A report on applying reinforcement learning techniques on the Cartpole problem using Q-Learning.

Sean Mcguire

**Abstract.**

Reinforcement learning (RL) is a subcategory of machine learning (ML) and Machine learning is also a subcategory of Artificial Intelligence (AI). The purpose of this project was to use RL and understand what RL can be applied to. The main algorithm I focused on was the Q-Learning algorithm which is based on the Bellman equation which is a reward-based algorithm. What makes Q-learning different to these common machine learning techniques is eliminating the labels, training sets and supervisors and solely relies on being able to monitor the response of taking an action, and measuring the reward returned from the action taken.

This report is based on the work I completed for my final year project and what I have learned in the process of completing this project. The initial idea for my project was to apply a RL algorithm to an environment provided by the gym library developed by Open AI. The technologies used in this project are Python, Java with eclipse and GitHub. The python is used for the server and the GUI to display the Cart-Pole. Then the java code is used to connect to the server on the localhost sending and receiving information on the cart pole.

**Keywords:** Reinforcement learning, Machine Learning, Artificial Intelligence, labels and training set, Q-Learning, Bellman equation.

1. Introduction

Reinforcement learning is a relatively new machine learning technique. It has qualities from the well know supervised and unsupervised learning techniques however, it learns from mistakes rather than applying labels to the data. RL is a reward-based system. First researched in 1989 by Christopher Watkins in a paper called “Learning from delayed rewards”. Watkins devised an algorithm referred to as Q-learning which greatly improved the practicality and feasibility of RL [1]. Q-Learning has been used in different ways and is the most common reinforcement learning algorithm.

1.0 Background

In this section we will cover some background information on reinforcement learning, Q-Learning, the gym library and the java implementation used for this project.

* 1. Reinforcement Learning (RL)

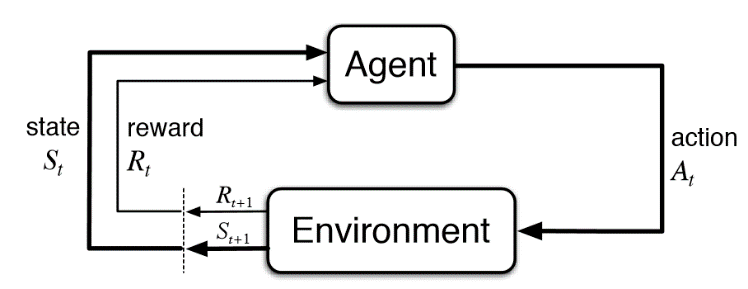
RL is a powerful machine learning tool where an agent learns to behave in an environment, executing actions and observes the rewards it receives from the actions. The agent generally has no prior knowledge on what actions are good or bad initially. An example of an agent could be a character in a video game and the action could be jump or move right. In RL there is a discount factor used to calculate the future reward. The discount factor essentially makes the future reward worth less than the immediate reward in order to fight against delayed gratification. The environment is the world in which the agent interacts with.

* 1. Q-Learning

Q-Learning is an off-policy RL algorithm and is one of the most common RL algorithms used for RL problems. Q-Learning uses Q-values (action values) to iteratively improve the actions taken of the agent. Q-learning does not require the agent to have any information about the environment, it works on estimating the state-action values in what’s called a Q-value. The Q-values are defined for states and action pairs. When using the Q-Learning algorithm we use a Q-Table to store all Q-values in a matrix.

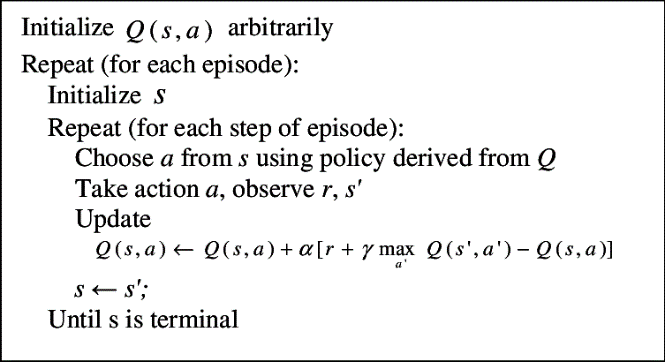
Initially we can set every Q-value to zero, iteratively updating each state action pair Q-value by selecting actions. The action taken is based on the reward taken by a specific state action pair and chooses the best action for that pair. Upon the transition from one state to another if it ends up in one of the termination states this means there are no more transitions or actions to be taken. This is referred to as completing an episode. After a while of exploration, the Q-table gets filled with values that can referred to as the most optimal actions to be taken in each state action pair.

**Fig. 1.**



This is also considered a Markov decision process (MDP). An MDP consists of a reward function, a set of states, a set of actions and a transition function. In this specific example, the sets of states, actions and rewards all have a finite number of elements. The learning agent will attempt to find a balance between exploration and exploitation when taking actions in the environment. The agent will improve its knowledge of the environment and will make better decisions in the next action.

**Fig. 2.**



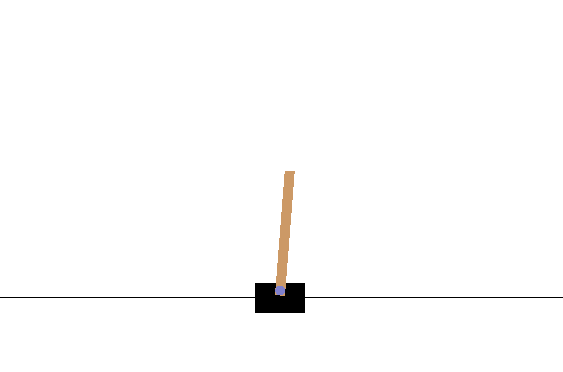
If all state-action pairs are given a Q-value in a certain environment, the Q-values are guaranteed to converge to the optimal values. It will always choose the highest Q-value from the current state it’s in.

* 1. OpenAI’s gym library

The OpenAI gym is a library that includes several environments including the cartpole, mountain car, a swinging pendulum and many others. The gym library used for this project is the gym http API as it allows you to run the gym library in a python server separate to the client. The aim of the library is to develop and compare different reinforcement learning algorithms. The reason why I chose to use the cartpole environment was that it was one of the earliest and most common environments. In this environment there is a simple cart that can either go left or right on a single 2D plane with a pole attached to the cart at its center. In this example the agent is the cartpole object in this environment, taking actions to either go left or right.

Gym provides the traditional cart-pole problem which is widely used in testing RL algorithms. The cart-pole was initially developed to test adaptive control and became very popular tool first researched in 1986. The Cart Pole environment (Fig. 3) allows testing of RL algorithms. This binary classification problem has 4 inputs: cart position, velocity, pole angle and pole velocity at the tip of the pole. These inputs are used as the current state matrix. The agent in this example must produce an action of either push the cart to the left or to the right based on the Q-Value in that current state of the 4 states.

**Fig. 3.**



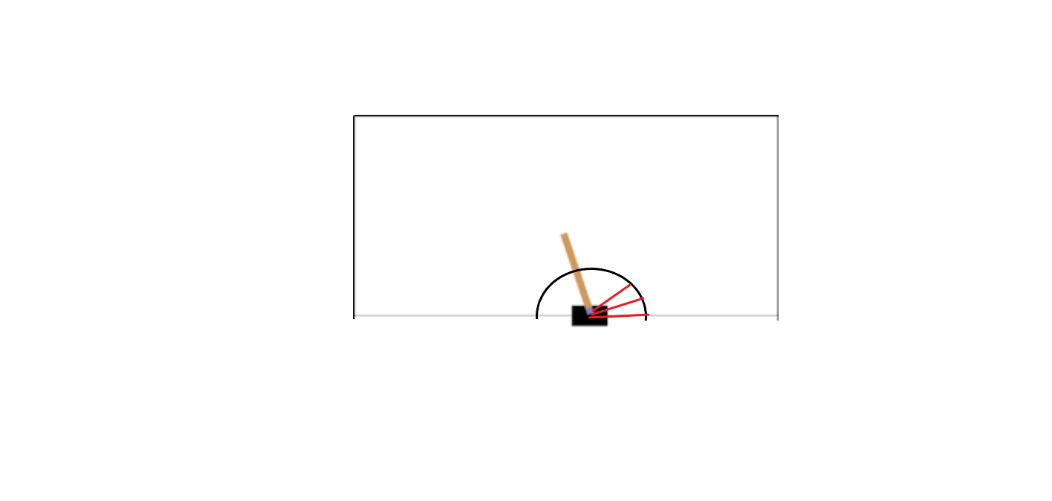
The Wiki of CartPole v0, “A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The pendulum starts upright, and the goal is to prevent it from falling over by increasing and reducing the cart's velocity”. The four observations used are: Cart Position, Cart Velocity, Pole Angle and Pole velocity at tip. The actions are binary 0 or 1, push the cart left or right. The reward is 1 for each step taken. The environment will reset for each of these outcomes: the pole angle being greater than 12 degrees on either the left or right side, the cart position is greater than 2.4 on left or right (the center of the cart reaches the edge of the display) or if the episode length is greater than 200 (the best possible value). The CartPole problem is considered solved if the CartPole reaches the value 195 or greater iteratively 100 times in a row.

* 1. Java implementation

The core part of this project is the development of the client side connecting to the gym library python server using JavaScript Object Notation (JSON) calls. Once we run the python server, we can connect to the server using any programming language that can send and receive JSON objects. Initially we run the python server locally on a specific port (default is 5000), then we run the java client that connects to the server on that port. The java client then creates an instance of an environment (Fig. 3) and an agent (the CartPole). Initially setting the Q-table values to zero. The client will iteratively send the action to take based on the previous state, current state, the selected action and the reward, in doing so filling the Q-table with optimal values. The updated Q-value method uses a temporal difference (TD) learning technique, which is an example of a model-free RL approach.

Sorting the observation values (returned from the server) from their original values into buckets was required as the values returned were very large decimal values. Each observation has their own minimum and maximum value, so in order to sort these into their own buckets you need the minimum value, maximum value, the observed value and the number of buckets for that observation. For example, the Pole angle observation could be 0.55815… and we could say we want 40 buckets that we want to have to deal with, this would sort this observed value into bucket 5. Below the image (Fig 4) depicts the pole angle buckets (in red). If the pole enters inside one of the three red areas it will only have to deal with that specific bucket and not the long decimal number. This bucket sorting method is also used for the 3 other observation values.

**Fig. 4.**



Over time the values in the Q-table become visibly more optimal and the average reward from each one hundred episodes will increase linearly. Ideally the average episode value over 100 iterations would be 195 or greater. In my java implantation, after around 2000 episodes on average the agent will reach an average of 200 (maximum reward) over 100 episodes. Once the java client completes every episode specified, it will print to a CSV file to later graph the overall average reward.

1. Methodology

The following section describes the different stages of the development of this project. Followed by the different implementations and the development process. Furthermore, the technologies used to carry out the project.

* 1. Agile Approach

The approach to this project was an agile approach. Every week I referred to my project supervisor to hold scrum meetings. initially getting the client server up and running was the main task and so the development process for this stage took up most of the time. Once the client and the server could send and receive information it was onto the second part of the project which was getting the java client to send and receive states and actions. Consistent testing was completed before pushing to GitHub to make sure every commit was a fully functioning and tested release. The third stage of the project was figuring out the optimal values for each observation bucket, this required a lot of research. The final stage then was using Excel to generate multiple graphs with different values for alpha and gamma and if either are decaying over time. also graphing different values for the buckets.

For each stage of the project there was a requirements analysis carried out between myself and my supervisor. In order to complete each stage, it had to work with the previous stage, so it was crucial to always have a current working version available on GitHub. The feedback from my project supervisor was very useful in order to carry out the next stage. Having weekly meetings allowed for weekly scrum meetings, these meetings usually lasted around 30 minutes and consisted of discussing what objectives and tasks have been completed and what needs to be done next.

2.2.1 Requirements Gathering & Analysis

The first sprint initially was investigating what different paths we could go down in the reinforcement learning topic. The initial idea was to build a 3D unity game where there would be a car that used RL to navigate around in the virtual world which later changed to using the Gym open AI library, which already has an environment built readily available. Extensive research of reinforcement learning was required initially in order to get an understanding the extent of the project scope. Once we figured out what technologies we wanted to use it was onto the next task of building a java client using JSON.

2.2.2 Design

After extensive research and analyzing the requirements it was onto the design and implementation phase of the project. The technologies and tools that we needed to use had to be tested to work together, rough sketches of how the project was going to be designed was carried out between myself and my supervisor. The core aspects of the project required extensive research onto how the implementation would be carried out. Organizing the tasks into different sprints was carried out in order to give an estimate on which parts of the project would require more time.

The core design of the project was then devised to include a java client using JSON send and receiving requests to interact with OpenAI’s http gym python server, the client would contain the Agent and Environment classes and within the Agent class would hold the Q-learning algorithm. Another class for file handling was also necessary for outputting the data in CSV format to Excel graphs.

2.2.3 Research

Extensive research on the topic of reinforcement learning and Q-Learning was required to pursue this project. Initially looking towards basic google searches towards countless research papers aided in the understanding of RL. My project supervisor also provided a wide range of material for development.

The initial stage of the research consisted of gathering multiple research papers and blogs on applying Q-learning, using the Gym API and definitions of RL. Whilst ongoing research was required in order to progress the project testing of code samples online was also conducted, usually in python code, converting parts of this code into java to work with the project.

2.2.4 Version control

Throughout the development of the project GitHub was used to manage, maintain and track the progress of what has been completed. GitHub is a free online version control web-hosting service which uses git. GitHub was useful for controlling the current state of the project for, if for any reason, data was to be lost locally, GitHub would provide a constant working version always readily available to revert to. GitHub also provided a visual side by side code comparison which would allow for comparisons between different pushes to the master.

2.2.5 Testing

Testing of individual parts of the program was carried out continuously. In the event of adding more code to the project testing would make sure it was functioning as it was designed to function. For example, when it came to test the function of sorting the raw inputs into buckets, testing on maximum value and minimum value handling was carried out. At times the maximum value would be too large for say the carts position and so thinking out how to handle larger value sizes was a task to be completed after testing.

2.2.6 Implementation

In the implementation phase of the project a large portion was designing the java development process. Initially developing the client to send and receive basic inputs from the Cart Pole server and later moving onto applying the Q-learning algorithm to update the Q-table. Upon the implementation phase several issues occurred, it was not clear why the Q learning algorithm was not actually learning, testing was required in order to figure out why this issue was happening. With the use of excel we ran the program for several hours at times changing several factors that could affect the learning rate of the algorithm. After testing each factor, we graphed the average reward over 2500 episodes to visually see if the program was learning, what we expected was the trendline to sloping upwards with respect to the number of episodes. It was clear that the learning rate (alpha) had a huge impact on how the RL algorithm learns. After fixing this issue we focused on reaching the goal of the Cartpole which was reaching an average score of 195 or greater for 100 consecutive runs. After testing different values for alpha, gamma, epsilon and the bucket sizes, we found values ideal for the program and implementing an alpha decay function also hugely affected the outcome.

1. Technology Review

This section is dedicated towards the review of technologies used. There will be a brief description of all the technologies and why they are used for this project.

* 1. Python

Python is an interpreted, high-level, general purpose programming language. Created by Guido van Rossum and first released in 1991. Python allows for development in building GUI applications, websites, web applications and many more. Python is a user-friendly language yet is also very powerful. Python is used extensively for data analysis, machine learning and large frameworks.

Large corporations such as Google, Spotify, Facebook and even NASA use Python for carrying out tasks. Because of its flexibility, Python can be integrated with many other programming languages. The IEEE ranked Python as the #1 programming language in 2018. GitHut is a visualizing tool to view the rankings of various languages and ranked Python 3rd on the list.

* 1. PIP

Pip is a package-management system used to install and manage software packages written in Python. Many packages can be found in the default source for packages and their dependencies — Python Package Index. Python 2.7.9 and later, and Python 3.4 and later include pip by default. PIP is used to install the libraries used for the Gym HTTP API including Flask, numpy, gym, requests and pytest.

* 1. Gym Library

<https://github.com/openai/gym>

The Gym library is a freely available toolkit used to compare RL algorithms. The library contains virtual environments that can allow an agent to learn a wide range of different challenges from Atari games, learning to walk, CartPole and many more. Open AI is a nonprofit research company that is focused on developing AI tools. Founded by Elon Musk and Sam Altman, their mission is to “build safe AGI, and ensure AGI’s benefits as widely and evenly distributed as possible”. Developers will tend to use the gym library directly by importing it into their python code.

* 1. Gym’s HTTP API

https://github.com/openai/gym-http-api

Gym’s HTTP API provides a rest API to the gym open source library. This allows to interact with the gym library with any other language that supports sending and receiving HTTP calls. Initially getting started with using the gym HTTP API we first need Python installed with pip. The [gym\_http\_server.py](https://github.com/openai/gym-http-api/blob/master/gym_http_server.py) file is all we need to run in order to get the server up and running.

The core interactions between the client and the server are a series of HTTP requests. These include POST and GET requests, for example creating an instance of a specific environment, getting a list of environments running on the server, resetting the state of the environment and step through an environment with an action which returns the valuable information to allow the agent to learn.

* 1. RESTful implementations with JSON

REST (Representational state transfer) is a software architectural style that defines a set of constraints to be used for creating web services. Web services that conform to the REST architectural style, termed RESTful Web services, provide interoperability between computer systems on the Internet. Some examples for the use of REST is for when we send and receive values from Gym’s HTTP API.

* 1. Git

Git is a distributed version-control system for tracking changes in source code during software development. It is designed for coordinating work among developers and can also be used for tracking changes in any file. Git stores information in a data structure called a repository. Git allows you to locally commit stages to store the version of your project and can be referred to using git logs and will give a list of commits.

* 1. GitHub

GitHub is a web-based hosting service for version control using Git. This cloud-based publishing tool allows for users and developers to view, share and update Git repositories online and keep tack of changes made to specific files. GitHub provides a GitHub Pro account to allow the use of unlimited private repositories whereas the free version of GitHub will limit users to only publishing public repositories and adding only up to three other users as repository collaborators.

* 1. Java

Java is a programming language and computing platform first released by Sun Microsystems in 1995. Java is a high-level, general purpose programming language that’s class-based, object-oriented and is designed to compile and run on any machine, regardless of architecture or platform (WORA principle). Java is fast, secure and reliable and is one of the most commonly used programming languages in the world. Java can be found in laptops, datacenters, game consoles, scientific supercomputers, cell phones the internet and many more.

The speed and robustness of Java makes for ideal language of choice for this project, the popularity and prior experience of the language also made for an ideal choice of language. Java is being consistently updated with new versions, currently at Java SE 12, along with new features being added and some bug fixes. In comparison to C or C++, Java does not use pointers which be a security feature, if in the event accessing a block of memory that should not have authorization for. Java has its own memory management feature which deals with the pointers for the developer. Java allows for development for fast web application development and JSON integration making it ideal for this project.

* 1. Eclipse IDE

Eclipse is an integrated development environment used in Java. It is also the most widely used IDE available. It contains a base workspace and a wide range of plug-in systems for customizing the environment. Initially released in 2001, Eclipse provides a common user interface model for Java client development, which is perfect for the use in this project.

* 1. Visual Studio Code

References

1. Watkins, C.J.C.H., 1989. *Learning from delayed rewards* (Doctoral dissertation, King's College, Cambridge).
2. http://www.scholarpedia.org/article/Temporal\_difference\_learning