Comparing Deep Reinforcement Learning Algorithms on CarRacing-v3:

A Study of Discrete vs. Continuous Action Spaces

Author:
BOUDRIKA ILIAS

Table of contents

Motivation

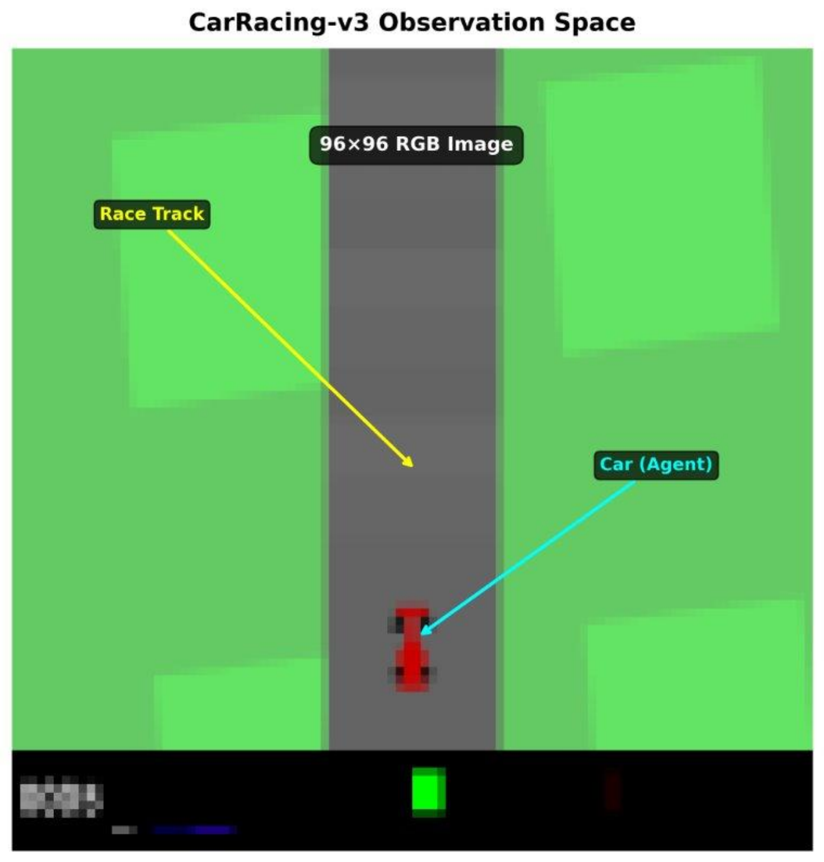
- Fundamental choice in RL: discrete vs continuous actions
- CarRacing-v3: visual control task supporting both representations
- Research Questions:
  - How do discrete and continuous actions compare in performance?
  - What are the stability characteristics of each approach?
  - What practical insights for practitioners?

Environment

- **Observation:** 96×96 RGB images
- **Discrete Actions** (DQN):

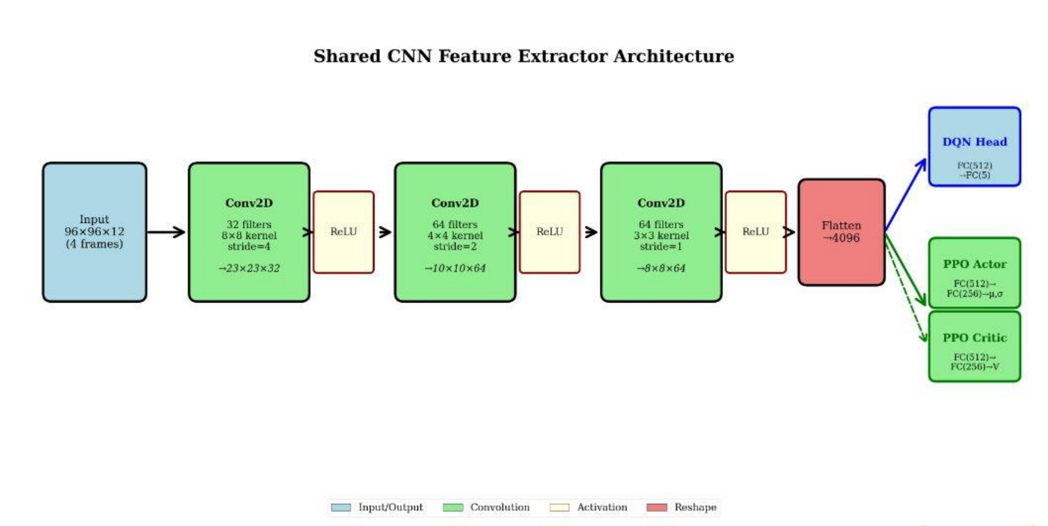
  - 5 actions: no-op, left, right, gas, brake
- **Continuous Actions** (PPO):

  - 3D vector [steering, gas, brake]
- **Reward:** -0.1 per frame + 1000/N pertile
- **Preprocessing:** frame stacking (4) +skipping (2)



Shared CNN Architecture

- Same feature extractor for faircomparison
- Nature DQN architecture:
  - **Conv2D**(32, 8×8, stride=4) → 23×23×32
  - **Conv2D**(64, 4×4, stride=2) → 10×10×64
  - **Conv2D**(64, 3×3, stride=1) → 8×8×64
  - **Flatten** → 4096 features
- Algorithm-specific heads:
  - **DQN**: FC(512) → FC(5)
  - **PPO**: Actor & Critic networks



DQN vs PPO

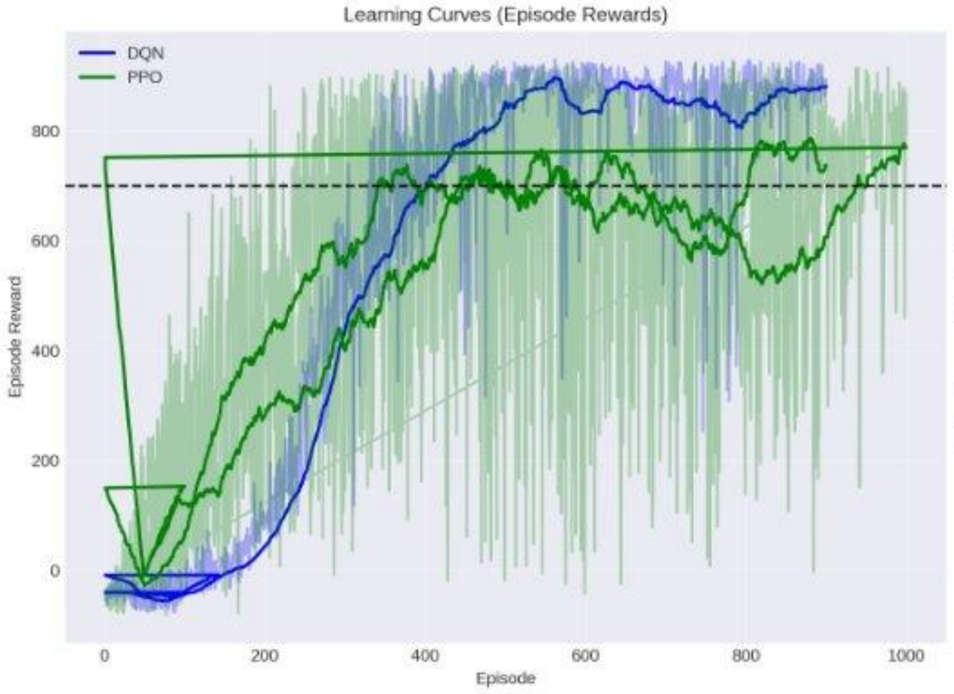
DQN (Deep Q-Network)

- Value-based method
- Off-policy learning
- Experience replay (100K)
- Target network (update every 1K steps)
- ε-greedy exploration (1.0→0.01)
- Huber loss
- Batch size: 32
- Learning rate: 1e-4

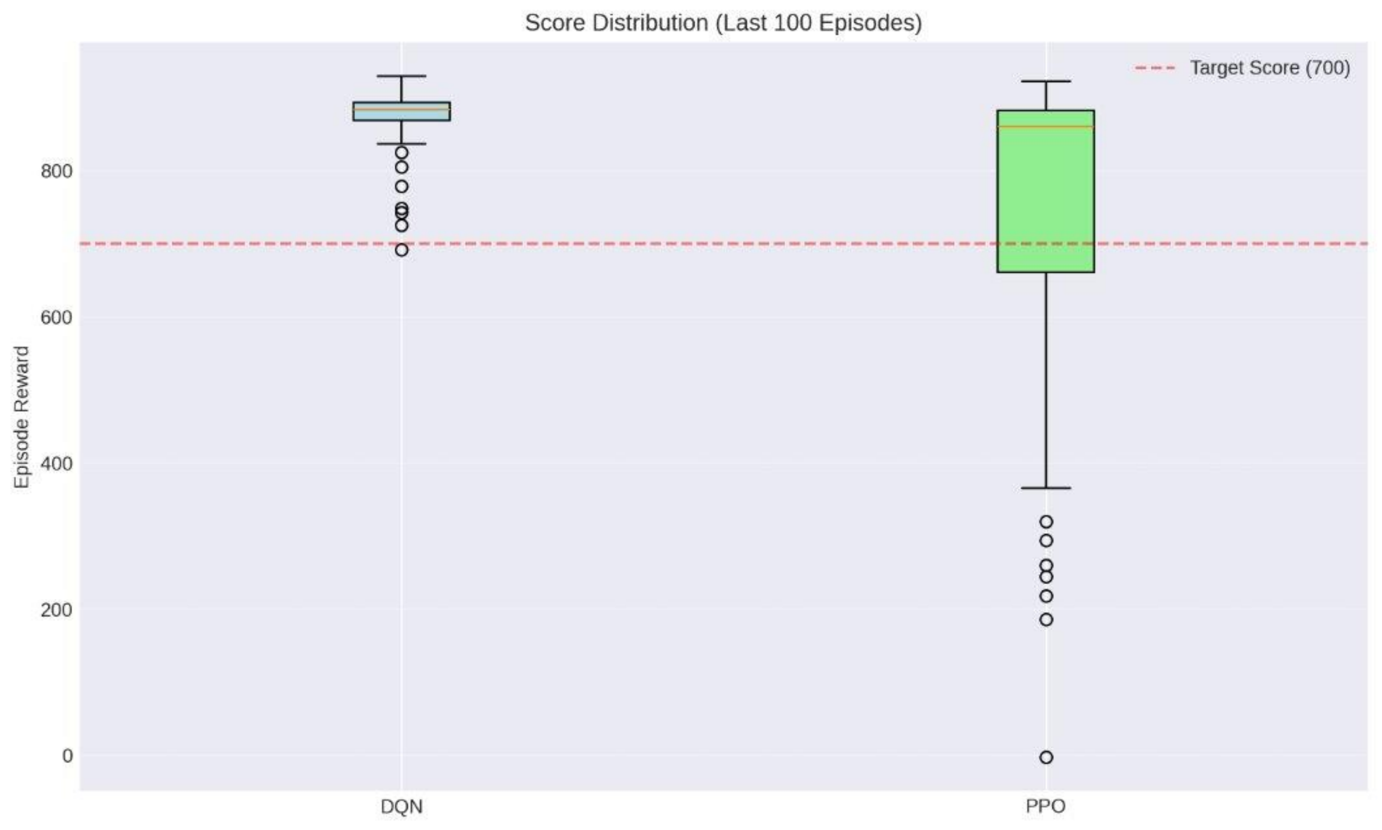
PPO (Proximal Policy Optimization)

- Policy gradient method
- On-policy learning
- Rollout buffer (2048 steps)
- Clipped surrogate objective
- GAE for advantages (λ=0.95)
- Multiple update epochs (10)
- Batch size: 64
- Learning rate: 3e-4

Learning Dynamics



Stability Analysis



Conclusion

- DQN achieved superior final performance (877.86 vs 752.43)
- DQN demonstrated better stability (std 40.24 vs 199.79)
- PPO showed smoother early learning but higher variance later
- Action space representation matters significantly
- Discrete actions can be effective when properly designed
- Choice should be guided by task characteristics and constraints

Thanks!

Do you have any questions?

Github Link: https://github.com/brk-ilias/reinforcement-learning