# A comparative study of temporal difference methods on self driving cars

Team 15

# Our Team!

Ashwin Shenolikar
MEng Computer Science
NCR Campus

Himanshu Singhal
MEng Computer Science
NCR Campus

Rithvik Gottimukkala
MEng Computer Science
NCR Campus

Srija Gurijala
MEng Computer Science
NCR Campus

# Motivation

- Self driving cars - Autonomous vehicles capable of perceiving environment without human intervention
- Gateway to re-imagining future transportation systems
- Taxi System - Passenger-based reward system.
- Environment perception done using various machine learning domains such as Computer Vision and Reinforcement learning to use the various sensors

# Problem statement

The job of the cab is to pick up a passenger and drop him off along with this task we are focusing on 3 other sub tasks to achieve this goal:

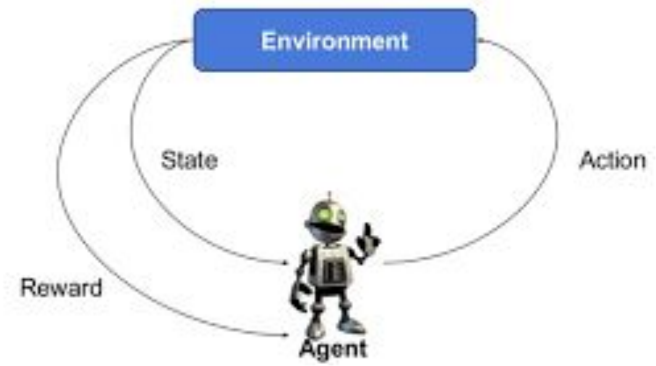
1. The pickup and drop off location should be the right location as set by the passenger.

2. Take the shortest time routes

3. Take safe routes

# Reinforcement Learning

Reinforcement Learning is a machine learning technique that is concerned with how agents should take action in an environment based on the rewards they receive after taking an action at a state.



# Temporal Difference Learning

Temporal difference (TD) learning refers to a class of model-free reinforcement learning methods which learn by bootstrapping from the current estimate of the value function.

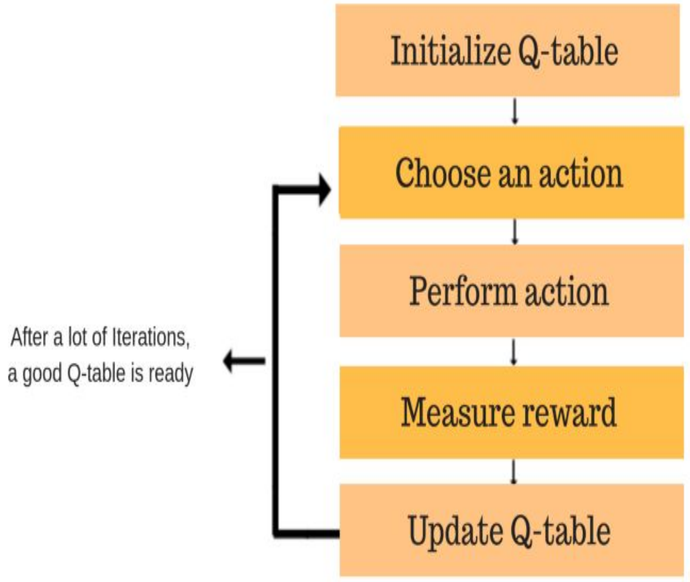
Unlike the monte-carlo methods, where the policy is updated at the end of the episode, Temporal Difference learning is updated at the end of every step.

Q-learning and Expected Sarsa are two TD methods that do not require model knowledge, only the observed rewards from any experiment returns.

# Q-learning

- Model-free: estimates optimal policy without the need for any transition or reward functions from the environment.
- Value-based: updates value functions based on Bellman equations rather than estimating  value function with greedy policy.
- Off-policy: learns from its actions and doesn’t depend on current policy.

$$Q(s_t, a_t) = (1 - \alpha) * Q(s_t, a_t) + \alpha * (r_t + \gamma * \max_a Q(s_{t+1}, a))$$



# SARSA and Expected SARSA

SARSA is called *on-policy* learning because new action a′ is chosen using the same epsilon-greedy policy as the action a, the one that generated s′.

$$Q(s_{t}, a_{t}) = Q(s_{t}, a_{t}) + \alpha*(r_{t} + \gamma*Q(s_{t+1}, a_{t+1}) - Q(s_{t}, a_{t}))$$

Expected SARSA is a variation of SARSA that takes in the expectation (mean) of Q values for every possible action in the current state.

$$Q(s_{t}, a_{t}) = Q(s_{t}, a_{t}) + \alpha*(r_{t} + \gamma*E[Q(s_{t+1}, a_{t+1})|s_{t+1}] - Q(s_{t}, a_{t}))$$

# OpenAI Gym - Taxi

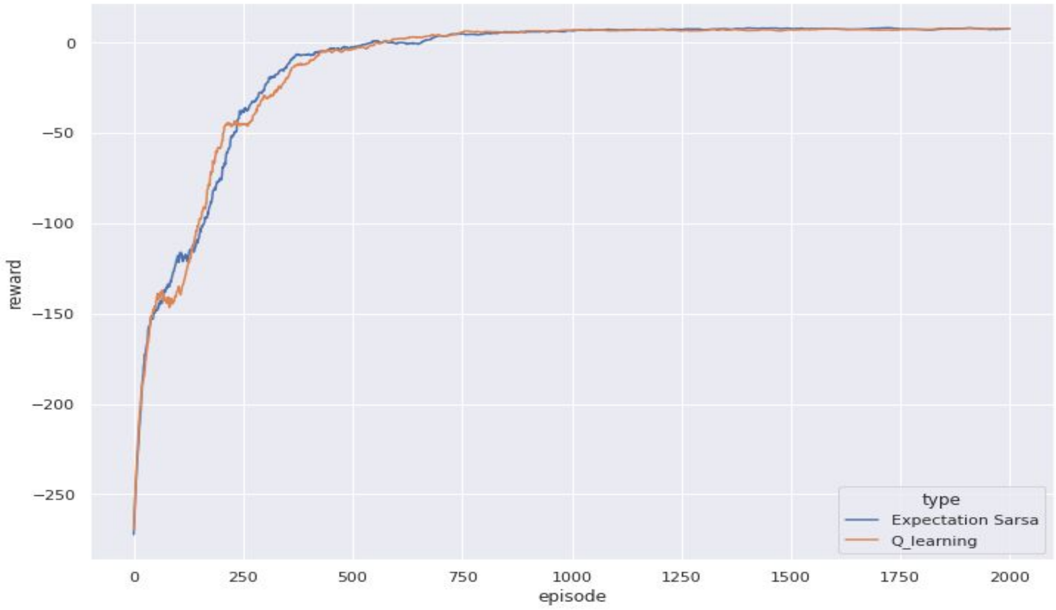
- OpenAI Gym - Reinforcement Learning Toolkit
- Taxi Environment:

  Taxi Agent which can pick-up and drop passengers

  in minimum number of moves.

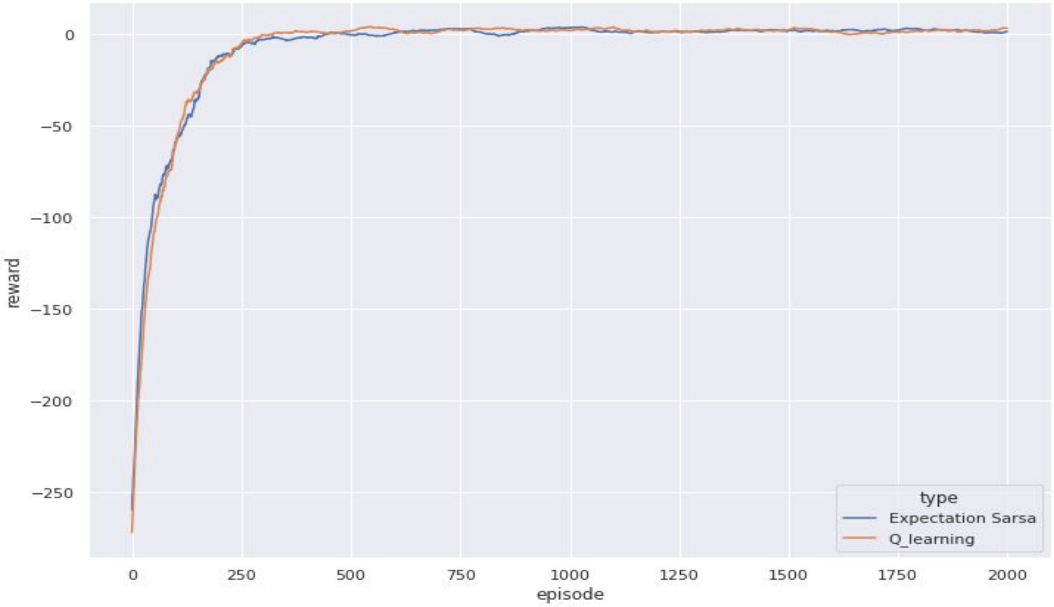
# Model Parameters and Results obtained

```
'alpha': 0.25
'epsilon_cut': 0
'epsilon_decay': 0.9
'start_epsilon': 0.99
'gamma': 0.7
'episodes': 2000
```



# Model Parameters and Results obtained

```
'alpha': 0.6
'epsilon_cut': 0.1
'epsilon_decay': 0.9
'start_epsilon': 0.99
'gamma': 0.9
'episodes': 2000
```



# Results

- Plot of Rewards vs Episodes demonstrating how both the algorithms learn with more training

- We can observe a faster convergence by Q learning algorithm that can be justified due to it being a greedy algorithm

- Higher Learning Rate also leads to faster convergence

# Evaluation metrics

We ran both the algorithms against 10 different environments and observed that Expected SARSA gave better results for 60% of the environments.

# Lesson Learned and Future Work

**Lessons:**

- Hands on experience with reinforcement algorithms
- Explored different and learnt new RL algorithm - Expected SARSA

**Future Work:**

- Figure out the most optimal parameters
- Try more algorithms
- Try with different environments and introduce additional obstacles making it closer to the real life scenarios.

# Thank You!