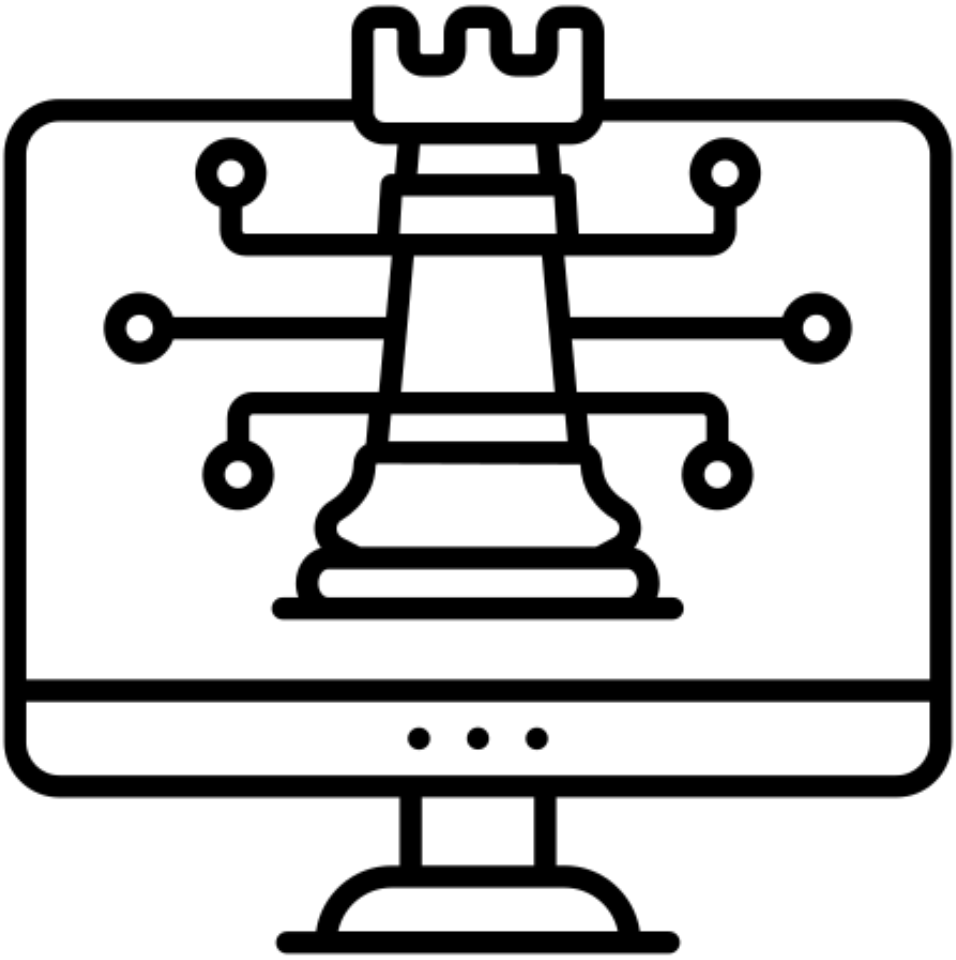
Research document : Game Reinforcement Learning

By Mickey Krekels

Class: AI7-MC2



Monitor Screen free icon: flaticon.com

VERSION HISTORY

Version	Date	Author(s)	Changes	State
0.1	12-09-2022	Mickey Krekels	Added the main structure of the document.	In progress
0.2	23-09-2022	Mickey Krekels	Added chapters 1 and 2	In progress
0.3	27-09-2022	Mickey Krekels	Added chapters 3	In progress
0.4	15-10-2022	Mickey Krekels	Added chapters conclusion	In progress
1.0	17-10-2022	Mickey Krekels	Fixed some grammatical errors	done

TABLE OF CONTENTS

GLOSSARY

In this part of the document the overall terms , acronyms and abbreviations will be explained.

ACRONYMS AND ABBREVIATIONS

Abbreviation	Meaning
MoSCoW	Must have, Should have, Could have, Won't have
CNN	Convolutional Neural Network
AI	Artificial Intelligence
GDPR	General Data Protection Regulation
RL	Reinforcement Learning

INTRODUCTION

1.1 ABOUT ME

My name is Mickey Krekels and I am currently in semester 7 as an IT student at Fontys University of Applied Sciences. Before Fontys however, I studied Game development at the MBO SintLucas, which gave me a good understanding of programming and UX design. During my specialization on Fontys, I studied the subject of AI with the main focus set on neural networks.

1.2 THE DOCUMENT CONTENT

For the semester 7 assignment the open programme, I decided to focus on learning Reinforcement Learning. To make thing more challenging this will be done in combination with a game or simulation. The document will contain the research and project results/findings.

1 RESEARCH QUESTIONS

In this part of the report, I will describe the research questions. This will be done by using the Dot Framework research methodology (1). Based on the project goal and the amount of focus on improving the project, the following main question became:

1.1 MAIN RESEARCH QUESTION:

- How is it possible to learn an agent to play a video game or simulation with deep learning?

To provide an answer to the main research question, the techniques and algorithms used must be analysed and researched. This way I could understand the underlining structure and from that point, I could decide what options for improvements are available. Therefore to provide an answer to the main question these sub-questions are required.

1.2 SUB QUESTIONS:

1.2.1 What is reinforcement learning?

1.2.2 What reinforcement learning algorithms are there?

1.2.3 What types of frameworks are already available?

Sub Question	Explanation
1.2.1	This question is needed, In order to understand how reinforcement learning works.
1.2.2	There are many types of reinforcement learning algorithms available. The answer to this question provides a perfect chance to find out which algorithm is most suited for me to learn
1.2.3	Reinforcement learning is not a new technology, this means that there are other sources on the internet available, that going to help me understand how to implement it in a video game or simulation.

2 INITIAL PHASE

2.1 WHAT IS REINFORCEMENT LEARNING

Reinforcement Learning or RL in short is an environmental-based feedback-based machine learning approach. The AI-driven coordinator or agent is trained by using a penalty and reward system.

This is typically done by running the agent in an environment for a certain amount of cycles. With each cycle, the agent is rewarded or punished for its achievements during that run. This could be a goal like a score or a sum of sub-goals. There are many types of algorithms available that use their own type of system, but generally, there are four key components.

The four key components of reinforcement learning are as follows:

1. Agent. the system you train with the intention of doing a certain job.
2. Environment. the physical or digital world in which the agent operates.
3. Action. is the movement performed by the agent that modifies the environment's condition.
4. Rewards. An action's evaluation, which may be positive or negative.

(see figure 1 below for a visualization of these components)

To add diversity random values are used on the network of the agent to further encourage its evolution. Algorithms like Epsilon-Greedy (2) are mostly used for this task.

This setup allows reinforcement learning, to learn difficultly problems which are impractical for training standard ANN or CNN networks.

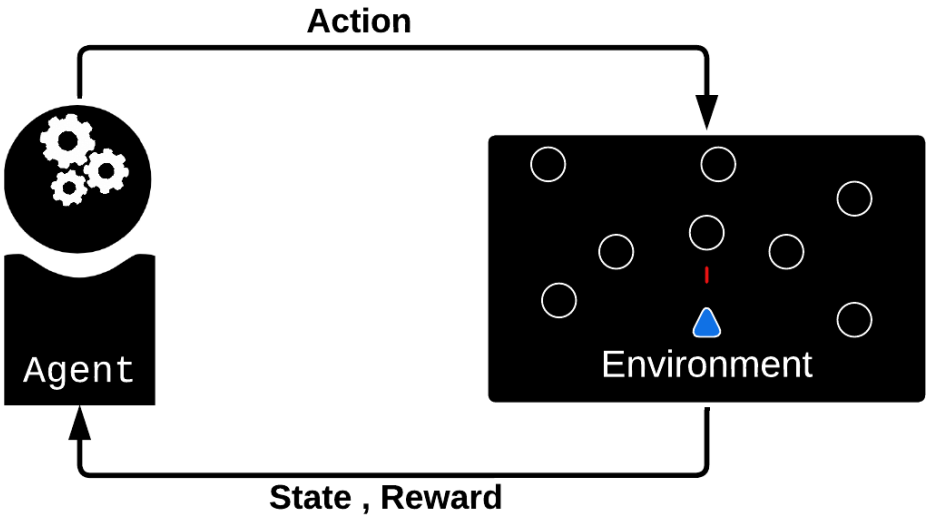


Figure 1 visualization of essential elements of reinforcement learning

2.2 WHAT TYPE OF REINFORCEMENT LEARNING ALGORITHMS ARE AVAILABLE

In this section of the document, we're going to look at the different types of reinforcement learning techniques available. Each of the advantages and disadvantages of this topic is going to be explained. Hopefully, this research will draw a better understanding of what algorithms are appropriate for this project to use.

2.2.1 Q-Learning

The Q-Learning RL technique is based on the well-known Bellman Equation, it is named after the mathematician Richard E. Bellman. The reshaped version of this is currently used in the Q-learning technique, the goal of this algorism is to maximize the overall Q-value (figure 2, displays the formula below).

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{current value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{current value}} \Big)}^{\text{temporal difference}}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

Figure 2 Q-learning formula: https://en.wikipedia.org/wiki/Q-learning

In code, the values are saved in a table where the trained model can look up these values at any given state during the simulation in their environment (such as a game).

2.2.2 Deep Q-learning

Deep Q-learning is a variation of the regular Q-learning, it used the same technique in combination with a CNN or convolutional neural network. It used Experience Replay, which saves the given state of the previous steps taken by the game or simulation. This data is then used in small portions by the model for training.

2.2.3 Deep Double Q-Learning

In order to address the overestimation issues with classic Q-learning, the double Q-learning method uses double estimation. It is an off-policy reinforcement learning technique. The same parameters are used for the selection and evaluation of an action by the max operator in both regular Q-learning and DQN.

2.2.4 Conclusion

There are also other examples that have their own implementations of Q-learning. Such as distributional Q-learning, delayed Q-learning and Greedy GQ which have their own advantages in certain situations. But for this project, the technique of Deep Q-learning is going to be applied. This is the most common type and it is a combination of deep learning, which alines with the learning outcomes of this AI semester.

2.3 WHAT TYPES OF FRAMEWORKS ARE ALREADY AVAILABLE

Now that we have a clear understanding of what technique is going to be used, the next step is going to find out if there are any tools/frameworks available.

2.3.1 Gym library

The Python library Gym (3) is a diverse collection of game and simulation environments. The environments can be easily downloaded with the help of their API. The collection includes Atari (arcade games), MuJoCo (3D physics-based simulations), Toy Text (simple games suitable for debugging implementations), Classic Control (2D physics-based simulations), Box2D (simulations using box2d based physics) and Third Party Environments (such as the game Doom or Unity ML-Agents).

2.3.2 RLlib

RLlib (4) is an open-source library for reinforcement learning. It is widely used in combination with game engines such as Unity, unreal engine and others types. The library shows lots of promise for this project, but if most of the complex is already taken care of this project would not be that learn full. Therefore this could be a great tool for just testing.

2.3.3 Unity Machine Learning Agents

This unity module (5) removes the need to code emergent behaviours. Therefore the tool allows game developers to focus on creating gameplay and an enhanced game experience. This has the same problem as the RLlib library but on an even bigger level, even if the agents are able to learn and perform great. The goal of the project is to learn and implement deep-reinforcement learning. Because of this, the technique will not be used in this project.

2.3.4 Conclusion

The chapter has provided a deep look at the current tools available for reinforcement learning on the market. The library Gym provided an excellent structure to learn how to implement RL in different types of environments. This will be using full for the early stages of the project. The open-source library RLlib could be investigated further, but it should not solo be used as the main block of the project. And finally, Unity Machine Learning Agents are indeed great to have when building a game, but the simplification does not allow for learning how the RL techniques truly work. Therefore the gym library is the best suited, for learning this AI technique.

3 CONCLUSION

In the previous chapters, we explained what reinforcement learning is and how to implement it. The first chapter explained the basic theory behind reinforcement learning, this structure is common among most implemented examples. The environment provides a state and a reward value on each step, and agents provide an action based on a decision-making algorithm (neural network, q-table).

The second chapter covers most of the common algorithms available for deep reinforcement learning. Lots of different types of algorithms, but the one most suited for this AI semester was chosen to be Deep Q-learning.

The last section dives deeper into frameworks that are available and support reinforcement learning. Most of the frameworks that I found focus more on providing the end users with as little amount of coding as possible. It meant I couldn't look into the code itself, which was a problem. Out of the 3 frameworks, the Gym library came out to be the best suited for learning. This provided me with full control of the code, and it was made in Python, which I was also familiar with.

Based on these findings, I can honestly say that I've answered the main research question, “How is it possible to learn an agent to play a video game or simulation with deep learning?”. With these steps done, I cannot start working on my first prototype.

BIBLIOGRAPHY

1. **The DOT Framework.** ***ictresearchmethods.*** **[Online] https://ictresearchmethods.nl/The_DOT_Framework.**

2. samishawl. epsilon-greedy-algorithm-in-reinforcement-learning. ***geeksforgeeks.*** **[Online] https://www.geeksforgeeks.org/epsilon-greedy-algorithm-in-reinforcement-learning/.**

3. Gym is a standard API for reinforcement learning, and a diverse collection of reference environments. ***gymlibrary.*** **[Online] https://www.gymlibrary.dev/.**

4. Team, The Ray. RLlib: Industry-Grade Reinforcement Learning. ***docs.ray.*** **[Online] 2022. https://docs.ray.io/en/latest/rllib/index.html.**

5. Unity Machine Learning Agents. ***unity.*** **[Online] https://unity.com/products/machine-learning-agents.**