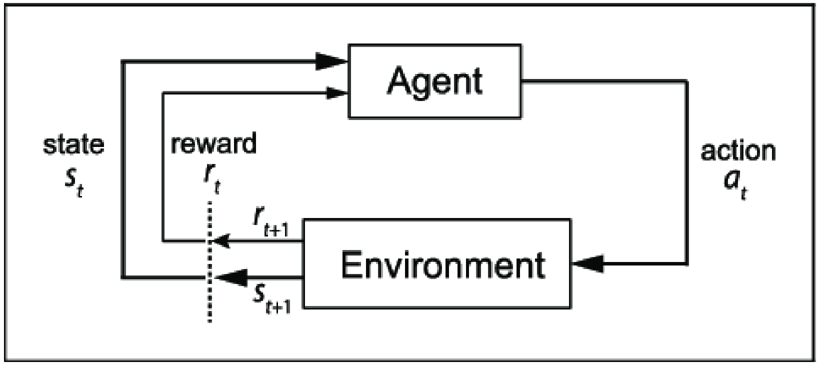
Comparison of Deep Reinforcement Learning Models for Automated Trading on Heterogeneous HPC System

Nabil Shadman

MSc High Performance Computing with Data Science

Introduction

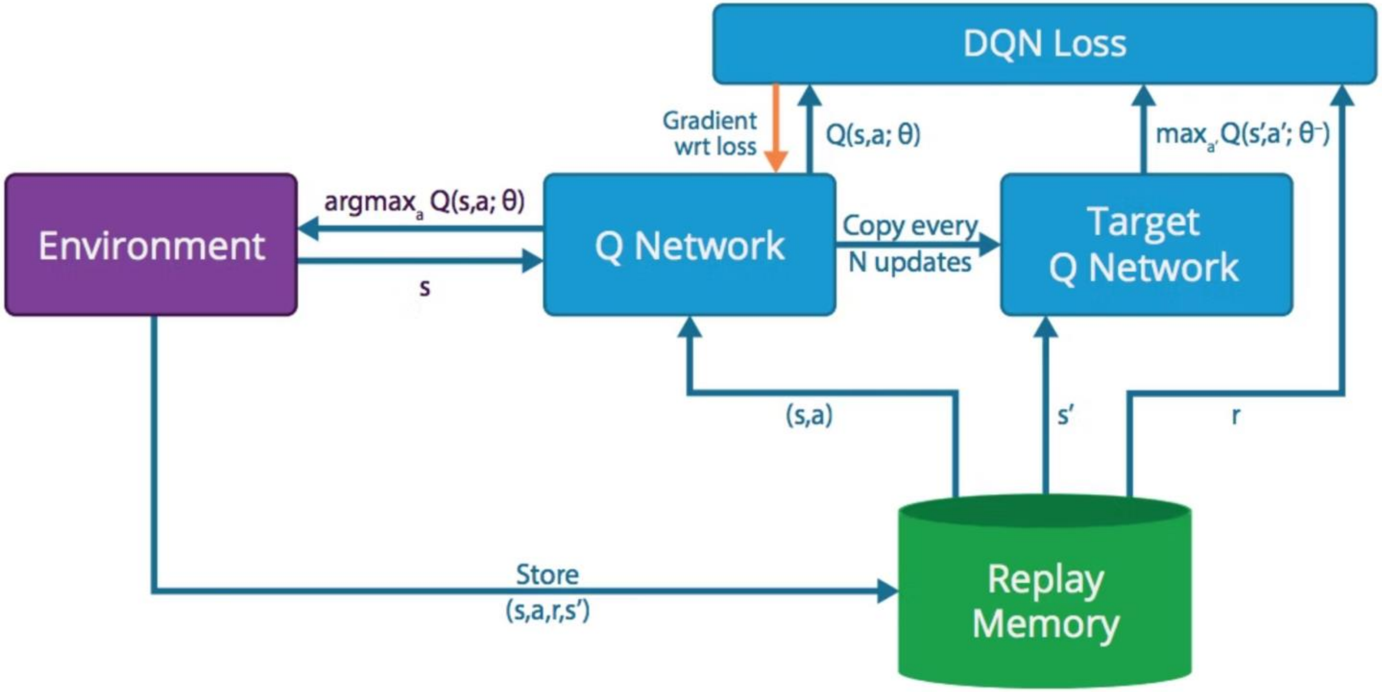
- **Background:** DRL in automated trading, need for HPC
- **Research Question:** Optimising DRL algorithms (DQN and PPO) for automated trading on HPC systems
- **Objectives:** Implement, optimise, compare performance, analyse scalability and generalisation



Picture taken from [1]

DQN

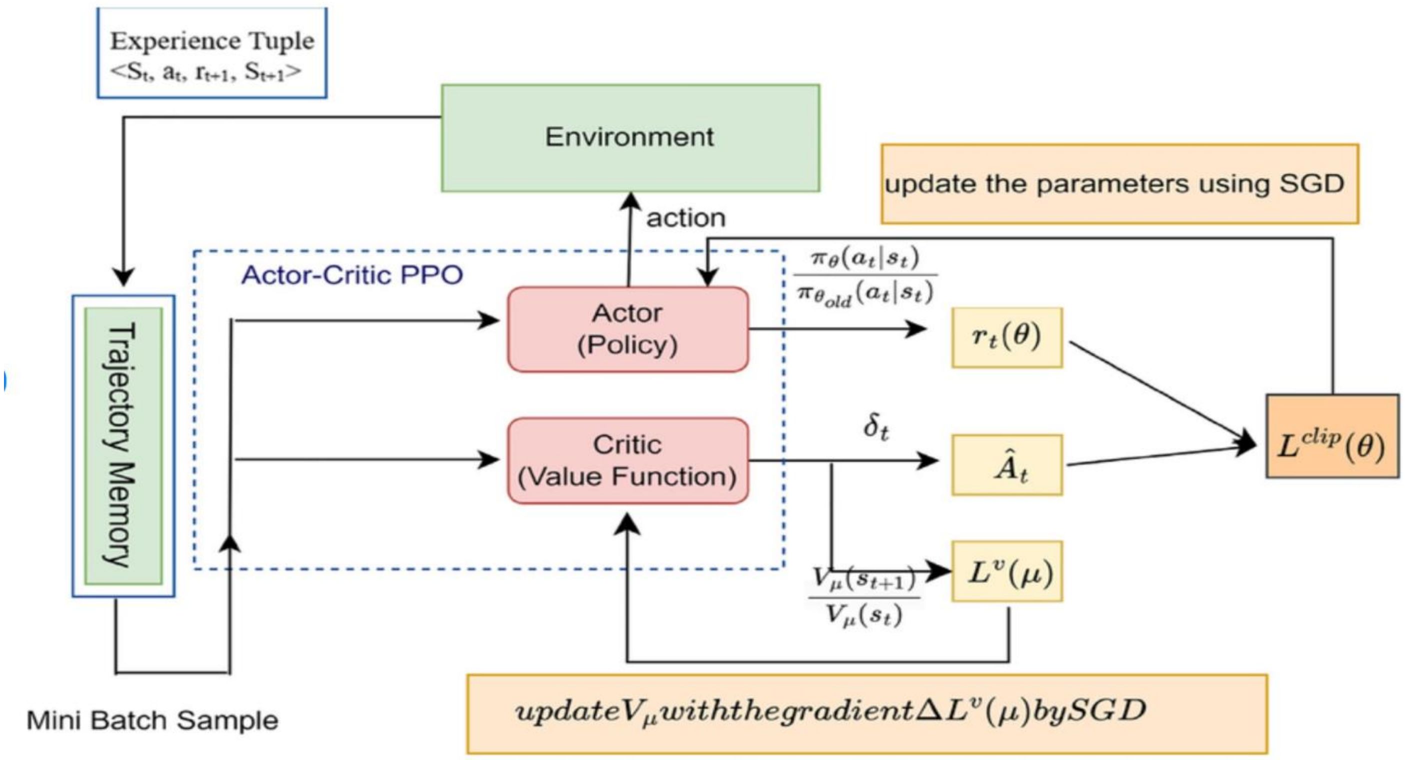
- Deep Q-Network



Picture taken from [2]

PPO

- Proximal Policy Optimisation



Picture taken from [3]

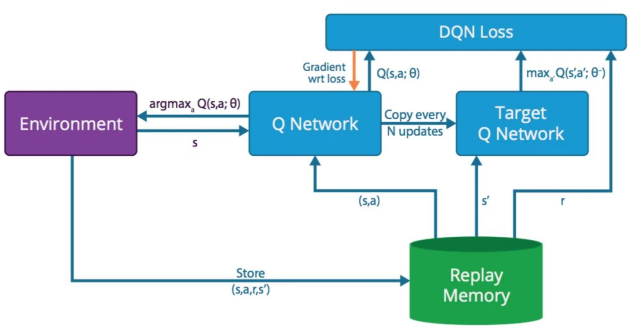
Methodology

- **DRL Algorithms:** DQN and PPO
- **Implementation:** PyTorch framework
- **HPC System:** Cirrus (CPU + GPU nodes)
- **Data:** Daily equity + ETF data – US markets (2018-2023)
- **Key Evaluation Metrics:** Portfolio value, execution time
- **Others:** Memory, energy, power, GPU utilisation

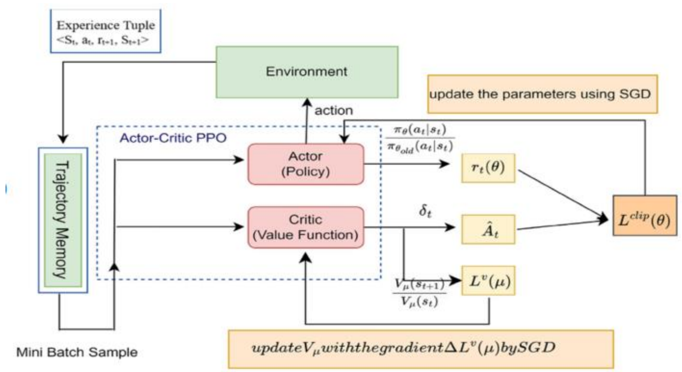
Cirrus *- Picture taken from [4]*

Implementation Details

- **Environment:** Multi-asset trading with transaction costs
- **DQN Architecture:** Q-Network with 2 hidden layers
- **PPO Architecture:** Separate actor and critic networks
- **Optimisation Techniques:** GPU acceleration, profiling, hyperparameter tuning



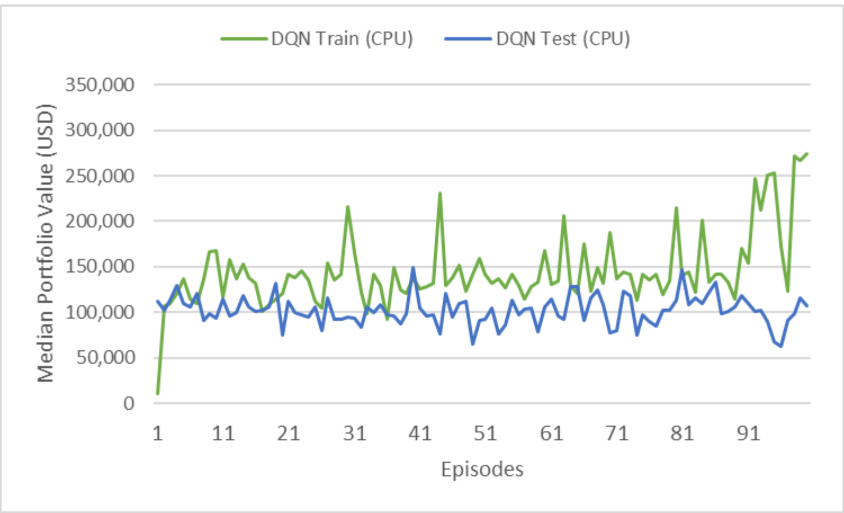
DQN *- Picture taken from [2]*



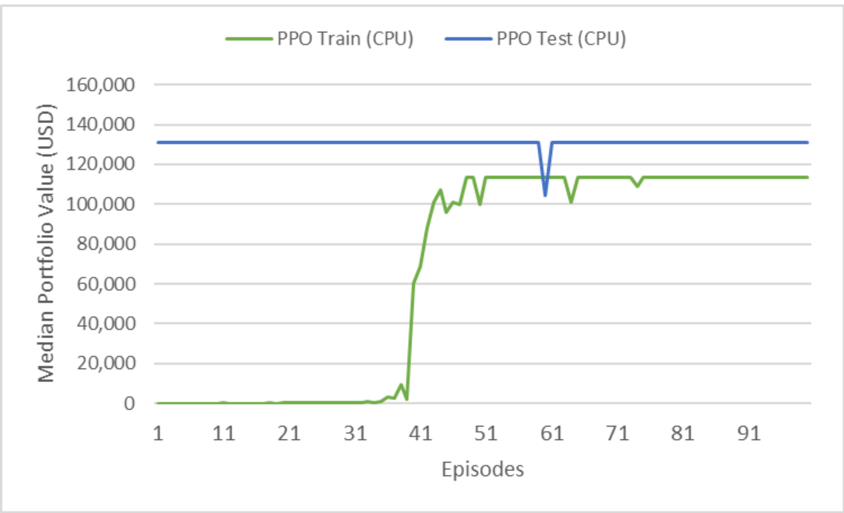
PPO *- Picture taken from [3]*

Baseline Performance

- **Comparison** of DQN and PPO on CPU and GPU
- **Metrics:** Portfolio value, execution time, resource utilisation



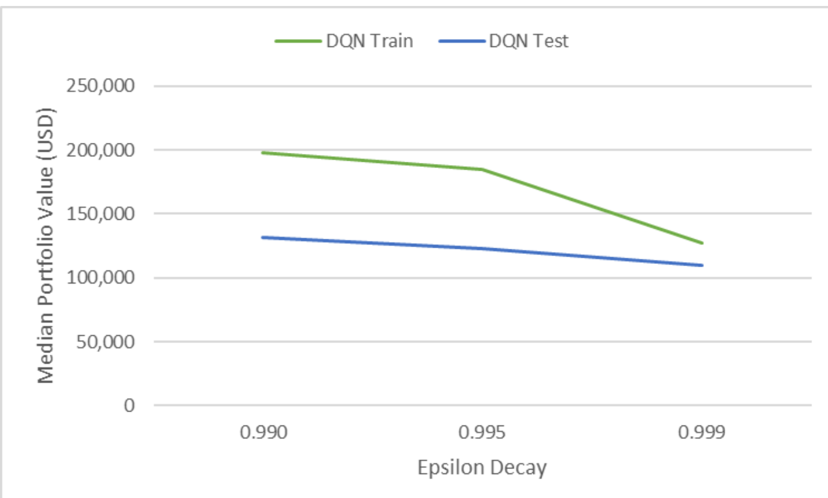
DQN (on CPU)



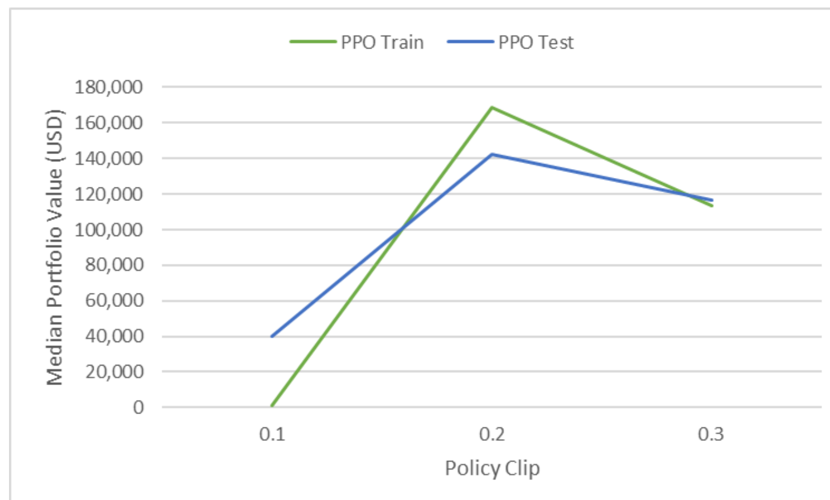
PPO (on CPU)

Hyperparameter Optimisation

- **Grid search** approach for key parameters
- **DQN:** Learning rate, gamma, **epsilon decay**
- **PPO:** Learning rate, epochs, **policy clip**



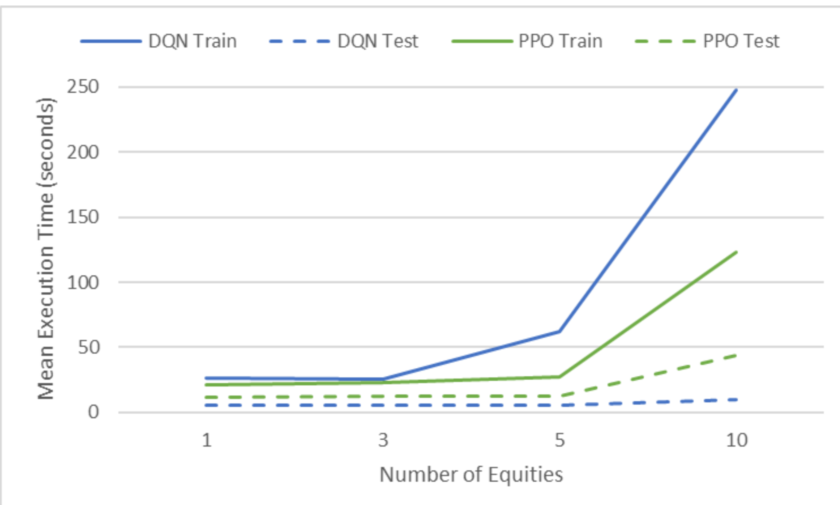
DQN (on GPU)



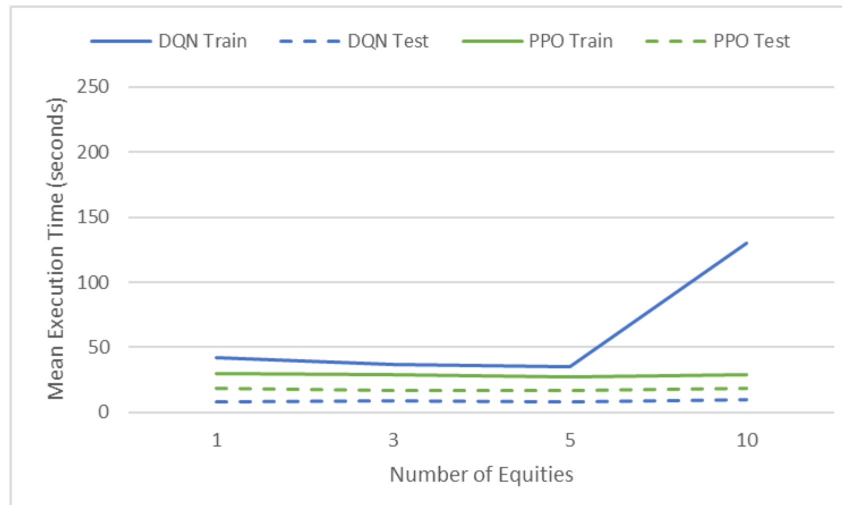
PPO (on GPU)

Scaling Assets

- **Testing** with increasing number of assets (1, 3, 5, 10)
- **Focus** on execution time
- **Comparison** between CPU and GPU performance



CPU Node



GPU Node

Robustness / Generalisation

- **Training** on one set, testing on different sets
- **Datasets:** Baseline, similar equities, commodities ETFs
- **Model** adaptability and performance stability

Dataset	Buy-and-Hold	DQN	PPO
Train	24.1%	20.5%	6.3%
Test	9.3%	6.7%	7.6%
Similar	30.3%	3.7%	18.6%
Commodities	6.7%	0.1%	0.0%

Compound Annual Growth Rates (CAGRs)

Key Findings

- **PPO** outperformed DQN in generalisation and scalability
- **Both** algorithms profitable, often underperformed buy-and-hold strategy
- **Hyperparameter tuning** and GPU acceleration vital for performance (especially with large portfolios)
- **Limited** transferability to different asset classes

Review of Objectives

- **Implemented,** compared DQN + PPO on HPC for trading
- **Gained** insights → performance factors, scalability
- **Demonstrated** potential of DRL for trading on HPC
- **Identified** limitations, areas for improvement in DRL trading models

Conclusions

- **DRL** programs show promise for automated trading, requires further refinement
- **Limitations:** Daily data, PyTorch, specific HPC system
- **Future Directions:** TorchRL, specialised hardware, risk management, forward testing

Q&A

- Thank you for listening!
- Any questions?

References

[1] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, Massachusetts: The MIT Press, 2018.

[2] A. Nair et al., “Massively parallel methods for deep reinforcement learning,” arXiv.org, https://arxiv.org/abs/1507.04296 (accessed August 24, 2024).

[3] N. Mohi Ud Din, A. Assad, and N. Firdous, "An imbalanced classification approach for establishment of cause-effect relationship between Heart-Failure and Pulmonary Embolism using Deep Reinforcement Learning," Engineering Applications of Artificial Intelligence, 2023. [Online]. Available: https://doi.org/

[4] The University of Edinburgh, "High Performance Computing services," https://www.epcc.ed.ac.uk/high-performance-computing-services, 2024.

Appendix A: Baselines (CPU)

CPU Node	DQN		PPO	
	Train	Test	Train	Test
Median Portfolio Value (USD)	135,324	111,366	113,413	131,217
Mean Execution Time (seconds)	138.64	28.33	113.62	58.28
Mean CPU Memory Usage (total) (MB)	1,014.72	4.43	851.42	10.20
Mean Consumed Energy (joules)	29,810	13,310	27,940	14,460

CPU baseline metrics

Appendix B: Baselines (GPU)

GPU Node	DQN		PPO	
	Train	Test	Train	Test
Median Portfolio Value (USD)	161,144	126,406	120,193	124,652
Mean Execution Time (seconds)	170.61	31.28	136.22	77.61
Mean CPU Memory Usage (total) (MB)	94.22	1.34	3.94	0.31
Mean GPU Memory Usage (total) (MB)	89,678.63	22,198.89	43,165.49	41,484.13
Mean Consumed Energy (joules)	55,630	12,990	43,070	26,380
Mean Power (watts)	62.37	62.00	61.96	61.93
Mean GPU Utilisation (%)	11.40	6.14	11.48	11.47

GPU baseline metrics

Appendix C: Initial Hyperparameters

Hyperparameter	Value
Number of assets	3
Initial investment ($)	100000
Transaction cost rate	0.02
Batch size	32
Discount factor (gamma)	0.99
Learning rate (alpha)	0.0003

Common hyperparameters

Hyperparameter	Value
Replay buffer size	500
Initial epsilon	1
Minimum epsilon	0.01
Epsilon decay rate	0.995

DQN-specific

Hyperparameter	Value
Steps between learning updates (N)	128
GAE lambda	0.95
Policy clip	0.2
Number of epochs	4

PPO-specific

Appendix D: Best Hyperparameters

Algorithm	Hyperparameter	Value
DQN	Learning Rate	0.001
	Gamma	0.95
	Epsilon Decay	0.99
PPO	Learning Rate	0.0003
	Epochs	4
	Policy Clip	0.2

Best hyperparameters found from tuning DQN and PPO