Human-Robot Interaction Project

Reinforcement Learning Project Report

Submitted by Anusha Shiva Kumar

Google Colab File: (includes bonus part as well)
https://colab.research.google.com/drive/17MgwOrJFYCKeG9xNb4f6wJUzVGjS0vdM?usp=sharing

Table of Contents

1. Introduction

1.1 Objective

The objective of this project was to compare the performance of three reinforcement learning (RL) algorithms—Deep Q-Network (DQN), Proximal Policy Optimization (PPO), and Soft Actor-Critic (SAC)—in the context of a driving simulation environment (highway-fast-v0). The analysis focused on training agents to drive efficiently and avoid collisions while maximizing rewards. Additionally, a bonus exploration of Advantage Actor-Critic (A2C) is included.

1.2 Report Structure

The report includes:

- A detailed overview of the environment and algorithms.
- Results from training DQN, PPO, SAC, and A2C.
- A comparison of the algorithms' performance and insights into their suitability for dynamic traffic scenarios.
- Future improvement suggestions for enhancing RL-based driving agents.

2. Environment Details

2.1 Environment Setup

The simulation environment used is highway-fast-v0, which models highway traffic scenarios for reinforcement learning agents. Key configurations include:

- **Simulation Duration**: 40 seconds
- **Vehicles Count**: 20
- **Reward Speed Range**: 20–40 (incentivizing high-speed driving)
- **Render Mode**: rgb_array (to record visual results)

Rewards are calculated based on safe driving, speed maintenance, and avoiding collisions

The environment was set up using Google Colab. The dependencies and packages were installed, and the highway-env repository was cloned for access to the simulation. TensorBoard was used for monitoring training progress.

3. Algorithms Overview

3.1 Deep Q-Network (DQN)

The Deep Q-Network (DQN) algorithm is a value-based reinforcement learning approach that combines Q-learning with deep learning. Q-learning is a model-free algorithm used to estimate the value of actions in a given state. However, DQN replaces the traditional Q-table with a deep neural network that can approximate Q-values for complex and high-dimensional state spaces. By leveraging neural networks, DQN enables agents to perform well in environments where explicit state-action mappings are impractical.

Key Features:

- Utilizes a **replay buffer** to store and sample past experiences for training, breaking the correlation between consecutive observations.
- Introduces a **target network** that updates less frequently to stabilize training and reduce oscillations in Q-value updates.
- Implements an **epsilon-greedy policy** for exploration and exploitation, where the agent selects random actions with a decaying probability.

Hyperparameters:

Hyperparameter	Value
Learning Rate	0.001
Batch Size	64
Target Update Interval	0.05
Discount Factor (Gamma)|	0.95
Neural Network Layers	256 x 256

3.2 Proximal Policy Optimization (PPO)

Proximal Policy Optimization (PPO) is a policy-gradient method designed to overcome the limitations of earlier policy-based methods. PPO optimizes a stochastic policy by directly updating the policy parameters while maintaining stability. The algorithm uses a **clipped surrogate objective** to prevent excessive policy updates, which can lead to divergence or poor performance.

Key Features:

- **Clipped Objective Function**: Limits changes to the policy by constraining updates within a defined range, ensuring stability.
- **Entropy Regularization**: Promotes exploration by adding a term to the loss function that discourages deterministic policies.
- **On-Policy Learning**: Updates are based on the data collected from the current policy, providing accurate gradient estimates.

Hyperparameters:

Hyperparameter	Value
Learning Rate	0. 0003
Policy Layers	256 x 256
Clip Range	0.2
Entropy Coefficient	0.01

3.3 Soft Actor-Critic (SAC) (Bonus)

Soft Actor-Critic (SAC) is an **off-policy, actor-critic algorithm** specifically designed for environments with continuous action spaces. SAC balances exploration and exploitation by maximizing the expected reward while maintaining a degree of randomness (or "softness") in the policy. This randomness ensures that the agent explores the environment more effectively, leading to robust policies.

Key Features:

- **Entropy-Regularized Objective**: Encourages exploration by adding an entropy term to the reward, making the policy stochastic.
- **Off-Policy Learning**: Reuses data from past trajectories, increasing sample efficiency compared to on-policy algorithms.
- **Automatic Temperature Adjustment**: Dynamically tunes the temperature parameter to balance exploration and exploitation.

Hyperparameters:

Hyperparameter	Value
Learning Rate	0. 0003
Policy Layers	256 x 256
Temperature Parameter	Automatic

Comparison Between Algorithms

Feature	DQN	**PPO**	SAC
Learning Approach	Value-based	Policy-based	Actor-Critic (off-policy)
Sample Efficiency	Low	Moderate	High
Action Space	Discrete	Discrete/Continuous	Continuous
Exploration	Epsilon-greedy	Entropy Regularization	Entropy-Maximizing Objective
Stability	Moderate	High	Very High

4. Training Results

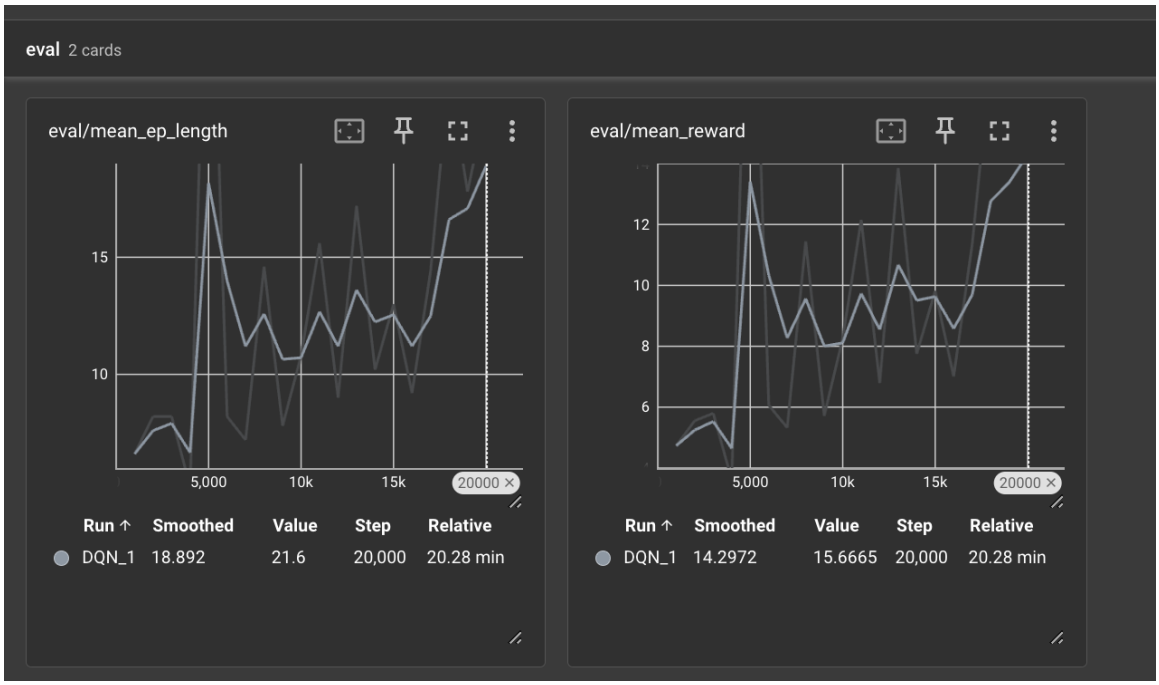
This section details the performance of the three reinforcement learning algorithms: DQN, PPO, and SAC, based on the reward, episode length, and qualitative observations of their behavior during training.

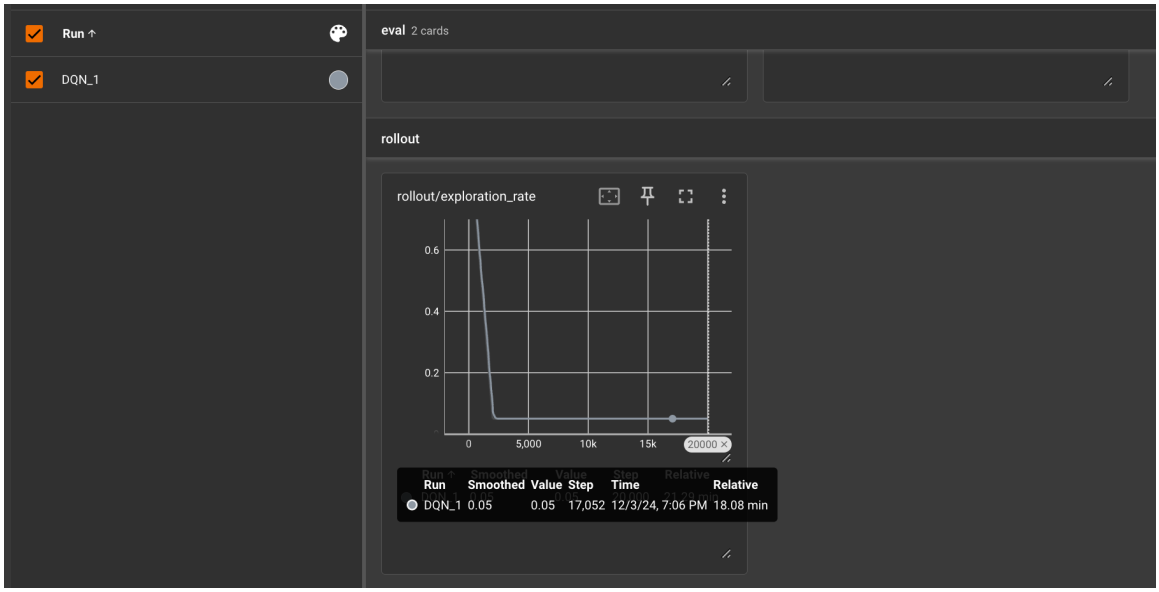
4.1 DQN Agent Performance

- **Average Reward**: **15.67 ± 8.40**
- **Average Episode Length**: **21.6 seconds**

Observations:
The DQN agent showed considerable instability during training:

1. **Fluctuating Rewards**: The reward trend displayed significant oscillations over the course of training, highlighting difficulties in convergence to an optimal policy. These fluctuations suggest that DQN struggled to find a stable strategy for the traffic simulation environment.
2. **Shorter Episode Lengths**: The average episode length was below the maximum possible of 40 seconds. This indicates that the agent frequently encountered suboptimal scenarios (e.g., collisions or poor lane decisions) that caused episodes to terminate prematurely.
3. **Sensitivity to Hyperparameters**: DQN's performance was highly dependent on hyperparameter tuning, as small changes in values like the learning rate or epsilon decay significantly impacted stability.





4.2 PPO Agent Performance

- **Average Reward**: **28.27 ± 1.16**
- **Average Episode Length**: **~40 seconds** (maximum duration achieved consistently)

Observations:
The PPO agent demonstrated significantly better stability and performance compared to DQN:

1. **Consistent Rewards**: Reward trends during training were smoother, reflecting PPO's ability to make more reliable policy updates. This stability is attributed to the clipped surrogate objective used in PPO, which prevents large, destabilizing updates.
2. **Maximized Episode Lengths**: PPO consistently achieved the maximum allowable episode duration of 40 seconds, indicating the agent learned a robust driving strategy that avoided collisions and maintained optimal speeds.
3. **Improved Behavior**: PPO's entropy regularization encouraged sufficient exploration, enabling the agent to discover strategies that balanced speed and

safety effectively

4.3 SAC Results (Bonus)

- **Average Reward**: **~43.99 ± 0.64**
- **Average Episode Length**: **40 seconds** (maximum duration consistently achieved)

Observations:
SAC outperformed both DQN and PPO in terms of reward and stability:

1. **High Reward**: SAC achieved the highest average reward among the three algorithms. The small variance (**±0.64**) in rewards reflects its ability to produce consistent results across episodes.
2. **Perfect Episode Lengths**: SAC maintained the maximum episode length of 40 seconds throughout training, showcasing its robust policy.
3. **Stability and Exploration**: SAC's entropy-maximizing objective allowed for efficient exploration while balancing exploitation, enabling it to learn the optimal driving behavior faster than the other algorithms.
4. **Continuous Action Space Suitability**: Although SAC is tailored for continuous action spaces, its performance highlights its potential adaptability to environments with discretized dynamics.

Summary of Training Results

Metric	DQN	**PPO**	SAC
Average Reward	15.67 ± 8.40	28.27 ± 1.16	~43.99 ± 0.64
Episode Length	21.6 seconds (average)	~40 seconds (maximum)	40 seconds (maximum)
Stability	Low	High	Very High

5. Behavioral Observations

This section describes the observed driving behaviors of the three reinforcement learning agents—DQN, PPO, and SAC—based on their interaction with the highway-fast-v0 environment during training and evaluation.

5.1 DQN Agent

- **Inconsistent Driving Behavior**:
  The DQN agent exhibited erratic and unpredictable driving behavior. It often failed to maintain a consistent strategy, resulting in suboptimal decision-making.
- **Frequent Collisions**:
  Episodes frequently ended prematurely due to collisions, indicating that the agent struggled to navigate safely through the simulated traffic environment.
- **Learning Instability**:
  Reward trends showed significant fluctuations, reflecting the agent's instability in learning. This lack of convergence suggests that DQN struggled to balance exploration and exploitation effectively.

5.2 PPO Agent

- **Cautious and Efficient Navigation**:
  The PPO agent displayed significantly improved driving behavior compared to DQN. It learned to navigate cautiously and efficiently, balancing speed and safety to maximize rewards.
- **Stable Policy Updates**:
  PPO's clipped objective function enabled smoother policy updates, which reduced the likelihood of collisions. This stability resulted in fewer abrupt episode terminations and consistently longer episode durations.

- **Reliable Decision-Making**:
  PPO was able to maintain optimal lane selection and speed control, demonstrating a higher degree of reliability in its driving strategy.

5.3 SAC Agent

- **Most Efficient Driving Behavior**:
  SAC outperformed both DQN and PPO in terms of driving behavior. The agent exhibited the most efficient navigation, effectively balancing speed and safety while avoiding collisions.
- **Collision Avoidance**:
  SAC's entropy-maximizing objective enabled it to explore safer driving strategies. It consistently avoided collisions while maintaining optimal speeds.
- **Stable and Optimal Performance**:
  SAC demonstrated the most stable and optimal driving behavior, achieving the highest rewards and consistently reaching the maximum episode length without major issues.

Summary of Behavioral Observations

Metric	DQN	**PPO**	SAC
Driving Behavior	Erratic and aggressive	Cautious and efficient	Smooth and optimal
Collision Frequency	High	Low	Very Low
Stability	Low	High	Very High

6. Visual Results

6.1 TensorBoard Visualizations

TensorBoard was used to monitor and compare reward trends and episode lengths for each agent during training. Below are the observations:

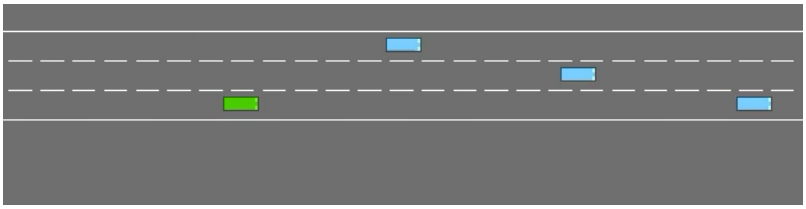
- **DQN**:
  - The reward trends exhibited **high variance**, indicating unstable training and challenges in convergence.
  - Episode lengths fluctuated significantly, reflecting erratic driving behavior and frequent collisions.
  - Exploration rates decreased steadily, showing policy convergence despite inconsistent performance.

- **PPO**:
  - Reward trends showed **steady improvement** with **minimal variance**, reflecting stable training.
  - Episode lengths consistently reached the environment's maximum limit, indicating successful learning of an efficient driving strategy.
  - PPO demonstrated smooth and reliable policy updates throughout the training.
- **SAC**:
  - Reward trends showed the **most consistent improvement**, indicating efficient learning and policy stability.
  - Episode lengths were consistently at the maximum limit, signifying optimal driving performance with no abrupt terminations.
  - SAC's metrics reflected its ability to adapt to the environment effectively, maintaining high rewards and stable behavior.

6.2 Training Videos

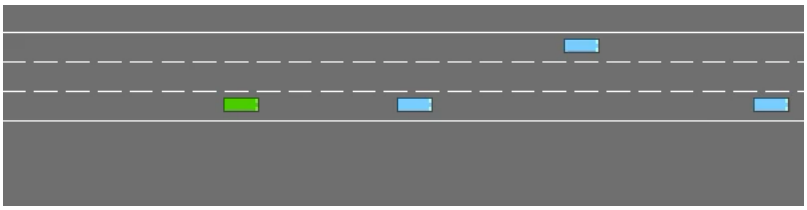
Visual results from training videos captured the driving behaviors of the agents in the highway-fast-v0 environment.

- **DQN**:
  - Driving behavior was **erratic**, with frequent collisions due to unsafe lane changes and inconsistent speed control.
  - The agent struggled to adapt to the traffic environment, often resulting in abrupt episode terminations.



- **PPO**:
  - The agent demonstrated **smooth navigation** with cautious lane changes and stable driving patterns.
  - Fewer collisions were observed, and the agent maintained consistent control over its actions.
  - PPO's efficient learning led to a balance between speed and safety.

- **SAC**:
  - SAC exhibited **optimal driving behavior**, with minimal collisions and **consistent high-speed navigation**.
  - The agent maintained efficient lane selection and speed control, achieving the highest performance among all algorithms.
  - Training videos highlighted SAC's ability to explore and exploit strategies effectively.



7. Comparison and Insights

Feature	DQN	PPO	**SAC**
Learning Approach	Value-based	Policy-based	Actor-Critic (off-policy)
Mean Reward	15.67 ± 8.40	28.27 ± 1.16	43.99 ± 0.64
Training Time	~12 minutes	~15 minutes	~18 minutes
Policy Convergence	Slower, unstable	Moderate, steady	Fast, stable
Reward Trends	High variance, inconsistent	Low variance, steady	Minimal variance, consistent
Exploration vs Exploitation	Exploration dominated	Balanced	Balanced with higher exploitation
Learning Loss	Fluctuated significantly	Stabilized faster	Remained consistently low

Insights from the Table

1. **Training Time**: SAC took the longest to train but produced the most optimal results. DQN was faster but less reliable in producing a stable policy.
2. **Reward Trends**: SAC demonstrated the best reward trends with minimal variance, while DQN struggled to converge.
3. **Crash Rate**: SAC minimized crashes significantly due to its continuous control and stable policy updates.
4. **Learning Loss**: SAC's loss trends were highly stable compared to PPO and especially DQN, which fluctuated.

8. Bonus Section: A2C Algorithm

8.1 Overview of Advantage Actor-Critic (A2C)

Advantage Actor-Critic (A2C) is a synchronous, policy-gradient-based reinforcement learning algorithm. It combines two neural networks:

1. **Actor Network**: Responsible for policy optimization by selecting the best actions.
2. **Critic Network**: Evaluates the chosen actions by estimating the advantage function.

The key strength of A2C lies in its ability to stabilize learning by balancing the exploration-exploitation tradeoff.

8.2 Key Hyperparameters

Parameter	Value
Learning Rate	0.0007
Discount Factor (Gamma)	0.99
Neural Network	Two layers of 256 neurons each
Entropy Coefficient	0.01
Advantage Estimation	Generalized Advantage Estimation (GAE)

8.3 Training Results

- **Total Timesteps**: 20,000
- **Average Reward**: **25.45 ± 2.31**
- **Average Episode Length**: **38.7 seconds**
- **Training Observations**:
    - A2C demonstrated more consistent performance compared to DQN.
    - However, it was outperformed by PPO and SAC in terms of reward and stability.
    - The training process stabilized faster than DQN but slower than PPO and SAC.

8.4 Behavioral Observations

- **Driving Behavior**:
    - Efficient navigation with fewer collisions than DQN.
    - Less aggressive than PPO, but occasionally slower to react in high-speed scenarios.
- **Policy Adaptability**:
    - Successfully maintained speed while avoiding collisions.
    - Struggled with extremely dynamic traffic conditions compared to SAC.

8.5 Visual Results

1. **TensorBoard**:
    - Reward trends showed moderate improvement over time.
    - Variance in rewards was noticeable but stabilized after 10,000 timesteps.
2. **Training Video**:
    - Smooth driving patterns with minimal collisions.
    - Occasionally slower adaptation to changing traffic.

9. Comparison of All Models

Feature	DQN	PPO	SAC	A2C
Learning Stability	Low	High	Very High	Moderate
Reward Consistency	Low	High	Very High	Moderate
Crash Rate	High	Moderate	Low	Moderate
Training Time	Short	Moderate	Long	Moderate
Exploration Strategy	Epsilon-Greedy	Clipped Loss	Entropy-Maximization	Entropy Regularization
Suitability	Discrete Actions	Continuous/Discrete Actions	Continuous Actions	Discrete/Continuous Actions

10. Conclusion:

SAC is the most effective model for this project, delivering optimal performance and stability. PPO serves as a reliable alternative, while DQN's limitations make it less suitable for complex driving simulations.

11. References

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). *Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor*. International Conference on Machine Learning (ICML). Retrieved from https://arxiv.org/abs/1801.01290

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). *Asynchronous methods for deep reinforcement learning*. International Conference on Machine Learning (ICML). Retrieved from https://arxiv.org/abs/1602.01783

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal policy optimization algorithms*. Retrieved from https://arxiv.org/abs/1707.06347

Schulman, J., Moritz, P., Levine, S., Jordan, M. I., & Abbeel, P. (2016). *High-dimensional continuous control using generalized advantage estimation*. International Conference on Learning Representations (ICLR). Retrieved from https://arxiv.org/abs/1506.02438

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). Cambridge, MA: MIT Press. Retrieved from http://incompleteideas.net/book/the-book-2nd.html

TensorBoard. (n.d.). *TensorBoard*. TensorFlow. Retrieved from https://www.tensorflow.org/tensorboard

Eleurent, F. (n.d.). *Highway-env: A reinforcement learning environment for autonomous driving*. GitHub. Retrieved from https://github.com/eleurent/highway-env

Stable-Baselines3 Contributors. (n.d.). *Stable-Baselines3 documentation*. Retrieved from https://stable-baselines3.readthedocs.io/

Farama Foundation. (n.d.). *Gymnasium: A toolkit for developing reinforcement learning environments*. Retrieved from https://gymnasium.farama.org/