Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Reinforcement Learning with Taxi-Traveller Agent

CS7IS2 Project (2019-2020)

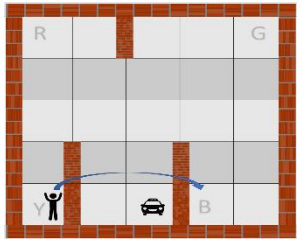
Date 14/04/2020

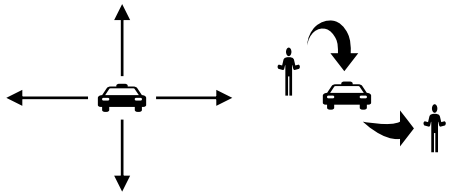
Submitted by:
Ankit Taparia 19302722
Rocky Bilei 15325585
Siddhartha Bhattacharyya 19301936
Srijan Gupta 17317499
Tanmay Bagla 19300702

Introduction

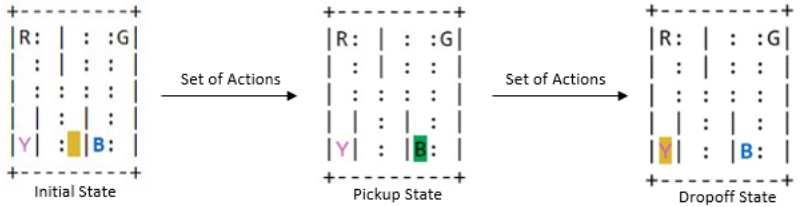
Problem Definition

- Reinforcement learning algorithms are implemented to design a simulation of a self-driving taxi which involves picking up and dropping of the passenger to the correct location.
- The OpenAI Gym environment consists of 500 state spaces and six primitive actions. These actions include four navigations i.e. North, South, East and West and two actions i.e pickup and drop.
- There is a reward of -1 for each action and an additional reward of +20 for successfully delivering the passenger. There is a reward of -10 if the taxi attempts to execute the drop or pickup actions illegally.
- The taxi problem requires an algorithm that supports temporal abstraction, state abstraction and subtask sharing which makes Reinforcement learning an ideal candidate for solving the task.





North
South
East
West
Pickup
Drop-off



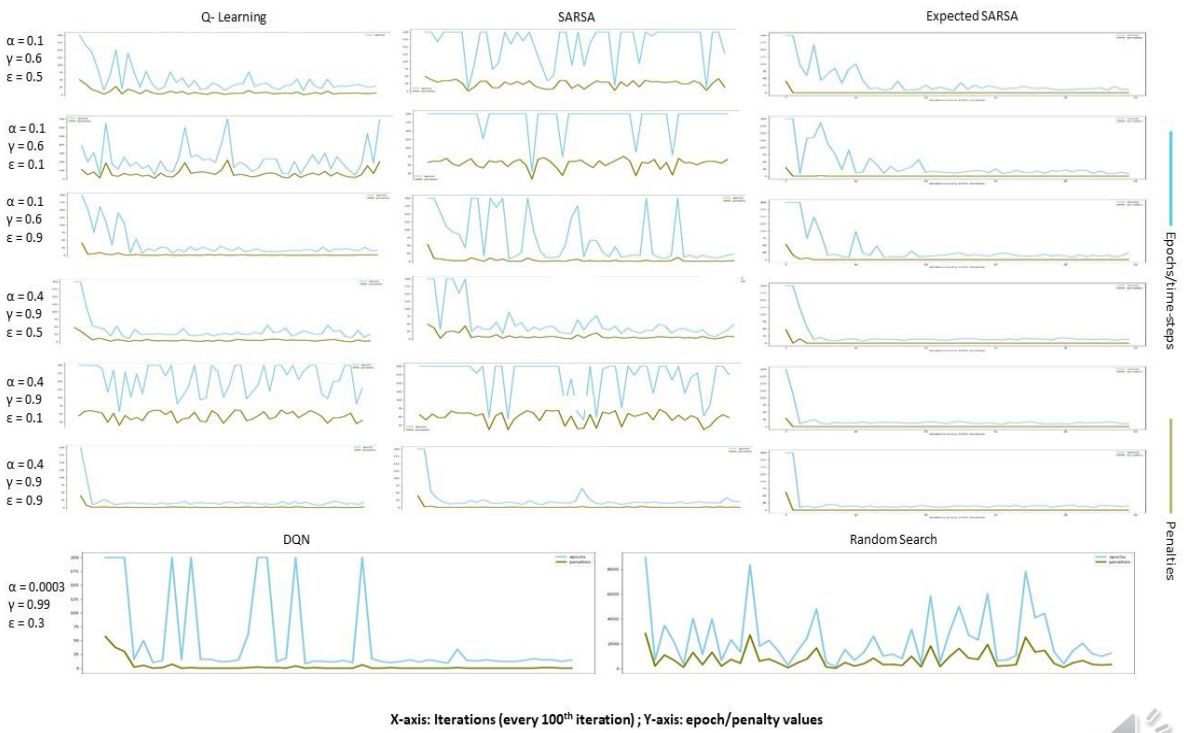
Algorithms

Random Search	Q-Learning	State Action Reward State Action (SARSA)	Expected-SARSA	Deep Q-Learning
Used as a Baseline for comparing state-of-the-art algorithms. A local search technique with very low complexity. Does not guarantee and optimal solution.	An iterative offline learning algorithm Agent learns from "rewards and penalties" it receives from the environment. Uses ε-greedy approach to choose next action. Hyperparameters = α, γ and ε	Improved version of Q-Learning using online learning. SARSA learns actions relative to the action it follows (max,mean etc) SARSA computes difference between Q(s,a) & weighted sum of average action value and maximum. Hyperparameters = α, γ and ε	Variant of SARSA that bases update on expected value of Q. Reduces variance leading to faster convergence. Action selection can occur after update unlike SARSA. Advantageous for returning actions like in Taxi problem. Hyperparameters = α, γ and ε	Variant of Q-learning that uses a feed-forward neural network as a non-linear approximator of the Q-function. Network parameters are updated at each timestep using stochastic gradient descent. Uses Experience Replay to train the network using a random sample of past transitions. Uses ε-greedy approach to choose next action.

Comparison of Algorithms

Q-Learning, SARSA, Expected-SARSA, Random Search, DQN

- The results indicate clearly that the Expected-SARSA algorithm worked best giving the least average time steps for the agent to reach its final destination, followed by Q-Learning, DQN and SARSA.
- Comparing these state of-the-art algorithms with our baseline Random Search results clearly shows immense improvement.
- Moreover, it can also be observed better results are achieved using the exploitation approach as compared to the exploration.
- Thus, it is prudent to give more weightage to the iteratively learned values as compared to letting the agent explore the environment



Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Thank You