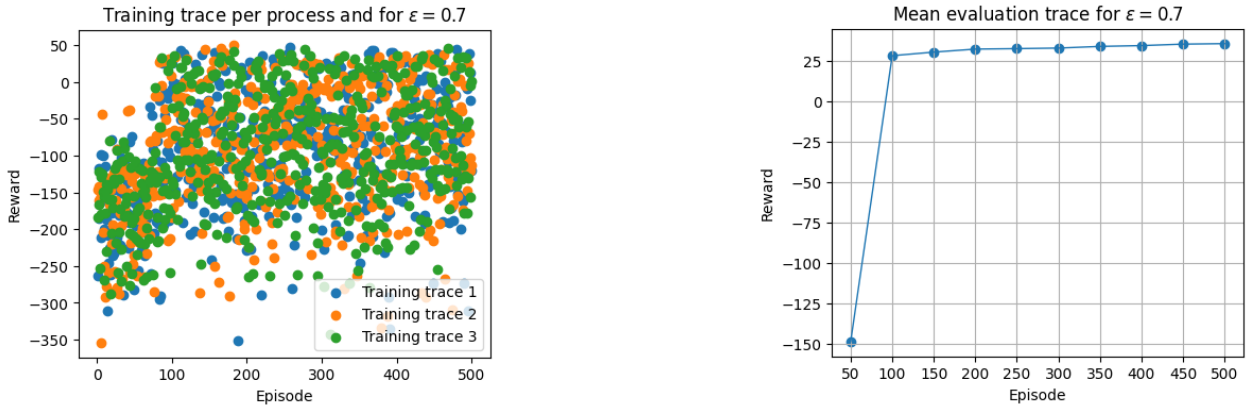

Figure 3.1: Training and evaluation traces with constant exploration using binary action-space

As can be seen on the evaluation trace plot, the cumulative reward converges very quickly and increases until reaching a high value of approximately 38. This suggests that the agent learns a meaningful policy.

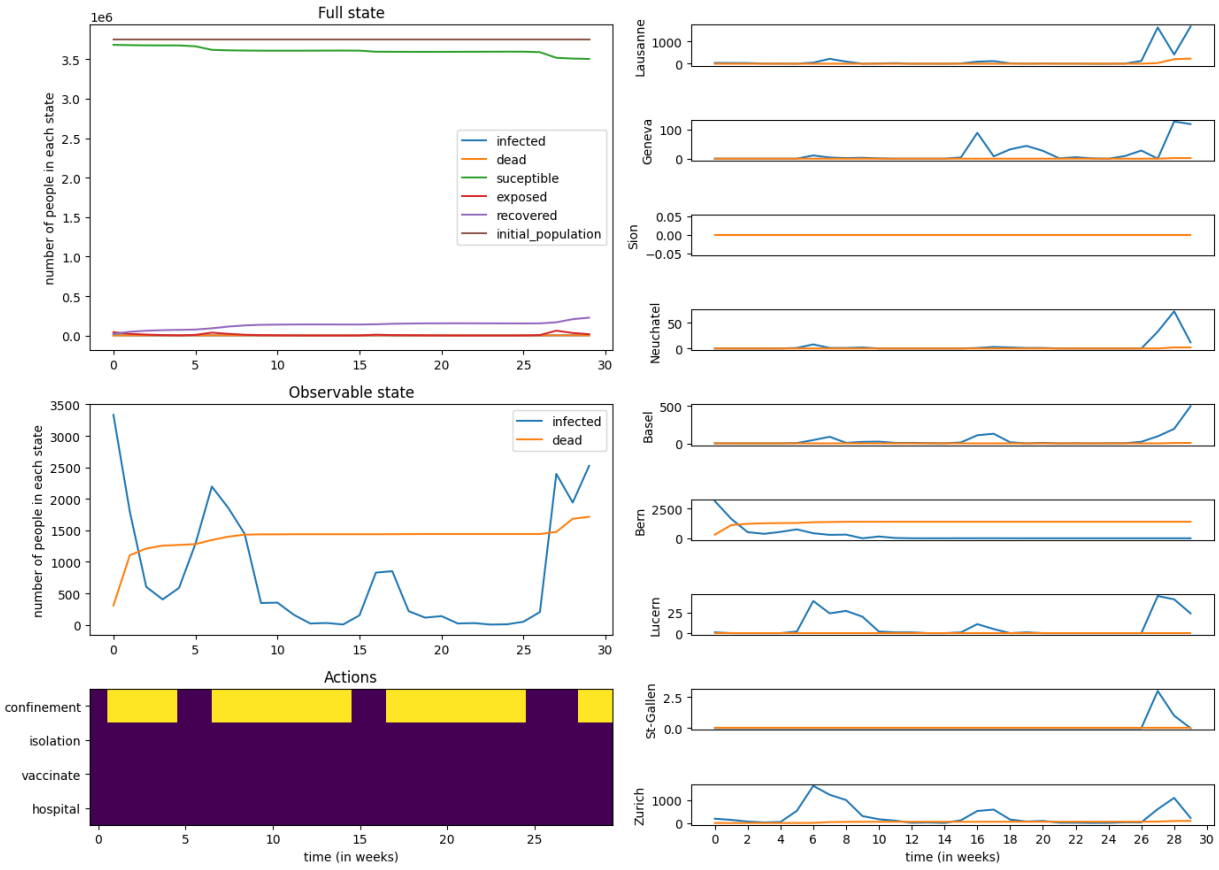

Figure 3.2: One episode simulation with the best DQN policy using constant exploration π^*_{DQN}

Figure 3.2 shows one of three episodes evaluated using the best policy π^*_{DQN} learned. The policy seems similar as the one defined by Pr. Russo. Indeed, the agent chooses to confine the country when infections are too high. However, the threshold seems to be lower than the one used by Pr. Russo and the confinement period is now not fixed. Doing so, the numbers of deaths and of infections are significantly lower than with π_{Russo}. The policy learned seems, therefore, to make sense and to be meaningful.

3.2 Decreasing exploration

The goal now is to evaluate the increase of performance achievable with the ϵ-greedy algorithm, considering now decreasing exploration (from 70% to 20%, gradually). The training and evaluation traces using constant or decreasing exploration are compared in Figure 3.3.

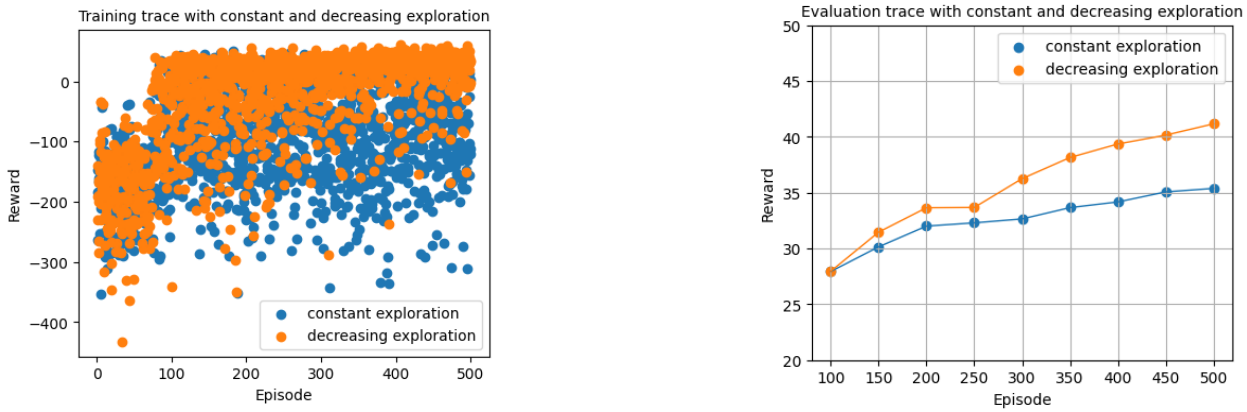

Figure 3.3: Training and evaluation traces with constant and decreasing exploration using binary action-space

The evaluation trace shows that the policy using a binary action-space but decreasing exploration outperforms the policy with constant exploration during evaluation. Indeed, with decreasing exploration, the agent relies less on random exploration which generally leads to low reward and focuses more on exploiting the best policy learned, accumulating knowledge on it and improving it.

3.3 Evaluate the best performing policy against Pr. Russo's Policy

The best performing policy π^*_{DQN} is here evaluated using the process defined in section 2.2. The results are shown in the following histograms.

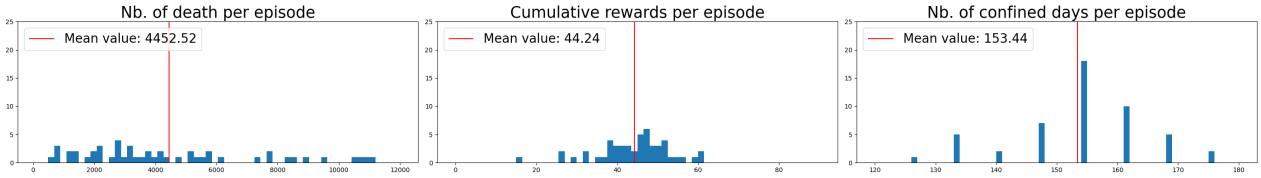

Figure 3.4: Histograms evaluating the best DQN policy with decreasing exploration π^*_{DQN}

As can be observed, the best policy π^*_{DQN} outperforms Pr. Russo's policy in terms of number of deaths (4,453 vs. 59,053 in average respectively) and cumulative reward (44 vs. -71 in average respectively) per episode. To achieve these results, the best policy π^*_{DQN} uses a higher number of confined days per episode (153 vs. 99 in average respectively).

4 Dealing with a more complex action-space

The goal now is to use the whole action-space to try to develop a better performing policy. First, a toggle action-space is considered and, then, the whole action-space is considered at once. Please note that decreasing exploration was considered and, for better results, the neural network was trained using a learning rate of 10^{-1} or $7.5 \cdot 10^{-2}$ for the toggle action-space or the factorized Q-Values, respectively.

4.1 Toggle-action-space multi-action agent

4.1.1 Action-space design

In order to consider the new action-space, several modifications must be applied to the network's architecture. The input dimension is now 130 ((nb. cities) · (nb. of days in a week) · (infections or deaths) + (nb. decisions) $= 9 \cdot 7 \cdot 2 + 4 = 130$) as the state of each action must be added to the observation space. Moreover, considering the number of possible actions, the output of the network is now of dimension 5 (*do nothing*, *confinement toggle*, *isolation toggle*, *vaccination toggle*, *hospitalization toggle*).

The advantage of a toggle action-space is a network with less weights than one computing directly the Q-Value of each possible action. It also makes sense in the framework considered here as actions are binary. However, toggle actions can introduce discontinuities in the Q-Values space, potentially leading to instabilities during training.

4.1.2 Toggle-action-space multi-action policy training

Let's first plot the training and evaluation traces using a toggle action-space using the same procedure as in section 3.1.

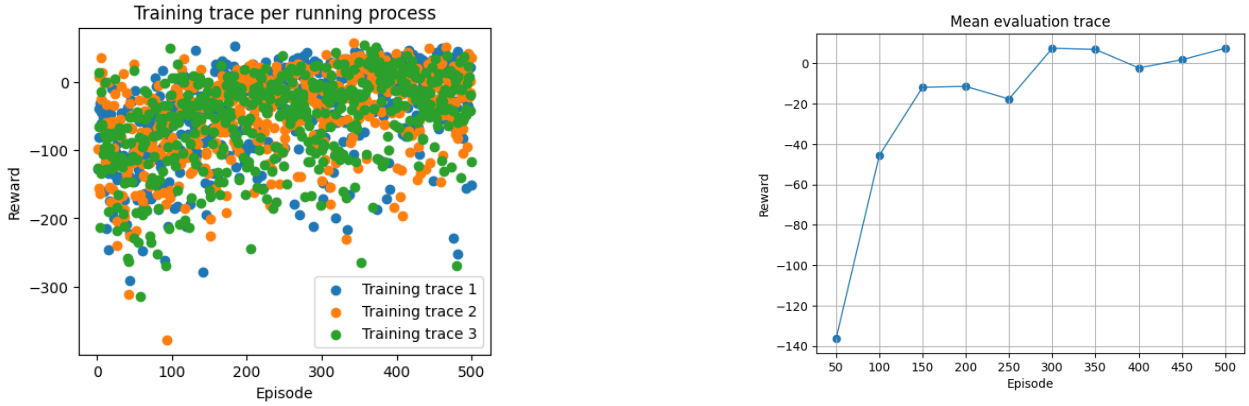

Figure 4.1: Training and evaluation traces with decreasing exploration using toggle action-space

As can be observed in the plot of the evaluation trace, the cumulative reward during evaluation encounters some instabilities due to the use of a toggle action-space. Indeed, the binary nature of toggle actions can create discontinuities in the Q-Values space, leading to convergence issues and training instabilities. Therefore, we can assume that the agent struggles on learning a meaningful policy. As a consequence, a higher number of training episodes might be required for the policy to fully converge.

Figure 4.2 shows that the toggle policy reacts well to increases in the number of infected persons, allowing for a quick decrease in the number of infections by using a confinement period. Moreover, we can see that the infection peaks are associated with an increase in the number of deaths. In order to limit this increase, the agent decides to add hospital beds (as it has the effect of reducing the mortality).

It is interesting to note that, in this episode, the policy does not take advantage of the isolation or vaccination actions as shown on the bottom left plot. Those only have an indirect impact on the number of infections or deaths which might not be beneficial when discounted.

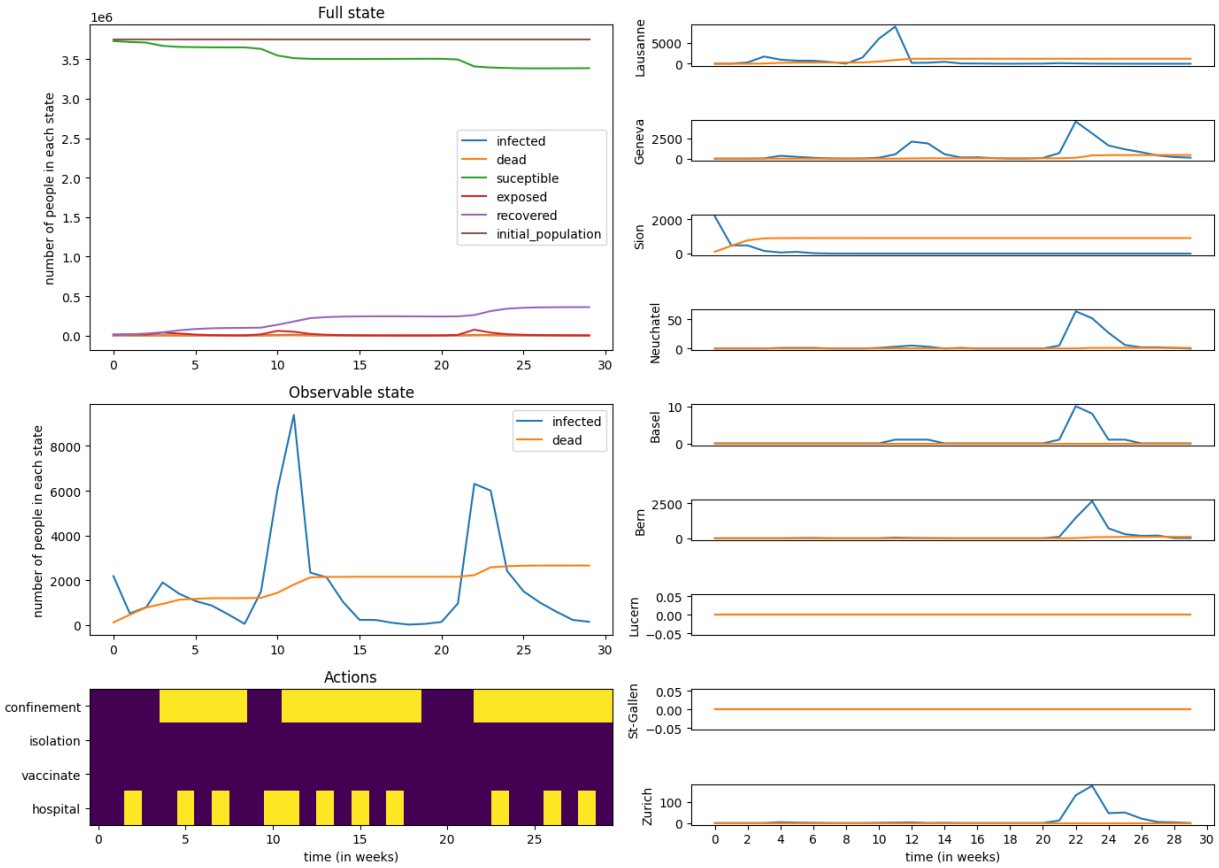

Figure 4.2: One episode simulation with the best policy under a toggle action-space π^*_{Toggle}

4.1.3 Toggle-action-space multi-action policy evaluation

The best performing toggle policy π^*_{Toggle} is now run on 50 episodes and an evaluation is provided in Figure 4.3.

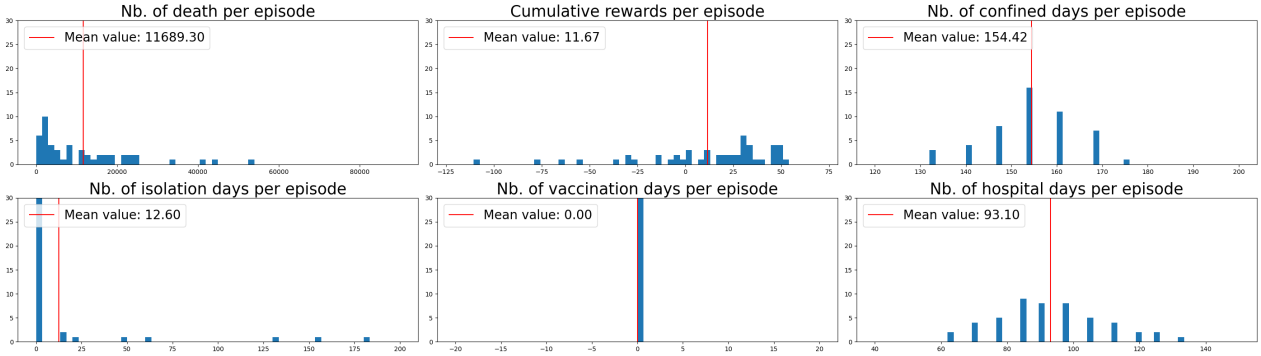

Figure 4.3: Histograms evaluating best policy under a toggle action-space π^*_{Toggle}

In comparison to the histograms in Figure 3.4, one can be see that the π^*_{Toggle} is outperformed by π^*_{DQN}. Indeed, the latter has a higher cumulative reward as well as a lower number of deaths. As can be observed, π^*_{Toggle} rarely considers isolation and never takes advantage of vaccination.

4.1.4 Question about toggled-action-space policy, what assumption does it make?

Using toggle action-space is only possible when the action-space is binary. The agent needs to learn a policy to determine when to toggle which action. Considering the higher number of possible actions compared to the DQN setting described above, a higher number of training episodes might be required for the agent to properly learn a policy. Moreover, it is sometimes assumed that the toggling actions are independent from each other in order to simplify the learning process and allow for a separate modeling of each toggle action but it does not reflect the reality.

Although a toggle action-space is suitable for the case considered here, it is not applicable to cases in which the action-space is continuous or multi-dimensional. For example, a continuous action-space would be required for the control of a robotic arm.

4.2 Factorized Q-Values, multi-action agent

4.2.1 Multi-action factorized Q-values policy training

Let's first compare the training and evaluation traces when using a toggle action-space or factorized Q-Values (see Figure 4.4.

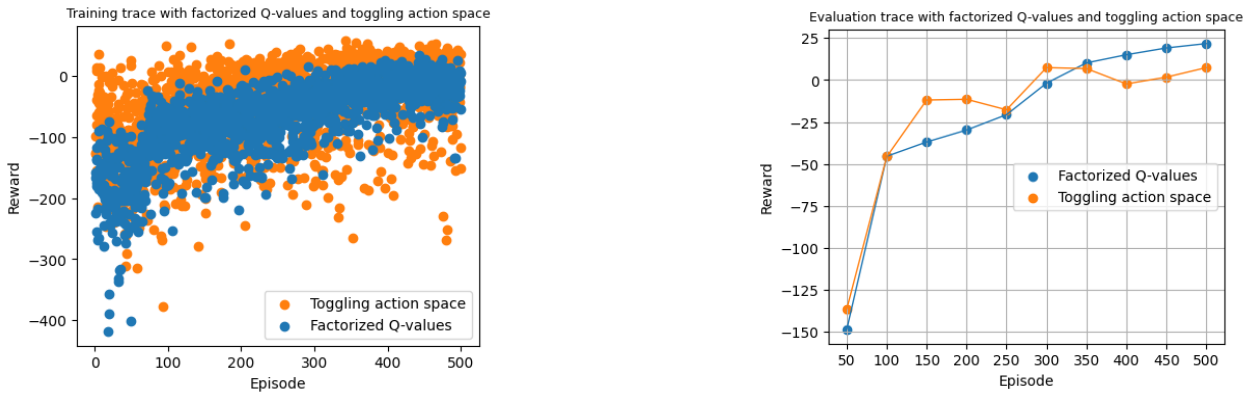

Figure 4.4: Training and evaluation traces with factorized Q-Values and toggle action-space

As can be observed in the right plot, when using factorized Q-values, the evaluation trace does not encounter the aforementioned instabilities and seems to converge to a higher cumulative reward value. The network's structure allowing an independent choice of the action seems to compensate for the instabilities observed during the training of toggling action-space policy. However, using factorized Q-Values leads to a slower increase of the evaluation trace compared to the other policy. This implies that the agent is learning the policy slower. However, it achieved a cumulative reward of around 25 at the end of the evaluation process but only around 7 using a toggle action-space. Therefore, based on the above plots, we can assume that the agent learns a meaningful behavior.

Figure 4.5 shows one of the episodes evaluated using the best policy using factorized Q-Values π^*_{Fac} learned. We can see that the number of infected drastically decreases due to a simultaneous activation of confinement, isolation, and vaccination. The number of deaths is pretty stable due to the activation of additional hospital bed days.

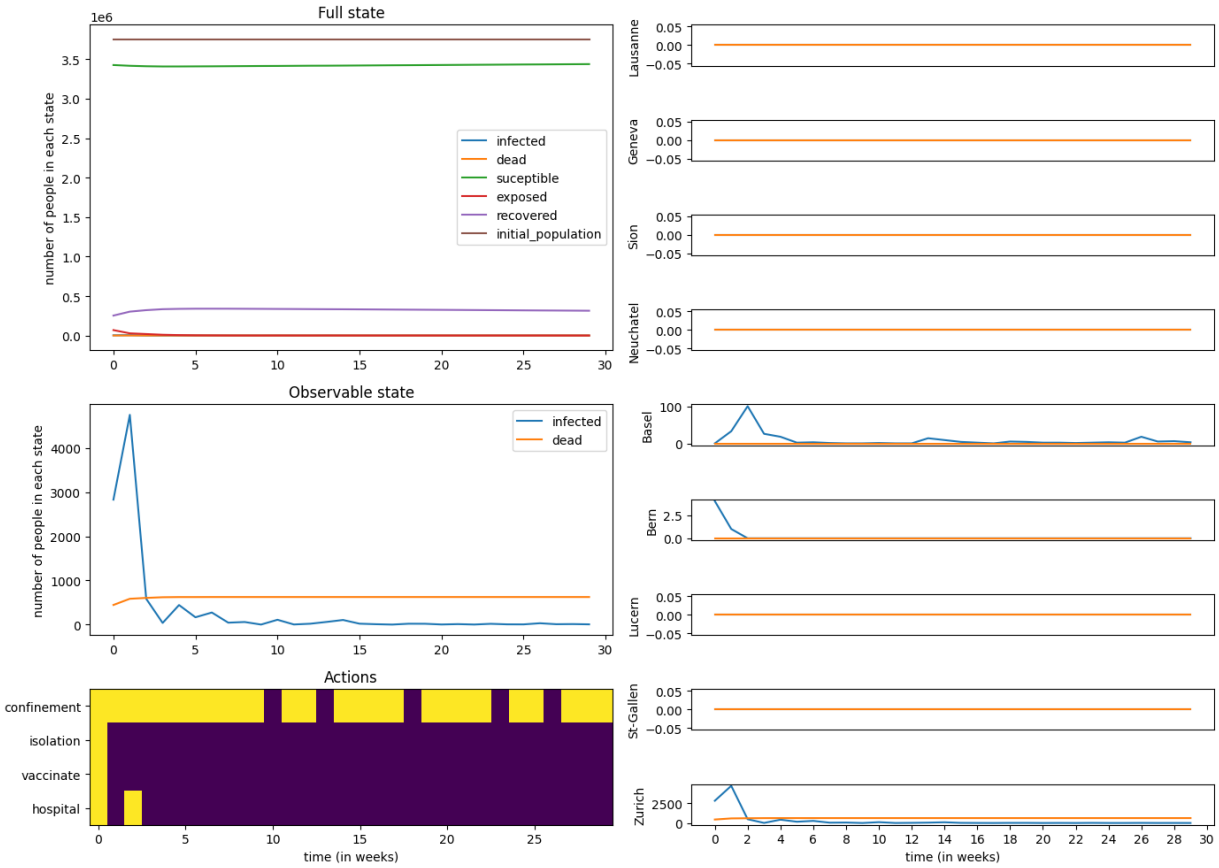

Figure 4.5: One episode simulation with the best policy using factorized Q-Values π^*_{Fac}

4.2.2 Multi-action factorized Q-values policy evaluation

To better evaluate the best policy learned using factorized Q-Values, let's run 50 episodes following this policy π^*_{Fac} and plot histograms of the quantities of interest. The results of these evaluations are summarized in Figure 4.6.

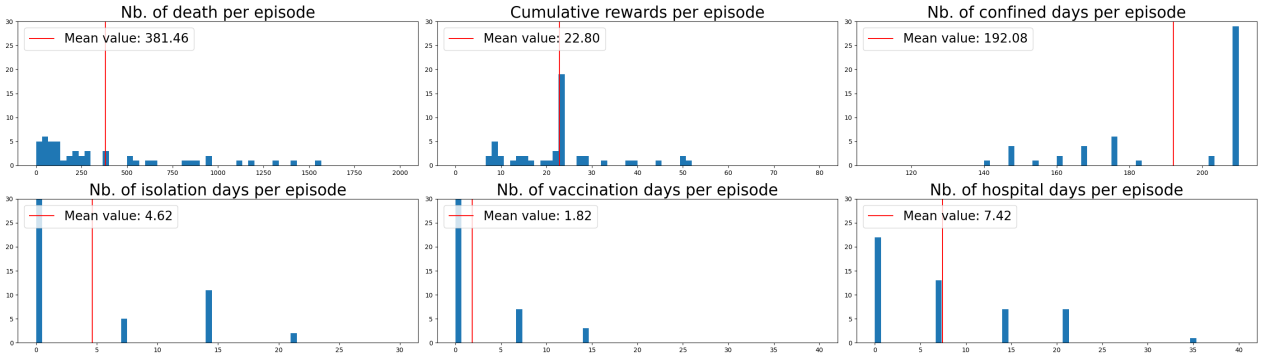

Figure 4.6: Histograms evaluating the best policy using factorized Q-Values π^*_{Fac}

When compared to the policy π^*_{Toggle}, one can see that the policy π^*_{Fac} performs much better. Indeed, the mean cumulative reward is higher and the mean number of deaths is around 30 times lower. To achieve these results, the policy π^*_{Fac} uses a higher number of confined days per episode (192 vs. 154 in average respectively) and of vaccination days per episode (1.82 vs. 0.00 in average respectively). Those performances can be explained by the fact that the policy π^*_{Fac} takes advantage of every action as can be seen in the histograms to better adapt to specific situations.

4.2.3 Factorized Q-Values, what assumption does it make?

The use of factorized Q-Values implies that the action-space can be represented as a set of decisions, where each decision is associated with a binary value (True or False). Moreover, it is assumed that each decision can be made independently of others. This assumption allows the factorized Q-Values model to compute the

Q-Value of the complete action as a sum of the Q-Value of each decision at a certain state.

However, factorizing Q-Values is not suitable for action-spaces where the decisions are not independent or exhibit strong dependencies. For example, in a game, an action might involve a series of movements or interactions. In such cases, factorized Q-Values might not be stable, as it does not account for the temporal dependencies and the extended nature of the actions.

5 Wrapping up

Let's now take a step back and compare the performance of the different policies learned by the agent.

5.1 Comparing the training behaviors

Now that five different policies were applied to mitigate the MARVIN23 epidemic, their comparison is relevant in order to determine which is the best performing one. The advantage of Pr. Russo's policy is that it does not need training and is, as a consequence, quick to implement. However, the number of deaths and of infections is rather high as explained earlier (see Figure 2.1 and Figure 2.2). We saw earlier that implementing a Neural Network with Deep Q-Learning using the same action-space as Russo's policy leads to an improvement (see Figure 3.4) highlighted by a higher cumulative reward and a lower number of deaths. Although using a Deep Q-Learning requires a non-deterministic training process, we saw (see Figure 3.3) that using decreasing exploration reduces the noise during training and leads to a higher cumulative reward by exploiting more the policy that the agent learns as being the best.

Leaving the binary action-space, we saw that using a toggle action-space including the five possible actions in this setting (*do nothing*, *confinement toggle*, *isolation toggle*, *vaccination toggle*, *hospitalization toggle*) might not bring improvements to the DQN strategy as the cumulative reward is lower and the number of deaths is higher (see Figure 4.3). However, the toggle nature of the action-space leads to instabilities during the training. Moreover, as the action-space is more complex, a higher number of training episodes might be required to achieve convergence of the policy. As a consequence, it would be interesting to train the agent on more episodes to see if the toggle policy can match the simple DQN policy performance.

Finally, the agent was given the possibility to use any possible combination of actions in the action-space to mitigate as well as possible the MARVIN23 epidemic in the factorized Q-Values setting. Although the training was slower than using the toggle action-space, it achieved a higher performance than the toggle policy in the end (see Figure 4.3, Figure 4.4, and Figure 4.6). This shows that the agent was able to take advantage of these new possibilities to learn an effective policy to mitigate the epidemic. However, the action-space is even more complex than in the toggle action-space case with 16 (2^4) possible actions at each time-step (every possible combination of *confinement*, *isolation*, *vaccination*, *hospitalization*). Again, in this case, a higher number of training episodes might allow to achieve an even higher performance as shown by the evaluation trace (see Figure 4.4) still growing at the end of the 500 training episodes.

5.2 Comparing policies

As a summary of the previous section, Table 5.1 shows a quick comparison of the different policies learned by the agent. This table was built averaging the different quantities of interest over 3 evaluation episodes of each best policy (π^*_{Russo}, π^*_{DQN}, π^*_{Toggle}, & π^*_{Fac}). Looking at the cumulative reward (written in the table simply as "Reward" to save space), the best performing policy is the π^*_{DQN} one.

Policy	Deaths	Reward	Confined Days	Isolated Days	Vaccination Days	Hospital Days
Russo	59053.52	-70.62	**98.98**	*N/A*	*N/A*	*N/A*
DQN	4452.52	**44.24**	153.44	*N/A*	*N/A*	*N/A*
Toggle	12204.68	15.49	154.56	4.76	**0.00**	46.48
Factorized	**381.46**	22.80	192.08	**4.62**	1.82	**7.42**

Table 5.1: Comparison between the different policies learned by the agent

In Table 5.1, the best policy according to each metric is easily recognizable by looking at the numbers in bold. The π^*_{Fac} policy achieves the lowest number of deaths but uses the most confinement days and vaccination days to reach it. The number of additional hospital bed days is low for the policy using factorized Q-Values as it achieves a quick decrease in the number of infections and, indirectly, a low number of deaths (see Figure 4.5). Comparing π^*_{DQN} and π^*_{Russo} which both use the same action-space, one can observe that π^*_{DQN} achieves a higher cumulative reward by using more confined days than π^*_{Russo}.

5.3 Q-values interpretability

It is now interesting to analyze the evolution of the simplest policy using Deep Q-Learning and a binary action-space (π^*_{DQN}) in comparison to the most complex one using factorized action-space (π^*_{Fac}). These can be seen looking at the heat-maps provided below (Figure 5.1 & Figure 5.2). The heat-maps provide a quick look on the evolution of the Q-Values of the different actions (vertical axis) at each time step of an episode (horizontal axis).

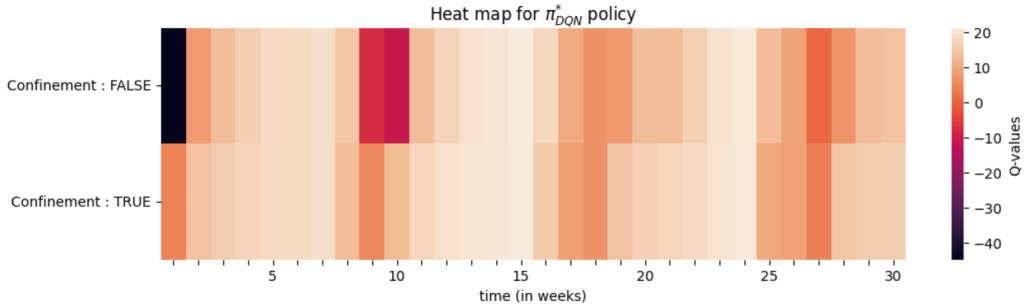

Figure 5.1: Heat-map of the evolution of the Q-Values for the best DQN policy π^*_{DQN}

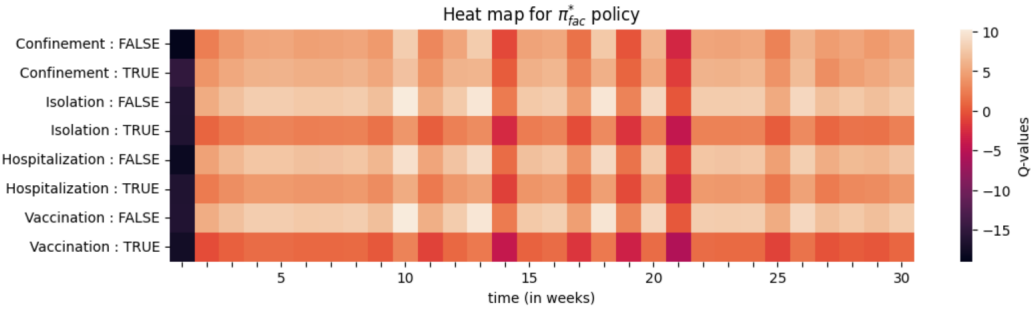

Figure 5.2: Heat-map of the evolution of the Q-Values for the best policy using factorized Q-Values π^*_{Fac}

The π^*_{DQN} policy is rather simple to interpret considering the binary action-space. The agent confines the country when the infections are above a certain threshold. On the other hand, it is more difficult to interpret the π^*_{Fac} policy as the agent has a higher flexibility between different actions. However, as can be observed, the agent takes advantage of the whole action-space to try and find the most effective policy. Figure 4.5 and Figure 4.6 provide a more detailed view on what happens in the background. It seems that the policy most often does not need isolation, vaccination, or hospital bed days to mitigate the epidemic but it rather prefers to use confinement.

Figure 4.5 shows that the agent first activates all actions to reduce as much as possible the initially high number of infected people. Indeed, the primary focus of any policy is to minimize the number of deaths as it is the element having the heaviest impact on the reward (highest cost) and infected people are, in this model, the only ones that can die. It then, uses confinement which has a high cost but strongly reduces the exposition to the virus leading to a low number of infections. It also uses (rarely) hospital days which does not impact infections but decreases the number of deaths. Isolation and vaccination seem to be used when the number of infections is high and the agent needs to quickly contain a peak of infections.

5.4 Is cumulative reward an increasing function of the number of actions?

The cumulative reward is not necessarily an increasing function of the number of actions as can be seen in Table 5.1. Indeed, more actions translates into a more complex training process which, in turn, translates into a cumulative reward still growing after the 500 training episodes considered here. Moreover, the impact of new actions depends on the nature of these actions. Adding a repetition of the already existing actions would lead to a higher number of actions but not impact the reward as the achievable output stays the same. Adding actions that have no impact on the reward (either directly or indirectly) would also not impact the cumulative reward as those actions would simply never be chosen.

However, given a high enough number of training episodes, an increased number of meaningful actions (in the sense that they bring more control over the state space) would lead to a higher cumulative reward after convergence as the agent can better adapt its policy to different states.