UNIVERSITÀ DEGLI STUDI DI TRIESTE

Analysis of RL Algorithms for a Simulated Hill Climb Racing Agent

July 28, 2025

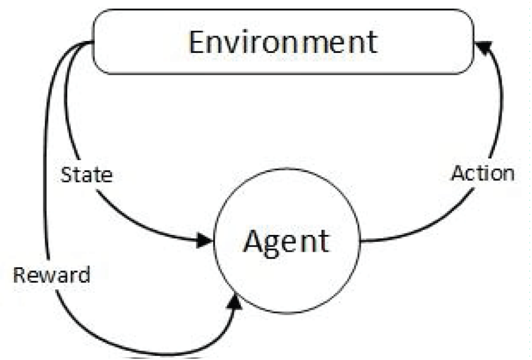
1. Problem Definition
2. Deep Q-Network
3. Expected SARSA
4. Proximal Policy Optimization
5. Results

Markov Decision Process

A MDP is a **stochastic model for sequential decision making** defined by a tuple:

$$(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$$

.



Reward Function ($\mathcal{R}$)

Event	**Value**
Forward Progress (per meter)	+5.0
Coin Collection	+20.0
Air Time (per second)	+5.0
Time Penalty (per step)	-0.1
Crash (Episode End)	-50.0

Policy (π) and Discount Factor (γ)

The agent's goal is to learn an optimal **policy** **(π^*)**.

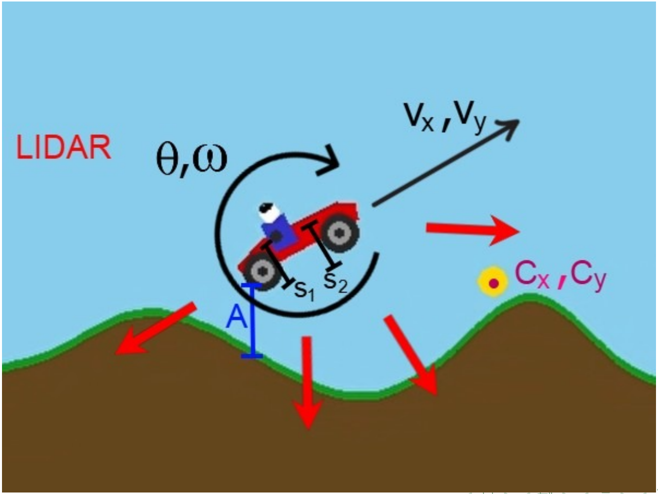
- In this project, the policy is approximated by a **neural network** due to the high-dimensional, continuous state space.

Future rewards are weighted by the **discount factor** **(γ)**.

- It balances the importance of immediate versus long-term rewards.
- We chose a high value of $\gamma = 0.99$ to create a "far-sighted" agent and encourages long-term returns.

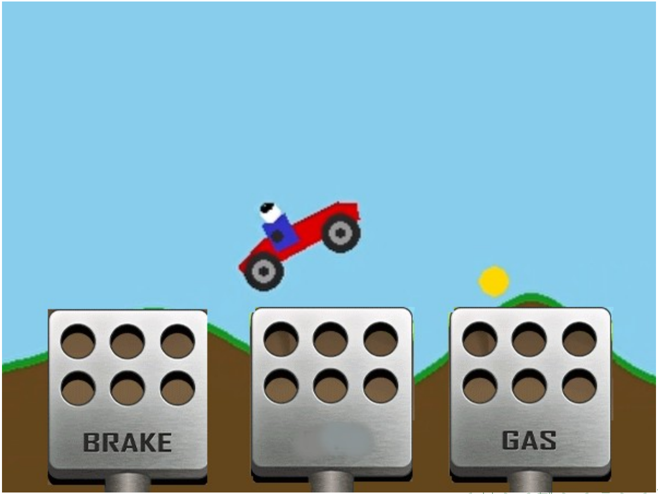
State Space ($\mathcal{S}$)

- The agent has perfect knowledge of the state, leading it to a **perfect observability** scenario.

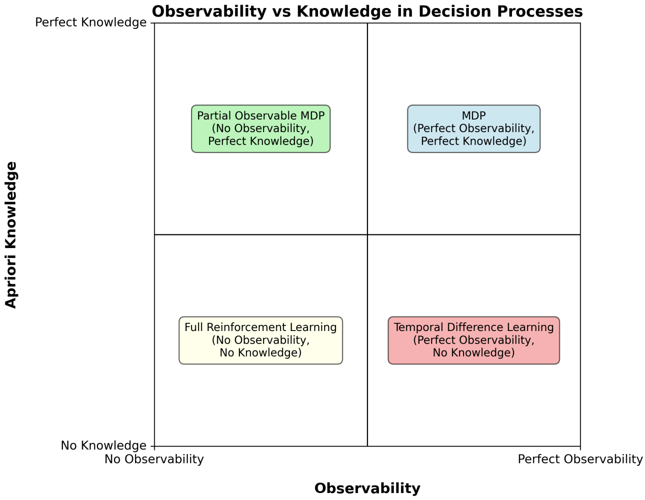


Action Space ($\mathcal{A}$)

- The agent does **not** have prior knowledge of this model, this puts it in a **model-free** context.



Problem Classification

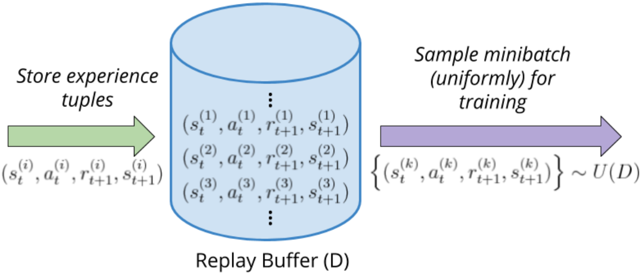


Characteristics of DQN

DQN combines the principles of **deep neural networks** with **Q-learning**.

- **Off-policy:** learning from actions taken by different policies.
- **Offline:** it collects a batch of experiences.

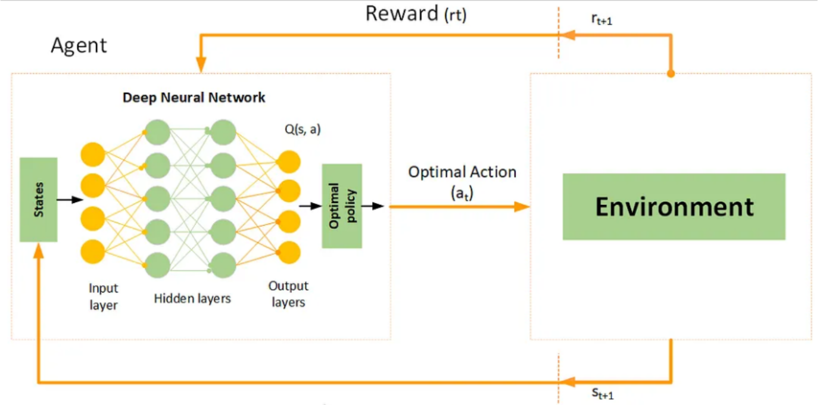
DQN Training



$$L(\theta) = \left(\underbrace{r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta)}_{\text{Target}} - \underbrace{Q(s, a; \theta)}_{\text{Prediction}} \right)^2$$

DQN Algorithm

•



Characteristics of Expected SARSA

Expected SARSA combines the principles of statistical expectation with the on-policy learning algorithm SARSA.

$$(s, a, r, s', a')$$

- **On-policy:** the update is based on the expected value according to the policy being followed.
- **Online:** An update after each single step.

From SARSA to Expected SARSA

- **SARSA**

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma Q(s',a') - Q(s,a) \right]$$

- **Expected SARSA**

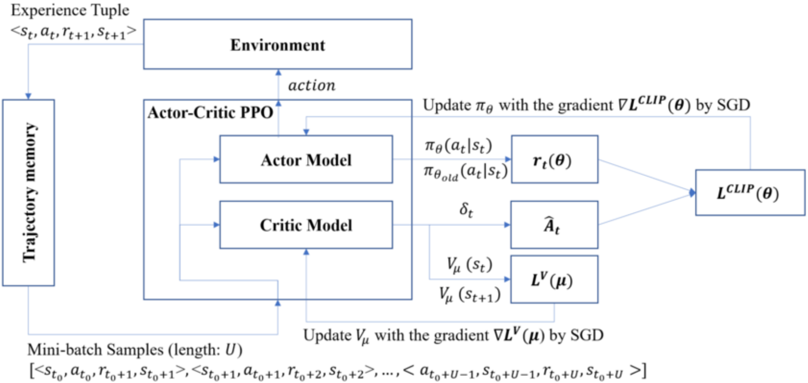
$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \mathbb{E}[Q(s',a')] - Q(s,a) \right]$$

$$\underbrace{\mathbb{E}[Q(s',a')]}_{\text{Expected value over all possible next actions}} = \sum_{a' \in \mathcal{A}} \pi(a'|s') Q(s',a')$$

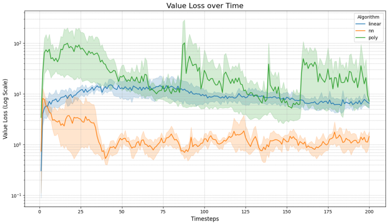
Loss

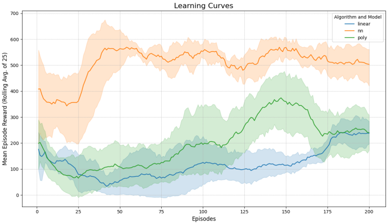
$$L(\theta) = \left(\underbrace{\left(r + \gamma \sum_{a' \in \mathcal{A}} \pi(a'|s') Q(s', a'; \theta^-) \right)}_{\text{Target}} - \underbrace{Q(s, a; \theta)}_{\text{Prediction}} \right)^2$$

PPO Algorithm

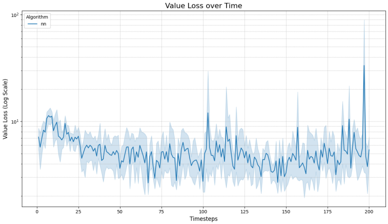


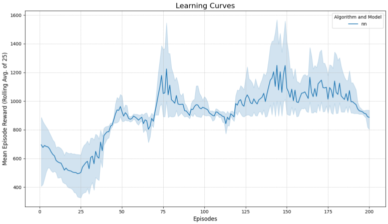
DQN - Loss vs Reward



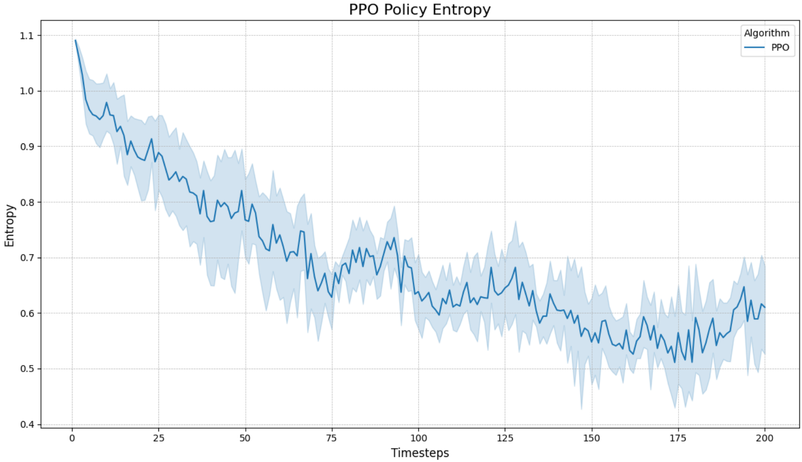


PPO - Loss vs Reward

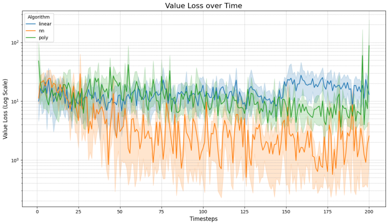


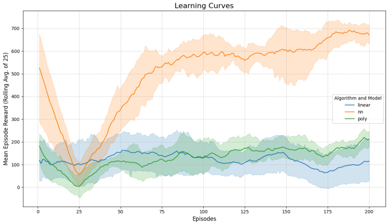


PPO - Entropy

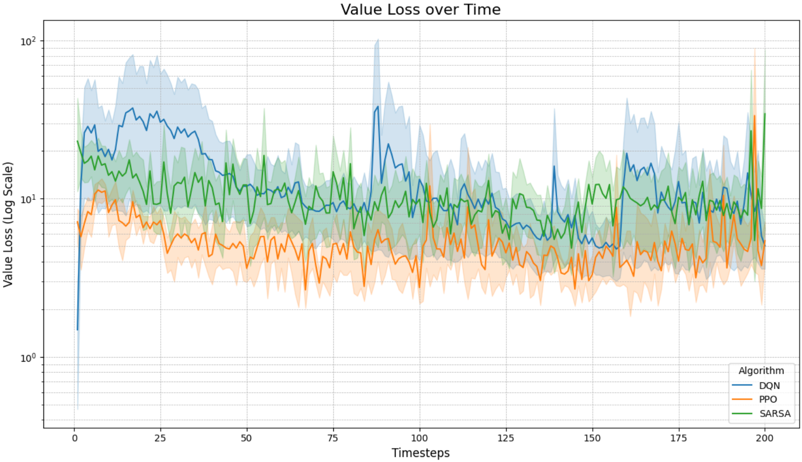

PPO Policy Entropy
Algorithm
PPO
Entropy
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
25
50
75
100
125
150
175
200
Timesteps

SARSA - Loss vs Reward

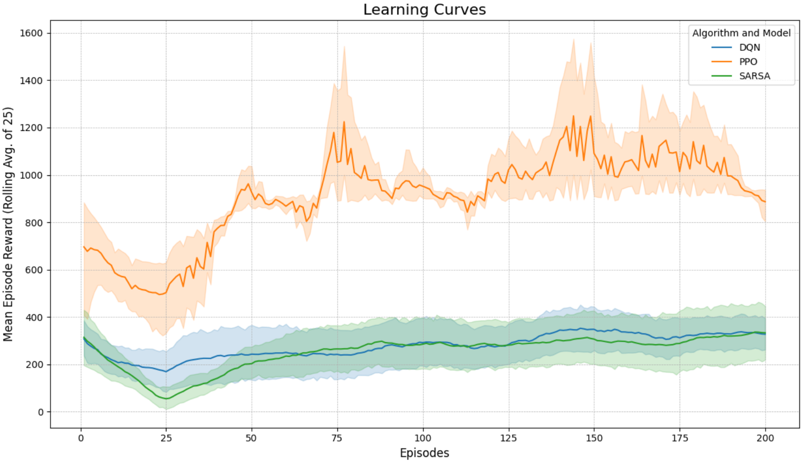




All vs All - Loss



All vs All - Reward



Thank you!